THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)*

Présentée et soutenue le *15/05/2013* **par :**

MAHUNA AKPLOGAN

Approche modulaire pour la planification continue.
Application à la conduite des systèmes de culture

JURY

MARTIN COOPER	Professeur des Universités	Président
ALAIN DUTECH	Chargé de Recherche HDR	Rapporteur
FRÈDÈRICK GARCIA	Directeur de recherche	Directeur de thèse
DAVID HILL	Professeur des Universités	Examinateur
ERIC JACOPIN	Maître de conférences HDR	Invité
ANNE MEROT	Chargé de recherche	Examinateur
MARC TCHAMITCHIAN	Chargé de Recherche HDR	Rapporteur

École doctorale et spécialité :
Mathématiques, Informatique, Télécommunications de Toulouse (M.I.T.T) : Intelligence Artificielle

Unité de Recherche :
INRA - Unité de Mathématiques et Informatique Appliquées de Toulouse (MIAT-UR0875)

Directeur(s) de Thèse :
Frèdèrick GARCIA, *Alexandre JOANNON* et *Gauthier QUESNEL*

Rapporteurs :
Alain DUTECH et *Marc TCHAMITCHIAN*

Je dédie ce travail à **Bernard** *et* **Marie-Claude AKPLOGAN** *mes parents. Tous les mots du monde ne sauraient exprimer ma profonde gratitude pour tous les efforts et les sacrifices que vous avez consentir pour mon instruction et mon bien-être. J'espère avoir répondu aux espoirs que vous avez fondés en moi. Je vous rends hommage par ce modeste travail en guise de ma reconnaissance éternelle et de mon infini amour. Vous résumez si bien le mot parents qu'il serait superflu d'y ajouter quelque chose que « merci d'illuminer mon chemin »*

Remerciements

Mes remerciements iront tout d'abord à Frédérick Garcia, mon directeur de thèse. Fred, tu m'as non seulement offert l'occasion de vivre cette aventure, mais tu t'es investi, pour que j'obtienne un maximum de soutien. Il est certain que les idées constructives que tu as émis durant les moments de discussion ont beaucoup contribuées à l'aboutissement de ce travail. Tes conseils ont bien souvent été décisifs.

Comment pourrais-je ne pas remercier mes deux co-directeurs Gauthier Quesnel et Alexandre Joannon ? Votre soutien durant ses années de thèse m'a permis de mener à bien cette aventure. L'impact que vous avez eu sur ce travail de recherche a largement dépassé le cadre de vos fonctions. Je tiens à vous exprimer ma plus profonde gratitude, pour avoir dirigé ces travaux et m'avoir soutenu même dans les moments les plus difficiles. Votre patience m'a beaucoup aidé. Vos conseils ont été indispensables à la concrétisation de cette recherche. Cette thèse est aussi la vôtre.

Je remercie également mes deux rapporteurs Alain Dutech et Marc Tchamitchian. Je sais combien vos agendas sont remplis, et pourtant vous n'avez ménagé aucun effort pour rapporter ce travail. Vous m'avez fait l'honneur de lire ce manuscrit et j'en suis reconnaissant. Vos remarques positives et vos critiques m'ont permis de prendre du recul sur mon travail.

Je tiens aussi à remercier Martin Cooper, David Hill, Anne Merot et Eric Jacopin pour avoir accepté de faire partie du jury. Vous m'avez fait l'honneur de participer à l'évaluation de ce travail et j'en suis reconnaissant. Je te remercie doublement Eric pour ta franchise, ta disponibilité, tes conseils et les nombreuses critiques qui tu as formulé tout au long de la thèse.

Comment pourrais-je ne pas exprimer ma plus profonde gratitude à Simon de Givry ? Tu m'as donné la possibilité de rehausser la qualité de ce travail. Tu ne sauras jamais oh combien a été précieuse ton aide. « Merci infiniment Simon ».

J'adresse également ma reconnaissance à Thomas Schiex, Roger Martin-Clouaire, Régis Sabbadin, David Allouche pour les conseils que vous m'avez apportés tout au long de la thèse. Je n'oublie pas l'ensemble des membres de l'unité mathématiques et informatique appliquées de l'INRA Toulouse pour l'ensemble des échanges que nous avons eu.

Merci également à Aurélie Favier, Ronan Trépos, Nguyen Hiep, Éric Audemard, Jimmy Vandel, Mathieu BONNEAU et Damien Leroux, Jérôme Dury et Morayo Adédjouma, qui de par leur collaboration ont joué un rôle non négligeable dans l'aboutissement de ce travail.

Je tiens à faire savoir aux personnes ci-dessous combien précieuse a été leur présence et à quel point je leur sais gré de ne pas s'être trop plaint de l'être trop absent que je suis devenu lors de ces derniers années. Je pense à Mikkael, Koura, Esprit (Brice), William, ... Il serait sans doute trop long de faire une liste exhaustive. Mes pensées vont à l'endroit de tous ceux dont les noms ne figurent pas sur cette liste.

Avant de terminer, je remercie les deux êtres chers qui m'ont accompagnés nuit et jour, qui ont tant souffert de mes absences répétées qui ne se sont jamais (... plutôt rarement) plaint. Clarice et Bwerani cette thèse est aussi la vôtre ; merci pour votre patience.

Enfin, je remercie ma famille (Bernard, Marie-Claude, Honorat et Clarisse) pour l'affection et la patience que vous m'avez accordées depuis le début de mes études. Vous m'avez toujours soutenu. Merci pour ce travail qui ne fût pas le mien, mais le nôtre.

Table des matières

Table des matières **vii**

Notations **xiii**

Liste des figures **xiv**

Introduction générale **1**
Contexte 1
Question de la thèse 4
Contributions majeures de la thèse 4
Organisation de le thèse 5

1 Dynamique de la décision dans les systèmes de culture à l'échelle de l'exploitation **7**
1.1 Description des concepts relatifs à la conduite de systèmes de culture par un agent agriculteur 8
1.1.1 Pour commencer. 8
1.1.2 Dynamique de la décision chez un agent agriculteur 14
1.2 Problème d'allocation des cultures : décision stratégique 16
1.2.1 Description du problème 16
1.2.2 Formulation du problème 17
1.2.3 Utilité des plans stratégiques : les critères d'analyse 18
1.3 Choix du mode de conduite des cultures : décisions tactiques 19
1.3.1 Description du problème 19
1.3.2 Formulation du problème 20
1.3.3 Utilité des plans tactiques : les critères d'analyse 23
1.4 Organisation du travail : décision opérationnelle 23
1.4.1 Description du problème 23
1.4.2 Formulation du problème 26
1.4.3 Utilité des plans opérationnels : les critères d'analyse 26
1.5 Quelques travaux sur la simulation bio-décisionnelle en agronomie 27
1.5.1 Décision stratégique 27
1.5.2 Décisions tactique et opérationnelle 27
1.6 Exemple d'exploitation virtuelle 28
1.6.1 Description structurelle de l'exploitation virtuelle 28
1.6.2 Les ressources de l'exploitation virtuelle 29
1.6.3 Opérations agricoles et modes de conduite 30
1.6.4 Profil de l'agriculteur 35
1.7 Résumé du chapitre 36

2 Concepts généraux sur la planification et la modélisation & simulation en DEVS **39**
2.1 Planification de tâches et contrôle d'exécution 41
2.1.1 Introduction générale sur la planification 41
2.1.2 Planification par satisfaction de contraintes pondérées 46
2.1.3 Planification hiérarchique 49
2.1.4 Architecture robotique pour la planification dans les systèmes complexes autonomes 52
2.1.5 Exécution de plan en planification 56
2.1.6 Quelques approches orientées modèles 59
2.1.7 Discussion sur la planification et l'exécution de plans temporels . . 63
2.2 Multi-modelisation & Simulation 64
2.2.1 Modèle atomique DEVS 64
2.2.2 Modèle couplé DEVS 66
2.2.3 Limites de DEVS classique 67
2.2.4 Simulation d'agent en DEVS 67
2.2.5 Plateforme de simulation VLE 68
2.3 Résumé du chapitre 70

3 Description du cadre de simulation : Safihr **73**
3.1 L'architecture des systèmes de production agricole 74
3.1.1 Vue d'ensemble 74
3.1.2 Système biophysique 74
3.1.3 Système opérant 74
3.1.4 Système agent 75
3.1.5 Interaction et message entre systèmes 75
3.2 Le système de décision 77
3.2.1 Caractérisation du cycle de décision à simuler 77
3.2.2 Définition de l'agent 78
3.2.3 L'architecture Safihr 79
3.3 Les états de croyance de l'agent $\mathcal{B}$: description et formalisation DEVS 81
3.3.1 Caractéristique des variables d'état 81
3.3.2 Représentation des variables d'état de croyance 82
3.3.3 Les opérateurs sur les relations 83
3.3.4 Fonctions de correspondance sur les objets 85
3.3.5 Formalisation DEVS du système $\mathcal{B}$ 85
3.4 Résumé du chapitre 86

4 Planification d'allocation des cultures : approche basée sur les WCSP **87**
4.1 État de l'art sur les contraintes globales 88
4.1.1 Les réseaux de contraintes pondérées 88
4.1.2 La contrainte globale Regular 89
4.1.3 La contrainte globale de cardinalité Gcc 92
4.1.4 La contrainte globale de cardinalité relaxée Soft-gcc 95
4.1.5 La contrainte globale Same 97
4.2 Description détaillée des contraintes pour la planification stratégique 99
4.2.1 Caractéristiques et hypothèses de notre approche de résolution . . 99
4.2.2 Description des contraintes 100
4.2.3 Variables et domaines du problème d'allocation de cultures 103

4.3 Formalisation des contraintes agronomiques . . . 104
4.3.1 Contraintes de zone cultivable - h-SCC . . . 104
4.3.2 Historique des parcelles élémentaires - h-HST . . . 104
4.3.3 Délai de retour - h-TSC . . . 105
4.3.4 Rotation culturale - h-CCS . . . 107
4.3.5 Collection de cultures par bloc - h-SCA . . . 108
4.3.6 Qualité des séquences de cultures : effet précédent - s-CSQ . . . 110
4.4 Formulation des contraintes organisationnelles . . . 111
4.4.1 Interdépendance entre parcelles élémentaires - h-EQU . . . 111
4.4.2 Préférence topologique - s-TOP . . . 111
4.4.3 Capacité de ressources - h-RSC . . . 112
4.5 Les objectifs de production . . . 113
4.5.1 Proportion annuelle par culture - s-SBC . . . 113
4.5.2 Proportion pluri-annuelle des cultures par parcelle élémentaire - s-TBC . . . 114
4.6 Application . . . 114
4.6.1 Description des instances du problème d'allocation de cultures . . . 114
4.6.2 Analyse des résultats . . . 118
4.6.3 Analyse des solutions trouvées pour l'instance B1[1-4]-LU15(*) . . . 120
4.6.4 Analyse des solutions trouvées pour l'instance B1[1-4]-LU30(*) . . . 121
4.6.5 Analyse des solutions trouvées pour l'instance B1[1-4]-LU60(*) . . . 124
4.7 Conclusion et discussion . . . 127

5 Planification des opérations agricoles et gestion des ressources **131**
5.1 Résolution des problèmes de décisions tactique et opérationnelle . . . 133
5.1.1 Rappel du problème de décision tactique . . . 133
5.1.2 Rappel du problème de décision opérationnelle . . . 134
5.1.3 Choix des approches de résolution . . . 135
5.2 État de l'art sur la planification hiérarchique temporelle 135
5.2.1 Les réseaux de tâches hiérarchiques . . . 135
5.2.2 Propagation des contraintes temporelles en planification . . . 138
5.2.3 Prise en compte de contraintes de ressources . . . 145
5.2.4 Bilan . . . 151
5.3 Représentation du domaine de planification tactique . . . 151
5.3.1 Prise en compte des effets . . . 151
5.3.2 Représentation des itinéraires techniques . . . 151
5.4 Affectation des ITKs et propagation des contraintes temporelles . . . 157
5.4.1 Définition des buts de la planification tactique . . . 157
5.4.2 Heuristique de décomposition du réseau de tâches . . . 158
5.4.3 Algorithme de planification tactique . . . 163
5.5 Application de l'approche de planification tactique . . . 165
5.5.1 Contexte des expérimentations . . . 166
5.5.2 Analyse des résultats . . . 168
5.5.3 Bilan sur la planification tactique . . . 175
5.6 Planification de la décision opérationnelle . . . 175
5.6.1 Modélisation du problème de décision opérationnelle . . . 176
5.6.2 Algorithme de résolution . . . 178
5.7 Application de l'approche de planification opérationnelle 179
5.7.1 Contexte des expérimentations . . . 181

5.7.2 Analyse des résultats . . . 181
5.8 Bilan du chapitre . . . 186

6 Exécution distribuée de plans temporels en DSDE **189**
6.1 État de l'art sur les formalismes PDEVS et DSDE . . . 191
6.1.1 Formalisme DEVS parallèle . . . 191
6.1.2 Formalismes DEVS à structures dynamiques DS-DEVS et DSDE . 192
6.2 Interactions entre le coordinateur et l'exécutif de plan 194
6.2.1 Différence terminologique entre exécutive DEVS et exécutif de plan 195
6.2.2 Coordinateur et l'exécutif de plan . . . 195
6.2.3 Description des tâches opérationnelles $\mathcal{A}$ et de planification $\mathcal{P}$ . . . 196
6.3 Formalisation PDEVS des tâches opérationnelles $\mathcal{A}$ et de planification $\mathcal{P}$ . . . 198
6.3.1 Interface des modèles génériques de tâches . . . 198
6.3.2 L'état d'un modèle de tâches . . . 199
6.3.3 Initialisation du modèle et transition de *Init* à *Wait* . . . 200
6.3.4 Transition du statut *Wait* vers *Started* ou *Failed* . . . 201
6.4 Formalisation PDEVS des éléments de l'interpréteur du plan courant π . . . 203
6.4.1 Types de relation entre les tâches du plan courant π . . . 203
6.4.2 Différentes interprétations des relations basées sur les événements **S** et **F** . . . 206
6.4.3 Types de modèles et interprétation en ligne des relations temporelles 207
6.4.4 Formalisation DEVS des modèles de contraintes de précédence . . 208
6.4.5 Formalisation DEVS des modèles de contraintes de synchronisation 210
6.4.6 Quelques exemples de relations temporelles entre tâches . . . 210
6.5 Fonctionnement du coordinateur : $\langle \chi, M_\chi \rangle$ . . . 212
6.5.1 Interface du modèle du coordinateur M_χ . . . 212
6.5.2 Description des états partiels s_χ^α . . . 213
6.5.3 Formalisation du coordinateur M_χ . . . 214
6.6 Application . . . 218
6.6.1 Contexte des expérimentations . . . 219
6.6.2 Quelques résultats de simulation . . . 219
6.7 Bilan sur le chapitre . . . 225

Conclusion générale **227**
Démarche suivie . . . 227
Aperçu global des apports de la thèse . . . 228
Perspectives . . . 230

A Annexes **233**
A.1 Description des opérations agricoles . . . 234
A.2 Discrete EVent system specification . . . 236
A.2.1 Fermeture sous couplage de DEVS classique . . . 236
A.2.2 Fermeture sous couplage de DEVS parallèle . . . 236
A.3 Analyse des solutions d'assolement : proportions de cultures . . . 239
A.3.1 Instance « B1234-LU15-ALL-star » : blocs fonctionnels 1,2,3 et 4 en interdépendance . . . 239
A.3.2 Instance « B1234-LU30-ALL-star » : blocs fonctionnels 1,2,3 et 4 en interdépendance . . . 239

A.3.3 Instance « B1234-LU60-ALL-star » : blocs fonctionnels 1,2,3 et 4 en interdépendance 242
A.4 Langage de description du domaine 248
A.4.1 Représentation de tâches primitives non duratives 248
A.4.2 Représentation de tâches primitives duratives 249
A.4.3 Les tâches composées 250
A.4.4 Exemple d'un domaine décrivant l'ITK de Maïs 250
A.5 Plans opérationnels 254
A.5.1 Instance « B[1-4]-LU30(*) » 254
A.5.2 Instance « B[1-4]-LU60(*) » 256

Index **257**

Bibliographie **259**

Notations

CSP	constraint satisfaction problem
DEVS	discrete event system specification
DS-DEVS	dynamic dtructure DEVS
DSDE	parallel dynamic structure discrete event system specification
DTJ	durée de travail journalier
HTN	hierarchical task network
ITK	itinéraire technique
PDEVS	parallel discrete event system specification
STN	simple Temporal Networks
TCSP	temporal constraint satisfaction problem
WCSP	weighted constraint satisfaction problem
δ	fonction de transition
$Is^t_{b,c}$	ensemble des parcelles élémentaires du bloc b affectées à la culture c à la date t
Is^t	ensemble des îlots fonctionnels de l'exploitation à la date t
$\mathbb{N}, \mathbb{N}^*$	ensemble des entiers naturels, des entiers strictement positifs
$\mathbb{R}, \mathbb{R}_+$	ensembles des réels et des réels positifs
$\|X\|$	cardinal de l'ensemble X
$x^t_{b,i}$	parcelle élémentaire i du bloc fonctionnel b à la date t
$x^{t,k}_{i,j}$	variables de décision binaires qui vaut 1 si à la date t, la ressource r_k est affectée à la tâche j de la parcelle i.
I	ensemble des entrées d'un système
O	ensemble des sorties d'un système
S	ensemble des états
π	plan courant

Liste des figures

1 Structure du manuscrit 6

1.1 Ontologie simplifiée des systèmes de culture au sein d'une exploitation agricole 8
1.2 Schéma illustratif de la production de l'orge de printemps OP ($dr(OP) = 3$) et du blé d'hiver BH ($dr(BH) \in [2-3]$) 11
1.3 Exemple de classes de rotations extrait de Castellazzi et al. (2008) . 12
1.4 Différence entre îlot fonctionnel et bloc fonctionnel. 13
1.5 Les niveaux d'abstraction de la décision dans les systèmes de culture au sein de l'exploitation agricole 14
1.6 Motif d'ITK décrivant une séquence fixe 22
1.7 Motif d'ITK décrivant une séquence flexible 22
1.8 Motif d'ITK décrivant des branches parallèles fixes 22
1.9 Motif d'ITK décrivant des branches parallèles flexibles 22
1.10 Motif d'ITK décrivant une itération 22
1.11 Calendrier simplifié de l'organisation du travail dans les exploitations agricoles du bassin versant de Bourville (Joannon, 2004) 25
1.12 Exploitation virtuelle de 4 blocs fonctionnels, 15 parcelles de 12ha chacune. Les parcelles grises ont accès à des ressources en eau 29
1.13 Itinéraires techniques du blé d'hiver 32
1.14 Itinéraires techniques de l'orge de printemps 32
1.15 Itinéraires techniques de maïs 33
1.16 Itinéraires techniques de colza d'hiver 35

2.1 Relations primitives d'Allen. 43
2.2 Représentation PDDL d'une opération de semis de maïs 45
2.3 Réseau de valeurs de la contrainte $(x_0 + x_1 + x_2) < x_3$ 47
2.4 Architecture *Sense/Plan/Act* 52
2.5 Architecture *trois tiers* 54
2.6 Architecture *CLARAty* 55
2.7 Planification successive vs planification continue 57
2.8 Horizon de planification hiérarchique Chien et al. (2000) 59
2.9 Traduction TPN des concepts de base de RMPL Kim et al. (2001) . 61
2.10 Structure d'un agent *IDEA* selon Dias (2003) 62
2.11 Structure d'un agent *IDEA* selon Finzi et al. (2004) 63
2.12 Représentation graphique d'un modèle atomique DEVS 65
2.13 Échantillonnage d'un signale sinusoïdale 65
2.14 Représentation graphique d'un modèle couplé DEVS 66
2.15 Graphe de dépendances entre les bibliothèques, programmes et composants de la plateforme VLE. 69

3.1 Modèle d'interaction des éléments d'un système de production agricole 74
3.2 Format des messages du système agent vers le système opérant . . . 76

3.3 Format des messages de notification du système opérant vers le système agent . . . 77
3.4 Format des messages d'observations du système opérant vers le système agent . . . 77
3.5 Cycle de décision à simuler . . . 78
3.6 Architecture du système de décision SAFIHR . . . 79
3.7 Interface d'une modèle DEVS définissant une relation . . . 85

4.1 Exemple d'automate à états finis déterministe . . . 89
4.2 Graphe en couches pour l'automate de la Figure 4.1 . . . 90
4.3 Graphe en couches pour l'automate de la Figure 4.1 après la phase ascendante . . . 91
4.4 Graphe en couches pour l'automate de la Figure 4.1 après la phase descendante . . . 91
4.5 Exemples de réseau de transport (a), un flot de valeur 7 (b) et un réseau résiduel (c) . . . 93
4.6 Exemple d'un réseau de transport représentant une contrainte GCC . 95
4.7 Example d'un réseau de transport représentant une contrainte SAME 98
4.8 Représentation schématique des concepts spatio-temporels du problème de décision (t_i : année, b : bloc, p_j : parcelle, $x_{b,i}$: parcelle élémentaire, k_p : effet précédent) . . . 99
4.9 DFA décrivant le langage associé au délai de retour de la culture CH avec ($dr(CH) = 3$ et $h = 5$). a désigne toutes les valeurs de $D_{b,i}$. La notation $\overline{CH}$ correspond à $D_{b,i} \setminus \{CH\}$. Le langage associé accepte toutes les séquences sur $\mathcal{H}$ pour lesquelles le délai de retour est respecté pour l'ensemble des variables du futur (ex : CH-OP-CH-OP-CH-BH-OP-CH-BH). . . . 107
4.10 DFA décrivant le langage associé au délai de retour du colza d'hiver (CH) en mode rotation ($dr(CH) = 3$) . . . 109
4.11 Représentation sous forme de graphes bipartis des collections de cultures par bloc fonctionnel . . . 110
4.12 Qualité des successions culturales . . . 110
4.13 Exemple de préférence topologique sur un ilot structurel . . . 111
4.14 Exploitation virtuelle échantillonnée en 15 parcelles élémentaires de 12ha . . . 115
4.15 Exploitation virtuelle échantillonnée en 15 parcelles élémentaires de 6ha . . . 115
4.16 Exploitation virtuelle échantillonnée en 15 parcelles élémentaires de 3ha . . . 116
4.17 Exploitation virtuelle échantillonnée en 15 parcelles élémentaires de 12ha . . . 116
4.18 Proportion annuelle des cultures pour les différentes années. . . . 120
4.19 Allocation des cultures pour l'instance B1[1-4]-LU15(*) . . . 121
4.20 Allocation des cultures pour l'instance B1[1-4]-LU30(*) . . . 123
4.21 Bloc 1 : séquences de cultures sur les parcelles élémentaires 1, 8, 9 et 16 . . . 124
4.22 Bloc 3 : séquences de cultures sur les parcelles élémentaires 25, 32, 33 et 40 . . . 125
4.23 Bloc 2 : séquences de cultures sur les parcelles élémentaires 17, 20, 21 et 24 . . . 125
4.24 Bloc 4 : séquences de cultures sur les parcelles élémentaires 41, 43, 47 et 60 . . . 125

4.25 Distribution des proportions annuelles de BH sur l'ensemble de solutions . . . 126
4.26 Distribution des proportions annuelles de MA sur l'ensemble de solutions . . . 126
4.27 Distribution des proportions annuelles de OP sur l'ensemble de solutions . . . 127
4.28 Distribution des proportions annuelles de CH sur l'ensemble de solutions . . . 127
4.29 Illustration de 4/136 plans stratégiques, solutions de l'instance B[1-4]-LU60-ALL(*) : ■ Blè d'hivers, ■ Orge de printemps, ■ Maïs, ■ Colza d'hivers . . . 128

5.1 Principe du lien entre la planification stratégique et la planification tactique. . . . 133
5.2 Principe du lien entre la planification tactique et la planification opérationnelle . . . 134
5.3 Réseau de contraintes temporelles simples STN. . . . 141
5.4 (a) Transformation des relations d'un STN en contraintes dans le graphe de distance, (b) Graphe de distance associé au STN de la Figure 5.3. . . . 141
5.5 Exemple d'un SPPRC avec une ressource. . . . 147
5.6 Application de l'algorithme 8 sur le graphe de la Figure 5.5. . . . 150
5.7 Graphe représentant le réseau de contraintes temporelles simples (STN) de l'ITK ↓ *travail* du maïs . . . 156
5.8 Graphe multivalué représentant l'ESPPRC à résoudre pour l'affectation des ITKs aux îlots fonctionnels . . . 162
5.9 Plans stratégiques : ■ Blé d'hiver, ■ Orge de printemps, ■ Maïs, ■ Colza d'hiver . . . 166
5.10 Modes de conduite des cultures par îlots fonctionnels : B[1-4]-LU15(*) 169
5.11 Modes de conduite des cultures par îlots fonctionnels : B[1-4]-LU30(*) 170
5.12 Modes de conduite des cultures par îlots fonctionnels : B[1-4]-LU60(*) 170
5.13 Consommation de ressources pour les années 6, 7, 8 et 9 : ■ B[1-4]-LU15(*), ■ B[1-4]-LU30(*), ■ B[1-4]-LU60(*), ■ Capacités limites des ressources . . . 171
5.14 Modes de conduite des cultures par îlots fonctionnel pour 45% de jours disponibles : B[1-4]-LU60(*) . . . 173
5.15 Modes de conduite des cultures par îlots fonctionnels et pour 35% de jours disponibles : B[1-4]-LU15(*) . . . 173
5.16 Exemple d'un modèle de graphe pour la planification opérationnelle 179
5.17 Performance pour la recherche des plans opérationnels . . . 182
5.18 Instance B1[1-4]-LU15(*) : plan opérationnel obtenus pour 6 semaines de l'année 1961 . . . 183
5.19 Évolution de l'état des ressources sur la semaine du 13 au 26 février 185
5.20 Évolution de l'état des ressources sur la semaine du 3 au 9 juillet . . 185

6.1 Coordinateur et exécutif distribué de plans temporels dans SAFIHR . 195
6.2 Automate de transition entre statut. . . . 197
6.3 Les quatre configurations des contraintes temporelles absolues d'une tâche opérationnelle. . . . 197
6.4 Interface générique des tâches opérationnelles et de planification. . . 199
6.5 Interface générique des modèles de relation temporelle . . . 208
6.6 Statut des modèles de relations temporelles. . . . 209

6.7 $(a, \prec \mathbf{FS}, b)$. L'événement de fin f_a précède le début s_b. 211
6.8 $(a, \prec \mathbf{SF}, b)$. L'événement de début s_a précède la fin f_b. 211
6.9 $(a, \prec \mathbf{SS}, b)$. L'événement de début s_a précède le début s_b. 211
6.10 $(a, \prec \mathbf{FF}, b)$. L'événement de fin f_a précède la fin f_b. 211
6.11 $(a, \prec \mathbf{FS} \succ, b)$. Synchronisation sur les événements f_a et s_b. 211
6.12 $(a, \prec \mathbf{SS} \succ, b)$. Synchronisation sur les événements s_a et s_b. 211
6.13 $(a, \prec \mathbf{SF} \succ, b)$. Synchronisation sur les événements s_a et f_b 211
6.14 $(a, \prec \mathbf{SS} \succ, b)$. Synchronisation sur les événements f_a et f_b 211
6.15 $((a \wedge b) \prec \mathbf{FS}, X) \bigwedge ((C \wedge D) \prec \mathbf{SS}, X) \vee (E \wedge F) \prec \mathbf{FS}, X))$ 212
6.16 Interface du modèle M_χ du coordinateur. 213
6.17 Dates et horizons de planification pour l'instance B1[1-4]-LU15(*) . 224
6.18 Dates et horizons de planification pour l'instance B1[1-4]-LU30(*) . 224
6.19 Dates et horizons de planification pour l'instance B1[1-4]-LU60(*) . 224

A.1 Ensemble des solutions de l'instance B1234-LU15-ALL(*) : ■ Blè d'hivers, ■ Orge de printemps, ■ Maïs, ■ Colza d'hivers 239
A.2 Sept des douze des solutions de l'instance B1234-LU30-ALL(*) : ■ Blè d'hivers, ■ Orge de printemps, ■ Maïs, ■ Colza d'hivers 241
A.3 7 des 136 des solutions de l'instance B1234-LU60-ALL(*) : ■ Blè d'hivers, ■ Orge de printemps, ■ Maïs, ■ Colza d'hivers 247
A.4 Instance B[1-4]-LU30(*) : plan opérationnel obtenus pour 6 semaines de l'année 1961 255
A.5 Instance B[1-4]-LU60(*) : plan opérationnel obtenus pour 6 semaines de l'année 1961 256

Introduction générale

Contexte

De nos jours, les enjeux environnementaux, en partie liés aux changements climatiques, imposent une remise en question des modes de production agricoles conventionnels. Les processus de décision usuels des agriculteurs, relatifs à l'organisation spatio-temporelle des cultures et des techniques de conduite des cultures à l'échelle de l'exploitation agricole, doivent être profondément réexaminés. En effet, les décisions d'un agriculteur impactent directement les ressources naturelles (ex : l'érosion du sol, la biodiversité, la dissémination des OGM) et la qualité des produits. De plus, la tendance actuelle va vers une réduction des intrants chimiques (Plan Ecophyto). Celle-ci pousse à une reconception en profondeur des modes de production agricole. En l'absence de solutions viables et compatibles avec les nouveaux défis, on assistera à l'accentuation de la vulnérabilité, en terme de production, de la plupart des exploitations agricoles. Pour faire face à ces nouveaux enjeux, l'adoption de techniques de production innovantes est devenue incontournable.

Au cours des dernières années, les chercheurs en agronomie se sont intéressés à ces problèmes émergents. Ils ont développé des modèles conceptuels relatifs à une gestion des systèmes de production agricole, dans l'objectif de favoriser une adaptation durable des systèmes agricoles aux changements environnementaux et économiques.

Depuis quelques années, de nombreux travaux en informatique (Attonaty et al., 1993; Leroy et al., 1997; Bergez et al., 2001; Muetzelfeldt et Massheder, 2003; Bolte et al., 2003; Chatelin et al., 2005; Martin-Clouaire et Rellier, 2009) ont été menés afin d'exploiter les modèles conceptuels agronomiques et aider les agronomes à analyser, évaluer et concevoir par simulation des systèmes intégrant l'agriculteur et les systèmes physiques qu'il contrôle. Ces travaux sont essentiels aujourd'hui où l'utilisation de la simulation devient de plus en plus prépondérante dans l'étude des systèmes complexes comme le sont les systèmes de production agricole.

Cependant, à ce jour, seules les règles de décision sont communément utilisées dans les modèles de simulation proposés en agronomie. Les approches d'agents à base de plans, de gestion des ressources, d'optimisation, etc. restent faiblement exploitées dans les études centrées sur la simulation. Or, ces approches sont tout à fait pertinentes voire indispensables pour les problématiques visées.

Le projet RECORD

Face aux différents défis que doivent relever les scientifiques et surtout dans l'optique de renforcer les capacités d'innovation de ses chercheurs, l'INRA [1] a lancé un projet de plate-forme informatique permettant de faciliter le développement, le partage et la réutilisation des modèles développés par les agronomes travaillant sur les systèmes de culture.

1. Institut national de la recherche agronomique http://www.inra.fr

Nommé RECORD[2] (Chabrier et al., 2007; Bergez et al., 2012), ce projet a pour objectif d'offrir aux agronomes une plate-forme informatique de modélisation et simulation pour l'aide à la conception et l'évaluation de systèmes de culture innovants. La finalité de ce projet est de faciliter l'étude d'un ensemble de systèmes complexes qui, mis en relation constitue un nouveau système global, avec de nouvelles fonctionnalités qui peuvent être :

- hétérogènes (ex : bio-physique, climat, décision, etc.),
- autonomes c'est-à-dire des systèmes qui fonctionnent de manière indépendantes.

Afin d'appréhender ces différentes caractéristiques, la plate-forme RECORD repose sur l'environnement de modélisation et de simulation VLE (*Virtual Laboratory Environment*[3] Quesnel (2006); Quesnel et al. (2009)) qui a été retenu pour son cadre théorique DEVS (*Discrete EVent System specification* (Zeigler et al., 2000)) formalisant la modularité et le couplage de modèles multi-formalismes, selon une représentation systémique des systèmes dynamiques. Les principes de modularité, de hiérarchisation et de couplage des modèles font de DEVS un formalisme bien adapté pour la modélisation et la simulation des systèmes complexes agronomiques dans lesquels des modèles issus de formalismes différents doivent être simulés dans un même moteur de simulation.

Cette thèse s'inscrit explicitement dans le cadre de ce projet. En accord avec les choix effectués au niveau du projet RECORD, le formalisme DEVS est au centre des développements méthodologiques de la thèse. Ainsi, nous allons modéliser et simuler les processus de décision d'un agriculteur en mobilisant des approches issues de l'intelligence artificielle, afin de représenter le comportement de l'agriculteur face à ses choix techniques et organisationnels, en tenant compte de ses contraintes et de ses objectifs au sein d'un contexte de production partiellement contrôlé. Nous considérons tout au long de cette thèse que la structure de l'exploitation est fixe. Nous ne considérons pas les décisions stratégiques liées au renouvellement et à la modification des ressources de l'exploitation.

Enjeux et objectifs

Dans le cadre de cette thèse, nous nous intéressons à la reproduction de la prise de décision relative à la conduite des systèmes de culture chez un agent agriculteur. Les problèmes de décision qui se posent concernent plus particulièrement la *planification dans le temps et l'espace* des cultures sur l'exploitation, la *planification des tâches de l'agriculteur* et l'*allocation dynamique des ressources matérielles et humaines* aux différentes tâches journalières. Ces décisions sont prises sur la base d'observations partielles de l'environnement et impactent directement l'évolution du système. Notre objectif est de concevoir un agent capable de reproduire par simulation le processus de décision d'un agriculteur. Cet agent doit pouvoir prendre en compte les aspects réactifs, planifiés et adaptatifs à la dynamique de l'environnement de l'agriculteur. Dans cette thèse, nous considérons que la reproduction du processus de décision d'un agriculteur passe avant tout par la résolution en ligne et l'entrelacement des problèmes de décision auxquels il se confronte.

Quel type d'agent pour représenter un agriculteur

De façon synthétique, nous appelons *agent* toute entité autonome physique ou logicielle, ayant des objectifs à atteindre et dotée de facultés sensori-motrices per-

2. http://www.inra.fr/record
3. http://www.vle-project.org

mettant de percevoir et de modifier son environnement. Selon la complexité de l'architecture interne des agents, on distingue classiquement des agents *réactifs* et des agents *délibératifs*.

Les agents réactifs sont régis par des règles comportementales réflexes (stimulus-réponse) et sont incapables d'anticiper ou de planifier leurs propres actions. Les agents réactifs n'ont qu'une vision à court terme sur la résolution des problèmes. N'ayant pas de représentation du monde, ils ne peuvent avoir des buts explicites.

Les agents délibératifs sont capables de raisonner et sont très souvent dotés d'une base de connaissances. Ils ont une représentation explicite de l'environnement. Ces agents sont qualifiés d'*intentionnels* au sens où ils ont un but défini et possèdent un plan explicite pour atteindre ce but. Ces caractéristiques nous poussent à admettre que l'approche délibérative est plus adaptée la conception d'un agent capable de reproduire les processus de décision d'un agriculteur.

Objectifs

... qu'allons nous faire ?

Dans cette thèse, nous allons montrer que le processus de décision d'un agriculteur peu être représenté sous la forme d'un un système hiérarchique dynamique et distribué. En conséquence, l'objectif de nos travaux de recherche sera de concevoir et de construire des systèmes distribués et autonomes. Autrement dit, nous allons nous intéresser à la reproduction du comportement d'un agriculteur pour résoudre de manière automatique différents problèmes de décision. La notion d'**autonomie** que nous soulignons dans cette étude est celle introduite par Russell et Norvig (2010). L'agent impliqué dans notre système sera qualifié d'autonome, si *son comportement est déterminé par ses propres expériences*. L'aspect **distribué** des systèmes que nous souhaitons construire repose sur la capacité de chaque entité à pouvoir s'exécuter indépendamment des autres.

Deux principaux domaines de recherche se retrouvent ainsi confrontés. D'une part, la simulation de systèmes complexes qui offre de nombreux cadres formels pour l'étude ces systèmes. D'autre part, l'intelligence artificielle (IA) qui propose des techniques avancées pour la décision. Pour ce dernier domaine, les problématiques de recherche les plus adaptées sont celles abordant la modélisation d'un agent intelligent observant, décidant, agissant sur un environnement dynamique et incertain. Les méthodes de planification sont largement utilisées pour la résolution des problèmes de décision. Les principaux domaines d'application visés par cette communauté sont la robotique autonome et animale, la simulation d'agents virtuels pour les jeux vidéos ou la réalité virtuelle. Aussi bien dans ces classes de problème que dans les nôtres, la résolution des problèmes ne se limite pas à la planification. Elle impose le contrôle d'exécution des plans produits et la replanification en situation d'échec.

L'une des particularités des problèmes issus du monde agronomique est que l'agriculteur ne dispose pas du modèle des systèmes qu'il tente de contrôler. Il dispose cependant de connaissances spécifiques sur la conduite de son système de production agricole. De manière générale, l'espace de tâches est de taille moyenne et l'horizon de simulation infini. À cela s'ajoute, la nécessité de construire des plans parallèles. En effet, les systèmes de production agricoles sont par essence constitués de plusieurs parcelles pour lesquelles un plan spécifique doit être construit. Pour de nombreux planificateurs, ce *parallélisme* cause des problèmes de temps de calcul ou des limites dû au fait que ces derniers ne peuvent produire que des plans sous la forme de séquences. En considérant l'horizon de planification et la dynamique des systèmes à piloter, il ne nous semble pas pertinent de rechercher des méthodes

classiques de planification. De plus, l'inaccessibilité, par l'agent, du modèle des systèmes à piloter à pour corollaire l'incapacité de l'agent à disposer des effets réels de ses actions.

Dans ces conditions, l'exploitation des connaissances du domaine peut permettre d'améliorer les limites des planificateurs classiques. Les approches intégrant des méthodes capables d'exploiter des modèles de connaissances riches et complexes sont plus adaptées aux problèmes de la conduite des systèmes de culture. Dans cette logique, les méthodes de planifications hiérarchiques semblent être plus appropriées et permettent de *planifier à différents niveaux d'abstraction.*

... qu'allons nous éviter de faire ?

Pour représenter un agent décideur en environnement dynamique et incertain, il existe un grand nombre de méthodes et d'outils en IA. Nous sommes convaincus qu'en caractérisant les problèmes de décision agronomique, nous pouvons les ramener à des problèmes classiques d'IA. Ce faisant, notre objectif ne sera donc pas de développer des techniques « ad hoc » de planification pour l'agronomie mais, d'adapter les approches existantes afin de répondre aux problèmes de décision en agronomie.

Question de la thèse

Peut on réaliser un système capable de simuler un agriculteur dans ses décisions relatives :
- *à l'organisation spatio-temporelle des cultures au sein de l'exploitation agricole (décision stratégique) ?*
- *au choix du mode de conduite des cultures (décision tactique) ?*
- *à l'organisation de la charge de travail au sein de l'exploitation agricole (décision opérationnelle) ?*

Tout l'enjeu de cette thèse se résume à cette question. Cette problématique peut se diviser en deux grands types de questions en informatique. D'une part, il s'agira d'apporter des réponses aux questions de *résolutions des problème de décisions stratégiques, tactiques et opérationnelles en fonction des connaissances de l'agriculteur.* D'autre part, il s'agira de proposer un cadre qui répond à la question de la *reproduction de la dynamique d'interaction entre ces trois types de décision.* L'objectif est également d'exploiter l'aspect modulaire proposé par DEVS qui est une spécification à événements discret pour la modélisation systémique.

Afin d'aborder cette question, nous avons fait le choix dans cette thèse des simuler le processus de décision en lui même. Ceci revient à modéliser les différents problèmes de décision et embarquer dans la simulation les méthodes de résolution. Cette approche nous permet de considérer dans son intégralité le problème de conduite de systèmes agricoles.

Contributions majeures

Cette thèse propose un cadre pour la reproduction du processus de décisions multi-échelles spatiales/organisationnelles et temporelles en agronomie. Elle montre les mécanismes de raisonnements numériques sur le temps et les ressources en agronomie. Les contributions majeures de la thèse peuvent être divisées en quatre catégories :

- ▷ *une architecture modulaire* SAFIHR *:* cette architecture d'agent est conçue pour l'entrelacement en ligne de plusieurs planificateurs spécifiques. SAFIHR intègre les mécanismes permettant de faire coopérer différents planificateurs spécifiques au sein d'un même système. Elle repose sur le cadre des systèmes à événements discrets (DEVS) et permet de voir l'agent comme un système hiérarchique , dynamique et distribué en interaction avec un ensemble de systèmes complexes. Chacun des planificateurs est perçu comme un système de contrôle indépendant.
- ▷ *une formulation « CSP pondérés » pour l'allocation de culture :* cette approche de résolution permet d'aborder l'allocation des cultures comme un problème de satisfaction de contraintes pondérées dans lequel l'utilité de l'allocation est évaluée par une fonction de coût global. Cette approche permet de prendre en compte simultanément et de manière explicite les dimensions spatiales et temporelles du problème d'allocation de culture.
- ▷ *une approche de décomposition globale des réseaux de tâches :* nous proposons un nouvel heuristique permettant d'expliciter les interdépendances entre différentes décompositions d'un réseau de tâches hiérarchiques.
- ▷ *un exécutif distribué de plan temporel en DEVS :* nous proposons une approche systémique pour l'exécution du plan courant de l'agent. L'exécutif est vu comme un système dynamique distribué dans lequel chaque tâche s'exécute de manière indépendante. L'approche proposée se base sur le formalisme DEVS. Elle permet de représenter chaque tâche d'un agent comme des systèmes dynamiques autonomes. Ces systèmes dynamiques interagissent afin de tester en ligne la consistance des contraintes temporelles du plan courant.

ORGANISATION DU MANUSCRIT

La suite de ce manuscrit est divisée en 7 chapitres. Le chapitre 1, introduit certains concepts agronomiques indispensables à la compréhension du manuscrit (cf. section 1.1). Nous y présentons principalement trois problèmes de décision liés à la conduite des systèmes de culture à l'échelle de l'exploitation agricole (cf. sections 1.2, 1.3 et 1.4). Ces problèmes sont ceux abordés dans le cadre de cette thèse. Nous finissons ce chapitre en présentant un cas d'étude (cf. section 1.6) reposant sur une exploitation virtuelle de 180 ha, cultivant 4 cultures annuelles, chacune pouvant être réalisée selon 3 modes de conduites différents.

Le chapitre 2, présente de manière sommaire les concepts généraux d'intelligence artificielle et de modélisation & simulation, nécessaires à la reproduction du comportement d'un agriculteur. L'idée maîtresse est d'apporter l'essentiel de l'information permettant de mieux positionner nos travaux. L'état de l'art sur les méthodes utilisées pour la résolution des problèmes de décision est présenté en début des chapitres correspondant. Ainsi, la section 2.1 est consacrée à la description des méthodes d'IA sur lesquelles nous nous basons pour la résolution des problèmes de décision de l'agriculteur. Nous introduisons brièvement les techniques de planification hiérarchique et de planification par satisfaction de contrainte. Nous consacrons une grande partie de la section à la description des architectures robotiques destinées à la planification dans les systèmes autonomes complexes. La section 2.2 est quant à elle consacrée à la description du cadre de simulation de systèmes dynamiques, distribués et hiérarchiques. Nous y présentons également les travaux dans lesquels l'agent est perçu comme un système dynamique.

Dans le chapitre 3, nous partons d'une architecture classique des systèmes de production agricole et décrivons notre modélisation des interactions entre l'agent

agriculteur et son environnement (cf. section 3.1). La section 3.2 présente une vue d'ensemble de l'architecture SAFIHR. Elle décrit les différents modules qui permettent de modéliser un agent comme un système hiérarchique dynamique.

Le chapitre 4 présente notre approche de résolution du problème d'allocation de culture basée sur les réseaux de contraintes pondérées. Dans un premier temps, nous détaillons les éléments théoriques relatifs à la modélisation et aux tests de cohérence des contraintes globales « REGULAR» , « SAME» , « GCC» et « SOFT-GCC» (cf. section 4.1). Nous décrivons ensuite chacune des contraintes liées aux facteurs agronomiques, organisationnels et de productions (cf. 4.2). La formalisation des contraintes est présentée dans les sections 4.3, 4.4 et 4.5. Enfin, l'application de notre approche sur le cas d'étude est détaillée dans la section 4.6.

Dans le chapitre 5 nous présentons notre approche de résolution des problèmes de décisions relatives au choix du mode de conduite des cultures et l'ordonnancement des opérations agricoles journalières. Pour ce faire, nous détaillons, dans la section 5.2, les éléments théoriques de la planification hiérarchique sous contraintes temporelles et quelques approches classiques pour la prise en compte des ressources en planification. Ensuite, les sections 5.3, 5.4 et 5.6 présentent respectivement nos propositions relatives à la représentation du domaine de planification, l'heuristique de décomposition du réseau de tâches et l'ordonnancement de tâches journalières. Enfin, les sections 5.5 et 5.7 décrivent l'application de notre approche sur le cas d'étude.

Le chapitre 6 est destiné à la présentation du contrôleur d'exécution de plan basé sur le formalisme DEVS. Il introduit dans la section 6.1 deux extensions essentielles du cadre de modélisation et de simulation DEVS (parallèle DEVS et DEVS à structure dynamique). La section 6.2 décrit les mécanismes d'interactions entre le coordinateur et l'exécutif de plan. Les sections 6.3, 6.4 et 6.5 introduisent le fonctionnement du module d'interprétation du plan. Enfin, la section 6.6 décrit l'application de notre approche sur le cas d'étude.

La Figure 1 reprend la structuration du document.

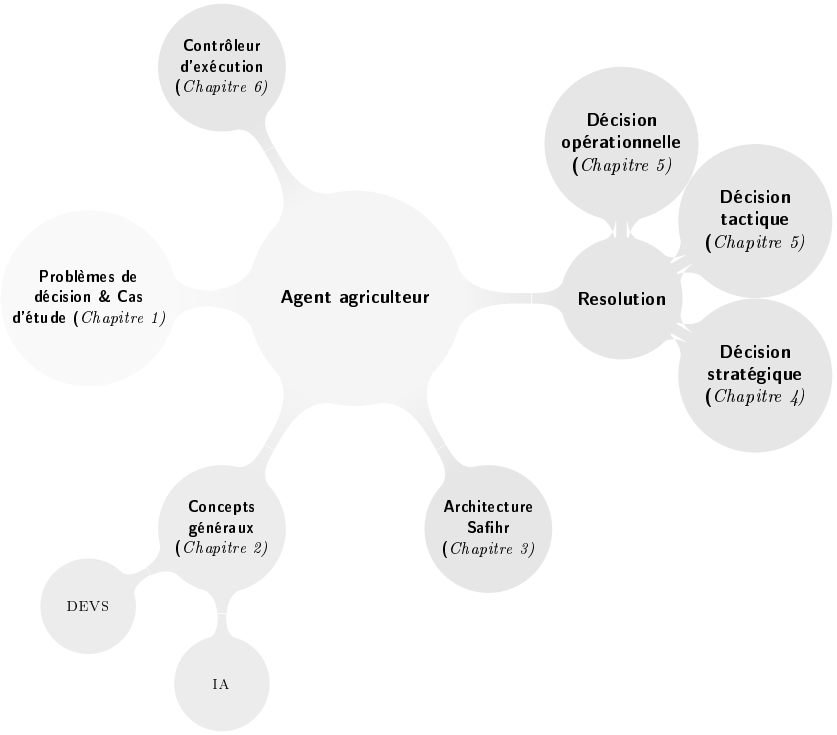


FIGURE 1 – *Structure du manuscrit*

Dynamique de la décision dans les systèmes de culture à l'échelle de l'exploitation

1

Sommaire

Contexte 1
Question de la thèse 4
Contributions majeures de la thèse 4
Organisation de le thèse 5

Ce chapitre introduit les différents problèmes de décision auxquels sont soumis les agriculteurs au sein d'une exploitation agricole. Nous y décrivons également l'exploitation virtuelle qui nous sert de cas d'étude tout au long du manuscrit. Dans un premier temps nous définissons quelques concepts relatifs à la conduite des systèmes de culture. Cette description permet de présenter successivement les trois classes de problèmes de décision qui sont : la décision stratégique, la décision tactique et la décision opérationnelle. Enfin, nous présentons les travaux existants qui abordent des problèmes similaires en agronomie.

1.1 Description des concepts relatifs à la conduite de systèmes de culture par un agent agriculteur

Il existe en agronomie de nombreuses terminologies et définitions permettant de décrire les objets nécessaires à la décision dans les systèmes de culture. Nous parlerons ici de concepts agronomiques relatifs à la conduite de systèmes de culture. Afin de positionner la sémantique que nous accordons à ces concepts, nous définissons dans cette section ceux qui sont déterminants pour cette thèse. Ainsi, avant d'aborder les classes de problèmes de décision présentées dans les sections 1.2, 1.3 et 1.4 nous proposons dans cette section une représentation ontologique simplifiée des concepts relatifs à la conduite de systèmes de culture par un agent agriculteur au sein d'une exploitation agricole.

1.1.1 Pour commencer...

Les concepts nécessaires à la conduite des systèmes de culture par un agriculteur portent d'une part sur les objets, cibles de la décision, et d'autre part sur les connaissances agronomiques inhérentes aux conduites pluriannuelle, intra-annuelle et journalière des systèmes de culture. Dans le premier cas, les concepts pris en compte sont relatifs à la structure de l'exploitation agricole. Dans le deuxième cas, les concepts considérés sont de nature fonctionnelle car ils déterminent les connaissances agronomiques de l'agriculteur permettant de raisonner sur les modalités de production agricole et la dynamique temporelle des capacités de production de l'exploitation agricole.

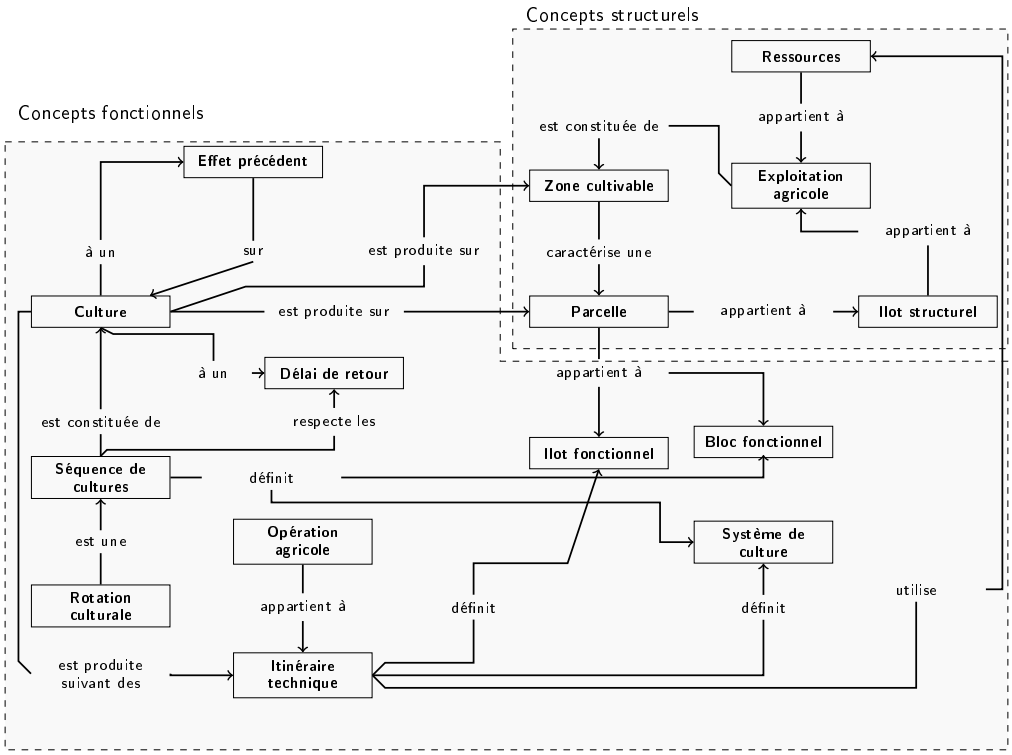


FIGURE 1.1 – *Ontologie simplifiée des systèmes de culture au sein d'une exploitation agricole*

1.1.1.a Description des concepts structurels

La Figure 1.1 regroupe les concepts structurels et fonctionnels et caractérise les relations entre chacun de ces concepts. Ainsi au niveau structurel, on retrouve l'EXPLOITATION AGRICOLE qui est un élément de territoire géré par un agriculteur, ayant une surface agricole et des caractéristiques biophysiques (ex : types de sol, topologie, climat). L'exploitation agricole est constituée de plusieurs ÎLOTS STRUCTURELS (cf. Définition 1.2) qui sont à leur tour constitués de PARCELLES (cf. Définition 1.1).

Définition 1.1 (Parcelle) *Une parcelle dans notre cadre de modélisation est une entité spatiale continue d'une exploitation agricole qui est homogène en terme de culture produite une année donnée mais qui peut présenter une hétérogénéité de ses propriétés (Nesme et al., 2010).*

Il résulte de cette définition que la structure des parcelles d'une exploitation agricole ; encore appelée le *parcellaire* de l'exploitation a une dynamique annuelle. La cause principale de la dynamique spatiale du parcellaire est liée aux découpages et fusions éventuels des parcelles dans l'optique de satisfaire les contraintes de production de l'exploitation. Il existe toutefois des limites physiques qui contraignent la structure du parcellaire. Ces limites sont celles des ÎLOTS STRUCTURELS.

Définition 1.2 (Îlot structurel) *Un îlot structurel (bloc de parcelles de Joannon et al. (2005b)) est un sous-ensemble de parcelles spatialement continues, appartenant à une seule et même exploitation agricole, délimité par des éléments permanents du paysage ou clairement identifiables tel qu'un chemin, une route, un ruisseau, une autre exploitation agricole ou des limites administratives.*

Sur les parcelles sont gérées des cultures. Aubry et al. (1998a) expliquent que la délimitation de la ZONE CULTIVABLE d'une culture (cf. Définition 1.3) c'est-à-dire la délimitation de la zone sur laquelle elle peut être produite, est réalisée par l'agriculteur en amont de la décision du mode de conduite des cultures.

Définition 1.3 (Zone cultivable) *La zone cultivable d'une culture comprend les parcelles de l'exploitation jugées favorables à la culture et éventuellement, celles où la présence de la culture est, selon l'agriculteur, tolérable (Aubry et al., 1998a).*

Ainsi, les zones cultivables sont déterminées par l'agriculteur sur la base de :
- sa connaissance des caractéristiques du sol (ex : profondeur du sol, pierrosité, hydromorphie, pente) et de son parcellaire (ex : forme et taille des parcelles, distance du siège de l'exploitation),
- sa connaissance des besoins des cultures,
- l'accessibilité de la parcelle,
- ses objectifs de production,
- contraintes exogènes relatives à la structure de l'exploitation telle que l'accessibilité d'une ressources (ex : l'eau pour l'irrigation).

L'exploitation agricole dispose généralement d'un ensemble de RESSOURCES (cf. Définition 1.4) humaines et matérielles (ex : tracteurs, semoir, eau, azote) indispensables à la production.

Définition 1.4 (Ressource) *Une* RESSOURCE *est une entité nécessaire à la réalisation d'une culture. Elle est généralement soumise à une capacité, à une disponibilité et à une accessibilité, contraignant ainsi la manière de produire les cultures.*

En fonction des modes de production et de consommation, les ressources peuvent être considérées comme consommables ou réutilisables pour reprendre le vocabulaire de Smith et Becker (1997) et Martin-Clouaire et Rellier (2009). Elles ont des capacités fixes ou variables et sont généralement soumises à des contraintes d'usages. Nous présentons plus en détail les ressources dans les sections 5.6. Par ailleurs, une parcelle peut être perçue, dans l'absolu, comme une ressource terre. Cependant, dans notre cas, nous considérons les parcelles comme des supports de la production de cultures plutôt que des ressources.

1.1.1.b Description des concepts fonctionnels

Au-delà des concepts structurels présentés ci-dessus, il en existe d'autres qui définissent le fonctionnement des systèmes de culture au sein de l'exploitation. Dans l'ontologie simplifiée décrite par la Figure 1.1, ils sont regroupés sous la dénomination de concepts fonctionnels. Ainsi, pour le fonctionnement de son exploitation, un agriculteur doit simultanément contrôler plusieurs processus biophysiques continus relatifs à la production de CULTURES sur ses parcelles. Pour chaque parcelle, la production d'une culture modifie les caractéristiques biophysiques et chimiques du sol. Exemple en est de la production fréquente d'une culture sur une parcelle qui peut accroître l'apparition de pathogènes dans le sol ou y favoriser certains types d'adventices (Doucet et al., 1999). De la même manière, une production fréquente de tubercules poussant sous le sol (ex : pommes de terre) peut nuire à la structure de ce dernier et accroître son érosion (Edwards et al., 1998). Plus généralement, les modifications des caractéristiques inhérentes à la production de culture entrainent des conséquences sur les conditions de réalisation (ex : croissance, mode de conduite) de la culture suivante. En agronomie, on parle généralement de l'EFFET PRÉCÉDENT (cf. Définition 1.5) d'une culture sur une autre.

Définition 1.5 (Effet précédent) *L'effet précédent se réfère à une parcelle et détermine la variation de l'état du sol (caractéristiques biologiques, chimiques et physiques) entre le début et la fin de la culture considérée, sous l'influence combinée du peuplement végétal et des techniques qui lui sont appliquées, l'ensemble étant soumis au climat (Sebillotte, 1990).*

Leteinturier et al. (2006) représentent l'effet précédent par un indicateur k_p qui détermine le bénéfice ou le déficit d'une culture sur son suivant. Cet indicateur prend en compte la dégradation de la structure du sol, les maladies, les résidus de pesticide, d'azote et les mauvaises herbes. Nous ne décrirons pas dans le cadre de cette thèse, les méthodes d'évaluation de ces effets précédents. Le lecteur peut toutefois se référer à l'article cité plus haut. Par ailleurs, dans Aubry et al. (1998a), les auteurs affirment que la prise en compte de ces effets précédents est réalisée par l'agriculteur sur la base de deux principes qui sont : le DÉLAI DE RETOUR (cf. Définition 1.6) et les SÉQUENCES DE CULTURE admissibles (cf.Définition 1.7).

Premièrement, chacune des cultures est associée à un délai de retour.

Définition 1.6 (Délai de retour) *Le délai de retour définit le nombre d'années entre deux implantations [c-à-d deux instances successives] d'une même culture sur une même parcelle (Aubry et al., 1998a; Dogliotti et al., 2003).*

Considérons t et t' deux différentes années telle que $t < t'$. Soit c une culture dont le délai de retour est $dr(c)$. Soient p^t et $p^{t'}$ respectivement l'occupation de la parcelle p à t et t'. Le délai de retour est respecté pour $p^t = p^{t'} = c$ si et seulement si la différence $(t' - t) \geq dr(c)$. Dans le cas où le délai de retour est d'une année, on

parle de monoculture sur la parcelle considérée. Nous faisons toutefois la différence entre les monocultures et les monocultures partielles. En effet pour les cultures annuelles, le délai de retour est appliqué après chaque occurrence de la culture sur la parcelle. La Figure 1.2 présente l'exemple de l'orge de printemps OP avec un délai de retour de trois ans. Certaines cultures peuvent être en monocultures partielles c'est-à-dire qu'elles imposent des itérations successives de la culture avant l'application du délai de retour. C'est le cas du blé d'hiver BH de la Figure 1.2 qui doit être répété au moins deux et au plus trois fois avant la contrainte de délai de retour qui est de trois ans.

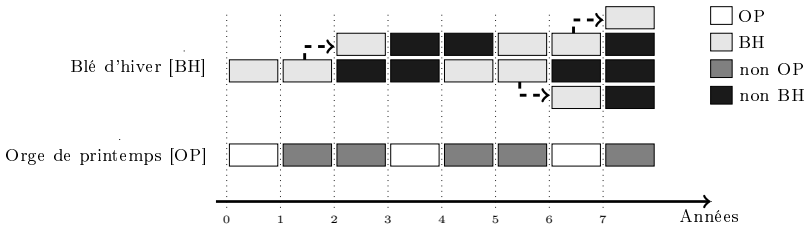


FIGURE 1.2 – *Schéma illustratif de la production de l'orge de printemps OP ($dr(OP) = 3$) et du blé d'hiver BH ($dr(BH) \in [2-3]$)*

La Définition 1.6 peut être généralisée afin d'introduire la notion de culture répétitive. Nous dirons dans ces conditions que le délai de retour définit le nombre d'années entre deux instances successives d'une même série homogène de cultures sur une même parcelle.

Le deuxième concept identifié par Aubry et al. (1998a) pour la prise en compte des effets précédents est la SÉQUENCE DE CULTURE. Des couples de cultures successives peuvent être pénalisés sur la base des impacts négatifs qu'ils engendrent sur l'état du sol. Selon Dogliotti et al. (2003), certaines successions peuvent être interdites en fonction de la sensibilité de la suivante à l'état du sol laissé par le précédent. Soulignons par exemple l'interdiction de produire du lin après une culture de pois. Le stock d'azote laissé par le pois est trop important et cet excès entrainerait un phénomène de verse du lin.

Considérant un horizon déterminé de quelques années, l'ensemble des cultures produites sur une parcelle est appelé SÉQUENCE DE CULTURE (cf. Définition 1.7). Ainsi une culture appartient à une séquence de culture. Plusieurs occurrences de la même culture peuvent appartenir à la même séquence de culture dès l'instant où le délai de retour est respecté.

Définition 1.7 (Séquence de culture) *La séquence de culture définit l'ordre d'apparition des cultures sur une parcelle donnée durant une période déterminée (Aubry et al., 1998b; Leteinturier et al., 2006). Dans une séquence de culture, les délais de retour doivent être strictement respectés.*

Il existe des séquences de culture particulières nommées ROTATIONS CULTURALES (cf.Définition 1.8) dans lesquelles l'ordre et la nature des cultures sont conservés ou répétés cycliquement.

Définition 1.8 (rotation culturale) *La rotation culturale est une séquence de cultures sur une parcelle donnée (Bullock, 1992). Elle est définie par une liste ordonnée sur une parcelle qui se reproduit identiquement au cours du temps (Sebillotte, 1990). La rotation culturale se caractérise par un cycle tandis que la séquence de culture se limite à l'ordre d'apparition des cultures dans la séquence (Leteinturier et al., 2006).*

La définition de Leteinturier et al. (2006) précise clairement la dimension cyclique d'une rotation. Castellazzi et al. (2008) présentent quelques classes de rotations que nous reprenons sur la Figure 1.3. Ces rotations sont caractérisées par leur taille, leur flexibilité annuelle et leur répétitivité.

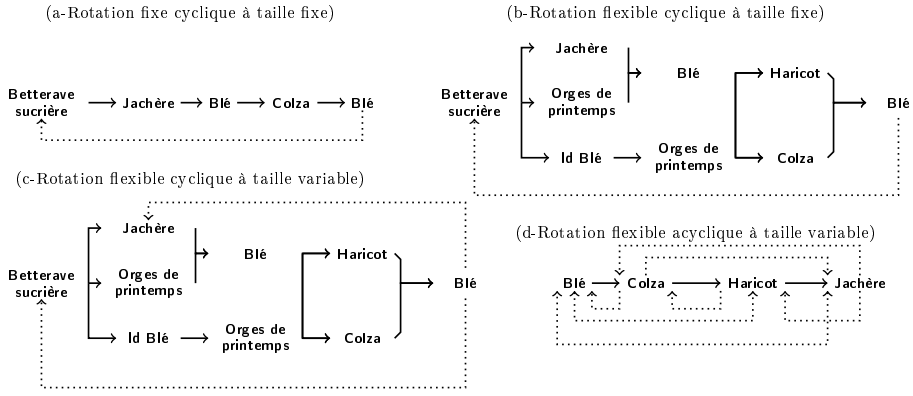


FIGURE 1.3 – *Exemple de classes de rotations extrait de Castellazzi et al. (2008)*

Dans cet exemple, on retrouve d'une part des rotations cycliques (Figure 1.3-a-b-c), au sens où elles sont constituées de séquences de culture qui se répètent, et d'autre part des rotations acycliques (Figure 1.3-d). Comme l'indique la Figure 1.3, la rotation (a) est une séquence fixe et de taille fixe qui se répète. La rotation (b) est également de taille fixe. Elle se distingue de la précédente par la flexibilité qu'elle autorise d'une année sur l'autre en fonction du précédent cultural. En plus de la flexibilité, la rotation (c) illustre la variation de la taille des séquences répétées. Enfin, certaines rotations telles que (d), sont moins structurées et très flexibles permettant ainsi, d'avoir un grand nombre de séquences de culture. Dans tous les cas, la cyclicité de la séquence de culture prédétermine le choix des cultures dans le temps (Dogliotti et al., 2003) car cette décision se détermine par une observation rétrospective des précédentes décisions. Ces rotations permettent également de garantir la stabilité et la durabilité (Leteinturier et al., 2006) des SYSTÈMES DE CULTURE (cf. Définition1.12) dans le temps.

Notre définition de rotation (cf. Définition 1.8) ne prend en compte que les rotations dites fixes (cf. Figure 1.3-a Castellazzi et al. (2008)). L'ensemble des trois autre motifs (Figure 1.3-b-c-d) étant de fait des combinaisons de plusieurs rotations.

Plus généralement, les parcelles sur lesquelles la même séquence de culture (rotation ou pas) est appliquée, forment ce qu'on nomme un bloc de cultures encore appelé BLOC FONCTIONNEL (cf. Définition 1.9).

Définition 1.9 (Bloc fonctionnel) *Un bloc fonctionnel est un sous ensemble de parcelles ayant la même séquence de cultures (Aubry et al., 1998b). Il caractérise un système de culture (Sebillotte, 1990; Ittersum et Rabbinge, 1997), c'est-à-dire une séquence de culture et l'utilisation du même itinéraire technique (cf.Définition 1.10) pour la conduite de ses cultures.*

De cette dernière définition, ressort une chose essentielle. Au-delà de la structure physique de l'exploitation définie par les parcelles et les îlots structurels, il existe des structurations fonctionnelles dans les exploitations agricoles. On parle généralement d'unité de gestion (Aubry, 2000). Ces unités de gestions structurent les parcelles de l'exploitation et permettent de mieux appréhender l'allocation des cultures et des ressources ainsi que l'organisation des opérations agricoles. Un exemple d'unité de gestion peut être un bloc d'irrigation que Bergez et al. (2001)

définissent comme étant une zone de l'exploitation irriguée par un équipement d'irrigation et soumise à des contraintes d'accès à la ressource.

Les blocs fonctionnels font partie de ces unités de gestion, car ils définissent des sous ensembles de parcelles. Les parcelles appartenant à un bloc fonctionnel, bien que pouvant appartenir à des îlots structurels différents, respectent toutes la même séquence de culture.

Chacune des cultures d'une séquence est produite suivant un même ITINÉRAIRE TECHNIQUE. (cf. Définition 1.10). Autrement dit, pour produire ses cultures, l'agriculteur décide d'un ensemble d'OPÉRATIONS AGRICOLES à exécuter. Ces dernières appartiennent à un plan que l'on appelle l'ITINÉRAIRE TECHNIQUE (cf. Définition 1.10) de la culture.

Définition 1.10 (Itinéraire technique) *Un itinéraire technique (ITK) est un ensemble de techniques combinées pour conduire une culture, y compris le choix de la variété, en vue d'atteindre des objectifs divers (rendement, qualité, effet sur l'environnement). Dans un ITK, le type de séquence [type de plan] et la mise en œuvre des opérations agricoles dépendent de règles de décision (Sebillotte et Soler, 1974; Sebillotte, 1990).*

Au-delà des blocs fonctionnels, il existe une autre unité de gestion appelé ÎLOTS FONCTIONNELS (cf. Définition 1.11).

Définition 1.11 (Îlots fonctionnels) *Un îlot fonctionnel est un ensemble de parcelles de l'exploitation agricole sur lesquelles une même culture est produite en utilisant un même itinéraire technique pour une année donnée. (Aubry et al., 1998c).*

Les îlots fonctionnels sont définis par les itinéraires techniques des cultures. Ils permettent de regrouper les parcelles pour lesquelles la conduite quotidienne coïncide. La structuration des parcelles en îlots fonctionnels permet essentiellement de gérer l'organisation du travail au jour le jour. Afin de mieux marquer la différence entre les îlots fonctionnels et les blocs fonctionnels, considérons la Figure 1.4 présentant deux systèmes de culture définis par les blocs fonctionnels 1 et 2 qui sont respectivement constitués des parcelles $\{1, 2\}$ puis $\{3, 4, 5\}$. Sur l'ensemble de ces blocs fonctionnels sont produites deux cultures : *maïs, blé*. Chaque parcelle est représentée par son occupation du sol sur trois années. On distingue pour cet exemple deux îlots fonctionnels 1 et 2 par an qui sont respectivement associés aux cultures de *maïs* et *blé*.

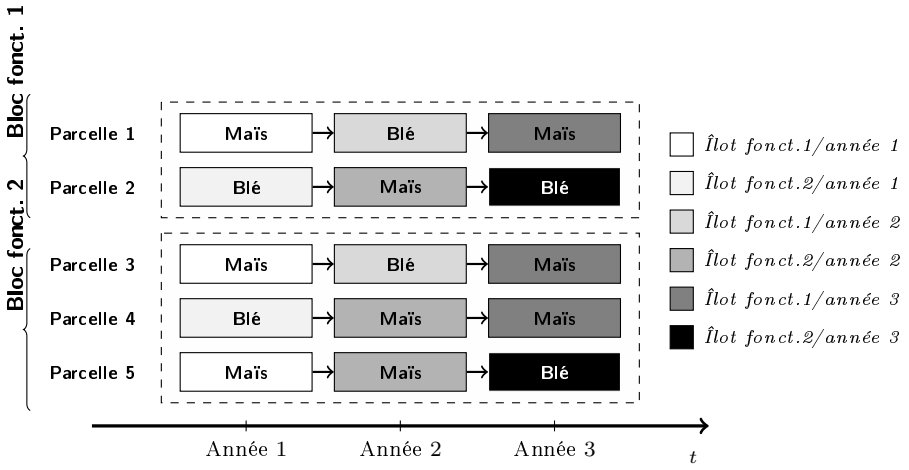


FIGURE 1.4 – *Différence entre îlot fonctionnel et bloc fonctionnel.*

Contrairement aux blocs fonctionnels, les îlots fonctionnels ont une dynamique temporelle car ils sont redéfinis chaque année en fonction des allocations de culture. Ainsi dans notre exemple, l'îlot fonctionnel 1 est défini par les parcelles $\{1, 3, 5\}$, $\{2, 4, 5\}$ et $\{1, 3, 4\}$ respectivement pour les années 1, 2 et 3.

Comme l'indique l'ontologie simplifiée de la Figure 1.1 une séquence de culture associée à un itinéraire technique permet de définir un SYSTÈME DE CULTURE (cf. Définition1.12).

Définition 1.12 (Système de culture) *Un système de culture est un ensemble des modalités techniques mises en œuvre sur des parcelles traitées de manière identique. Chaque système de culture se définit par : (i) la nature des cultures et de la séquence de culture (ii) les itinéraires techniques (cf. Définition 1.10) appliqués à ces différentes cultures, y compris le choix des variétés (Sebillotte, 1990).*

1.1.2 Dynamique de la décision chez un agent agriculteur

Nous rappelons que dans le cadre des applications qui nous intéressent, nous considérons que la délimitation des blocs fonctionnels et les ressources de l'exploitation sont fixes.

Le processus de décision d'un agriculteur dans les systèmes de culture au sein de l'exploitation agricole est une combinaison de tâches de planification à différents niveaux d'abstraction. Comme l'indique la Figure 1.5, l'agriculteur est confronté à trois niveaux différents de décisions qui sont : stratégique, tactique et opérationnelle. Chacune de ces décisions porte sur des échelles de temps variables allant de plusieurs années à quelques jours.

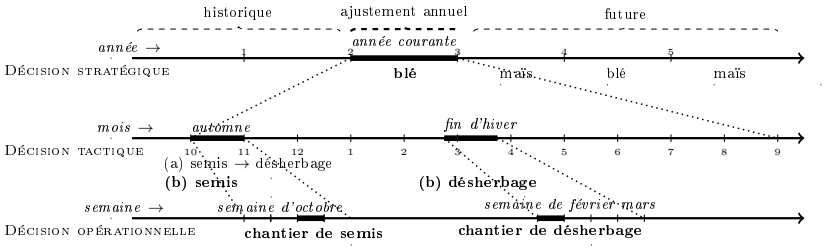


FIGURE 1.5 – *Les niveaux d'abstraction de la décision dans les systèmes de culture au sein de l'exploitation agricole*

1.1.2.a Décisions stratégiques

La DÉCISION STRATÉGIQUE consiste à définir pour chacun des blocs fonctionnels une séquence de culture. Elle détermine ainsi la dynamique de l'organisation spatio-temporelle des systèmes de culture de l'exploitation agricole. Sur la base des ressources disponibles et de la structuration de l'exploitation, elle permet de construire un plan à long terme (plusieurs années) permettant d'atteindre les objectifs de production de l'agriculteur. Pour chacune des années appartenant à l'horizon de la décision stratégique, l'agriculteur décide la sole c'est-à-dire la superficie consacrée à chacune des cultures et l'allocation de ces dernières aux parcelles : c'est ce qu'on appelle l'ASSOLEMENT. Par exemple, sur la Figure 1.5, l'assolement de l'année courante consiste à produire du blé sur certaines parcelles de l'exploitation agricole.

1.1.2.b Décisions tactiques

Considérant l'assolement d'une année donnée, l'agriculteur prend un ensemble de DÉCISIONS TACTIQUES qui consiste à choisir l'itinéraire technique et les variétés des cultures de l'année courante. Comme l'indique la Figure 1.5, il existe plusieurs ITK ((a) et (b)) pour la conduite du blé. Il en est de même pour la plupart des cultures. Par exemple, pour un blé semé à l'automne, l'itinéraire technique (a) consiste à réaliser un désherbage juste après le semis. Pour l'ITK (b) du blé, le désherbage est réalisé en sortie d'hiver à la reprise de la végétation. Le choix des itinéraires techniques dépend des objectifs de production de l'agriculteur (ex : l'espérance de rendement, réduction des intrants) et des ressources disponibles.

Par ailleurs, du fait de la dynamique du contexte climatique et économique, la production des cultures est soumise à une part d'incertitude. Dans ces conditions, des ajustements tactiques à moyen terme (quelques mois, une année) sont indispensables à la conduite de cultures. L'agriculteur réajuste dynamiquement les itinéraires techniques de chacune des cultures.

Ainsi, les décisions tactiques sont intra-annuelles et permettent à l'agriculteur de choisir, au début de chaque campagne culturale, les itinéraires techniques (ITK) dans l'optique de déterminer la façon d'appliquer la stratégie décidée au niveau d'abstraction supérieur. L'élément central de la phase tactique est la détermination pour chacune des parcelles, et en fonction des objectifs, du le travail requis et des conditions de réalisation. Ainsi, un plan d'action (les itinéraires techniques ITKs) est défini par culture et par ilots fonctionnels. Ces plans sont adaptés tout au long de l'année de manière opérationnelle en fonction des états de milieu et de la culture (ex : croissance des plantes, hauteur des mauvais herbes).

1.1.2.c Décisions opérationnelles

Les décisions tactiques bien que servant de cadre à la conduite annuelle ne sont pas pour autant assez précises pour la conduite journalière. Comme l'indique la Figure 1.5, elles sont combinées à court terme (quelques jours) à un ensemble de DÉCISIONS OPÉRATIONNELLES. Cette phase de décision permet de mettre en avant les moyens d'exécution du plan tactique déterminés dans la phase précédente. En effet, les contraintes d'organisation du travail sur l'ensemble des parcelles de l'exploitation ne se limitent pas aux modes de conduite des cultures. L'organisation opérationnelle du travail prend en compte :

- les adaptations de l'itinéraire technique à la dynamique de l'environnement (intempérie, croissance etc.) en modifiant les dates de réalisation des opérations agricoles (Sebillotte, 1988),
- la disponibilité des ressources humaines et matérielles (Aubry, 1995),
- les règles de priorité en cas de concurrence entre opérations agricoles (Papy, 2001).

La gestion des opérations agricoles est donc le point central de la décision opérationnelle. Cette phase de décision opérationnelle permet de définir l'ordre et la date de réalisation des opérations, l'allocation des ressources, le mode de mise en œuvre, et les conditions de réussite. Elle vise à s'assurer que chacune des opérations agricoles sera réalisée dans les meilleurs conditions de délais (ex : période de réalisation optimale), de coûts (ex : consommation des ressources) et de qualité (ex : vitesse de réalisation, adéquation entre l'opération et la spécialité de la ressources humaines). Ainsi, pour chacun des îlots fonctionnels de l'exploitation, l'agriculteur détermine des chantiers d'opérations agricoles (ex : chantier de semis) et définit un ordonnancement.

Il existe, selon Papy (2001), des périodes de concurrence au cours desquelles plusieurs chantiers doivent être conduits simultanément. Selon lui, dans les cas où les ressources de l'exploitation ne permettent pas d'exécuter en parallèle tous les chantiers, des règles de priorités existent et permettent d'arbitrer l'organisation du travail.

En résumé, on distingue trois types de décisions : stratégiques, tactiques, et opérationnelles. Nous avons présenté les interactions entre ces différents types de décision. Dans les sections suivantes nous décrivons puis caractérisons chacune d'elles.

1.2 Problème d'allocation des cultures : décision stratégique

1.2.1 Description du problème

L'ensemble des cultures à produire est défini à priori par l'agriculteur. Dans ces conditions, la décision stratégique se résume (i) à *la détermination des proportions de chacune des cultures* et (ii) à *l'allocation des cultures aux parcelles*.

Généralement, ces décisions dépendent :

- des *objectifs de production* de l'agriculteur en termes de rendement, de qualité et de demande en ressource fourragère,
- des *allocations des années précédentes*, car celles-ci imposent des contraintes agronomiques liées aux règles de succession culturales (cf. section 1.1) tenant compte de l'effet précédent,
- des *règles de succession culturale*, car celles-ci déterminent la qualité des séquences de culture et conditionnent la stabilité de l'assolement d'une année à l'autre,
- des *capacités des ressources* de l'exploitation, car celles-ci contraignent les types de cultures et les proportions annuelles admissibles,
- d'un *ensemble de facteurs spatiaux* (ex : zones cultivables, caractéristiques des îlots structurels) interagissant à différentes échelles de l'exploitation agricole.

De cette première caractérisation, on dira que ce problème de décision est un problème de planification de l'organisation spatio-temporelle de l'exploitation agricole. Elle permet à l'agriculteur de planifier sur plusieurs années sa stratégie d'occupation du sol et de gestion des cultures. La dimension spatiale de la planification stratégique est nommée décision d'*assolement* tandis que la dimension temporelle est connue sous la terminologie de *choix des séquences de culture*. Ces deux types de décisions sont étroitement liées dans la mesure où le choix des séquences de culture pour chaque parcelle prédétermine l'assolement des parcelles. Nous caractérisons chacune de ces décisions dans les sections suivantes.

1.2.1.a Choix de séquences de culture

Le choix des séquences de culture représente la dimension temporelle de la décision stratégique. Elle consiste à déterminer les successions de culture sur chacune des parcelles de l'exploitation. Cette décision porte sur une échelle de temps allant d'une à plusieurs années. Sa pertinence agronomique réside dans le fait qu'elle permette de maintenir la qualité et la productivité des parcelles (Johnson et Toensmeier, 2009). Elle assure par ailleurs des proportions annuelles de chacune des cultures produites tout en tenant compte de l'effet précédent.

Pour prendre cette décision, l'agriculteur a besoin d'un ensemble de facteurs

généraux relatifs à la décision stratégique et de quelques facteurs spécifiques au choix des séquences de culture. Les facteurs généraux sont :

- les cultures et le parcellaire de l'exploitation,
- les caractéristiques biophysiques et topologiques des parcelles (ex : type de sol, distance de la parcelle au siège de l'exploitation, accessibilité des parcelles),
- les zones cultivables pour chacune des cultures,
- les objectifs de production annuels et pluriannuels de l'agriculteur en terme de surface.

Exceptés ces différents facteurs, ceux qui sont spécifiques au choix de séquences de culture sont :

- le délai de retour de chacune des cultures,
- l'estimation des effets précédents de chaque couple de cultures,
- l'historique des parcelles,

Les séquences de culture recherchées peuvent être ou non des rotations.

1.2.1.b Décision d'assolement

Pour une année donnée, l'*assolement* détermine la proposition de chacune des cultures (Maxime et al., 1995; Wijnands, 1999) et leur répartition spatiale sur l'ensemble de l'exploitation (Aubry et al., 1998c). Dans cette formalisation du problème, on identifie d'une part l'idée de la superficie consacrée à chaque culture et d'autre part, celle de l'allocation explicite de ces cultures aux parcelles de l'exploitation.

Pour cette décision, en plus des facteurs généraux relatifs à la décision stratégique (cf. section 1.2.1.a), l'agriculteur a besoin d'un ensemble de facteurs spécifiques à la décision d'assolement. Ces facteurs spécifiques sont :

- l'accessibilité des parcelles,
- les capacités des ressources,

Nous reviendrons plus en détail dans la section 1.5.1 sur les approches existantes permettant d'appréhender cette question. Il est toutefois important de noter qu'à ce jour, il n'existe pas de travaux permettant de concilier à la fois le choix des séquences de culture et la décision d'assolement spatialement explicite (Dury et al., 2011).

1.2.2 Formulation du problème

L'allocation de culture à l'échelle de l'exploitation est un problème de planification spatio-temporelle dans lequel des cultures sont allouées aux parcelles de manière à satisfaire un ensemble de contraintes agronomiques tout en respectant les objectifs de production de l'agriculteur. L'utilité des affectations peut être estimée par une fonction objective globale de l'agriculteur. Le plan recherché porte sur un horizon de plusieurs années et peut être soumis à des adaptations en cas de situation d'échec.

1.2.2.a Les variables de décisions

Pour chaque parcelle, les variables qui interviennent dans les contraintes liées à l'allocation de culture sont les suivantes : les délais de retour des cultures, l'historique des allocations, les caractéristiques du sol, la localisation des parcelles dans les blocs fonctionnels, la disponibilité des ressources et les types de culture possibles.

1.2.2.b Les actions possibles

Pour cette planification, les actions envisageables sont les suivantes :

1. *définir les successions* de cultures admissibles en fonction des contraintes de succession culturale et de la nature des séquences recherchées (rotations ou pas),
2. *assigner les cultures* aux parcelles en fonction des contraintes agronomiques et de ressources,
3. *découper et/ou fusionner* les parcelles en fonction des objectifs globaux de production et les sous contrainte des précédents culturaux, de capacités et d'accessibilité de ressource.

1.2.2.c Les contraintes

Les contraintes liées à l'allocation de culture sont de deux ordres. D'une part on retrouve des contraintes liées à la dimension temporelle du problème d'allocation de cultures. Ces contraintes portent sur le délai de retour des cultures, l'historique des parcelles et les effets précédents des couples de cultures. D'autre part, on retrouve des contraintes liées à la dimension spatiale du problème. Ces contraintes sont liées à la zone cultivable de chacune des cultures, à la topologie de l'exploitation, aux objectifs de production de chacune des cultures, aux interdépendances spatiales entre les parcelles, puis aux capacités de ressources.

Ces différentes contraintes sont définies à différents niveaux d'organisation qui sont :

- les *parcelles* afin d'exprimer (i) si les parcelles peuvent être découpées ou fusionnées, ou (ii) si elles doivent être maintenues fixes tout au long de l'horizon de planification afin de définir leur caractère statique.
- les *blocs fonctionnels* afin d'exprimer pour chaque parcelle et culture leur compatibilité spatiale, les délais de retour et les effets précédent.
- les *îlots structurels* afin d'exprimer l'organisation spatiale des allocations souhaitée par l'agriculteur,
- l'*exploitation* afin d'exprimer les préférences de l'agriculteur ou l'utilisation des ressource.

1.2.3 Utilité des plans stratégiques : les critères d'analyse

Il existe généralement plusieurs options possibles pour un problème d'allocation de culture. De plus la mesure de l'utilité d'une allocation par rapport à une autre dépend de différents facteurs qui sont parfois antagonistes. Considérons par exemple une exploitation où la culture principale est le maïs. Si l'utilité de l'allocation est évaluée par une maximisation de la superficie en culture principale, les allocations optimales seront certainement celles qui proposent des mono-cultures de maïs. On sait par ailleurs que les qualités d'une séquence de culture dépendent de la diversité des cultures qui la composent. En effet, la monoculture de maïs sur une parcelle peut accroître l'apparition de pathogènes dans le sol et entraîner une prépondérance de mauvaises herbes sur la parcelle. Cet exemple illustre très bien l'impossibilité d'une maximisation de la superficie en culture principale et une qualité de séquences de culture optimale.

Dans ces conditions, l'utilité de l'allocation ne peut être qu'un compromis entre les différents facteurs pris en compte dans les objectifs globaux de l'agriculteur. En ce qui concerne les applications qui ont été étudiées dans le cadre de cette thèse, nous avons identifié trois classes de facteurs permettant d'évaluer l'utilité d'une allocation. Ces classes de facteurs sont : la *qualité des séquences de culture*, la *préférence structurelle* des allocations et les *objectifs de production* de l'exploitation.

Premièrement, la *qualité des séquences de culture* est évaluée en utilisant l'indicateur *kp* de Leteinturier et al. (2006) représentant l'effet précédent d'un couple de culture. Grâce à ceci, il est possible d'évaluer la nature d'une séquence de culture en prenant en compte la dégradation de la structure du sol qu'elle entraine, les maladies, les résidus de pesticides, d'azote et les mauvaises herbes.

Deuxièmement, la *préférence structurelle* des allocations est évaluée en fonction de la continuité spatiale des parcelles auxquelles une même culture a été assignée une année donnée. Autrement dit, l'allocation est d'autant plus intéressante que les parcelles allouées à une même culture sont spatialement groupées. Cette préférence structurelle est intéressante car elle minimise les déplacements entre parcelles et facilite implicitement le travail au quotidien de l'agriculteur. Notons cependant que cette préférence n'est pas toujours possible. L'une des raisons est que deux parcelles côte à côte peuvent appartenir à des zones cultivables différentes induisant ainsi l'impossibilité de produire la culture sur l'une des deux parcelles. On dénombre d'autres raisons liées à l'historique de l'exploitation, aux capacités et à l'accessibilité des ressources.

Enfin, les *objectifs de production* de l'exploitation sont évalués en fonction de l'adéquation entre les surfaces cultivées et les besoins de fonctionnement de l'exploitation. Ces besoins peuvent être imposés par des contrats de l'agriculteur ou fixés en fonction des exigences internes de l'exploitation (ex : production de fourrage pour les troupeaux).

En résumé, le problème d'allocation de culture tel que nous le percevons dans cette thèse est un problème de planification spatio-temporelle de l'occupation du sol dans lequel l'utilité de l'allocation est évaluée par une fonction multicritères dépendant de la qualité des séquences de culture, des préférences structurelles de l'agriculteur et des objectifs production de l'exploitation. Les plans solutions d'un problème d'allocation de culture sont des plans abstraits dont l'exécution nécessite une expansion.

1.3 Choix du mode de conduite des cultures : décisions tactiques

1.3.1 Description du problème

Une fois les cultures affectées aux parcelles, l'agriculteur décide annuellement la façon de conduire chacune d'elle : c'est la décision tactique. Il combine, pour ce faire, les opérations agricoles : semer, désherber, fertiliser, récolter etc. Chacune de ces opérations agricoles nécessite des ressources spécifiques et est soumise à un ensemble de contraintes temporelles telles que : (i) la durée d'exécution, (ii) la fenêtre d'activation et de fermeture.

L'enchaînement des opérations agricoles définit le mode de conduite adopté pour la production des cultures allouées durant la phase de planification stratégique. Notons dès le départ que l'agriculteur ne réinvente pas à chaque fois la manière de conduire les cultures à partir d'un objectif final qui serait de produire une culture. Généralement il dispose via ses itinéraires techniques (ITK) de connaissances sur les différentes manières de résoudre le problème. Il existe pour chaque culture, plusieurs ITKs possibles. Chacun permet d'atteindre un objectif spécifique (ex : maximiser le rendement, minimiser les intrants etc.). Ainsi, considérant l'allocation des cultures d'une année donnée, le problème de décision tactique relatif au mode de conduite consiste à associer à chaque culture allouée un plan décrivant un itinéraire technique lui permettant d'atteindre ses objectifs de production. Au vue de la dynamique de l'environnement, ce plan ne peut être

que partiellement instancié. En effet une planification très précise sur une année de la date et des ressources associées à une opération agricole ne pourraient prendre en compte l'incertitude intrinsèque de la conduite des systèmes de culture (ex : le climat).

Par ailleurs, le choix du mode de conduite des cultures est raisonné par système de culture.

Pour appréhender cette décision, l'agriculteur dispose :
- d'un ensemble de parcelles de l'exploitation affectées à des cultures,
- d'un ensemble d'ITKs par culture,
- des plans courants en cours d'exécution sur les parcelles,
- d'un ensemble de variables observées décrivant la dynamique de l'environnement,
- des capacités de ressources de l'exploitation,
- d'objectifs de production annuels.

1.3.2 Formulation du problème

Le choix du mode de conduite des cultures, caractéristique de la décision tactique, est un problème de planification d'actions duratives dans l'espace de plans partiellement ordonnés que sont les itinéraires techniques (ITKs). Le problème de décision sous-jacent est l'affectation d'ITKs à des couples parcelle/culture de manière à respecter la sémantique des systèmes de culture tout en satisfaisant les contraintes de ressources, les contraintes temporelles et de précédences entre opérations agricoles et ceci dans l'optique d'atteindre les objectifs de production de l'agriculteur. Les plans recherchés sont des plans temporellement consistants, partiellement instanciés et portant sur un horizon de quelques mois voire une année au plus. L'exécution de ce plan temporel peut être soumise à des adaptations tout au long de l'année.

1.3.2.a Les variables de décisions

Pour chaque plan tactique, les variables de décision sont relatives (i) aux ressources (consommation et production), (ii) au temps (durée, intervalle d'activation/fermeture et précédence entre opérations agricoles) et (iii) aux états du milieu (ex : croissance de la plante, état du sol, climat, etc.).

1.3.2.b Les actions possibles

Pour cette phase de planification, les opérations agricoles envisageables recouvrent l'ensemble du domaine des opérations agricoles possibles.

De manière générale ces opérations agricoles sont caractérisées par un ensemble de conditions d'exécution, de contraintes temporelles et de ressources auxquelles elles sont soumises (cf. Tableau 1.1).

Nous présentons dans l'Annexe A.1 une description simplifiée et non exhaustive des principales opérations agricoles pour la conduite des cultures annuelles. Le lecteur y trouvera notamment une description des opérations agricoles de :
- travail du sol,
- semis,
- buttage,
- désherbage,
- fertilisation de croissance,
- protection ravageurs,
- régulation de croissance,
- irrigation,

TABLE 1.1 – *Caractéristiques générales des opérations agricoles*

Caractéristiques	**Descriptions**
NOM	*le nom de l'opération agricole*
PRÉCONDITION	*les conditions sur l'état du système propices à l'exécution de l'opération agricole*
DURÉE	*la durée d'exécution estimée*
CONTRAINTES TEMPORELLES	*les contraintes sur les périodes de l'année où l'opération peut être exécutée*
RESSOURCES	*les contraintes sur la disponibilité des ressources nécessaires à la réalisation de l'opération*
EFFETS	*les effets directs et attendus de l'opération agricole sur les ressources de l'exploitation et la dynamique de l'environnement chimique et biophysique de l'exploitation.*

– défanage,
– récolte.

Les dates de réalisation ces opérations agricoles sont spécifiques aux types de culture. Nous décrirons pas l'ensemble des dates possibles. Cependant, dans la section 1.6 nous présenterons celles liées aux opérations agricoles utilisées dans le cas d'étude de la thèse.

1.3.2.c Les plans partiels décrivant les itinéraires techniques

L'itinéraire technique d'une culture est un plan partiellement ordonné constitué d'opérations agricoles. Ce plan décrit l'enchainement des opérations. Une culture peut être conduite de plusieurs manières en fonction des objectifs de l'agriculteur. Par exemple, pour conduire le blé, un agriculteur peut avoir deux itinéraires techniques :

1. *Labour* → *Semer* → *Désherbage mécanique* → *Fertilisation Azote* → *Récolter*.
2. *Labour* → *Semer* → (*Désherbage chimique*/ *Fertilisation Azote* → *Fertilisation Azote*/ *Fongicide*) → *Récolter*.

Le premier est compatible avec les enjeux environnementaux car il introduit très peu d'apport de substances chimiques. Le second quant à lui, offre des garanties de rendement par l'introduction *Fertilisation Azote* et de *Fongicide* en parallèles. Dans les deux cas, l'itinéraire technique est décrit par une séquence d'opérations agricoles totalement ordonnées.

Généralement, les ITKs sont plus flexibles que les séquences fixes décrites ci-dessus. Comme l'indique Martin-Clouaire et Rellier (2006), les ITKs sont décrits d'une part par des *séquences* d'opérations agricoles, spécifiant que deux ou plus de deux opérations agricoles doivent se suivre sans recouvrement lors de leurs exécutions sur une même parcelle. Selon les auteurs, la nature de l'enchainement des opérations agricoles s'étend également à la synchronisation entre la fermeture et l'ouverture de deux opérations consécutives. D'autre part, les ITKs peuvent contenir des *itérations* qui spécifient qu'une opération agricole doit être répétée dans l'intervalle de temps correspondant à son intervalle d'ouverture et de fermeture. Enfin, les itinéraires techniques peuvent contenir des *optionalités* qui expriment

le fait qu'un défaut de réalisation de certaines opérations agricoles n'entraine pas systématiquement la poursuite de l'ITK initial.

En nous basant sur les travaux de Martin-Clouaire et Rellier (2006) nous avons identifié cinq motifs élémentaires permettant de représenter les différentes types d'enchainements des opérations agricoles au sein d'un itinéraire technique de culture. Il s'agit d'itinéraires techniques décrits entièrement ou partiellement par des :

- *séquences fixes* pour lesquelles un ordre total existe entre les opérations agricoles (cf. Figure 1.6),
- *séquences flexibles* pour lesquelles l'ordre total existant entre les opérations agricoles n'exclut pas les optionalités dans la séquence (cf. Figure 1.7),
- *branches parallèles fixes* pour lesquelles un ordre partiel existe entre les opérations agricoles. (cf. Figure 1.8),
- *branches parallèles flexibles* pour lesquelles l'ordre partiel existant entre les opérations agricoles n'exclut pas les optionalités dans les différentes branches décrivant les enchainements d'opérations agricoles. (cf. Figure 1.9),
- *itérations* pour lesquelles des opérations agricoles peuvent être répétées plusieurs fois (cf. Figure 1.10).

Ainsi, l'itinéraire technique d'une culture est une combinaison de ces différents motifs élémentaires.

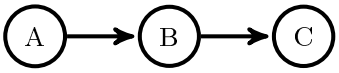


FIGURE 1.6 – *Motif d'ITK décrivant une séquence fixe*

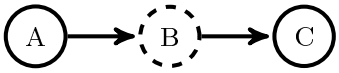


FIGURE 1.7 – *Motif d'ITK décrivant une séquence flexible*

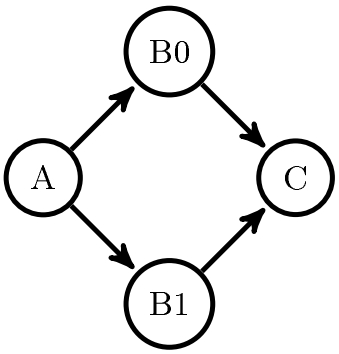


FIGURE 1.8 – *Motif d'ITK décrivant des branches parallèles fixes*

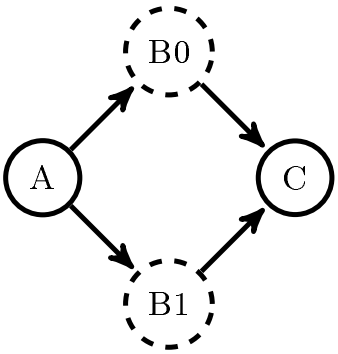


FIGURE 1.9 – *Motif d'ITK décrivant des branches parallèles flexibles*

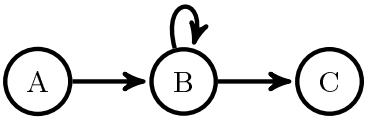


FIGURE 1.10 – *Motif d'ITK décrivant une itération*

1.3.2.d Les contraintes

Les contraintes liées au choix du mode de conduite des cultures sont de trois types. Premièrement nous retrouvons les contraintes liées aux aspects temporels des itinéraires techniques. Ces contraintes portent sur les périodes d'activation des opérations agricoles décrivant l'ITK. Elles concernent également des :

- relations temporelles d'enchaînement ou de synchronisation entre opérations agricoles (Martin-Clouaire et Rellier, 2006).

– relations symboliques de types itération ou optionalité d'une opération agricole.

Les contraintes temporelles sont définies pour chaque couple parcelle/culture de l'exploitation.

Deuxièmement, nous retrouvons les contraintes relatives à la consommation et la production de ressources par chacune des opérations agricoles. Ces contraintes de ressources sont spécifiées à l'échelle des *blocs fonctionnels* et de l'*exploitation* afin d'exprimer l'accessibilité des ressources sur les parcelles et leur disponibilité.

Enfin on retrouve une contrainte liée aux systèmes de culture. Elle impose pour une année donnée, dans un bloc fonctionnel donné, que chaque culture soit conduite suivant un même ITKs et ceci pour toutes les parcelles affectées à la culture considérée. L'itinéraire technique appliqué à une culture d'un bloc fonctionnel peut cependant changer d'une année à l'autre. Cette contrainte est de ce fait définie annuellement à l'échelle du *bloc fonctionnel*.

1.3.3 Utilité des plans tactiques : les critères d'analyse

Il existe plusieurs modes de conduite (ITK) pour chaque culture. L'utilité d'un ITK de culture par rapport à un autre dépend (i) des autres parcelles associées à la même culture et (ii) des effets de l'itinéraire technique sur les objectifs de production de l'exploitation (ex : rendement, charge de travail). Ainsi, l'affectation d'un ITK à une parcelle est d'autant plus intéressante qu'elle est réalisable sur l'ensemble des parcelles d'un bloc fonctionnel affecté à la même culture. L'ensemble des affectations d'ITK est évalué au niveau de l'exploitation en fonction de l'adéquation entre les espérances de rendement des modes de conduite choisis et les contraintes structurelles de l'exploitation. Ces contraintes peuvent être imposées par l'accessibilité et la disponibilité des ressources.

En résumé, le problème du choix du mode de conduite est un problème de planification d'actions duratives (opérations culturales) dans l'espace des plans partiels (ITK). L'utilité des ITK affectés aux parcelles dépend de la disponibilité des ressources, de l'assolement et des objectifs de production de l'exploitation. Les ITKs solutions sont des plans temporellement consistants dont l'exécution nécessite une allocation explicite de ressources.

1.4 Organisation du travail : décision opérationnelle

1.4.1 Description du problème

En décidant le mode de conduite des cultures, l'agriculteur dispose d'un plan tactique annuel c'est-à-dire d'un ITK pour chaque culture affectée à une parcelle. Les ITKs obtenus à ce stade de la décision ne sont pas directement exécutables. En effet, les informations issues de l'allocation des cultures et des itinéraires techniques ne servent qu'à déterminer les différentes opérations agricoles à réaliser. Afin de les exécuter, l'agriculteur doit déterminer un ordonnancement précis des opérations agricoles c'est-à-dire une organisation du travail sur l'ensemble de l'exploitation. Cette décision est nommée décision opérationnelle. Du fait de l'incertitude intrinsèque à la conduite des systèmes de culture, les décisions opérationnelles sont prises sur un horizon de temps de quelques jours.

En se basant sur Papy (2001), nous identifions quatre paramètres de décision essentiels pour raisonner l'organisation du travail. Ces paramètres sont :

1. la disponibilité effective des ressources humaines et matérielles,
2. la vitesse de réalisation des opérations agricoles sur les différents chantiers,

3. les itinéraires techniques et les périodes de concurrence des opérations agricoles,
4. les jours disponibles (Rounsevell, 1993) pour la réalisation d'une opération culturale.

Les deux premiers paramètres peuvent se passer d'explications dans la mesure où ; si disponibilité des ressources il y a, la vitesse de réalisation des opérations agricoles voire des chantiers sur les îlots structurels, est fonction du type de ressources utilisées (ex : puissance des tracteurs, largeur de travail des outils, capacité du pulvérisateur, etc.).

Les deux derniers paramètres mettent en parallèle l'ensemble des itinéraires techniques et les périodes de concurrences entre opérations agricoles. Afin, de bien cerner cette dimension de la décision opérationnelle, nous reprenons l'analyse de Joannon (2004) sur l'organisation du travail entre juillet et novembre, pour des systèmes de culture du pays de Caux en France (cf. Figure 1.11). L'auteur justifie cette tranche de l'année par le fait qu'on y retrouve d'une part la période d'interculture, où différentes opérations agricoles de travail du sol et semis de cultures intermédiaires doivent être réalisées. D'autre part, cette tranche de l'année est également propice à de nombreuses opérations agricoles critiques. On y retrouve entre autre la saison des récoltes (de 15 juillet à 30 novembre) et celle des semis d'automne (de 25 août à 30 novembre).

Comme l'indique la Figure 1.11, la tranche considérée dans cette étude de cas est divisée en quatre périodes. La première et la troisième sont des périodes peu chargées. Les deuxième et quatrième périodes comportent des charges de travail conséquentes avec des concurrences potentielles entre chantiers. Du fait des contraintes de ressources, les chantiers en concurrences ne peuvent être réalisés simultanément. Les agriculteurs établissent généralement des règles de priorité entre les différents chantiers. La récolte des cultures à maturité est souvent prioritaire sur les semis. À titre illustratif, nous détaillons la gestion des concurrences dans la deuxième période c'est-à-dire, celle allant de début août à début septembre. Le lecteur peut se référer à Joannon et al. (2005a) pour plus de détails sur les autres périodes.

Les principaux chantiers à réaliser durant cette deuxième période sont la récolte du blé et du pois et l'enroulage du lin. Ces chantiers représentent une charge de travail importante en fonction des surfaces cultivées (cf. Joannon (2004) pour plus de détails). Comme l'indique la Figure 1.11, la date de récolte du blé varie beaucoup. Cette variation est fonction de la date de semis de la variété et du climat. En cas de récolte du blé dans la première quinzaine d'août il y a concurrence entre les récoltes de blé et celles du pois. Par ailleurs, lorsque les récoltes du blé sont tardives (deuxième quinzaine d'août), elles se retrouvent en concurrence avec celles du lin.

Afin de gérer ces concurrences liées à l'allocation des ressources, les agriculteurs fixent des règles de priorité. À titre illustratif, dans le cas de cette étude, la récolte du lin est toujours prioritaire, car ce chantier est plus exigeant vis-à-vis des conditions climatiques et la marge brute dégagée de la récolte est supérieure à celle du blé ou du pois. Les gousses de pois étant plus fragiles que les épis de blé, ce dernier n'est pas prioritaire.

En résumé, l'organisation du travail est raisonnée sur l'ensemble de l'exploitation sous la forme de chantiers d'opérations agricoles sur les îlots fonctionnels. En cas de concurrence entre chantiers il existe de règles de priorité entre les chantiers d'opérations agricoles.

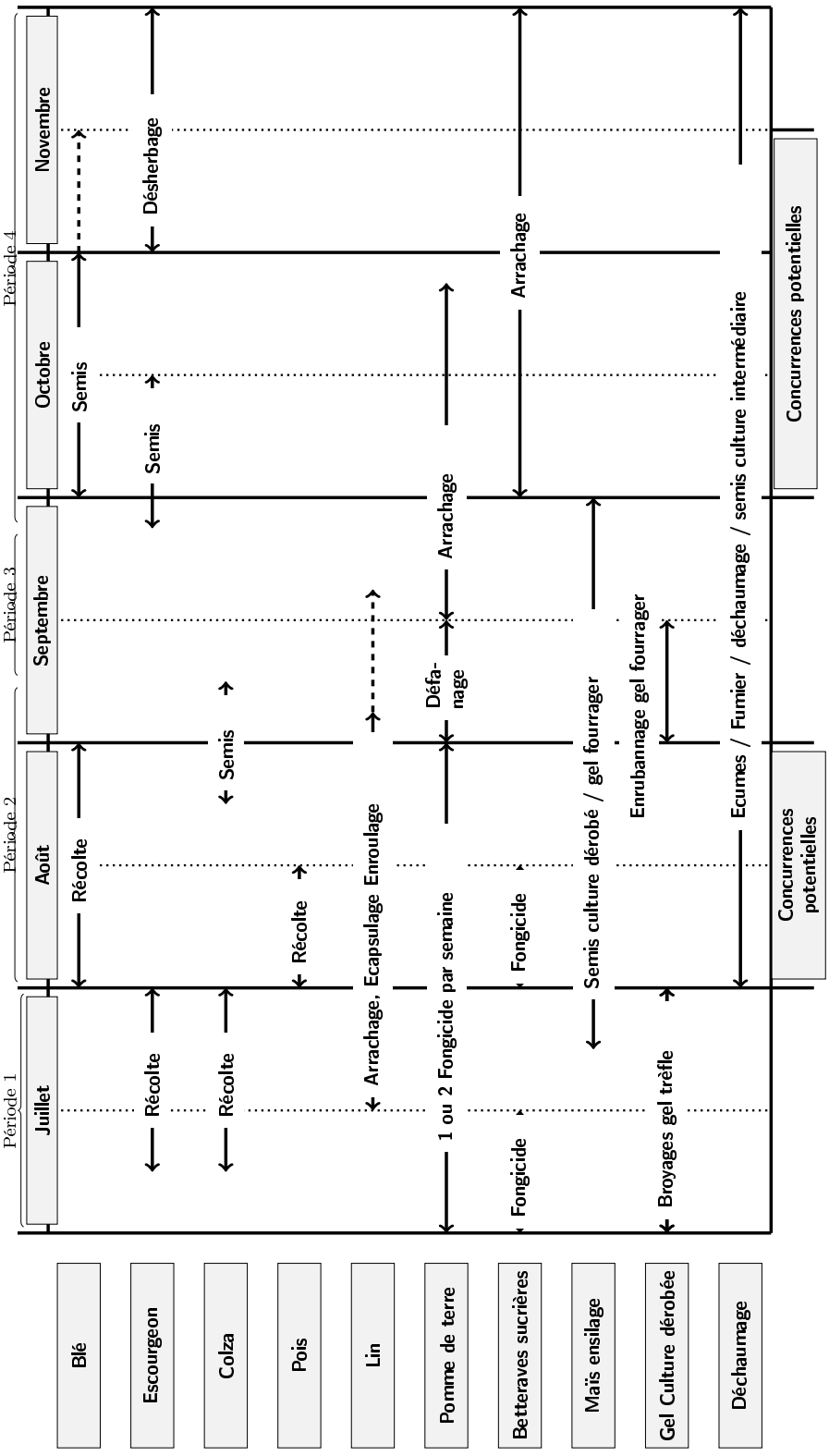


FIGURE 1.11 – *Calendrier simplifié de l'organisation du travail dans les exploitations agricoles du bassin versant de Bourville (Joannon, 2004)*

1.4.2 Formulation du problème

L'organisation du travail dans les systèmes de culture au sein de l'exploitation agricole est un problème d'ordonnancement d'actions durative dans l'espace des itinéraires techniques. Le problème de décision consiste à assigner des ressources aux différentes opérations agricoles de manière à respecter les contraintes temporelles des ITKs et les règles de priorité entre chantiers. Le plan opérationnel obtenu doit permettre d'atteindre les objectifs opérationnels. Ces derniers peuvent être :

- la minimisation du travail le weekend,
- la minimisation de la durée de réalisation des chantiers,
- le minimisation du temps écoulé entre deux opérations agricoles (inter-tâches).

L'ordonnancement recherché est un plan opérationnel totalement instancié sur un horizon de quelques jours. L'exécution du plan est soumise à des adaptations relatives aux contextes climatiques et aux états du milieu.

1.4.2.a Les variables de décisions

Les variables de décision sont les mêmes que celles de la décision tactique. À celles-ci s'ajoutent les règles de priorité entre les chantiers.

1.4.2.b Les actions possibles

Les actions envisageables sont strictement les mêmes que celles présentées dans la section 1.3.2.b pour la décision tactique.

1.4.2.c Les contraintes

À l'exception de celles liées aux règles de priorités entre les chantiers, les contraintes relatives à la décision opérationnelle sont les mêmes que celles de la décision tactique. Ils s'agit notamment des contraintes :

- d'accessibilité et de disponibilité des ressources,
- de durée des opérations agricoles,
- de dates d'activation et de fermeture des opérations agricoles,
- de précédence entre les opérations agricoles d'un même itinéraire technique,
- de priorités entre les chantiers.

1.4.3 Utilité des plans opérationnels : les critères d'analyse

Plusieurs ordonnancements sont possibles pour l'organisation du travail. L'utilité d'une organisation est évaluée en fonction des objectifs opérationnels de l'agriculteur (minimisation de la durée de réalisation des chantiers, de l'inter-tâche etc.) et la satisfaction des contraintes temporelles et de ressources.

En résumé, l'organisation du travail est un problème d'ordonnancement dans lequel l'utilité du plan opérationnel est évalué par une fonction multicritères dépendante des variables temporelles des opérations agricoles, des ressources affectées et des objectifs opérationnels de l'agriculteur. Le plan solution est totalement instancié et directement exécutable.

1.5 Quelques travaux sur la simulation bio-décisionnelle en agronomie

1.5.1 Décision stratégique

De nombreux travaux ont été initiés afin de résoudre le problème de planification stratégique. Depuis Heady Heady (1948), différentes approches ont été proposées pour la recherche de la meilleure affectation de cultures aux parcelles Kein Haneveld et Stegeman (2005). La plupart des travaux sont basés soit sur l'optimisation de la marge brute Annetts et Audsley (2002) des cultures soit sur l'utilisation de filtres agronomiques Bachinger et Zander (2007) (ex : règles de succession, quantités de produits phytosanitaires dans le sol, etc.) pour fournir des successions de cultures admissibles. Dans les différentes approches existantes, la planification stratégique a été abordée soit sous la forme d'une optimisation de proportions annuelles des cultures McCarl et al. (1977); Itoh et al. (2003); Sarker et Ray (2009) soit sous la forme d'une recherche des successions (rotations) de cultures admissibles El-Nazer et McCarl (1986); Dogliotti et al. (2003). Ces deux aspects constituent cependant deux dimensions (spatiale et temporelle) du problème.

Nombre d'entre ces travaux s'inspirent de méthodes de programmation linéaire afin de trouver des rotations optimales. Par exemple, Haneveld et Stegeman (2005) proposent de spécifier les successions inadmissibles en vue de trouver, pour chaque région, les séquences de cultures envisageables. Par ailleurs, Detlefsen (2004) présente une méthode intégrant des éléments d'optimisation et des approches prédicatives en utilisant une modélisation sous forme de *network flow*. Afin de proposer un cadre de représentation simple et rigoureux, Castellazzi et al. (2008) font l'hypothèse qu'on peut modéliser la rotation comme un processus Markovien dans lequel l'allocation d'une culture à une parcelle pour une année donnée ne dépend que de l'allocation réalisée à l'année précédente. Cette modélisation permet d'une part d'introduire plus de flexibilité dans le choix des rotations et favorise l'alternance entre plusieurs rotations. D'autre part elle permet de prendre en compte l'incertitude inhérente à la production agricole.

Toutefois, il existe très peu d'outils capables d'appréhender toute la complexité des problèmes d'assolement et de rotation. À notre connaissance, les plateformes ROTAT (Dogliotti et al., 2003), ROTOR (Bachinger et Zander, 2007) et LORA (Leroy et Jacquin, 1994) permettent de répondre dans une certaine mesure à la question. Les deux premiers se basent sur une approche statique consistant à générer en partant d'un ensemble de culture, toutes les rotations possibles puis utilisent des filtres agronomiques comme des règles de succession, des quantités de produits phytosanitaires dans le sol, etc. afin de fournir des rotations possibles. Le troisième, LORA, optimise l'assolement sur le périmètre irrigable d'une exploitation agricole en prenant en compte les facteurs climatiques, la ressource en eau, les capacités d'irrigation des équipements et la main-d'œuvre disponible.

À défaut d'être spatialement explicites, les solutions proposées sont des agrégations de propositions de cultures sur un ensemble de parcelles de différents types. Nous proposons d'une part, de prendre en compte simultanément les deux dimensions du problème et d'autre part, de rendre les solutions proposées spatialement explicites.

1.5.2 Décisions tactique et opérationnelle

Depuis la fin des années 80, les travaux autour de la simulation du choix des itinéraires techniques ne cessent de s'accroître. Qu'il s'agisse de travaux au niveau

des parcelles cultivées, comme dans DECIBLE (Chatelin et al., 2005) pour le blé d'hiver ou MODERATO pour le maïs irrigué (Bergez et al., 2001) jusqu'au niveau de l'exploitation agricole, comme dans OTELO pour les grandes cultures (Attonaty et al., 1993), IRMA (Leroy et al., 1997) pour l'irrigation ou SEPATOU (Cros et al., 2001) pour le pâturage, la simulation informatique est devenue un outil incontournable de recherche en agronomie. Ces simulateurs permettent d'une part de prédire les conséquences de différents modes de conduite de l'activité agricole, et d'autre part, de raisonner la politique de conduite afin d'élaborer de nouvelles lois agronomiques. Chacune des plate-formes précédemment citée est exclusivement attachée à un problème donné.

Dans OTELO (Attonaty et al., 1993) et DIESE (Martin-Clouaire et Rellier, 2009) les auteurs intègrent un modèle décisionnel capable de prendre en compte les dimensions tactique et opérationnelle. Le premier appréhende la problématique de décision sous la forme de comportements réactifs (règles de décision). Le second, quant à lui, permet de créer et d'exécuter des plans flexibles. Il autorise la spécification des contraintes sur et entre les activités. Grâce à son interpréteur il opère sur des plans construits manuellement afin de déterminer de manière itérative les activités à exécuter. Dans DIESE l'ordonnancement des tâches journalières est réalisé de manière empirique sur un horizon d'une journée.

Toutefois, aucune de ces plate-formes ne prend en compte actuellement les objectifs de l'agriculteur. Ils ne sont également pas capables aujourd'hui d'intégrer l'anticipation dans le processus décisionnel. Cependant les recherches en agronomie soulèvent la nécessité de prendre en compte l'ensemble de ces facteurs. Les récents travaux de Snow (Snow et Lovatt, 2008) montrent la pertinence de cette piste. Elle intègre la notion d'anticipation sous la forme d'une exploration d'arbre de décisions. Elle propose une approche consistant à générer un premier plan. Lors de l'exécution de ce plan, on lance de nouvelles explorations de l'arbre permettant de résoudre les conflits en choisissant l'action la plus intéressante suivant un critère donné.

1.6 Exemple d'exploitation virtuelle

Dans cette section nous décrivons une exploitation virtuelle en polyculture. Cette exploitation nous sert de cas d'étude pour la simulation des décisions stratégique, tactique et opérationnelle. Dans les deux premières sections nous décrivons successivement la structure et les ressources de l'exploitation. Nous présentons ensuite les opérations agricoles et les différents itinéraires techniques des cultures. Enfin, nous définissons le profil de décision de l'agriculteur.

1.6.1 Description structurelle de l'exploitation virtuelle

Considérons une exploitation virtuelle issue des travaux de Dury et al. (2011). Comme l'indique la Figure 1.12, cette exploitation est constituée de quatre îlots structurels. Chacun de ces îlots structurels définit également un bloc fonctionnel $b \in \{1, 2, 3, 4\}$. Les blocs fonctionnels de l'exploitation se subdivisent en 15 parcelles $p_{[1-15]}$ de 12 *ha*. Avec une superficie totale 180 *ha*, l'exploitation virtuelle est de taille importante. Chaque bloc fonctionnel a une surface fixe (cf. Figure 1.12).

Quatre différentes cultures sont cultivées sur l'exploitation. Il s'agit du *blé d'hiver* (BH), de *l'orge de printemps* (OP), du *maïs* (MA) et du *colza d'hiver* (CH). La seule culture irriguée productible sur l'exploitation est le maïs. Le tableau 1.2 indique les délais de retour de chaque culture. Il précise également les îlots structurels sur lesquels ces cultures sont productibles décrivant ainsi la zone cultivable

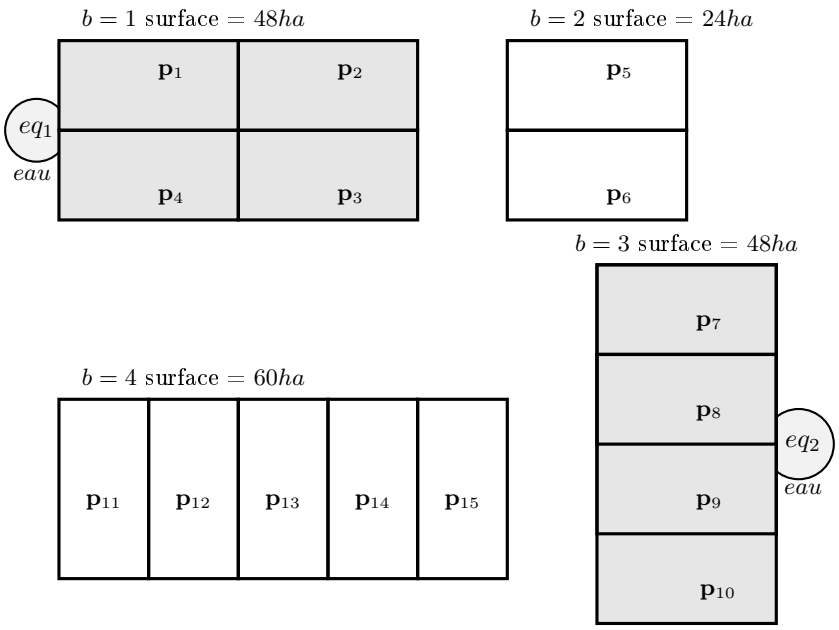


FIGURE 1.12 – *Exploitation virtuelle de 4 blocs fonctionnels, 15 parcelles de 12ha chacune. Les parcelles grises ont accès à des ressources en eau*

des cultures. L'indice des îlots structurels se confond avec celui des blocs fonctionnels.

TABLE 1.2 – *Caractéristiques des cultures produites sur l'exploitation virtuelle*

Culture	BH	OP	MA	CH
Délai de retour (nb. année)	2	3	2	3
Zone cultivable (blocs fct.)	{1,2,3,4}	{1,2,3,4}	{1,3}	{2,4}
Irrigable	Non	Non	Oui	Non

On distingue deux types de sol sur l'ensemble de l'exploitation. Le premier est celui des blocs fonctionnels 1 et 3 tandis que le second est celui des blocs fonctionnels 2 et 4.

Les assolements réalisés par l'agriculteur sur les cinq années précédentes sont décrites par le tableau 1.3. Cet historique des assolements révèle une chose majeure. Les délais de retour pratiqués par le passé ne sont pas les mêmes que ceux à utiliser pour le futur. En effet, sur le bloc fonctionnel 3 on observe une monoculture de maïs, tandis que la séquence de culture sur le bloc fonctionnel 1 était MA → MA → BH → OP. Ce type d'assolement est simplement impossible dans le futur. Il faudra donc trouver en fonction de l'historique de nouvelles séquences de culture qui satisfassent les nouvelles contraintes de délai retour.

1.6.2 Les ressources de l'exploitation virtuelle

L'exploitation dispose de :

- *2 ouvriers* qui travaillent 8h par jour,
- *1 tracteur* de type 1 T1,
- *1 tracteur* de type 2 T2,
- *1 Charrue* Ch,
- *1 Semoir combiné à une herse rotative* S,
- *1 Pulvérisateur* P,

TABLE 1.3 – *Historique de l'exploitation représentant l'assolement des cinq dernières années*

Parcelle	Année 1	Année 2	Année 3	Année 4	Année 5	Bloc fct.
$\mathbf{p}_1$	MA	MA	BH	OP	MA	1
$\mathbf{p}_2$	OP	MA	MA	BH	OP	
$\mathbf{p}_3$	BH	OP	MA	MA	BH	
$\mathbf{p}_4$	MA	BH	OP	MA	MA	
$\mathbf{p}_5$	BH	OP	BH	CH	BH	2
$\mathbf{p}_6$	OP	BH	CH	BH	OP	
$\mathbf{p}_7$	MA	MA	MA	MA	MA	3
$\mathbf{p}_8$	MA	MA	MA	MA	MA	
$\mathbf{p}_9$	MA	MA	MA	MA	MA	
$\mathbf{p}_{10}$	MA	MA	MA	MA	MA	
$\mathbf{p}_{11}$	BH	CH	BH	OP	BH	4
$\mathbf{p}_{12}$	CH	BH	OP	BH	CH	
$\mathbf{p}_{13}$	BH	OP	BH	CH	BH	
$\mathbf{p}_{14}$	OP	BH	CH	BH	OP	
$\mathbf{p}_{15}$	BH	CH	BH	OP	BH	

- *1 Cultivateur* C,
- *1 Moissonneuse batteuse* MB,
- *2 équipements fixes pour l'irrigation*,
- *10000 m^3 de quota d'eau par an.*

À l'exception des équipements d'irrigation, les ressources peuvent être utilisées sur toutes les parcelles. Comme l'indique la Figure 1.12, seuls les blocs fonctionnels 1 et 3 sont équipés en matériel d'irrigation fixe. Les équipements eq_1 et eq_2 sont respectivement associés aux blocs fonctionnels 1 et 3. Le quota d'eau annuel destiné à l'irrigation est de $10000m^3$ dont $6000m^3$ pour la ressource eq_1 et $4000m^3$ pour la ressource eq_2.

1.6.3 Opérations agricoles et modes de conduite

Chaque opération agricole est décrite par les ressources nécessaires, la vitesse d'exécution et les fenêtres temporelles représentant le début au plus tôt et la fin au plus tard. Les tableaux 1.4, 1.5, 1.6 et 1.7 décrivent respectivement les caractéristiques des opérations agricoles du *blé d'hiver*, de *l'orge de printemps*, du *maïs* (MA) et du *colza d'hiver*. De la même manière, la mention "-" indique l'inexistence de contraintes temporelles sur le début ou la fin de l'opération. Lorsque la date de début au plus tôt est supérieure à la date de fin au plus tard, l'opération est considérée comme étant à cheval sur deux années consécutives.

L'agriculteur dispose de trois itinéraires techniques pour la conduite de chacune des cultures. Les Figures 1.13, 1.14 1.15 et 1.16 décrivent les différents ITKs. Les nœuds représentent les opérations agricoles. Les arcs indiquent la précédence entre les opérations agricoles et le nombre de jours entre la fin du précédent et le début du suivant.

Ces itinéraires techniques permettent d'atteindre des objectifs prédéfinis. L'ITK

↓ *travail* permet à l'agriculteur de produire ses cultures tout en réduisant la charge de travail qu'elles représentent pour l'exploitation. Les itinéraires techniques ↓ *travail* garantissent un rendement acceptable. L'ITK ↓ *intrant* permet de réduire les intrants. Enfin, l'itinéraire technique ↑ *rendement* permet à l'agriculteur de maximiser son rendement.

1.6.3.a Les opérations agricoles et mode de conduite du blé

Les opérations agricoles pour la conduite du blé sont celles décrites dans le tableau 1.4.

TABLE 1.4 – *Contraintes temporelles et de ressources associées aux opérations agricoles de blé*

BLÉ D'HIVER (BH)					
Opération		**Début au plus tôt**	**Fin au plus tard**	**Ressource**	**Vitesse (ha/j)**
Labour	Lab.	01/10	15/11	1 T1/ 1 Ch / 1 Ouv.	9
Semis	Sem.	01/10	15/11	1 T2/ 1 S / 1 Ouv.	12
Désherbage 1	Des.1	-	20/11	1 T2/ 1 P / 1 Ouv.	21
Désherbage 2	Des.2	10/02	10/03	1 T2/ 1 P / 1 Ouv.	21
Fertilisation 1	Fer.1	05/02	25/02	1 T2/ 1 P / 1 Ouv.	21
Fertilisation 2	Fer.2	10/03	30/03	1 T2/ 1 P / 1 Ouv.	21
Fertilisation 3	Fer.3	10/05	30/05	1 T2/ 1 P / 1 Ouv.	21
Récolte	Rec.1	15/07	15/08	1 T2/ 1 MB / 2 Ouv.	15
Déchaumage	Dec.1	15/07	15/09	1 T1/ 1 Cu / 1 Ouv.	12

L'itinéraire technique ↓ *travail* exclut les opérations de labour. Seul un déchaumage est prévu après la récolte. Contrairement à cet ITK, ↑ *rendement* introduit un labour avant semis et deux déchaumages après la récolte. Les déchaumages sont réalisés à quinze jours d'intervalle au minimum. Pour une conduite du blé qui minimise l'apport d'intrant, (ITK ↓ *intrant*), le premier désherbage et la troisième fertilisation de l'ITK ↑ *rendement* sont supprimés.

1.6.3.b Les opérations agricoles et mode de conduite de l'orge

Les opérations agricoles pour la conduite de l'orge de printemps sont celles décrites dans le tableau 1.5.

L'itinéraire technique ↓ *travail* n'intègre pas d'opérations de labour avant le semis. Un déchaumage est prévu après la récolte. L'ITK, ↑ *rendement* introduit un labour avant semis et deux déchaumages, espacés d'au moins quinze jours, après la récolte. La deuxième fertilisation est supprimée pour la conduite minimisant l'apport d'intrant.

1.6.3.c Les opérations agricoles et mode de conduite du maïs

Les opérations agricoles pour la conduite du maïs sont celles décrites dans le tableau 1.6.

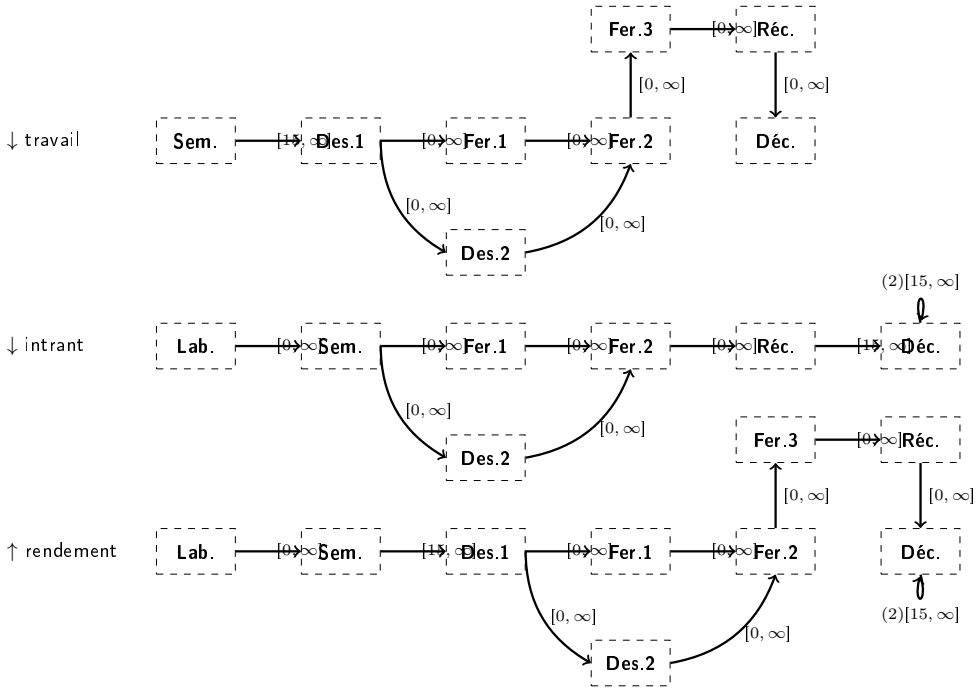


FIGURE 1.13 – *Itinéraires techniques du blé d'hiver*

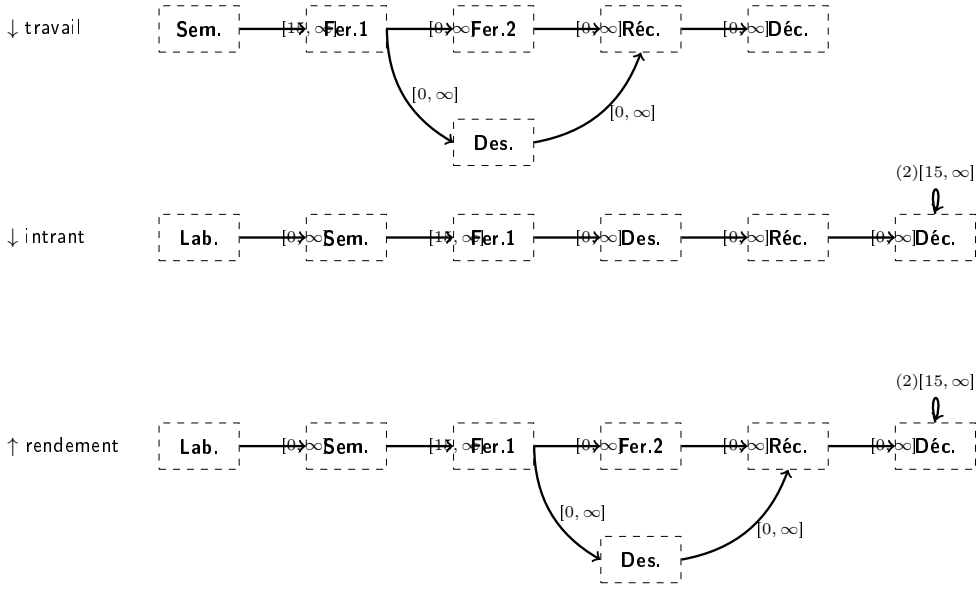


FIGURE 1.14 – *Itinéraires techniques de l'orge de printemps*

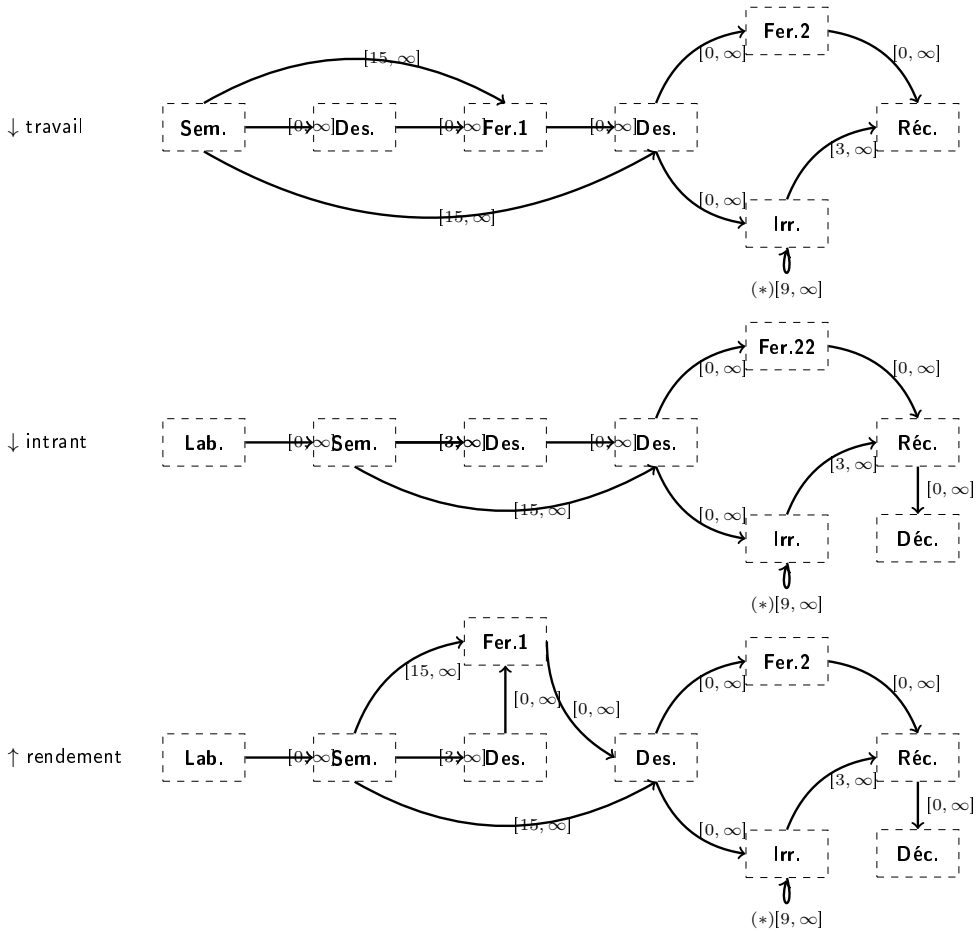


FIGURE 1.15 – *Itinéraires techniques de maïs*

Table 1.5 – *Contraintes temporelles et de ressources associées aux opérations agricoles d'orge*

Orge de printemps (OP)					
Opération		**Début au plus tôt**	**Fin au plus tard**	**Ressource**	**Vitesse (ha/j)**
Labour	Lab.	15/02	15/03	1 T1/ 1 Ch / 1 Ouv.	9
Semis	Sem.	15/02	15/03	1 T2/ 1 S / 1 Ouv.	12
Fertilisation 1	Fer.1	15/02	15/03	1 T2/ 1 P / 1 Ouv.	21
Fertilisation 2	Fer.2	05/04	25/04	1 T2/ 1 P / 1 Ouv.	21
Désherbage	Des.	15/04	30/04	1 T2/ 1 P / 1 Ouv.	21
Récolte	Rec.	01/07	15/07	1 T2/ 1 MB / 2 Ouv.	15
Déchaumage	Dec.	01/07	15/08	1 T1/ 1 Cu / 1 Ouv.	12

Table 1.6 – *Contraintes temporelles et de ressources associées aux opérations agricoles du maïs*

Maïs (MA)					
Opération		**Début au plus tôt**	**Fin au plus tard**	**Ressource**	**Vitesse (ha/j)**
Labour	Lab.	10/04	10/05	1 T1/ 1 Ch / 1 Ouv.	9
Semis	Sem.	10/04	10/05	1 T2/ 1 S / 1 Ouv.	12
Désherbage	Des.	-	-	1 T2/ 1 P / 1 Ouv.	21
Fertilisation 1	Fer.1	-	-	1 T2/ 1 P / 1 Ouv.	21
Fertilisation 2	Fer.2	05/06	25/06	1 T2/ 1 P / 1 Ouv.	21
Fertilisation 22	Fer.22	05/05	05/06	1 T2/ 1 P / 1 Ouv.	21
Irrigation	Irr.	15/05	15/11	1 Eq/ 16 Eau	21
Récolte	Rec.	15/10	15/11	1 T2/ 1 MB / 2 Ouv.	15
Déchaumage	Dec.	15/10	20/11	1 T1/ 1 Cu / 1 Ouv.	12

L'itinéraire technique ↓ *travail* n'intègre ni labour avant semis ni déchaumage après la récolte. L'ITK, ↑ *rendement* introduit ces deux opérations. Les deux fertilisations Fer.1 et Fer.2 sont supprimées pour la conduite minimisant l'apport d'intrants. Ces fertilisations sont par ailleurs remplacées par une nouvelle Fer.22 réalisée un peu plus tôt que Fer.2. De plus, le maïs étant la seule culture irriguée, une campagne d'irrigation est prévue dans l'ITK correspondant (cf. Figure 1.15). Les opérations agricoles d'irrigation sont espacées d'au moins 9 jours. Pour les trois types d'ITK, le nombre d'irrigations réalisé n'est pas connu d'avance.

1.6.3.d Les opérations agricoles et mode de conduite du colza d'hiver

Les opérations agricoles pour la conduite du colza d'hiver sont celles décrites dans le tableau 1.7.

À l'instar du blé d'hiver et de l'orge de printemps, l'itinéraire technique ↓ *travail* n'intègre pas d'opérations de labour avant le semis. Un déchaumage est prévu après la récolte. L'ITK, ↑ *rendement* introduit un labour avant semis et

TABLE 1.7 – *Contraintes temporelles et de ressources associées aux opérations agricoles du colza d'hiver*

COLZA D'HIVER (CH)					
Opération		**Début au plus tôt**	**Fin au plus tard**	**Ressource**	**Vitesse (ha/j)**
Labour	Lab.	15/08	15/09	1 T1/ 1 Ch / 1 Ouv.	9
Semis	Sem.	15/08	15/09	1 T2/ 1 S / 1 Ouv.	12
Désherbage 1	Des.1	-	-	1 T2/ 1 P / 1 Ouv.	21
Désherbage 2	Des.2	05/10	25/10	1 T2/ 1 P / 1 Ouv.	21
Fertilisation 1	Fer.1	10/09	20/09	1 T2/ 1 P / 1 Ouv.	21
Fertilisation 2	Fer.2	05/02	25/04	1 T2/ 1 P / 1 Ouv.	21
Récolte	Rec.	01/07	15/07	1 T2/ 1 MB / 2 Ouv.	15
Déchaumage	Dec.	-	-	1 T1/ 1 Cu / 1 Ouv.	12

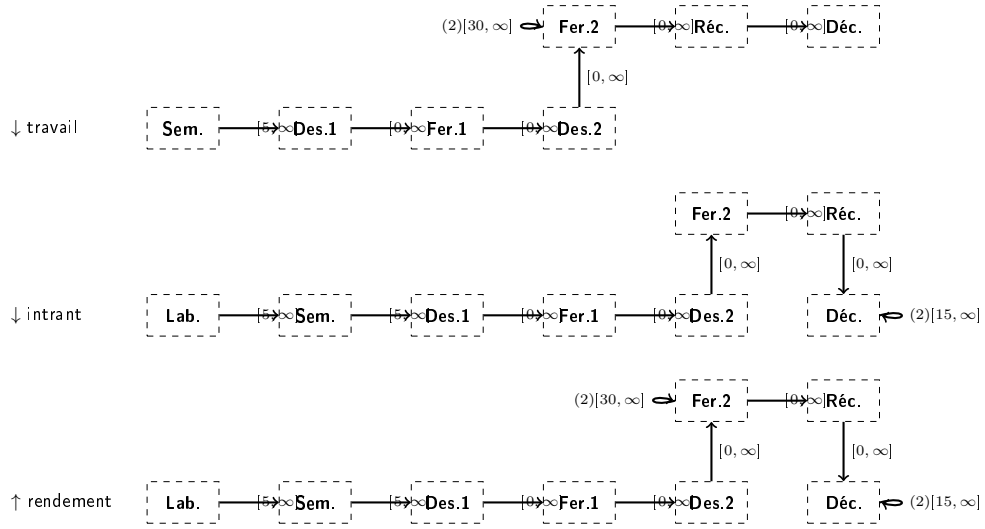


FIGURE 1.16 – *Itinéraires techniques de colza d'hiver*

rajoute un deuxième déchaumage quinze jours minimum après le premier. Pour ces deux ITK, deux fertilisations Fer.2 sont réalisées à trente jours d'intervalle. Une seule occurrence de cette deuxième campagne de fertilisations (Fer.2) est réalisée pour l'ITK ↓ *intrant*.

1.6.4 Profil de l'agriculteur

1.6.4.a ... pour la décision stratégique

La production annuelle de maïs doit être comprise entre $[40, 72]$ *ha* sur l'ensemble de l'exploitation. Cette production doit être répartie sur les blocs 1 et 3 de manière à avoir entre $[24, 48]$ *ha* et $[12, 24]$ *ha* respectivement sur les blocs fonctionnels 1 et 3. De la même manière, la production annuelle de blé d'hiver sur l'ensemble de l'exploitation est comprise entre $[70, 100]$ *ha*. Les autres cultures servent de variables d'ajustement.

La stratégie d'allocation de l'agriculteur est une fonction multicritère qui dépend de quatre facteurs.

1. Il préfère les séquences de culture de bonne qualité c'est-à-dire celles qui minimisent les effets précédents négatifs des cultures.
2. Il s'impose de respecter ces quotas de production annuelle.
3. Il préfère grouper au maximum ces îlots fonctionnels afin de réduire les déplacements liés à la réalisation des chantiers.
4. Il préfère, pour des raisons agronomiques liées au type de sol, les séquences de cultures admissibles sur les blocs fonctionnels 2 et 4 qui doivent contenir au moins une occurrence de colza.

1.6.4.b ... pour la décision tactique

L'agriculteur dispose d'un ensemble de préférences pour le choix des itinéraires techniques. Ainsi, pour les cultures principales de l'exploitation que sont le blé et le maïs, il préfère des conduites intensives. Les deux autres cultures secondaires doivent être conduites de manière à réduire les intrants tout en garantissant un rendement acceptable. Ainsi, l'ordre de préférence des ITK par culture est :

▷ *Blé d'hiver :* $\uparrow$ *rendement* - $\downarrow$ *intrant* - $\downarrow$ *travail*

▷ *Orge de printemps :* $\downarrow$ *intrant* - $\downarrow$ *travail* - $\uparrow$ *rendement*

▷ *Maïs :* $\uparrow$ *rendement* - $\downarrow$ *intrant* - $\downarrow$ *travail*

▷ *Colza :* $\downarrow$ *intrant* - $\downarrow$ *travail* - $\uparrow$ *rendement.*

1.6.4.c ... pour la décision opérationnelle

Pour cet agriculteur, plus une opération est réalisée tard par rapport à la date de début de période optimale de réalisation, plus cela aura un impact négatif sur le rendement de la culture. Il considère que les retards ont des conséquences égales pour toutes les opérations agricoles. De ce fait, la meilleure affectation des ressources aux opérations agricoles est celle qui minimise les inter-tâches.

Les récoltes sont prioritaires sur les semis et sur toutes autres opérations. Par ailleurs, les cultures principales sont prioritaires sur les secondaires. Le maïs est la culture la plus prioritaire de toutes.

1.7 Résumé du chapitre

Nous avons abordé dans ce chapitre les concepts spécifiques à la conduite des systèmes de culture au sein de l'exploitation agricole. Nous avons mentionné d'une part les concepts relatifs à la structure d'une exploitation agricole (ex : ressources, parcelles, zone cultivables etc.) et d'autre part les concepts de nature fonctionnelles permettant à l'agriculteur de raisonner la conduite de son exploitation (ex : bloc et îlot fonctionnel, délai de retour, séquence de culture, système de culture, itinéraire technique etc.).

La description des concepts agronomiques nous a permis d'aborder les trois types de problèmes de décision (stratégique, tactique et opérationnelle) auxquels nous nous proposons d'apporter une méthode de résolution dans cette thèse. Chacune de ces décisions porte sur des dimensions spatiales et temporelles variables allant de l'exploitation à la parcelle puis de plusieurs années à quelques jours. Notre caractérisation des trois différents problèmes de décision nous permet d'affirmer que :

– la DÉCISION STRATÉGIQUE est un problème de planification spatio-temporelle à long terme (plusieurs années) dans lequel des cultures sont affectées à

des parcelles de manière à atteindre les objectifs de production de l'agriculteur tout en satisfaisant un ensemble de contraintes et de préférences agronomiques.

- la DÉCISION TACTIQUE est un problème de planification d'actions duratives dans l'espace de plans partiellement ordonnés que sont les itinéraires techniques (ITKs). L'objectif de cette planification à moyen terme (plusieurs mois voir une année) est d'affecter des ITKs à des couples parcelle/culture de manière à respecter les préférences de conduite de l'agriculteur tout en tenant compte des contraintes temporelles et de ressources sur les opérations agricoles.
- la DÉCISION OPÉRATIONNELLE est un problème d'ordonnancement à court terme (quelques jours) d'un ensemble actions duratives et parallèles. Le but étant d'allouer des ressources réutilisables et consommables aux différentes opérations agricoles de manière à satisfaire les contraintes temporelles et les règles de priorités entre opérations.

Nous avons terminé ce chapitre en décrivant un cas d'étude basé sur une exploitation virtuelle. Ce cas d'étude nous permettra d'illustrer d'une part nos approches pour la planification stratégique, tactique et opérationnelle et d'autre part les mécanismes que nous proposons pour reproduire la dynamique d'interaction entre ces trois niveaux de planification.

Concepts généraux sur la planification et la modélisation & simulation en DEVS

2

Sommaire

1.1 Description des concepts relatifs à la conduite de systèmes de culture par un agent agriculteur 8
1.1.1 Pour commencer... 8
1.1.2 Dynamique de la décision chez un agent agriculteur 14
1.2 Problème d'allocation des cultures : décision stratégique 16
1.2.1 Description du problème 16
1.2.2 Formulation du problème 17
1.2.3 Utilité des plans stratégiques : les critères d'analyse 18
1.3 Choix du mode de conduite des cultures : décisions tactiques 19
1.3.1 Description du problème 19
1.3.2 Formulation du problème 20
1.3.3 Utilité des plans tactiques : les critères d'analyse 23
1.4 Organisation du travail : décision opérationnelle 23
1.4.1 Description du problème 23
1.4.2 Formulation du problème 26
1.4.3 Utilité des plans opérationnels : les critères d'analyse 26
1.5 Quelques travaux sur la simulation bio-décisionnelle en agronomie 27
1.5.1 Décision stratégique 27
1.5.2 Décisions tactique et opérationnelle 27
1.6 Exemple d'exploitation virtuelle 28
1.6.1 Description structurelle de l'exploitation virtuelle 28
1.6.2 Les ressources de l'exploitation virtuelle 29
1.6.3 Opérations agricoles et modes de conduite 30
1.6.4 Profil de l'agriculteur 35
1.7 Résumé du chapitre 36

Nous avons décrit dans le chapitre précédent les problèmes de décision dans les systèmes de culture à l'échelle de l'exploitation. Ces décisions imposent la prise en compte de contraintes agronomiques, temporelles et de ressources. Reproduire le processus de décision d'un agent agriculteur *in silico* se traduit par l'entrelacement des phases de décisions stratégiques, tactiques et opérationnelles.

Dans ce chapitre, nous présentons les concepts généraux nécessaires à la reproduction du comportement d'un agriculteur. Dans un premier temps nous présentons quelques approches de planifications (planification hiérarchique, planification par satisfaction de contrainte) et des architectures robotiques capables d'appréhender en intelligence artificielle des problèmes similaires au notre. Dans un deuxième temps, nous introduisons le cadre de simulation de systèmes dynamiques, distribués et hiérarchiques (DEVS). Nous nous baserons sur ce cadre pour la modélisation du problème et la simulation du comportement de l'agriculteur.

2.1 Planification de tâches et contrôle d'exécution

2.1.1 Introduction générale sur la planification

2.1.1.a Quel type de planification pour un agent agriculteur ?

La planification permet d'adopter un ensemble de décisions dans l'optique d'atteindre un objectif prédéfini. Elle consiste à raisonner l'organisation des décisions avant même que leur mise en œuvre ne soit réalisée. Ces décisions sont généralement appelées des *plans*. L'exécution d'un plan est en quelque sorte la mise en œuvre des décisions suivant un ordre établi par le plan. Dans le domaine de l'intelligence artificielle (IA), la planification occupe une partie très importante. De nombreux ouvrages d'IA (Russell et Norvig, 2010; LaValle, 2006; Ginsberg, 1994) présentent différentes techniques de planification. Toutefois, le terme planification peut s'avérer très ambiguë. En effet, il existe autant de planifications que de types de décisions auxquelles on pourrait être confronté. Pour exemple en robotique, Ghallab (2001) parle (i) de *planification de mouvement* pour la construction du chemin et de la trajectoire d'un robot dans l'espace, (ii) de *planification pour la communication* en référence aux interactions robot-robot voir homme-robot, (iii) de *planification de tâches* pour l'organisation des actions (tâches) d'un ou plusieurs robots etc.

Les problèmes de planification de tâches sont les plus proches de ceux auxquels nous nous intéressons pour la conduite des systèmes de culture au sein d'une exploitation agricole. Pour Ghallab (2001), c'est la forme la plus générale et la plus abstraite de la planification. Elle vise à déterminer et à organiser un ensemble de tâches dans le temps et à leur attribuer des ressources compte tenu des évolutions prévisibles de l'environnement. Ainsi, nous parlerons tout au long de ce manuscrit de planification en référence à la *planification de tâches*.

2.1.1.b Représentations des problèmes de planification

De manière générale, un problème de planification décrit les états de l'environnement et la manière dont ces états évoluent sous l'influence des actions d'un agent. Selon Ghallab et al. (2004), un problème de planification est décrit par un système de transition d'état $\langle S, \mathcal{A}, \delta, s_0, g\rangle$ dans lequel S est l'ensemble des états possibles du monde, $\mathcal{A}$ l'ensemble des actions et $\delta(s,a)$ [1] : $S \times \mathcal{A} \rightarrow S$ une fonction de transition qui à tous couples *état-action* fait correspondre un état d'arrivée, $s_0 \in S$ et $g \in S$ respectivement les états initiaux et buts.

Il existe deux approches de représentation des problèmes de planification.

Représentation classique : la *représentation classique* Ghallab et al. (2004) ou représentation STRIPS (*STanford Research Institute Problem Solver*) (Fikes et Nilsson, 1971) est basée sur une logique du premier ordre. Notons cependant qu'en dépit du fait que la représentation STRIPS ne soit pas strictement propositionnelle, les termes fonctionnels ne sont pas permis. Les arguments des prédicats ne peuvent être que des constantes ou des variables appartenant à des domaines finis. Ainsi, les actions sont dans ce cas représentées comme des opérateurs de transition instantanée entre deux états du système.

Une action est décrite d'une part par ses *préconditions* c'est-à-dire par un ensemble de propositions qui doivent être satisfaites ou non satisfaites avant l'exécution de l'action. D'autre part, la représentation d'une action intègre également

1. Dans Ghallab et al. (2004) δ est noté γ. Nous l'avons changé en δ pour garder la même notation pour les fonctions des transitions tout au long du manuscrit

les *effets*, l'ensemble des propositions qui sont rendues vraies (eff_a^+) ou rendues fausses (eff_a^-) par l'exécution de l'action. La fonction de transition peut donc s'écrire $\delta(s,a) = (s - eff_a^-) \cup eff_a^+$.

Représentation orientée variables d'états : elle exploite une notation fonctionnelle des états (Ghallab et al., 2004). L'état est dans ce cas représenté par des variables d'état de la forme $x(v_0, v_1, \ldots, v_k) \in X$ où x est le nom de la variable, v_i est un objet ou la valeur prise par une variable décrivant un objet. Chacune des variables est associée à une fonction de correspondance :

$$x \quad : \quad \mathcal{D}_0 \times \mathcal{D}_1 \times \ldots \mathcal{D}_k \times S \rightarrow \mathcal{D}_x$$

Où $\mathcal{D}_i \subseteq \mathcal{D}$ sont les sous domaines de la fonction de correspondance qui représente l'union d'une ou de plusieurs classes d'objets appartenant au domaine total $\mathcal{D}$. Les variables représentant des prédicats peuvent être associées à un domaine de booléen $\mathcal{D} = \{0, 1\}$.

Dans cette représentation, l'état S est constitué d'un ensemble s, d'états définis par $s = \{x \mapsto v | x \in X \wedge v \in \mathcal{D}_x\}$. La représentation est identique à la précédente à l'exception du fait que :

(i) les préconditions sont définies par un ensemble d'expressions sur les variables d'état,

(ii) les effets définissent des assignations de valeurs aux variables d'état.

En conséquence, la fonction de transition est définie par $\delta(s,a) = \{x \mapsto v | x \in \mathcal{D}\}$ avec c l'assignation $(x \leftarrow v) \in eff_a$.

Notons cependant que les problèmes du monde réel nécessitent généralement une représentation très expressive des systèmes et qu'ils imposent la prise en compte du temps et des ressources. Dans les sections 2.1.1.c et 2.1.1.d, nous détaillons les types de représentations généralement adoptées pour le temps et les ressources.

2.1.1.c Représentation du temps en planification

Le temps est sans doute la ressource la plus importante en planification. Dans les systèmes de planification classique basés sur les transitions d'état, les actions sont instantanées. La notion de temps est définie de manière implicite au travers du lien de causalité entre les effets et les préconditions des actions. Vere (1983) propose une alternative pour appréhender les actions duratives dans laquelle les préconditions doivent rester vraies tout au long de l'exécution de l'action. Les effets ne passent à vrai qu'à la fin de l'action. Malgré cela, il est courant de rencontrer des problèmes dans lesquels les actions sont non seulement duratives mais les effets attendus des actions ne sont pas instantanées. Plus encore, ces effets attendus peuvent être le fruit de plusieurs actions jointes. À cela s'ajoute le fait que certains événements liés à la dynamique de l'environnement peuvent être attendus à des dates spécifiques. Dans ces conditions, une représentation explicite du temps s'impose.

Deux approches existent pour la représentation du temps en planification

(i) *L'approche qualitative* pour exprimer les relations temporelles symboliques (*Algèbre des intervalles*) entre les actions, les événements de l'environnement.

(ii) *L'approche quantitative* pour exprimer des relations temporelles absolues ou des restrictions entre la distance entre des instants (*Algèbre des instants*).

Algèbre des intervalles : Allen (1984) propose 13 relations primitives (*before, meet, start, finish, equal, overlap, during, after, is-met-by, is-started-by, is-finished-by, is-overlapped-by, includes*) qui permettent de raisonner sur des intervalles et

2^{13} contraintes binaires qualitatives. La Figure 2.1, représente les sept premières relations primitives. Les six autres peuvent être retrouvées par symétrie.

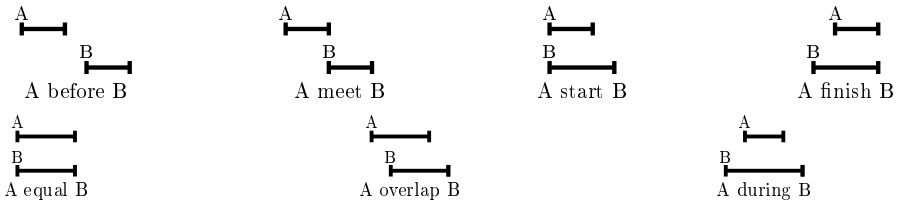


FIGURE 2.1 – *Relations primitives d'Allen.*

Algèbre des instants : proposée par Vilain et al. (1986), cette représentation du temps, basée sur des instants, exploite des relations primitives $\{=, <, >, \leq, \geq, \neq\}$ pour des raisonnements quantitatifs sur les instants. Marc Vilain définit son algèbre des instants continus comme une sous-classe de l'algèbre des intervalles d'Allen. Cette sous-classe est capable de représenter les relations temporelles sous la forme d'une disjonction de relations entre des instants appartenant à des intervalles. Cette représentation a plusieurs avantages comme celui de spécifier des intervalles sur les dates de début et de fin des actions ou celui d'exprimer des contraintes numériques liées aux décalages temporels (*timelag*) entre actions.

Dans les problèmes où le raisonnement quantitatif sur le temps est crucial, l'approche proposée par Dechter et al. (1991) (*Temporal Constraint Satisfaction Problem* (TCSP)) est généralement adoptée. Ce cadre permet d'exprimer dans un sous-ensemble de points, des contraintes unaires ou binaires.

Par exemple, notons $[S_A^{min}, S_A^{max}[$, $[S_B^{min}, S_B^{max}[$ (respectivement $[F_A^{min}, F_A^{max}[$, $[F_B^{min}, F_B^{max}[$) les intervalles d'ouverture des actions A et B (respectivement les intervalles de fermeture). On peut exprimer une relation composée de type *before-meet* par $(S_A^{min} \leq S_A < S_A^{max}) \wedge (F_A^{min} \leq F_A < F_A^{max}) \wedge (S_A < F_A) \wedge (F_A \leq S_B) \wedge (S_B < F_B)$ où S_i et F_i représentent respectivement les dates de début au plus tôt et de fin au plus tard. Le cadre général TCSP permet également d'exprimer des contraintes disjonctives portant sur les distances entre deux points. Par exemple, pour exprimer plusieurs fenêtres d'ouvertures pour l'action A on écrira $(S_A^{min} \leq S_A < S_A^{max}) \vee (S_A^{min'} \leq S_A < S_A^{max'}) \vee (S_A^{min''} \leq S_A < S_A^{max''})$.

Toutefois, dans la plupart des planificateurs temporels, une version simplifiée appelée *Simple Temporal Problem* (Dechter et al., 1991) est utilisée. Elle modélise le problème sous la forme d'un réseau temporel simple : *Simple Temporal Network* (STN). Ce réseau représente un graphe orienté dans lequel les nœuds sont les instants et les arcs sont les contraintes de distance entre les instants. Dans ce cas chaque arc est labellisé par un et un seul intervalle. Il existe des algorithmes efficaces (cf. section 5.2.2) pour l'ajout et la propagation des contraintes temporelles.

2.1.1.d Représentation des ressources en planification

Généralement, les applications issues du monde réel nécessitent, pour l'exécution des actions planifiées, des raisonnements numériques qui prennent en compte des ressources limitées. On parle généralement d'ordonnancement, en référence à la détermination précise de l'usage des ressources pour la mise en œuvre des actions.

Ces ressources sont représentées par des variables d'état portant sur des objets physiques nécessaires à l'exécution d'une tâche. Chaque ressource est disponible en quantité limitée c'est-à-dire qu'il existe pour chaque ressource une capacité maximale Q_{max}. Une ressource r est associée à deux fonctions particulières qui sont : la fonction de *consommation* $cons(q(r, a))$ et la fonction de *production* $prod(q(r, a))$

de la ressource (avec $q(r,a)$ la quantité de r consommée ou produite par la tâche a). Il existe une contrainte d'ordre entre les phases de consommation et production des ressources. Pour une tâche a, s'exécutant entre $[t_s, t_f]$, la consommation à lieu à la date de démarrage t_s de la tâche tandis que la production a lieu à la date de fermeture t_f.

Il existe différentes catégories de ressources qui sont : **consommables** et **réutilisables** (Smith et Becker, 1997; Bedrax-weiss et al., 2003). Elles se différencient principalement par leur mode de production et de consommation.

Les ressources **consommables** sont des ressources dont la capacité peut être diminuée par une tâche (ex : stock d'eau). Elles peuvent être potentiellement produites par une autre tâche du système. Ainsi, l'utilisation d'une ressource consommable r par une tâche a, s'exécutant entre $[t_s, t_f]$, est définie par $Use(r,a) \equiv cons(q(r,a)) \wedge prod(q(r,a))$ (avec $q(r,a) = 0$ à la date t_f de fermeture de a).

Les ressources **réutilisables** sont des ressources qui ne peuvent être que momentanément exploitées par des tâches. Elles sont consommées au démarrage de la tâche puis relâchées (produites) à la fin de celle-ci. (ex : tracteur, ouvrier). Ainsi, l'utilisation de la ressource réutilisable r par une tâche a, s'exécutant entre $[t_s, t_f]$, est définie par $Use(r,a) \equiv cons(q(r,a)) \wedge prod(q(r,a))$ (avec $q(r,a)$ à la date t_s égale à $q(r,a)$ à la date t_f).

Suivant les quantités produites ou consommées, une ressource peut être *discrète* (ex : tracteur) ou *continue* (ex : stock d'eau). Une ressource peut être à capacité unique ($q(r,a) = 0$ si consommé et $q(r,a) = 1$ sinon) ou multiple ($Q_{max}(r) > 1$). Dans le cas où une ressource est à usage multiple, on parlera de ressources *partageable* et de ressources *non-partageable* dans le cas contraire.

Il existe différentes approches (cf. section 5.2.3) permettant de raisonner sur les ressources. La plupart d'entre elles (Laborie, 2003) se basent sur la propagation de contraintes dans l'espace de plans ou d'actions.

2.1.1.e Langages de planification

Il existe deux principales approches de représentation des domaines de planification qui sont : STRIPS (*STanford Research Institute Problem Solver* (Fikes et Nilsson, 1971)) et ADL (Action Description Language (Pednault, 1994)).

La représentation STRIPS peut être considérée comme le premier langage de description de domaines de planification. Les opérateurs STRIPS sont décrits par les préconditions et les effets des actions. Le langage ADL est une extension de STRIPS qui autorise l'utilisation de quantificateurs et d'expressions conditionnelles dans la description de l'opérateur. Le langage le plus utilisé dans de nombreux planificateurs basés sur ADL est PDDL (*Planning Domain Description Language* Mcdermott et al. (1998)). Dans sa version 2.1 (Fox et Long, 2003), ce langage permet de prendre en compte :

- des variables discrètes et continues dans les préconditions,
- des préconditions et effets temporels,
- des opérateurs de relations $=, >, <, <=, >=$,
- des actions duratives.

La BNF (*Backus Normal Form*) complète du langage PDDL 2.1 est présentée dans Fox et Long (2003). Par exemple, la Figure 2.2 montre une représentation PDDL d'une opération de semis de maïs. On retrouve notamment le nom et les paramètres de l'opération. La contrainte de durée, dans cet exemple, est égale au produit de la superficie de la parcelle par la vitesse de semis. Les conditions et les effets portent aussi bien sur le début que la fin de l'opération de semis.

```
;; Opération de semis
(:durative-action Semis-MA
:parameters(?p - Parcelle ?t2 - Tracteur2 ?o - Ouvrier ?s - Seeddrill)
;; Durée de l'action
:duration (= ?duration (* (surface ?p) (vitesse-semis ?t2)) )
:condition
(and
   ;; Prédicat indiquant si le semis est réalisable
   (conditionSemis)
   ;; Contrainte temporelle d'ouverture de l'action
   (at start (>= (date)  99) )
   ;; Contrainte temporelle de fermeture de l'action
   (at end (<= (date)  129) )
  )
:effect (and
      ;; Consommation de ressources: un ouvrier, un tracteur et un semoir
      (at start (decrease (capacite ?t2) 1))
      (at start (decrease (capacite ?o) 1))
      (at start (decrease (capacite ?s) 1))
      ;; Production de ressources: un ouvrier, un tracteur et un semoir
      (at end (increase (capacite ?t2) 1))
      (at end (increase (capacite ?o) 1))
      (at end (increase (capacite ?s) 1))
    )
)
```

FIGURE 2.2 – *Représentation PDDL d'une opération de semis de maïs*

2.1.1.f Les classes de planificateurs

Pour appréhender les différents types de problème de planification, on distingue deux grandes classes de planificateurs (Ghallab et al., 2004).

La première est celle des planificateurs dits *indépendants du domaine* tel que HSP (Bonet et Geffner, 1999), FF (Hoffmann et Nebel, 2001), LPG (Gerevini et al., 2003) et FastDownward (Helmert, 2006). Ces planificateurs sont conçus dans l'optique de résoudre un grand nombre de problèmes pour lesquels les buts à atteindre sont explicites. Aucune limite n'est imposée sur les types de problèmes à résoudre, dès l'instant où le planificateur dispose d'une connaissance complète du monde et que le système est déterministe et statique. Il existe des planificateurs indépendants du domaine basé sur les réseaux de contraintes (Constraint Satisfaction Problem - CSP Montanari (1974)). Dans cette thèse, nous nous intéresserons particulièrement à ce type de planificateur pour les décisions stratégiques et opérationnelles. En effet, pour ces deux problèmes, nous disposons d'un modèle déterministe et complet de l'environnement de l'agent. Nous reviendrons plus en détail sur cette approche dans la section 2.1.2.

La seconde classe est celle des planificateurs dits *spécifiques au domaine* tels que ASPEN (Rabideau et al., 1999), TLplan (Bacchus et Ady, 2001), TALplan (Doherty et Kvarnström, 2001) et SHOP2 (Nau et al., 2003). Ces planificateurs sont, quant à eux, conçus pour des types de problèmes particuliers. L'idée principale étant d'exploiter les connaissances du domaine de planification afin de restreindre la taille de l'espace de recherche. Par exemple, dans Bacchus et Ady (2001) les auteurs suggèrent l'expression des connaissances sous la forme de règles de contrôle. Erol et al. (1994) et Nau et al. (2003) quant à eux proposent d'exprimer les connaissances sous la forme d'un réseau de tâches hiérarchiques (*Hierarchical task Network*, HTN). Dans notre cas, ce type de planificateur peut convenir à la planification tactique.

En effet, l'agriculteur ne réinvente pas à chaque fois ses itinéraires techniques de cultures à partir d'un objectif final qui serait de produire une culture donnée. Il dispose généralement de connaissances sur les différentes manières de résoudre le problème. Nous reviendrons plus en détail sur cette approche dans la section 2.1.3.

Dans la suite nous développons deux approches de planification que nous exploitons dans cette thèse. Il s'agira premièrement de développer une technique de planification basée sur les réseaux de contraintes à coûts. Ensuite nous aborderons les techniques de planification hiérarchique.

2.1.2 Planification par satisfaction de contraintes pondérées

Les techniques de planification basées sur la satisfaction de contraintes (Constraint Satisfaction Problem CSP Montanari (1974)) permettent d'appréhender certaines classes de problèmes généraux. Ces techniques offrent des algorithmes efficaces permettant de rechercher les solutions d'un problème. Nous décrivons dans cette section les CSP et une extension des CSP qui permet de modéliser non seulement des contraintes *dures* mais également des critères d'optimisation exprimés sous la forme d'une agrégation de fonctions de coût (Schiex et al., 1995).

2.1.2.a Problème de satisfaction de contraintes

Définition 2.1 (Problème de satisfaction de contraintes) *Un problème de satisfaction de contraintes est défini par un tuple* $\langle \mathcal{X}, \mathcal{D}, \mathcal{C} \rangle$ *avec*

- $\mathcal{X} = \{x_1, \cdots, x_n\}$ *: un ensemble fini de variables,*
- $\mathcal{D} = \{\mathcal{D}_1, \cdots, \mathcal{D}_n\}$ *: un ensemble fini de domaines de variables tel que chaque variable* $x_i \in \mathcal{X}$ *soit associée à un domaine fini de valeur* $\mathcal{D}_i \in \mathcal{D}$
- $\mathcal{C} = \{\mathcal{C}_{\mathrm{s}_1}, \cdots, \mathcal{C}_{\mathrm{s}_e}\}$ *: un ensemble contraintes dures (à satisfaire absolument). Chaque contrainte dure* $\mathcal{C}_{\mathrm{s}_j} \in \mathcal{C}$ *porte sur un sous-ensemble de variables* $\mathrm{s}_j \subseteq \mathcal{X}$*, appelé* **portée** *de la contrainte. La taille* $|\mathrm{s}_j|$ *de la portée c'est-à-dire, le nombre de variables qu'elle comporte est généralement appelé l'***arité** *de la contrainte.*

Les contraintes dures imposent des restrictions sur les affectations de valeurs aux variables de sa portée. Soit $l[\mathrm{s}_j]$ l'ensemble des combinaisons de valeurs sur la portée s_j. Chaque contrainte dure $\mathcal{C}_{\mathrm{s}_j}$ n'autorise qu'un sous ensemble des combinaisons de valeurs de $l[\mathrm{s}_j]$.

Définition 2.2 (Affectation d'une variable) *Affecter une variable* x_i *consiste à lui attribuer une valeur* v *appartenant à son domaine* $\mathcal{D}_i$*. L'affectation est généralement notée* $(x_i = v)$*.*

On parlera d'une *affectation complète* A pour signifier une affectation de toutes les variables à une valeur. Autrement dit, $\mathsf{A} \in l[\mathcal{X}]$ est un élément de l'ensemble des combinaisons de valeurs sur l'ensemble des domaines $\mathcal{D}$ des variables $\mathcal{X}$. Si l'affectation ne porte que sur un sous ensemble de variables s_j, on parlera d'*affectation partielle* notée $\mathsf{A}[\mathrm{s}_j]$.

Définition 2.3 (Projection d'une affectation) *La* projection *d'une affectation (partielle ou totale)* $\mathsf{A} = \{x_1 = v_1, \ldots, x_k = v_k\}$ *sur l'ensemble* $Y = \{x_{i_1}, \ldots, x_{i_p}\} \subset \{x_1, \ldots, x_k\}$ *est l'affectation partielle* $\{x_{i_1} = v_{i_1}, \ldots, x_{i_p} = v_{i_p}\}$*. Elle est notée* $\mathsf{A}[Y]$*.*

Un problème de satisfaction de contraintes peut être représenté par un graphe de contraintes.

Définition 2.4 (Graphe de contraintes) *Un graphe de contraintes est un graphe dans lequel les nœuds représentent les variables (un nœud par variable) et les arcs représentent des contraintes entre les deux variables.*

Pour représenter les domaines des variables appartenant à la portée d'une contrainte, on utilise la notion de réseau de valeurs.

Définition 2.5 (Réseau de valeurs) *Le réseau de valeurs permet de représenter graphiquement les domaines des variables appartenant à la portée d'une contrainte.*

Pour une contrainte $\mathcal{C}_{i,\mathrm{S}}$, le réseau de valeurs des représenté par un graphe bipartite $G = (\mathcal{X}, Y, E)$. Les nœuds de $\mathcal{X}$ représentent les variables de $x_i \in \mathrm{S}$ tandis que les nœuds de Y représentent les valeurs appartenant à l'union des domaines des variables x_i. Les arcs (x_i, v_k) sont définis tel que si $(x_i, v_k) \in E$ alors $v_k \in \mathcal{D}_i$. La Figure 2.3 représente le réseau de variables de la contrainte d'arité 4 $(x_0 + x_1 + x_2) < x_3$.

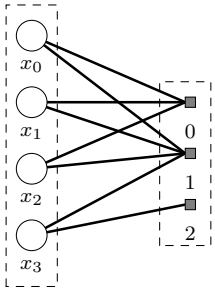


FIGURE 2.3 – *Réseau de valeurs de la contrainte* $(x_0 + x_1 + x_2) < x_3$

La solution d'un problème de satisfaction de contraintes est une affectation complète A qui satisfait toutes les contraintes dures $\mathcal{C}_{\mathrm{S}_j}$.

2.1.2.b Les réseaux de contraintes pondérées

Les CSP ne modélisent que l'autorisation ou l'interdiction de combinaisons de valeurs. Or certains problèmes issus du monde réel sont par nature sur-contraints, c'est-à-dire qu'ils ne possèdent aucune solution. Lorsqu'on cherche à modéliser ces types de problème, il est primordial de définir des contraintes dures (à satisfaire absolument) et des contraintes souples encore appelées *préférences*. Ces préférences définissent des contraintes que l'on souhaitent voir vérifier dans des solutions de bonnes qualités. Ainsi, l'objectif n'est plus de voir toutes les contraintes satisfaites mais plutôt de les satisfaire du mieux possible. En d'autres termes, il s'agit de satisfaire toutes les contraintes dures et de minimiser une agrégation des coûts des préférences insatisfaites.

Pour modéliser des préférences sur certaines combinaisons de valeurs, les réseaux de contraintes valuées (VCSP, *Valued Constraint Satisfaction Problem*) ont été proposés (Schiex et al., 1995). Cette extension de CSP permet d'appréhender plusieurs différentes classes de problèmes dont les réseaux de contraintes pondérées (WCSP, *Weighted Constraint Satisfaction Problem* Meseguer et al. (2006)).

Les réseaux de contraintes pondérées (WCSP) sont de ce fait, une extension des réseaux de contraintes (CSP) qui permet d'ajouter aux CSP une structure de valuation permettant de définir une structure algébrique caractérisant les coûts associés à certaines combinaisons de valeurs.

Définition 2.6 (Réseau de contraintes pondérées) *Un réseau de contraintes pondérées, WCSP est défini par un tuple* $\langle \mathcal{X}, \mathcal{D}, \mathcal{W} \rangle$ *avec*

▷ $\mathcal{X} = \{x_1, \cdots, x_n\}$ *: un ensemble fini de variables,*

▷ $\mathcal{D} = \{\mathcal{D}_1, \cdots, \mathcal{D}_n\}$ *: un ensemble fini de domaines de variables tel que chaque variable* $x_i \in \mathcal{X}$ *soit associée à un domaine fini de valeurs* $\mathcal{D}_i \in \mathcal{D}$

▷ $\mathcal{W} = \{\mathcal{W}_{\mathrm{s}_1}, \cdots, \mathcal{W}_{\mathrm{s}_e}\}$ *: un ensemble de fonctions de coûts. Soit* $l[\mathrm{s}_i]$ *l'ensemble des combinaisons de valeurs sur la portée* s_i*. Chaque fonction de coûts* $\mathcal{W}_{\mathrm{s}_i} \in \mathcal{W}$ *est définie par* $\mathcal{W}_{\mathrm{s}_i} : l(\mathrm{s}_i) \rightarrow [0, m]$ *avec* $m \in [1, \cdots, +\infty]$.

Le coût cost d'une affectation complète $\mathsf{A} \in l[\mathcal{X}]$ est :

$$\mathsf{cost}(\mathsf{A}) \quad = \quad \sum_{\mathcal{W}_{\mathrm{s}_i} \in \mathcal{W}} \mathcal{W}_{\mathrm{s}_i}(\mathsf{A}[\mathrm{s}_i])$$

où $\mathsf{A}[\mathrm{s}_i]$ est la projection d'une affectation de valeurs sur le sous ensemble de variables s_i.

La solution d'un WCSP est une affection complète $\mathsf{A} \in l[\mathcal{X}]$ de coût $\mathsf{cost}(\mathsf{A})$ telle que

$$\mathsf{cost}(\mathsf{A}) \quad = \quad \underset{\mathsf{A} \in l[\mathcal{X}]}{min} \left(\sum_{\mathcal{W}_{\mathrm{s}_i} \in \mathcal{W}} \mathcal{W}_{\mathrm{s}_i}(\mathsf{A}[\mathrm{s}_i]) \right)$$

2.1.2.c Technique de résolution pour les WCSP

Pour trouver une affectation des variables du WCSP qui minimise les coûts, on utilise généralement l'algorithme exact, de séparation évaluation nommé DFBB (*Depth First Branch and Bound* cf. l'algorithme 1). Cet algorithme consiste à énumérer récursivement les valeurs possibles des variables. Un nœud de l'arbre de recherche correspond à une affectation d'une variable à une valeur. À chacun de ces nœuds, on calcule un minorant $lb(\mathsf{A})$ et un majorant ub qui dépend du coût des solutions contenues dans le sous-arbre courant. Ce sous-arbre correspond à toutes les extensions possibles de l'affectation courante A. Le majorant est donné par la valeur minimum des coûts des solutions déjà trouvées. Si le minorant est plus grand que le majorant alors toutes les extensions de l'affectation, dans le sous-arbre ne peuvent pas être des solutions optimales. Il n'est donc pas nécessaire de continuer la recherche à partir de cette affectation. Dans ces conditions, l'algorithme retourne à la variable déjà affectée.

Algorithme 1 : DFBB (X, A, ub) : coût

```
début
    c ← lb(A)
    si c < ub alors
        si | A |= n alors retourner c
        Choisir Y dans X
        X ← X ∖ {Y}
        forall v ∈ D_Y do
            ub ← min(ub, DFBB(X, A ∪ {Y ← v}, ub)
        retourner ub
    retourner le majorant initial
fin
```

La complexité de DFBB est en $O(md^n)$ où m est le nombre de fonctions de coûts, d est la taille maximale du domaine des variables et n le nombre de variables.

Cependant, l'efficacité de l'algorithme dépend de la qualité du minorant $lb(\mathsf{A})$ utilisé. Un minorant trivial consiste à cumuler les coût des fonctions dont les variables sont déjà affectées. Généralement ce type de minorant est de mauvaise qualité. Pour trouver de meilleurs minorants des fonctions de coût d'un WCSP, on utilise des mécanismes de filtrage. Le but de ces mécanismes de filtrages est de retirer du domaine de variables, certaines valeurs qui n'appartiennent à aucune solution et augmenter par la même occasion le coût de la fonction qui servira de minorant. Par exemple, les algorithmes dit de *cohérence d'arc* Schiex (2000) permettent de filtrer localement les domaines des variables. Le réseau de contraintes qui résulte de ce filtrage est dit *arc-consistant* si toutes ses valeurs sont viables et que le domaine. De nombreuses techniques de filtrage existent. Quelques une sont détaillées dans de Givry (2011).

2.1.2.d Le solver Toulbar2

Toulbar2 est un solveur de WCSP libre et disponible sur à l'adresse `http://mulcyber.toulouse.inra.fr/projects/toulbar2/`. Il intègre un ensemble de techniques de cohérences locales et globales. Le solveur dispose également de méthode de recherche utilisant des décompositions arborescentes. Les expérimentations présentées dans le chapitre 4 sont réalisées en utilisant Toulbar2.

2.1.3 Planification hiérarchique

2.1.3.a Généralités

La planification hiérarchique se base sur le concept d'hiérarchie d'abstraction des actions dont le but est de réduire la complexité liée à l'espace de recherche.

Les niveaux d'abstraction caractérisent la granularité avec laquelle on décrit le problème. Il existe deux façons d'introduire des hiérarchies d'abstraction dans les processus de planification. On retrouve d'une part, la hiérarchisation des buts à accomplir et d'autre part la hiérarchisation des opérateurs disponibles.

La première approche a été introduit pour la première fois dans le planificateur Abstrips (Sacerdoti, 1974). Dans ce planificateur, la hiérarchisation consiste à définir une mesure d'importance entre les préconditions d'actions. La planification à plus haut niveau est autorisée à ignorer les préconditions d'actions de plus bas niveau afin de dériver la structure générale d'un plan exécutable. A chaque niveau d'abstraction, le planificateur ne prend en compte que les préconditions de niveau le plus élevé.

La seconde approche d'abstraction se base sur une hiérarchisation des opérateurs. Elle consiste à définir des opérateurs à différents niveaux d'abstraction. Cette approche permet de décrire des tâches très complexes qui sont constituées d'autres tâches plus élémentaires. La hiérarchisation des opérateurs se retrouve au départ dans le planificateur NOAH (Sacerdoti, 1975) avec ensuite une amélioration dans NONLIN (Tate, 1977). NOAH représente le problème de planification comme une liste de tâches partiellement ordonnées. Les informations sur la manière de décomposer les tâches sont encodées par des méthodes dans ce qui est appelé des *soup code*. Les relations entre les tâches sont appréhendées en introduisant un ordre ou des variables de restriction entre tâches. Dans NONLIN, les *soup code* de NOAH sont remplacés par des *opschemas* qui représentent la manière de décomposer les tâches. NONLIN introduit deux structures de données qui sont la TOME (*Table of Multiple Effects*) et le GOST (*Goal Structure*) pour garder la trace des effets des tâches et leurs influences sur les préconditions.

Dans les années 90, le système DIVISER (Vere, 1990), une extension de NOAH et NONLIN, a été développé afin de prendre en compte les tâches duratives et les tâches liées à des contraintes temporaires. Les travaux de Wilkins (1984) sur SIPE ont introduit la gestion de ressource et l'interaction avec l'utilisateur du planificateur. Le planificateur O-Plan de (Currie et Tate, 1991), quant à lui propose une extension plus générale de NONLIN pour la création et l'exécution de plan. O-PLan représente le plan en utilisant différents types de contraintes.

Les planificateurs hiérarchiques ont été utilisés pour des applications réelles. NOAH a été utilisé pour la supervision des apprentis ingénieurs en mécanique. NONLIN a été utilisé pour l'entretien des turbines électriques. DIVISER a été utilisé pour l'enchainement des missions du vaisseau spacial VOYAGER. SIPE a été appliqué pour la planification des missions sur les portes-avions.

2.1.3.b Principe de fonctionnement

Le principe de fonctionnement d'un planificateur hiérarchique consiste à produire à chaque niveau, un plan explicite qui résout le problème posé au « niveau d'abstraction » considéré. Le planificateur commence par dresser un plan à haut niveau, ce plan est ensuite enrichi au fur et à mesure que le planificateur progresse dans les différents niveaux, ceci jusqu'à l'obtention d'un plan uniquement composé de tâches de bas niveau.

2.1.3.c Avantage

Les avantages de la planification hiérarchique se décomposent en trois grandes lignes :

- l'exploitation des connaissances expertes spécifiques au domaine. Ces connaissances sont représentées dans les différentes hiérarchies.
- la réduction de l'espace de recherche au travers du raisonnement sur les différentes couches d'abstractions,
- la résolution des interactions entre sous-tâches représentée par les contraintes d'ordres partielles ou totales.

2.1.3.d Planification HTN

C'est au milieu des années 90 que Erol et al puis Nau et al développent une base formelle, des algorithmes complèts et une analyse de complexité de la planification hiérarchique. Dés lors, le terme planification hiérarchique fait référence à la planification HTN (*Hierarchical Task Networks*) qui est une approche de hiérarchisation des opérateurs dans laquelle le domaine de planification spécifie la façon de réaliser les buts. Les planificateurs HTN les plus connus sont UCMP (Erol et al., 1994), SHOP (Nau et al., 1999) et SHOP2 (Nau et al., 2003)

Éléments de base : les éléments de base proposés pour la formalisation des réseaux de tâches hiérarchiques (*Hierarchical Task Network* HTN) sont les *états du système*, les *tâches primitives* ou *composées* et les *méthodes de décomposition*.

ÉTATS DU SYSTÈME : un état est une liste de symboles propositionnels encore appelés atomes. Les atomes qui apparaissent dans cette liste sont tous satisfaits. Ceux qui n'apparaissaient pas sont tous non-satisfaits.

TÂCHE : une tâche est une liste de la forme $t(h_1, h_2, ..., h_n)$ où t représente le nom de la tâche et les h_i des arguments de la tâche. Il existe deux types de tâches dans un HTN. Les tâches *composées* et les tâches *primitives*. Les tâches

composées se décomposent en sous-tâches de façon récursive jusqu'à atteindre des tâches primitives directement exécutables par le système.

TÂCHES PRIMITIVES : elles sont de la forme $t(h_1, h_2, ..., h_n)$. Elles correspondent à des actions dans la planification de type STRIPS. Les préconditions et les effets des tâches primitives sont déclarés en utilisant des opérateurs. Ainsi, le nom d'une tâche *primitive* est celui d'un **opérateur** qui décrit son exécution.

OPÉRATEURS : les opérateurs sont de la forme $(Operatoro, pre, eff)$ où o est la tâche primitive, *pre* l'ensemble des atomes qui décrivent les conditions d'activation de o et *eff* l'ensemble des atomes qui décrivent les effets de o.

TÂCHES COMPOSÉES : elles sont également de la forme $t(h_1, h_2, ..., h_n)$. Elles correspondent à un réseau de tâches. Une tâche composée décrit un plan d'action en y ajoutant les conditions d'exécution des actions qu'elle comporte. Elle vise à informer le planificateur sur les sous solutions possibles. Les tâches *composées* sont associées à des **méthodes** qui permettent de décomposer des tâches en sous tâches. Les sous tâches peuvent être partiellement où totalement ordonnées.

MÉTHODES : les méthodes sont de la forme $(Methodem, pre, T)$ où m définit le nom de la tâche composée, *pre* l'ensemble des atomes qui décrivent les conditions d'activation de la méthode m et T une liste de sous tâches.

2.1.3.e Représentation d'un HTN

Le graphe de recherche du planificateur est un arbre **ET-OU**. Chaque sous-arbre représente un sous-problème. Les nœuds ET de cet arbre sont des sous-problèmes qui ont pour fils d'autres sous-problèmes plus élémentaires. Un nœud ET est résolu à condition que tous ses fils soient résolus. Un nœud OU de l'arbre est aussi un sous-problème. Chacun de ses fils représente une manière de résoudre le sous problème. Un nœud OU est résolu à condition qu'un au moins de ses fils soit résolu. Si plusieurs nœuds OU sont applicables, des heuristiques peuvent être utilisées pour choisir le nœud le plus adapté à la résolution du sous problème. Dans un HTN, les nœuds OU sont encodés dans les méthodes et un nœud ET est une réduction d'un nœud OU.

2.1.3.f Approche de planification dans les HTN

Les systèmes de planification HTN, sont des systèmes de planifications non linéaires combinés à une hiérarchisation des actions. Le planificateur prend en entrée l'état initial et un réseau de tâches. Le fonctionnement d'un planificateur HTN, est basé sur une expansion des actions de haut niveau en actions de plus bas niveaux. Pour chaque action abstraite il existe des décompositions dont l'application aboutit à des actions partiellement ordonnées qui doit être introduites dans le plan. Ainsi, une action abstraite du plan courant peut être décomposée de différentes manières. Ces décompositions sont généralement stockées dans une librairie de plans. Le choix d'une décomposition dépend d'un ensemble de conditions. Les planificateurs HTN, ignorent les préconditions, les effets et les liens causaux entre les actions d'une décomposition tant que celle-ci n'est pas choisie. Le fonctionnement d'un planificateur HTN est de ce fait une forme de réduction de problème auquel l'on ajoute du « backtracking ». Le principe de l'algorithme de fonctionnement peut être décrit comme suit :

1. *Choisir l'une des tâches possibles au niveau d'abstraction en cours.*
2. *S'il s'agit d'une tâche primitive, appliquer l'opérateur et passer en 4*
3. *S'il s'agit d'une tâche non-primitive appliquer les méthodes de décomposition en sous tâches et passer en 4*

4. *Appliquer les contraintes*
5. *Retour en arrière (*backtrack*) et essai d'autres décompositions si aucune méthode n'est applicable, ou s'il n'existe pas moyen de satisfaire une contrainte.*

Le plan solution est une séquence de tâches primitives.

Dans la suite, nous nous focaliserons sur les architectures robotiques permettant de concevoir les systèmes complexes autonomes.

2.1.4 Architecture robotique pour la planification dans les systèmes complexes autonomes

En IA, la plupart des architectures de systèmes complexes autonomes sont celles issues de la robotique. Dans la majorité des cas, la caractéristique de base des architectures proposées est leurs capacités à prendre en compte la dynamique de l'environnement en exploitant le couplage entre des modules délibératif et d'exécution. Ainsi, les architectures logicielles proposées en robotique autonome intègrent (i) un niveau décisionnel pour la planification des actions de l'agent et (ii) un niveau dit fonctionnel pour l'exécution des actions planifiées. Contrairement au premier niveau entièrement délibératif, le second est beaucoup plus réactif. Nous présentons dans la suite quelques architectures intéressantes. Cela nous permettra de positionner celle que nous proposons dans cette thèse.

2.1.4.a De l'architecture *Sense/Plan/Act*-SPA à nos jours

La première architecture proposée pour la planification et l'exécution dans les systèmes autonomes est basés sur la boucle *Perception/Planification/Action* (*Sense/-Plan/Act SPA* (Nilsson, 1982)). Comme l'indique la Figure 2.4, le module de perception (*Sense*) permet de capturer et de traduire les informations issues d'environnements en modèle d'environnement disponible pour l'agent. Sur la base de ce modèle de l'environnement, le module de planification (*Plan*) construit un plan permettant d'atteindre les objectifs que se fixe l'agent. Le plan résultant est exécuté par le module d'action (*Act*).

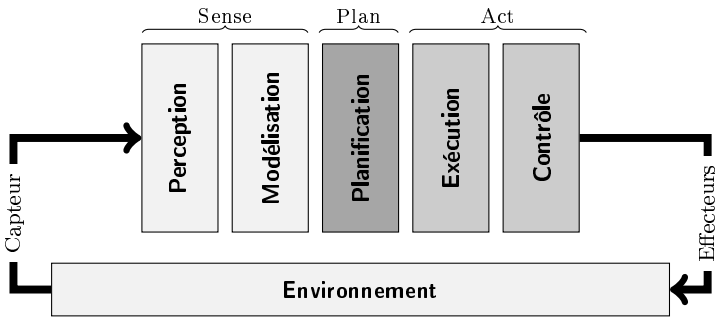


FIGURE 2.4 – *Architecture* Sense/Plan/Act

Le robot Shakey (Fikes et al., 1972) est le premier à avoir intégré la boucle perception, planification et action. Dans le cas de Shakey, le module de planification embarque un planificateur STRIPS (Fikes et Nilsson, 1971). Le module d'exécution utilise une table de triangulation basique pour la prise en compte des échecs c'est-à-dire, en cas d'échec d'une action, celle-ci est relancée si ses préconditions restent valides. Dans le cas contraire, un nouveau plan est généré.

Limites La première limite de SPA réside dans le fait que toute l'intelligence du système repose sur le module de planification. La seconde limite de SPA est qu'elle se base sur une architecture monolithique dans laquelle toute la boucle Sense/Plan/Act est réalisée à chaque étape. Cela peut entrainer des problèmes de réactivité du système car il est très difficile voir impossible de planifier à l'échelle du changement de l'environnement. À ce jour, l'approche SPA est pratiquement abandonnée au détriment de deux classes d'architectures dites *ascendantes* (architectures réactives) et *descendantes* (architecture délibératives).

Pour certains chercheurs, dont Brooks (Brooks, 1986, 1991) qui défendent les architectures réactives, il n'est pas nécessaire d'avoir un agent doté de module de décision complexe pour parvenir à des comportements intelligents. Selon lui, des règles comportementales de type stimulus-réponse dans les capteurs et effecteurs suffisent pour résoudre des problèmes complexes. Dans ce cas, nul besoin d'intégrer un module de planification. Cette caractéristique favorise une meilleure adaptation à la dynamique de l'environnement. Pour étayer ses propos, Brooks (1986) propose une architecture en plusieurs couches de contrôle nommée architecture de *subsumption*. Elle permet de se passer de la décomposition horizontale du cycle de décision dans SPA pour adopter une décomposition verticale ascendante en niveaux de compétence. Cependant, la simplicité de ce type d'architecture ne permet pas de prendre en compte les informations portant sur la globalité de l'environnement. En conséquence, les actions produites peuvent être sujettes à des incohérences au niveau global.

Contrairement aux architectures réactives, celles dites délibératives sont dotées de modules de décision complexe capables de prendre en compte aussi bien les objectifs que les informations explicites sur la globalité de l'environnement. Ces modules permettent de raisonner sur les actions au sens où ils ont des objectifs définis et sont capables de produire des plans explicites pour atteindre les objectifs. Les problèmes de réactivité liés à la planification demeurent valides dans cas. Toutefois, depuis le début des années 90, différentes architectures délibératives intéressante ont été proposées.

Dans la suite, nous décrivons plus particulièrement quelques architectures délibératives. En effet, les décisions de conduite des systèmes de culture au sein de l'exploitation agricole sont relativement complexes. Elles nécessitent la prise en compte d'objectifs et de connaissances explicites sur le domaine de l'agriculture et l'ensemble de l'environnement de l'agent.

2.1.4.b Architectures trois-tiers

La plupart des architectures existantes pour la planification et l'exécution dans les systèmes autonomes repose sur trois couches que l'on retrouve au départ dans les architectures 3T de Bonasso et al. (1997) et ATLANTIS de Gat (1992, 1998). Cette architecture a été reprise dans de très nombreux travaux (Muscettola et al., 1998; Alami et al., 1998; Park et al., 1999; Simmons et al., 2007) portant sur le couplage entre des modules de planification et des exécutifs réactifs pour la prise en compte de la dynamique de l'environnement.

Les trois couches ont été intégrées dans l'optique de répartir le traitement du problème suivant des niveaux de granularité différents. Ainsi, comme l'indique l'architecture ATLANTIS présentée sur la Figure 2.5-a, les trois couches sont constituées de trois composants qui sont :

- *délibération* : composant décisionnel chargé de produire le plan d'action qui permet d'accomplir les objectifs de l'agent,

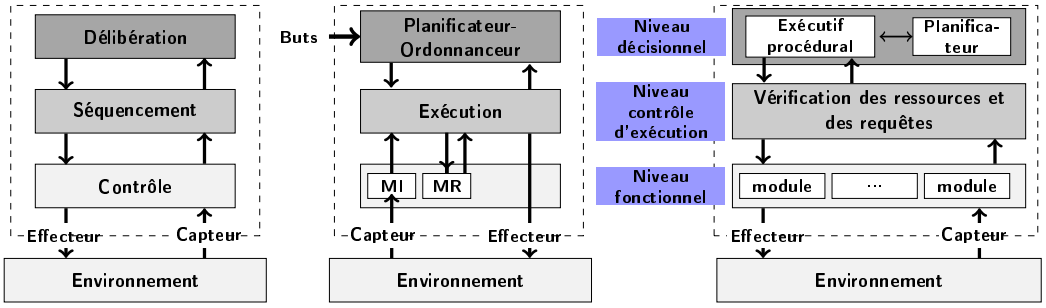


FIGURE 2.5 – *Architecture* trois tiers

- *séquencement* : composant réactif chargé de l'exécution du plan produit au niveau de délibération.
- *contrôle* : composant réactif capable d'exécuter les actions en faisant le lien avec l'environnement.

Il existe plusieurs variantes de l'architecture en trois couches. Nous décrivons deux d'entre elles. La première variante de l'architecture en trois couches que nous détaillons est celle adoptée en 1999 pour la sonde *Deep Space one*. Nommée *Remote Agent architecture* (cf. Figure 2.5-b), la couche délibérative embarque le système de planifications temporelles et d'ordonnancements sous contraintes de ressources nommé HSTS (*Heuristic Scheduling Testbed System* (Muscettola, 1993)). La couche de séquencement est quant à elle basée sur un exécutif de type EST (*Execution Support Language* (Gat, 1997)) capable d'envoyer de requêtes et d'exécuter des plans comportant des actions concurrentes et interdépendantes dont l'issue de la mise en œuvre peut être soumis à de l'incertitude. La couche de contrôle de l'architecture *Remote Agent* intègre un mécanisme de diagnostique (Mode identification MI & Mode Reconfiguration MR) basé sur le système *Livingstone* (Williams et al., 1996) qui est un noyau réactif orienté modèles pour l'auto-reconfiguration des systèmes autonomes.

A l'instar de la précédente, l'architecture LAAS (Alami et al., 1998) (cf. Figure 2.5-c) est une variante de l'architecture en trois couches. On y retrouve un niveau fonctionnel, un niveau de contrôle d'exécution et un niveau décisionnel. Le niveau décisionnel couple l'exécutif procédural OpenPRS (Ingrand et al., 1996) au planificateur temporel IxTeT (*Indexed Time Table* Ghallab et Laruelle (1994)). Le rôle de l'exécutif procédural est de lancer et superviser les actions puis de réagir aux événements provenant du niveau inférieur ou d'un opérateur.

Le planificateur est quant à lui chargé de construire et d'ordonnancer le plan global de l'agent, tout en prenant en compte les objetifs ainsi que les échecs d'exécution. Pour ce faire, IxTeT planifie dans l'espace des plans partiellement ordonnés et partiellement instanciés (Partial Order Causal Link POCL). Dans l'une de ses implémentations les plus récentes (IxTeT-eXEC Lemai (2004)) ce planificateur intègre également un exécutif temporel. Le niveau de contrôle d'exécution basé sur R2C (*Requests and Resources Checker* Ingrand et Py (2002)) est dédié d'une part à la vérification (i) de l'utilisation des ressources et (ii) des requêtes envoyées par l'exécutif procédural à la couche inférieure. D'autre part, il transmet à l'exécutif procédural les bilans d'exécutions envoyés depuis le niveau fonctionnel. Enfin, Le niveau fonctionnel contient des modules contrôlables, nécessaires à l'action et à la perception de l'agent. Cette couche peut intégrer de modules très évolués tel qu'un planificateur de déplacement etc.

Limites Selon Estlin et al. (2001), les architectures en trois couches posent trois problèmes. Premièrement, chacune des couches dispose de son propre modèle de l'environnement. Ces différents modèles du monde comportent généralement des informations redondantes qui doivent être cependant maintenues constamment. Deuxièmement, les couches sont contraintes de fonctionner à un certain niveau de granularité. Cette configuration ne permet pas aux techniques de planification et d'exécution d'être utilisées sur les problèmes pour lesquels elles sont plus appropriées. Troisièmement, l'architecture en trois couches ne permet pas d'appréhender les travaux (ex : système CLEaR Estlin et al. (2001)) dans lesquels la frontière entre la planification et l'exécution est considérablement réduite dans l'optique d'accroître le temps de réponse des systèmes complexes autonomes. Dans la section suivante nous développons une architecture alternative qui permet de palier à ces différentes limites.

2.1.4.c Architecture CLARAty

Afin de palier aux limites de l'architecture en trois couches, l'architecture en deux couches CLARAty (*Coupled Layer Architecture for Robotic Autonomy* (Volpe et al., 2000, 2001; Estlin et al., 2001)) a été proposée par le département Jet Propulsion Laboratory de la NASA. Comme l'indique la Figure 2.6, cette architecture est composée d'un niveau fonctionnel en interaction avec un niveau décisionnel.

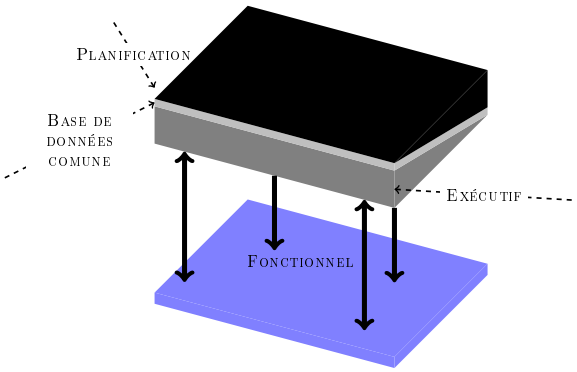


Figure 2.6 – *Architecture* CLARAty

Le niveau fonctionnel intègre les fonctionnalités de base permettant d'interagir avec le niveau décisionnel et de faire le lien avec l'environnement. Il est constitué de composants génériques ayant des comportements prédéfinis. Ces composants sont organisés de manière hiérarchique allant du bas niveau (ex : contrôle du moteur d'un robot) à des niveaux d'abstraction plus élevés (ex : navigation de robot avec évitement d'obstacle). Chaque composant est capable d'estimer sa consommation de ressources et d'informer le niveau décisionnel.

Le niveau décisionnel contient un planificateur ordonnanceur déclaratif en interaction avec un exécutif procédural. Ces deux composants sont liés par une sous couches comportant une base de données commune de plans représentés dans un langage combinant aussi bien les structures procédurales et déclaratives. Durant l'exécution du plan produit par le planificateur, les capacités du niveau fonctionnel sont appelées puis le résultat de l'exécution des actions est pris en compte dans l'optique d'adapter le plan à la dynamique de l'environnement.

Dans la première implémentation de l'architecture CLARAty, les auteurs utilisent le système CLEaR (*Closed-Loop Execution and Recovery* (Fisher et al., 2000; Estlin et al., 2001)) comme instance du niveau décisionnel. CLEaR combine un planificateur continu nommé CASPER (*Continuous Activity Scheduling,*

Planning, Execution and Re-planning (Chien et al., 2000)) à un système d'exécution basé sur TDL (*Task Description Language* (Simmons et Apfelbaum, 1998)). Notons cependant que dans cette implémentation, CASPER et TDL avaient des représentations différentes.

2.1.4.d Bilan sur les architectures de systèmes complexes autonomes et positionnement sur les approches de planification

Considérant les différents travaux présentés dans cette section, nous pouvons affirmer qu'il existe au moins deux niveaux dans les architectures de systèmes complexes autonomes. D'une part, on retrouve un niveau *décisionnel* chargé de la construction et de l'exécution du plan en fonction des objectifs du système. Pour son fonctionnement, ce niveau a besoin d'informations relatives à l'état global de l'environnement. D'autre part, on retrouve un niveau *fonctionnel* chargé de la mise en œuvre effective des actions et du lien avec l'environnement (système physique). Pour son fonctionnement, chaque composant de ce niveau a besoin d'informations partiels sur l'état de l'environnement.

Dans le cas des applications qui nous intéressent c'est-à-dire, celles relatives à la conduite des systèmes de culture au sein de l'exploitation agricole, il ne s'agit pas au niveaux fonctionnel, de piloter un robot. Par contre, l'idée ici est de contrôler un ensemble de processus opérationnels dont le but est de perturber la dynamique de processus biophysiques liés au sol (ex : niveau hydrique) et aux plantes (ex : l'indice foliaire, la biomasse, le stade de floraison). A l'instar de travaux en robotique autonome, le niveau décisionnel contient deux composants *planificateur/ordonnanceur* et *exécutif*.

Cette thèse s'intéresse particulièrement au niveau décisionnel c'est-à-dire :

- au fonctionnement délibératif relatif à la planification (décisions stratégique, tactique) et ordonnancement (décision opérationnelle) des actions d'un agriculteur,
- au fonctionnement réactif relatif à l'exécution du plan de l'agriculteur,
- aux interactions existantes entre les fonctions délibératives et réactives de l'agriculteur.

2.1.5 Exécution de plan en planification

L'idée maitresse des travaux portant sur la décision dans les systèmes complexes autonome se base sur l'hypothèse que l'autonomie d'un système dépend de sa capacité à manipuler et adapter des plans dans un contexte différent de celui pour lequel le plan a été initialement conçu.

Dans la littérature, trois grandes stratégies ont été développées pour appréhender le couplage planification et exécution : *planification réactive*, *planification successive* et *planification continue*.

La première approche (*planification réactive*) est centrée autour du composant d'exécution. La planification est dans ce cas vue comme une ressource pour l'exécution. La seconde approche (*planification par lot*) intègre les deux composants (planification et exécution) sous la forme de processus distincts. Le plan produite par le planificateur porte sur tout l'horizon de la planification c'est-à-dire de l'état initial à l'état but. Ce plan complet est ensuite passé au système d'exécution. En cas d'échec le planificateur génère un nouveau plan partant de l'état courant. Notons toutefois, qu'il existe des systèmes de planification successive tel que *Remote Agent* (Muscettola et al., 1998) dans lesquels le temps est subdivisé en horizon de planification (cf. Figure 2.7-a). Le plan est donc construit sur l'horizon courant et

exécuté. Ensuite la planification est réalisée successivement sur les horizons suivantes. Enfin, la troisième approche (*planification continue*) consiste à entrelacer continuellement les phases de planification et d'exécution. Dans ce cas, la planification reste active afin d'adapter le plan en cas de conflit ou lorsque de nouveaux buts sont ajoutés. Les actions exécutables sont lancées même si le plan n'est pas complet. Comme l'indique la Figure 2.7-b, la construction du plan est réalisée de manière incrémentale. L'horizon sur lequel on travaille est décalé au fur et à mesure de l'exécution de façon à ce que le nouvel horizon recouvre partiellement l'horizon précédent.

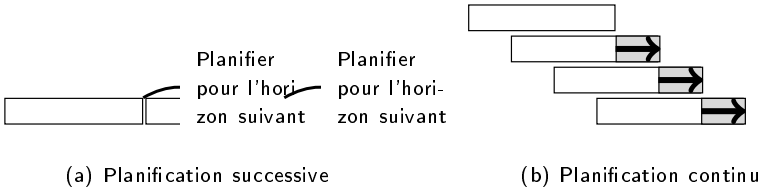


FIGURE 2.7 – *Planification successive vs planification continue*

Dans le cas des applications qui nous intéressent, l'horizon d'une simulation peut couvrir plusieurs années. Considérant, l'incertitude intrinsèque des systèmes de production agricole, il nous parait simplement aberrant d'aborder les différents problèmes de décision sous la forme de planification portant sur tout l'horizon de la simulation. A ceci s'ajoute le fait que les planifications stratégique, tactiques et opérationnelles portent sur des horizons très différents allant de plusieurs années à quelques jours. Dans ces conditions, la stratégie adoptée pour les phases de planification et d'exécution doit pouvoir permettre de prendre en compte cette variabilité de l'horizon de planification. Pour ces différentes raisons, les approches de planification successives ne nous semblent pas pertinentes. Nous proposons dans la suite de développer quelques systèmes de planification réactive et de planification continue.

2.1.5.a Stratégie de planification réactive

Les approches de planification réactives exploitent généralement un langage intermédiaire permettant de spécifier la façon d'exécuter un plan. Dans ces systèmes les plans sont pré-spécifiés. Tant que le plan reste valide, l'exécutif s'occupe de son interprétation. En cas d'échec, il sollicite le planificateur pour l'obtention d'un nouveau plan valide dans le contexte d'exécution courant. La planification est dans ce cas utilisée comme une ressource pour l'exécution. Dans cette classe d'exécutifs on retrouve d'une part, celles qui sont purement réactives (ESL Gat (1997), TDL Simmons et Apfelbaum (1998)). Généralement elle ne fonctionnent pas lorsque les systèmes échouent et qu'aucun plan et procédure explicites n'est pré-spécifié. D'autre part on retrouve celles qui intègrent des capacités délibératives dans l'exécutive (PROPEL Levinson (1995), Propice Despouys et Ingrand (1999).

Les exécutifs *Execution Support Language* ESL (Gat, 1997) et *Task Description Language* TDL (Simmons et Apfelbaum, 1998), respectivement utilisés pour la sonde *Deep Space one* avec l'architecture *Remote agent* puis dans l'implémentation CLEaR de l'architecture CLARAty peuvent être considérés comme des systèmes d'exécution purement réactifs.

ESL : ESL se base sur un exécutif procédural réactif nommé *Reactive Action Package* (RAP (Firby, 1994)) et propose dans le langage intermédiaire, des concepts permettant d'encoder les connaissances d'exécution permettant de prendre en compte :

- des événements de synchronisation sur les actions,
- des sélections et des décompositions de méthodes dans un sous ensemble de méthodes prédéfinis,
- des contingences relatives à la détection des échecs et l'appel des procédures de correction,
- des buts sur l'état du système à atteindre et les méthodes pour y arriver,
- des requêtes sur un ensemble de propositions logiques.

Dans, ESL, la prise en compte de la contingence est fondée sur la philosophie « cognizant-failure » c'est-à-dire l'idée selon laquelle, en cas de détection d'échec inévitable, le système est capable d'y répondre de manière appropriée. Cela suppose que les situations de succès et d'échecs des actions puissent être facilement identifiées.

TDL : TDL dispose de caractéristiques très proches de ESL. Les actions sont exécutées suivant un arbre généré en fonction de l'état du système. Chaque nœud de l'arbre est capable de détecter les situations d'échec. Il existe plusieurs autres systèmes d'exécution réactifs que nous ne présentons pas ici.

PROPEL : Contrairement à ces deux exécutifs purement réactifs, PROPEL (*PROcedure Planning and Execution Language* Levinson (1995)), intègre une capacité délibérative dans l'exécutif. Dans l'architecture de PROPEL on retrouve un exécutif et un planificateur, le tout associé à un moteur de recherche qui opère dans un espace de procédures prédéfinies. Ce moteur génère des procédures disjonctives représentées sous la forme d'un arbre OU. L'exécutif exploite cet arbre et utilise une technique de recherche prédictive afin d'évaluer les choix et génère le plan adéquat en fonction du contexte d'exécution. Dans PROPEL, l'exécutif n'autorise aucun « backtrack » dans l'arbre de recherche. De la même manière que l'exécutif, le planificateur utilise une technique de recherche prédictive pour l'anticipation et la prise en compte de situations d'échec. L'anticipation dans PROPEL consiste à simuler l'effet des actions et à produire un plan correspondant à un ensemble de règles de sélection associées à des procédures abstraites. Ces règles seront ensuite exploitées afin de guider l'exécution. En cas de d'échec imprévu, le planificateur opère des "backtrack" dans l'arbre de recherche initialement généré par l'exécutif afin de trouver de nouvelles règles exploitable par l'exécution. On retrouve des fonctionnalités similaires également dans Propice Despouys et Ingrand (1999) qui se base sur un exécutif procédural nommé OpenPRS (Ingrand et al., 1996).

2.1.5.b Planification continue

L'approche de planification continue consiste à entrelacer continuellement les phases de planification et d'exécution. La planification reste active afin d'adapter le plan en cas de conflit ou lorsqu'un nouveau but est ajouté. Dans cette approche, certaines actions sont exécutées même si la planification est en cours.

SIPE : afin d'entrelacer les phases de planification et d'exécution dans SIPE et SIPE-2 (*System for Interactive Planning and Execution monitoring* Wilkins (1988)), l'utilisateur définit les buts qui peuvent être reportés. En utilisant une technique de planification non-linéaire hiérarchique, un premier plan est alors construit sur la base des buts qui ne peuvent être reportés. L'exécution de ce plan est réalisé en parallèle avec la planification portant sur les buts reportés. En cas d'échec de situation inattendu, soit le plan construit en parallèle est mis en œuvre, soit le système de réparation de plan de SIPE est appelé afin de modifier le plan.

CLEaR : Basé sur l'architecture CLARAty le système CLEaR (*Closed-Loop Execution and Recovery* (Fisher et al., 2000; Estlin et al., 2001)) combine le planificateur continu nommé CASPER (*Continuous Activity Scheduling, Planning, Execution and Re-planning* (Chien et al., 2000)) à un système d'exécution basé sur TDL (*Task Description Language* (Simmons et Apfelbaum, 1998)).

CASPER (*Continuous Activity Scheduling, Planning, Execution and Replanning* Chien et al. (2000); Chien (2006)) se base sur le planificateur ASPEN (Rabideau et al., 1999) qui utilise une technique de planification par recherche locale afin de réduire le temps de replanification. En conséquence la réactivité du système se trouvera ainsi augmenté. Pour ce faire, un premier plan est construit sur un horizon donné. Ce plan sera ensuite étendu de manière incrémentale au fur et à mesure de l'exécution. Les auteurs préconisent de combiner cette planification continue avec une approche hiérarchique de planification. Cela consiste à opérer une planification à long terme portant sur un niveau d'abstraction élevé. Ensuite, des planifications sur des horizons de plus en plus courts sont réalisées afin d'obtenir des plans de plus en plus détaillés.

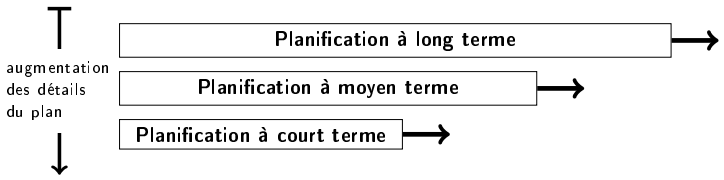


FIGURE 2.8 – *Horizon de planification hiérarchique Chien et al. (2000)*

Cette approche est schématisé par le Figure 2.8. L'idée de base étant de considérer qu'il est inutile de construire un plan détaillé sur du long terme. Pour chacune des couches, le planificateur opère de manière continue. La taille de l'horizon et la fréquence de la planification diffèrent en fonction du niveau de planification. En cas de situation inattendue, le plan est réparé en exploitant les capacités d'ASPEN. Pour assurer que le planificateur ne modifiera pas des actions encours d'exécution, une fenêtre d'engagement est définie. Ainsi, au sein de cette fenêtre, toutes les modifications du planificateur sont interdites. En fonction des applications, l'exécution peut être suspendue ou non durant la replanification.

Pour intégrer la planification et l'exécution dans CLEaR, les auteurs définissent trois classes de processus qui sont :

- ▷ *Processus de planification :* maintient le plan courant et répond au consigne de replanification des processus d'exécution.
- ▷ *Processus d'exécution :* maintient une copie du plan et se charge de l'engagement des actions
- ▷ *Processus d'estimation d'état :* maintient l'estimation de l'évolution de l'état du système et des ressources.

2.1.6 Quelques approches orientées modèles

Dans cette section nous présentons le langage RMPL (*Reactive Model-based Programming Language*) utilisé dans le système KIRK et l'architecture IDEA (*Intelligent Distributed Execution Architecture*) qui sont deux approches orientées modèles particulièrement intéressantes dont nous nous inspirons dans cette thèse. La première permet de concevoir des exécutifs réactifs capables de prendre en compte des contraintes temporelles quantitatives. La seconde quand à elle se base sur une organisation multi-agent orientée modèle dans laquelle un grand nombre de systèmes

y compris des systèmes de planification complexes peuvent être structurés sous la forme d'une collection d'agents en interaction.

2.1.6.a RMPL : un langage orienté modèle pour des exécutifs réactifs et temporels

L'ensemble des exécutifs procéduraux décrits dans la section 2.1.5.a ne permettent pas de prendre en compte des contraintes temporelles quantitatives durant l'exécution. Pour palier à cela, Kim et al. (2001) proposent un langage original nommé RMPL (*Reactive Model-based Programming Language*). L'approche proposée vise à écrire et exécuter des programmes de contrôle de robot basés sur une programmation orientée modèle. L'idée ici est d'écrire des programmes de contrôle qui encodent de manière compacte les capacités des robots, les objectifs d'une mission et l'environnement.

Pour ce faire, RMPL propose une représentation unifiée du système en utilisant un seul et même formalisme pour l'exécution, le diagnostic et la planification. Le langage est défini dans l'optique de prendre en compte :

(i) les raisonnements relatifs aux contingences et à l'ordonnancement des actions (Kim et al., 2001)

(ii) l'inférence de variables d'état cachée et le contrôle des variables d'états du système (Williams et al., 2001a).

Dans le système de planification KIRK par exemple, le planificateur prend en entrée des programmes exprimés en RMPL. Contrairement à CLEaR qui utilise un exécutif basé sur le langage ESL, KIRK ne dispose pas d'exécutif programmé dans un langage intermédiaire.

Concepts de base : Considérons c, a, A et B où c est un littéral définissant une condition, a une action commençant à la date courante et A, B des procédures RMPL. Les instructions RMPL suivantes définissent les concepts de base du langage RMLP :

- c : tester si la condition est *vrai* à la date courante,
- a : démarrer l'action à la date courante,
- A, B : Exécution *parallèle* des procédures RMPM A et B.
- $A; B$: Exécution *séquentielle* des procédures RMPM A et B.
- $A[l, u]$: Contraindre la durée d'exécution de la procédure A de manière à ce qu'elle soit comprise entre l et u.
- **if** c **thennext** A : lancer l'exécution de la procédure A si la condition $c = vrai$,
- **do** A **maintaining** c : exécuter la procédure A, et assurer que la condition $c = vrai$ tout au long de son exécution,
- **choose** $\{A, B\}$: exécuter l'une de procédure A ou B.

Ces éléments de base de RMPL permettent de spécifier des branchements conditionnels et des synchronisations, des exécutions concurrentes ou consécutives.

Couplage entre RMLP et TPN : Les programmes RMPL sont compilés par le système de planification KIRK sous la forme d'une représentation graphique du plan appelé TPN (*Temporal Plan Network* Williams et al. (2001b)). Un TPN correspond à un réseau de contraintes temporelles, exprimé en utilisant une généralisation du formalisme des STN. Cette généralisation autorise la prise en compte de contraintes symboliques et des branches conditionnelles. Les auteurs affirment que ces éléments additionnels permettent de traduire l'ensemble des concepts de base en TPL.

Comme dans les STN, les nœuds d'un TPL sont des instants associés à des événements. Les arcs (i, j) du réseau temporel sont étiquetés avec les contraintes des distances entres les instants i et j auxquelles s'ajoutent les deux contraintes symboliques qui sont $Tell(c)$ et $Ask(c)$. La contrainte symbolique $Tell(c)$ permet de tester si la condition $c = vrai$ durant l'intervalle de temps entre i et j. La contrainte symbolique $Ask(c)$ permet de spécifier que la condition $c = vrai$, est requise durant l'intervalle de temps entre i et j. La Figure 2.9 présente une traduction en TPN des différents concepts de base de RMPL.

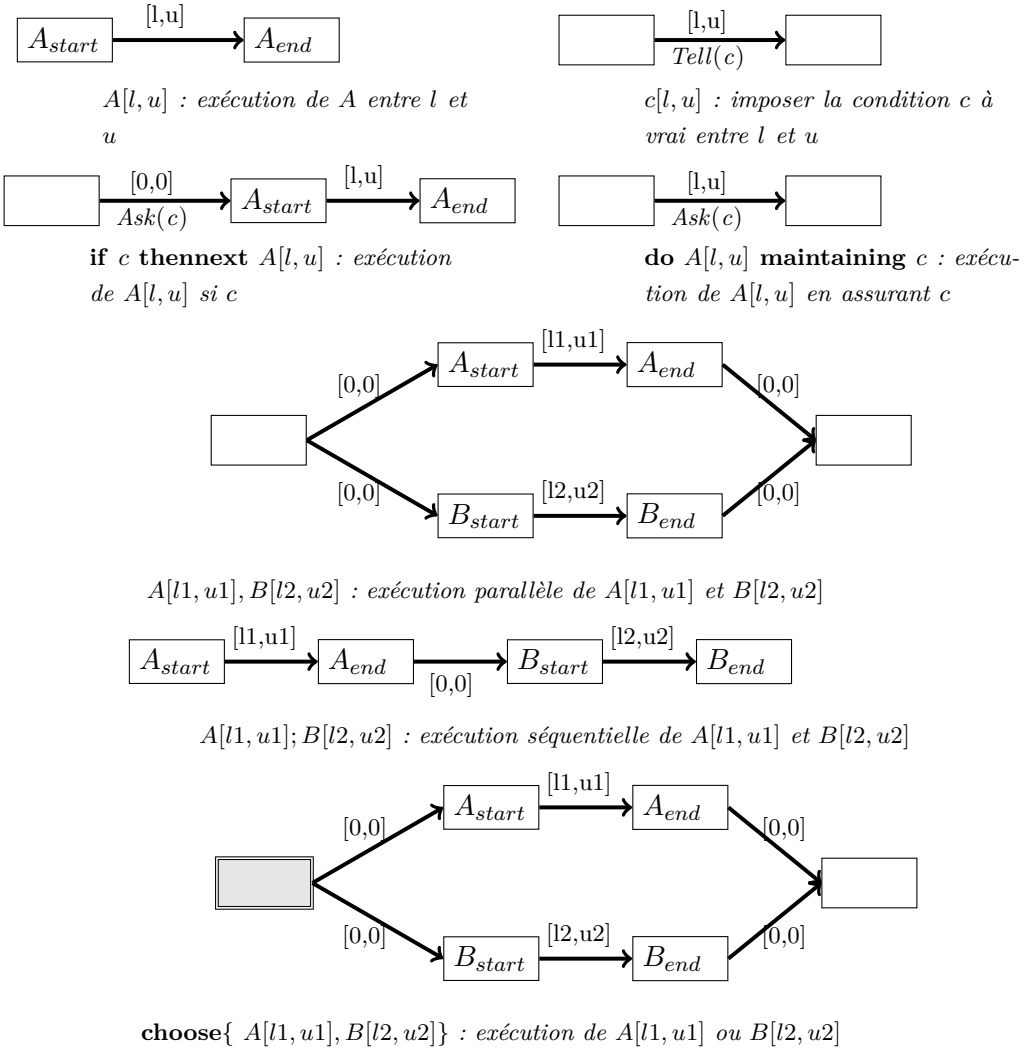


FIGURE 2.9 – *Traduction TPN des concepts de base de RMPL Kim et al. (2001)*

2.1.6.b IDEA : une architecture distribuée pour la planification réactive orientée modèle

L'architecture IDEA (*Intelligent Distributed Execution Architecture* Muscettola et al. (2002); Dias (2003)) se base sur une approche multi-agent orientée modèles dans laquelle un agent définit un module fonctionnel, un système de planification, un système de diagnostic, etc. Chaque agent dispose d'un modèle de contraintes et gère un ensemble de variables dont l'état est fonction du temps. L'idée principale de IDEA est qu'un grand nombre de système de contrôle peut être structuré sous la forme d'une collection d'agents en interaction.

La Figure 2.10, indique la structure d'agent IDEA. L' agent communique avec

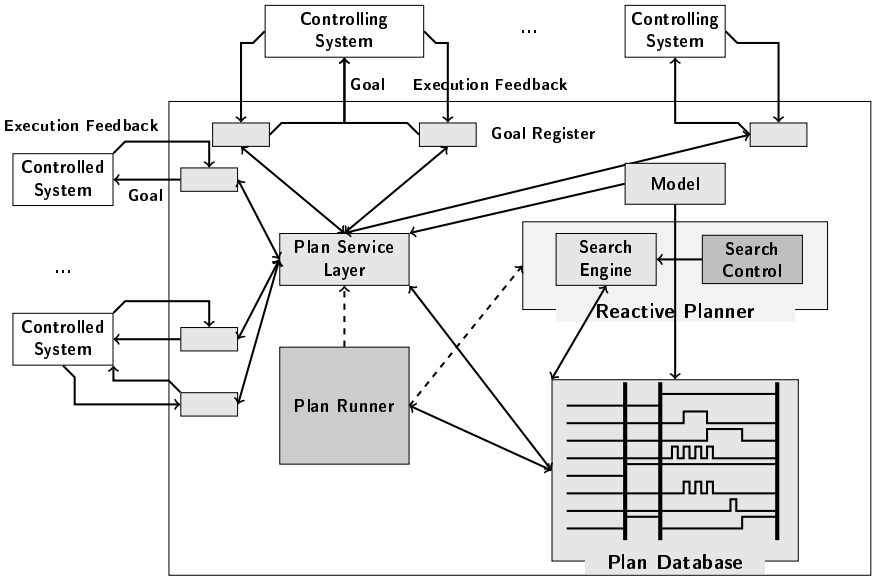


FIGURE 2.10 – *Structure d'un agent* IDEA *selon Dias (2003)*

les systèmes externes (contrôlés [2] ou contrôlant [3]) via un ensemble de *goal registers*. Ces *goal registers* maintiennent le contexte d'exécution de l'agent en envoyant ou en recevant des messages correspondant aux buts envoyés aux systèmes contrôlés ou aux buts reçus des systèmes contrôlant.

La structure d'un agent IDEA est constituée de quatre principaux éléments qui sont : un *plan service layer* PSL , un *plan database* PD, un *plan runner* PR, et un *reactive planner* RP.

plan service layer PSL : il gère les interfaces pour la communication entre les actions, les buts et les échanges de données entre les systèmes externes.

plan database PD : il contient un ensemble ordonné d'instants décrivant la dynamique temporelle des variables d'états du système. Les variables d'état définissent des processus parallèles décrivant :

(i) les changements ou les persistances de la valeur d'un attribut dans le temps
(ii) la séquence des actions exécutées et celles prévues pour une exécution dans le futur.

Les actions et les attributs sont constitués d'une structure particulière appelée *token*. Un token est un intervalle temporel durant lequel une procédure doit être exécuté. Les résultats de l'exécution sont soit envoyés aux systèmes externes, soit exploités afin de guider l'exécution des actions.

plan runner PR : il gère une boucle d'exécution simple de type SPA (sense/-plan/act cf. section 2.1.4.a). Le PR est activé à temps discret et synchronisé avec l'horloge interne de l'agent. Son principe de fonctionnement est :

- se réveiller suite à un message reçu d'un système externe, ou à la date de réveil prévue par l'horloge interne,
- mettre à jour les valeurs des *goal registers* en fonction des messages reçus,
- activer le RP et planifier sur la base du contexte courant et des observations,

2. Le système externe est considéré comme contrôlé si c'est l'agent IDEA qui initialise l'ensemble des valeurs du *goal register*

3. Le système externe est considéré comme contrôlant si l'ensemble des valeurs du *goal register*, l'agent IDEA est initialisé par le système externe

– mettre à jour le contexte d'exécution et envoyer des messages appropriés aux systèmes externes,
– invoquer le RP afin de déterminer la prochaine date d'expiration de l'exécution,
– s'endormir jusqu'à la prochaine date d'expiration de l'exécution.

reactive planner RP : chaque fois qu'une nouvelle information est perçue ou encore, lorsque la fenêtre de réalisation d'une action expire, le *plan runner* PR active RP. Ce dernier est chargé de prendre des décisions sur un horizon court portant sur la prochaine étape. Il détermine les procédures qui répondent aux buts et vérifie la consistance temporelle de tous les tokens suivants. Pour ce faire, RP consulte un *Model* décrivant les règles d'enchaînement des procédures et les mécanismes de synchronisation du démarrage et de la fin des procédures. Il met à jour le *plan database* PD en y représentant aussi bien les intervalles temporels durant lesquels les procédures ont été (exécutions passées) ou doivent être exécutées (exécutions futures).

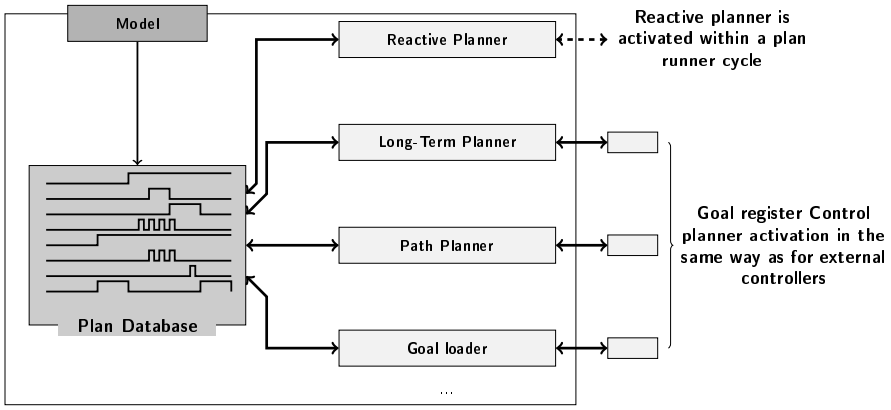


FIGURE 2.11 – *Structure d'un agent* IDEA *selon Finzi et al. (2004)*

Toutefois, des planificateurs spécifiques plus complexes fonctionnant sur des horizons différents peuvent être introduits dans la structure d'un agent IDEA (cf. Figure 2.11). Pour cela, il suffit d'inclure dans le *Model* les mécanismes permettant d'activer et de faire coopérer les différents planificateurs qui modifient le *plan database* PD. A l'instar des systèmes externes, cela se traduit par la spécification, pour chaque planificateur, d'un ensemble de tokens dont l'exécution active le planificateur. Ainsi, l'activation des planificateurs spécifiques peut être planifiée de manière appropriée en fonction de l'évolution des états interne et externe modélisés par l'agent.

2.1.7 Discussion sur la planification et l'exécution de plans temporels

Dans cette section, nous avons introduit quelques éléments de base nécessaires à la conception des systèmes complexes autonomes opérants dans un environnement dynamique et incertain. Pour cela, nous avons présentés différentes approches de représentation des problèmes de planification (cf. section 2.1.1). À ce stade, nous pouvons affirmer que la représentation orientée variables d'états (cf. section 2.1.1.b) est la plus adaptée à notre problème. En effet, en se basant sur une notation fonctionnelle, cette approche permet de représenter des connaissances complexes à l'instar de celles d'un agriculteur.

Nous avons également décrit dans cette section, les concepts de base de deux techniques de planification particulièrement intéressants pour cette thèse : la plan-

ification par satisfaction de contraintes (cf. section 2.1.2) et la planification hiérarchique (cf. section 2.1.3). Nous reviendrons plus en détails (cf. sections 4.1 et 5.2) sur la présentation des travaux majeurs qui ont guidé notre inspiration.

Cette introduction générale nous a toutefois permit de nous focaliser sur des architectures robotiques pertinentes pour la conception de systèmes complexes autonomes. L'architecture en deux niveaux (fonctionnel, décisionnel) CLARAty, utilisé dans le système CLEaR, nous semble particulièrement intéressante. En effet, elle intègre la planification et l'exécution au niveau décisionnel permettant ainsi de réduire la frontière entre ces deux éléments. Cela permet d'accroître les performances, en terme de temps de réponse, liées à l'entrelacement des phases de planification et exécution.

Nous nous inspirons également de la hiérarchisation de l'horizon de planification proposée dans CLEaR. Ainsi, les planifications stratégique, tactique et opérationnelle seront respectivement associées au niveau de planification à long, à moyen et à court terme. Étant le plus abstrait, le plan stratégique de l'agriculteur sera mis à jour moins souvent que les deux autres. Le plan opérationnel sera quand à lui plus détaillé et mis à jour régulièrement en fonction de la dynamique de l'environnement.

Les problèmes de décision stratégique, tactique et opérationnelle étant de nature différentes, il nous semble plus judicieux, d'utiliser pour chacun d'entre eux, des approches de résolution adaptées. Le problème sera donc de déterminer les mécanismes permettant de faire coopérer ces planificateurs spécifiques au sein d'un même système. Pour ce faire, nous exploiterons les approches proposées dans IDEA pour la construction et l'interaction entre différents planificateurs. Chaque planificateur sera perçue comme un système de contrôle indépendant.

Pour l'exécution du plan opérationnel, nous partons du concept d'exécutif réactif basé sur des modèles d'actions et de contraintes temporelles, proposé dans RMPL. Nous étendons l'approche de modélisation et d'interprétation du plan à un cadre événementiel de M&S (cf. section 2.2) de systèmes distribués et hiérarchiques.

2.2 Multi-modelisation & Simulation

Le terme *système dynamique* est un concept abstrait qui décrit le comportement et les interactions d'une entité dans le temps. Un système à évènements discrets est un système dans lequel les changements d'état sont exclusivement dus à des occurrences d'événements contrôlables par le système. Il existe de nombreux formalismes pour la modélisation et la simulation de systèmes à évènements discrets. L'un des plus connus est le formalisme DEVS (*Discrete EVent System Specification* (Zeigler, 1976; Zeigler et al., 2000)). DEVS est un formalisme abstrait pour la modélisation à événements discrets qui permet de décrire des systèmes complexes discrets et continus. Dans le formalisme *DEVS classique*, les systèmes sont constitués d'une collection d'un ou plusieurs composants. Ces composants sont de deux types : les modèles *atomiques* et les modèles *couplés*.

2.2.1 Modèle atomique DEVS

Un modèle atomique DEVS définit un système ayant un état, des entrées, des sorties et dont le comportement dépend de l'état courant, du temps passé dans cet état et des perturbations externes. De manière générale, on représente un modèle atomique DEVS M par un rectangle (cf. Figure 2.12) doté de ports d'entrées et de sorties sur lesquels transitent un ensemble de valeurs.

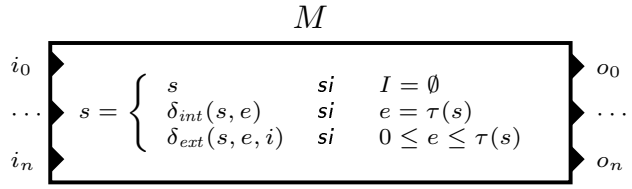


FIGURE 2.12 – *Représentation graphique d'un modèle atomique DEVS*

Définition 2.7 (Modèle atomique DEVS) *Formellement un modèle DEVS atomique M est décrit par un tuple $\langle I, O, S, \tau, \delta_{int}, \delta_{ext}, \lambda \rangle$ où :*

I *est l'ensemble des ports i et des valeurs d'entrées,*
O *est l'ensemble des ports o et des valeurs de sorties,*
S *est l'ensemble des états partiels du système,*
$\tau : S \to \mathbb{R}_0^+$ *est la fonction d'avancement du temps,*
$\delta_{int} : S \times S$ *est la fonction de transition interne,*
$\delta_{ext} : Q \times I \to S$ *est la fonction de transition externe tel que :*
Q *est l'ensemble des états totaux,*
$Q = \{(s,e) | s \in S, 0 \leq e \leq \tau(s)\}$,
e *est le temps écoulé depuis la dernière transition,*
$\lambda : S \to O$ *est la fonction de sortie.*

À tout instant t, le système est décrit par son état $s \in S$. En l'absence d'évènement sur les ports d'entrées I, le système conserve un état passif s durant $\tau(s)$ unité de temps. À l'issu de cette phase ($e = \tau(s)$) le système fait une sortie $\lambda(s)$ et passe ensuite dans l'état $\delta_{int}(s,e)$. Si durant l'état total (s,e), un événement externe apparaît sur l'un des ports d'entrées avant la fin de la phase transitoire ($t \leq e \leq t + \tau(s)$), l'état du système passe à $\delta_{ext}(s,e,i)$.

Ainsi, les transitions d'état doivent être entièrement déterminées par l'état courant, le temps passé dans cet état et les entrées s'il y en a. Les deux fonctions de transition $\delta_{int}(s,e)$ et $\delta_{ext}(s,e,i)$ permettent de séparer respectivement l'*autonomie* du système de sa *réaction* aux perturbations externes. Ainsi, pour un système qui ne reçoit aucun événement externe, sa dynamique est uniquement influencée par la fonction de transition interne. Celle-ci définit les transitions d'état liées aux changements internes du système et à l'avancement du temps. Si le système reçoit des événements externes, sa dynamique du système est également influencée par sa réaction aux événements d'entrée. Il n'existe donc pas d'état terminal. Tant qu'il y aura des perturbations externes, le système doit pouvoir réagir même si cela consiste à ignorer ces évènements. La Figure 2.13 indique un modèle atomique DEVS qui reçoit sur ses ports d'entrées deux signaux continu et discret. Il renvoie en sortie un échantillonnage du signal continu.

FIGURE 2.13 – *Échantillonnage d'un signale sinusoïdale*

2.2.2 Modèle couplé DEVS

Le formalisme DEVS propose également un mécanisme permettant de construire des modèles DEVS couplés en se basant sur d'autres composants DEVS. Chaque modèle atomique DEVS peut être combiné avec un ou plusieurs modèles afin de construire un modèle *couplé*. Cette opération peut être répétée afin d'obtenir une hiérarchie de modèles couplés. Ainsi, un modèle DEVS couplé peut être considéré comme un réseau hiérarchique de composants atomiques et couplés (cf. Figure 2.14). Le réseau encore appelé *structure* du modèle est caractérisé par ses ports d'entrées et de sorties, des modèles qui le constituent et les connections internes entre ces modèles.

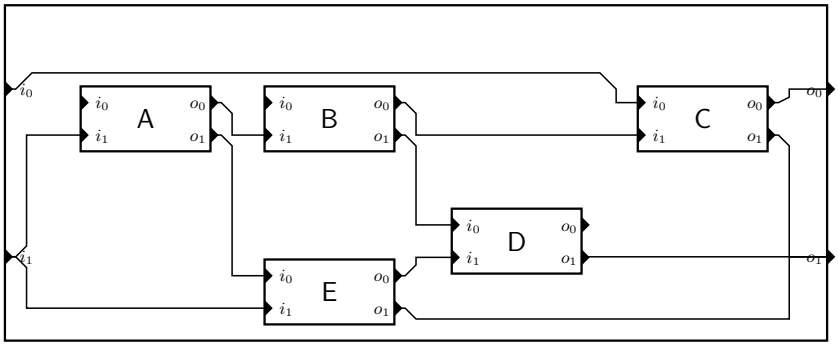


FIGURE 2.14 – *Représentation graphique d'un modèle couplé DEVS*

Définition 2.8 (Modèle couplé DEVS) *Plus formellement, un modèle DEVS couplé M est décrit par un tuple $\langle I, O, N, \{M_n\}, \{\zeta_n\}, \{Z_{n,n'}\}, \mathsf{select}\}\rangle$ où*

I *est l'ensemble des ports et des valeurs d'entrées,*

O *est l'ensemble des ports et des valeurs de sorties,*

N *est l'ensemble des identifiants des modèles constituant le modèle couplé M, y compris l'identifiant « self » de M lui même,*

M_n *est un modèle DEVS (atomique/couplé) constituant au réseau hiérarchique et indexé par $n \in N$,*

ζ_n *est l'ensemble des modèles qui influencent le modèle n,*

$Z_{n,n'}$ *est une famille de fonctions de transfert telle que :*

$Z_{n,n'} : O_n \to I_{n'}$ *si* $n, n' \in N$ *et* $n \in \zeta_{n'}$,

$Z_{\mathsf{self},k} : I \to I_k$ *si* $k \in N$ *et* $\mathsf{self} \in \zeta_k$,

$Z_{k,\mathsf{self}} : O_k \to O$ *si* $k \in N$ *et* $k \in \zeta_{\mathsf{self}}$,

$\mathsf{select} : 2^N \to N$ *est une fonction de sélection.*

Le réseau de modèles DEVS atomiques et couplés ainsi défini est caractérisé par des connexions définissant les *influenceurs* et les *influencés* de chaque modèle n. Soit un modèle nommé $n \in N$. ζ_n désigne les modèles dont les sorties sont connectées aux entrées de n. Si $\zeta_n \neq \emptyset$ alors, il existe une fonction de transfert $Z_{n',n} : O_{n'} \to I_n$ pour chaque influenceur $n' \in \zeta_n$, qui définit les sorties o ($o \subseteq O_{n'}$) de n' connectées aux entrées i ($i \subseteq I_n$) de n. Notons qu'un modèle donné peut être influenceur de plusieurs autres modèles. Le modèle couplé M indexé par $\mathsf{self} \in N$ est également influenceur de certains de sous-modèles le constituant. Ces connexions sont définies par les fonctions de transfert $Z_{\mathsf{self},k} : I \to I_k$. Celles-ci représentent la liste de ports d'entrées du modèle couplé M qui sont connectés aux ports d'entrées des sous modèles $k \in N$. De la même manière, certains sous-modèles peuvent être des influenceurs du modèle couplé M. La fonction de transfert associée $Z_{k,\mathsf{self}} : O_k \to O$ définit la liste des ports de sorties des sous modèles k connectés aux ports de sorties du modèle couplé M. Notons toutefois que dans la version classique de DEVS, les

fonctions de transfert $Z_{k,k} : O_k \to I_k$ ne sont pas définies. Une sortie d'un modèle ne peut donc pas être couplée à l'une de ses entrées. Cette contrainte a été supprimée dans l'extension DEVS Parallèle Chow et Zeigler (1994) que nous présentons dans la section 6.1.1.

Dans un modèle couplé, les sous-modèles fonctionnent de manière indépendantes. Leurs sorties sont communiquées à tous les modèles influencés y compris le modèle couplé M lui même. Si ce dernier reçoit une perturbation externe, le message est transmis à tous les modèles k influencés par self. De la même manière si $k \in \zeta_{\mathsf{self}}$ et que k génère une sortie alors, le modèle M génère également une sortie. Dans ces conditions, plusieurs transitions internes de différents sous modèles à une même date peuvent conduire à des conflits c'est-à-dire à des interférences entre des événements simultanés. Le sous ensemble de modèles en conflits est appelé les sous ensemble *imminent*. Ce sous-ensemble peut être vide si aucun des sous modèles de M n'est supposé faire une transition interne. Dans la version classique de DEVS, si à une date t, plusieurs sous-modèles du sous ensemble *imminent* sont en conflits, seul l'un d'entre eux est sélectionné pour réaliser sa transition interne. Cet arbitrage est réalisé par la fonction de sélection select décrite dans Zeigler et al. (2000).

D'après la propriété de fermeture sous couplage de DEVS (cf. A.2.1), un modèle couplé peut être associé à un modèle DEVS atomique $M_r = \langle I, O, S, \tau, \delta_{int}, \delta_{ext}, \lambda \rangle$ appelé la *résultante*.

2.2.3 Limites de DEVS classique

En pratique, plusieurs modèles pourraient avoir prévu une transition interne à la même date. Or, dans le formalisme DEVS classique le lien de causalité entre modèles peut être brisé en cas d'*événements simultanés*. En effet si plusieurs modèles en influencent un autre, les événements de sortie des influenceurs ne seront pas reçus par l'influencé à la même date. Ceci résulte du fonctionnement de la fonction « select » car cette dernière empêche toute possibilité de simultanéité.

De la même manière, un événement externe peut apparaître sur les ports d'entrées d'un modèle à la même date que celle prévue pour la transition interne. La prise en compte de ce conflit est fortement dépendante du modèle considéré. Le formalisme DEVS parallèle (Chow et Zeigler, 1994) a été proposé afin de palier à ces limites. Ce formalisme et ses extensions sont présentés plus en détail dans la section 6.1.

Dans la suite de cette section, nous présentons une approche systémique de la modélisation d'agent.

2.2.4 Simulation d'agent en DEVS

Un agent peut être perçu comme un système dynamique à événements discrets en interaction avec des systèmes hétérogènes qui décrivent l'environnement de l'agent voir d'autres agents. La reproduction par simulation des interactions entre de tels systèmes et la prise en compte explicite de la dimension temporelle du comportement de l'agent peuvent être facilitées par l'utilisation de formalismes à évènements discrets.

L'approche événementielle est de plus en plus utilisée dans les simulateurs d'agents. Par exemple, le système SPADES (Riley et Riley, 2003) qui est la base du simulateur RobotCup 3D et la plateforme JAMES (Uhrmacher et Kullick, 2000;

Schattenberg et Uhrmacher, 2001) qui est un environnement de modélisation et de simulation d'un agent et de son environnement basé sur les évènements discrets.

Dans JAMES plus particulièrement, les auteurs utilisent la spécification DEVS pour formaliser la modularité et le couplage de modèles hétérogènes, selon une représentation systémique. JAMES est à ce jour la plateforme de simulation d'agent en DEVS la plus connue. Cette plateforme propose un cadre expérimental pour la simulation des systèmes multi-agents. JAMES représente un agent par un modèle atomique DEVS. Le cycle de fonctionnement des agents est fortement inspiré de la boucle *Perception/Planification/Action* (SPA cf. section 2.1.4.a). Les entrées et les sorties modélisent les capteurs et les effecteurs de l'agent. La fonction de transition externe permet de traduire les informations issues d'environnement en modèle d'environnement disponible pour l'agent. La fonction de transition interne modélise l'autonomie de l'agent. Dans JAMES, la fonction de transition interne intègre un algorithme de planification capable d'exploiter les informations contenue dans le modèle de l'environnement. Enfin, les actions de l'agent sur son environnement sont modélisées par la fonction de sortie. Par ailleurs, Quesnel (2006), affine la modélisation d'agent en DEVS en intégrant les capteurs et les effecteurs de l'agent sous la forme de modèles atomiques.

Ainsi, du point de vue de DEVS, l'originalité de JAMES consiste à proposer et à implémenter des classes de modèles dédiées à la problématique des agents. Notre proposition s'inscrit dans cette approche DEVS car ces travaux ont été initiés par le projet de plate-forme RECORD (Chabrier et al., 2007) basé sur l'environnement VLE (Virtual Laboratory Environment Quesnel (2006)) qui utilise DEVS.

2.2.5 Plateforme de simulation VLE

VLE (*Virtual Laboratory Environment* Quesnel et al. (2009)) est une plateforme de modelisation & simulation dont le noyau repose sur le formalisme DEVS. La plateforme VLE développée en C++ (Stroustrup, 1986) repose sur un ensemble de bibliothèques (GLIBMM, GTKMM LIBXMLPP) du projet GNU (Free Software Fundation, 1984). Ces bibliothèques permettent à la plateforme d'assurer la portabilité et l'efficacité aussi bien au niveau rapidité de développement qu'en terme de temps d'exécution.

2.2.5.a Aperçu de l'API de VLE

VLE utilise un découpage modulaire qui fait interagir différents composants applicatifs. Chaque composant applicatif est destiné à un type de traitement spécifique. Pour se faire, les composants de VLE se basent sur un ensemble de programmes et de bibliothèques.

Comme l'indique la Figure 2.15 les différents programmes, bibliothèques partagées et composants sont interdépendants. On y retrouve notamment :

- *libutils* : une bibliothèque qui fournit des outils supplémentaires à la GLIBMM. Au nombre de ces outils on peut citer *libvalue* qui intègre un ensemble de classes encapsulant les types de base tel que les entiers, les réels etc. et les structures de données plus complexes tels que les listes, tableaux, tables associatives ;
- *libgraph* : un ensemble de classes permettant de manipuler une hiérarchie de modèles DEVS. Les principales entités sont les modèles atomiques DEVS, les modèles couplés DEVS, les ports d'entrées/ sorties des modèles puis les connexions entre ces modèles ;

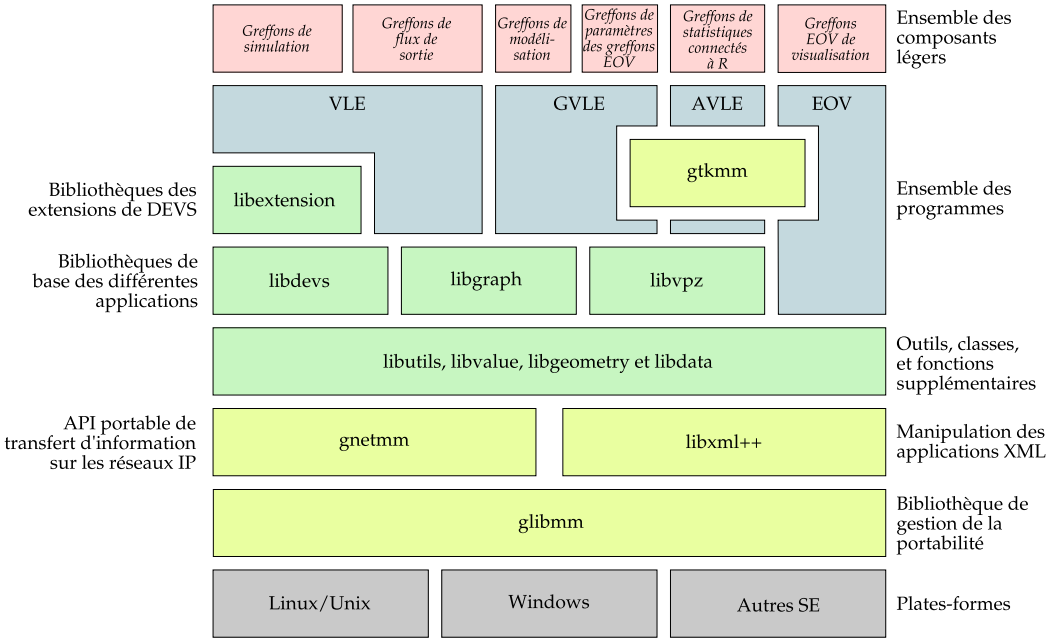


FIGURE 2.15 – *Graphe de dépendances entre les bibliothèques, programmes et composants de la plateforme VLE.*

- *libdevs* : un ensemble de classes destinées à la simulation (coordinateurs, simulateur, échéancier, événements etc.) ;
- *libvle* : un module contenant entre autre les différentes fonctions d'exécution de simulation ;
- *libextension* : une bibliothèque qui fournit différentes extensions du formalisme DEVS ;
- *libvpz* : une bibliothèque chargée de gérer les fichiers entrées/sorties de VLE décrits dans un langage spécifiques nommé VPZ pour *VLE file Project gZiped*. Ce langage intègre entre autre :
  - la hiérarchie de modèles : (i) la définition de chaque modèle atomique avec son nom, sa description, ses ports d'entrées/sorties, ses conditions d'initialisation, (ii) la définition des modèles couplés, leurs structures internes et les connexions internes et externes ;
  - la dynamique des modèles : la définition du comportement de chaque modèle atomique ;
  - l'expérience : la définition de la durée de la simulation, une liste de valeurs d'initialisation des modèles atomiques, le paramétrage des observations et les composants de visualisation associés etc.
  - la visualisation graphique : la définition de la représentation (taille, position, couleurs etc.) graphique des modèles dans le composant de visualisation GVLE.

2.2.5.b Simulateur VLE

Le simulateur VLE lit dans un premier temps le fichier VPZ. Il récupère l'ensemble des informations nécessaires à la simulation c'est-à-dire, (i) la structure des modèles, (ii) les dynamiques des modèles et (iii) les conditions expérimentales. Ensuite, le simulateur construit et initialise les réseaux de modèles couplés. Enfin, la simulation est lancée. Pour ce faire, VLE implémente les simulateurs abstraits de DEVS. Plus précisement, les simulateurs abstraits utilisés sont différents de ceux développés pour un simulateur DEVS classique. En effet, Ils se basent sur une

extension de DEVS capable de prendre en compte le parallélisme et les changements de structures en cours de simulation.

Pour plus de détail sur la plateforme VLE, le lecteur peut se référer à Quesnel et al. (2009) et Quesnel (2006).

2.3 Résumé du chapitre

Dans ce chapitre, nous avons introduit les éléments de base nécessaires à la conception des systèmes complexes autonomes. Nous avons décrit les fondamentaux de la planification par satisfaction de contraintes et de la planification hiérarchique. Nous utilisons ces deux approches dans le cadre de cette thèse. Nous nous sommes ensuite focalisés sur quelques architectures robotiques généralement employées pour la conception de systèmes complexes autonomes. De notre description, nous pouvons tirer trois conclusions qui portent sur :

1. l'architecture d'un système de planification et d'exécution de plans dans un environnement dynamique et incertain,
2. les mécanismes d'interaction entre différents planificateurs spécifiques opérant sur des horizons de temps variables,
3. l'exécutif à utiliser pour l'interprétation de plans temporels.

Au niveau architectural, CLARAty (Fisher et al., 2000; Estlin et al., 2001) (utilisée dans le système CLEaR) et IDEA (Muscettola et al., 2002; Dias, 2003) nous semblent être des architectures particulièrement intéressantes. La première intègre la planification et l'exécution au niveau décisionnel. Elle favorise, par la même occasion, la mise en œuvre des systèmes de planification continue. La seconde illustre la manière d'organiser des modules fonctionnels distribués afin de construire un système de planification à différents niveaux d'abstractions. Nous nous inspirons de ces deux architectures afin d'en proposer une qui réponde à notre problématique.

L'autre avantage de l'architecture IDEA est qu'elle propose des mécanismes permettant de mettre en interaction différents planificateurs spécifiques. Ces planificateurs sont perçus comme des systèmes de contrôles autonomes qui, coopèrent pour construire le plan et dont l'activation est gérée par un système particulier. Nous nous inspirons des mécanismes d'interactions de IDEA, puis de l'idée de la hiérarchisation des horizons de planification proposée dans CLEaR, afin de répondre à notre question de reproduction de la dynamique d'interaction entre les décisions stratégiques, tactiques et opérationnelles.

Enfin, au niveau de l'exécution du plan opérationnel résultant de la coopération des systèmes conçus pour la résolution des problèmes de décisions stratégiques, tactiques et opérationnelles, nous nous inspirons du fonctionnement de l'exécutif réactif, basé sur des modèles d'actions, et des contraintes temporelles, proposé dans RMPL Kim et al. (2001). Nous étendons cette approche de modélisation et d'interprétation du plan à un cadre événementiel de Modélisation & Simulation de systèmes distribués et hiérarchiques.

Le cadre de Modélisation & Simulation que nous utiliserons est nommé DEVS. Nous avons décrit dans ce chapitre les concepts de base et les principales caractéristiques du formalisme *DEVS classique* (Zeigler, 1976; Zeigler et al., 2000). Cette première description nous a permis d'ouvrir notre discours sur la pertinence d'un agent conçu comme système dynamique reposant sur le cadre DEVS. A ce jour, certains auteurs comme Uhrmacher et Kullick (2000) se sont intéressés à la question.

La plateforme de simulation agent nommée JAMES a été conçue exclusivement en DEVS. Cependant, dans JAMES, le cycle de fonctionnement des agents est basé sur une version *monolithique* de la boucle SPA (*Sense/Plan/Act*). Cette version *monolithique* est de plus en plus abandonnée de nos jours. En effet, elle entraîne des problèmes de réactivité du système car il est très difficile de planifier à l'échelle du changement de l'environnement.

Notre proposition s'inscrit dans cette approche DEVS. Contrairement à JAMES, nous nous basons sur des extensions DEVS capables de gérer :

- la parallélisation de l'exécution des modèles DEVS,
- les changements de structure des modèles couplés DEVS en cours de simulation.

Cette piste nous permet de repousser les limites de l'approche adoptée dans JAMES.

Description du cadre de simulation : Safihr

3

Sommaire

2.1 Planification de tâches et contrôle d'exécution 41
2.1.1 Introduction générale sur la planification 41
2.1.2 Planification par satisfaction de contraintes pondérées 46
2.1.3 Planification hiérarchique 49
2.1.4 Architecture robotique pour la planification dans les systèmes complexes autonomes 52
2.1.5 Exécution de plan en planification 56
2.1.6 Quelques approches orientées modèles 59
2.1.7 Discussion sur la planification et l'exécution de plans temporels . . 63
2.2 Multi-modelisation & Simulation 64
2.2.1 Modèle atomique DEVS 64
2.2.2 Modèle couplé DEVS 66
2.2.3 Limites de DEVS classique 67
2.2.4 Simulation d'agent en DEVS 67
2.2.5 Plateforme de simulation VLE 68
2.3 Résumé du chapitre 70

Dans ce chapitre, nous introduisons les composants de l'architecture modulaire que nous proposons dans cette thèse. Nous partons de l'architecture traditionnellement utilisée dans les systèmes de production agricole. Nous décrivons le couplage entre les différents éléments de cette architecture. Cela nous permet d'introduire les composantes de notre cadre de simulation (Safihr). Nous finissons ce chapitre par la formalisation du système de gestion des états de croyance dans Safihr.

3.1 L'ARCHITECTURE DES SYSTÈMES DE PRODUCTION AGRICOLE

3.1.1 Vue d'ensemble

Le système de production agricole est divisé en trois systèmes en interaction : le *système agent* (AS), le *système opérant* (OS) et le *système biophysique* (BS) (cf. Figure 3.1). Chacun de ces systèmes est constitué de processus indépendants qui modélisent la dynamique des éléments constituant le système. Ce découpage inspiré de Martin-Clouaire et Rellier (2009) et de Le Gal et al. (2007) permet de séparer les *processus de décision* de l'agent, des processus biophysiques qu'il contrôle. De plus, l'architecture proposée permet de séparer les processus opérationnels qui résultent de la mise en œuvre des consignes d'exécution de tâches ou d'observation de l'agent.

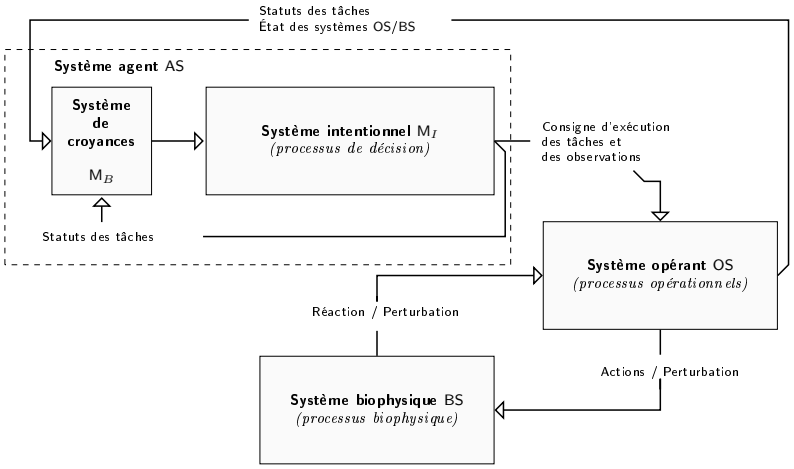


FIGURE 3.1 – *Modèle d'interaction des éléments d'un système de production agricole*

Ainsi, le système agent (AS) contrôle la dynamique du système biophysique (BS) via un système opérant (OS). Il envoie des consignes d'exécution de tâches ou d'observation au système opérant (OS). Ce dernier grâce aux *processus opérationnels* qui le constituent, simule l'exécution des consignes. Les processus opérationnels définissent ainsi le mode d'interaction avec le système biophysique. AS reçoit en retour des informations en provenance de l'OS sur les statuts d'exécution des processus opérationnels et de l'état du système biophysique.

3.1.2 Système biophysique

Le système biophysique (BS) est constitué d'un ensemble de *processus biophysiques* autonomes. Ces processus simulent en particulier la dynamique des paramètres caractéristiques :

- du sol tel que le niveau hydrique etc.
- des plantes tels que l'indice foliaire, la biomasse, le stade de floraison etc.

Le système biophysique est influencé par un ensemble d'événements externes incontrôlables liés à l'environnement (l'exposition au soleil de la plante, la température, les précipitations etc.) et par les perturbations des processus opérationnels issus du système opérant. Sous l'influence des perturbations de l'environnement et du système opérant, BS retourne à OS les valeurs d'un ensemble de variables d'état prédéfini.

3.1.3 Système opérant

Le système opérant (OS) reproduit le déroulement de l'exécution des tâches et celui de la dynamique des ressources physiques du système de production agricole. En

pratique cela revient à simuler, par exemple, un ouvrier qui fertilise un champs ou encore un tracteur labourant une parcelle. La mise en œuvre de ces actions est associée à un processus physique pouvant être modélisé sous la forme d'un système d'équations aux différences perturbant les processus biophysiques de BS. Nous appelons *processus opérationnels*, les processus physiques qui simulent la mise en œuvre d'une décision du système agent. Ainsi, OS est constitué d'un ensemble de processus opérationnels qui perturbent le système biophysique et qui retourne à l'agent le statut d'exécution des tâches et les observations sur BS et OS. Ces observations peuvent porter sur l'état réel des ressources ou celui des éléments du système biophysique.

3.1.4 Système agent

Le *système agent* (AS) reproduit le comportement de l'agriculteur. Il intègre d'une part un module de raisonnement symbolique sur son état propre et celui des systèmes opérant et biophysique (M_B). D'autre part, il se base sur un module délibératif (M_I) afin de mettre en cohérence sa planification d'action avec l'objectif global qu'il se fixe pour la conduite de son système de production. Deux processus majeurs permettent de garder cette cohérence. Les *processus de mise-à-jour de l'état de croyance* contenus dans M_B qui modélisent la perception de l'agent de l'évolution du système de production. Les *processus de décision* contenus dans M_I quant à eux représentent les mécanismes dont il dispose pour la construction de l'action. Ces processus de décision permettent de :

- définir sur la base des contraintes de l'exploitation et de préférences de l'agent, les objectifs de production atteignables à long terme,
- décider le mode de conduite des cultures permettant d'atteindre ses objectifs globaux,
- ordonnancer la conduite journalière en fonction des contraintes de l'exploitation et de préférence de l'agent,
- mettre à jour le statut des tâches à réaliser en fonction de l'évolution du système,
- déduire de l'état de croyance, les situations décisives pour le contrôle du système biophysique,
- détecter des situations considérées comme étant critiques puis de proposer une révision de la planification courante.

Ainsi, suivant les processus de décision considérés, les capacités de planification de l'agent peuvent varier dans des proportions allant du comportement purement réactif basé sur des règles de production agricoles à des comportements délibératifs basés sur des objectifs prédéfinis. Dans les deux cas, AS perçoit l'état des systèmes OS et BS envoyé par le système opérant et produit des consignes d'exécution de tâches qu'il envoie au système opérant.

3.1.5 Interaction et message entre systèmes

D'après la Figure 3.1, quatre types d'interaction existent entre les systèmes AS, OS, et BS. Chacune de ces interactions est caractérisée par un type de message allant du :

1. système agent (AS) vers le système opérant (OS),
2. système opérant (OS) vers le système agent (AS),
3. système opérant (OS) vers le système biophysique (BS),
4. système biophysique (BS) vers le système opérant (OS).

Dans ce manuscrit, nous ne décrivons pas la structure des messages entre le système opérant et le système biophysique. En effet, ces messages sont étroitement dépendants du système biophysique considéré. Il existe à ce jour un très grand nombre de modèles biophysiques ayant chacun leur propre spécificité.

3.1.5.a Du système agent vers le système opérant

L'interaction entre le système agent (cf. Figure 3.1) et le système opérant se caractérise essentiellement par des consignes d'exécution envoyées du premier vers le second. Ces consignes comportent les éléments de base permettant de déclencher des processus opérationnels du système opérant. Chacune des consignes envoyées par AS est associée à un processus opérationnel spécifique dans l'optique de contrôler un élément déterminé du système biophysique. Ces consignes prennent effet immédiatement et sont relatives à l'exécution, l'annulation ou la suspension de tâches. Il y a ainsi autant de messages que de consignes d'exécution envoyées.

Le format du message est décrit par la Figure 3.2. Les nœuds pleins représentent les données obligatoires du message tandis que les nœuds en pointillés représentent celles considérées comme étant optionnelles.

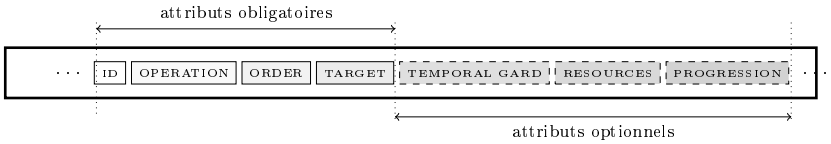


FIGURE 3.2 – *Format des messages du système agent vers le système opérant*

- ▷ *ID :* identifiant du message,
- ▷ *Operation :* la tâche concernée par le message. ex : *Semis*, *Récolte*.
- ▷ *Order :* le processus opérationnel associé à l'opération. Ces processus opérationnels sont : *start, stop*. Si l'ordre est de type *start*, le système opérant crée dynamiquement un processus opérationnel qui perturbe le système biophysique. L'ordre *stop* supprime un processus opérationnel en cours. Il relâche systématiquement toutes les ressources mobilisées pour l'exécution de la tâche.
- ▷ *Target :* la cible sur laquelle l'opération sera réalisée. Cette cible correspond à un objet du système biophysique (ex : parcelles).
- ▷ *Temporal guard :* attribut optionnel, qui définit la durée maximale d'exécution de l'instruction. Au delà de cette durée, l'instruction sera considérée comme étant en situation d'échec.
- ▷ *Resources :* attribut optionnel qui spécifie les ressources à utiliser et le paramétrage requis pour leur utilisation.
- ▷ *Progression :* attribut optionnel qui spécifie le mode de notification. Ces modes de notification définissent la manière dont le système opérant doit rendre compte de l'avancement de la tâche. Deux modes de notification sont possibles : discret ou événementiel.

3.1.5.b Du système opérant vers le système agent

L'interaction allant du système opérant vers le système agent (cf. Figure 3.1) se caractérise par des *notifications* relatives à l'avancement des processus opérationnels et des *observations* des variables d'états des systèmes biophysique et opérant. Ainsi, le système agent reçoit deux types de message du système opérant : *notifications*

et *observations*. Le premier, décrit par la Figure 3.3, informe AS qu'un processus opérationnel se termine convenablement ou qu'une évolution exceptionnelle du processus opérationnel s'est produite.

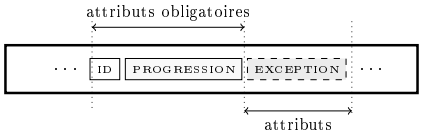


FIGURE 3.3 – *Format des messages de notification du système opérant vers le système agent*

▷ *ID :* identifiant de la consigne envoyée par AS à OS.

▷ *Progression :* information relative au temps passé (elapsed time (E.T)), L'estimation de la fin de la tâche (Estimated Time of Arrival E.T.A).

▷ *Exception :* attribut optionnel qui définit qu'une exception est survenue et indique l'origine de l'exception (ressources, temps d'exécution écoulé).

Le second type de message, défini par la Figure 3.4, informe AS de l'évolution des systèmes biophysique et opérant.

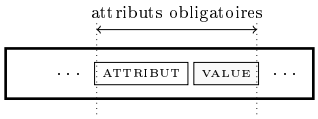


FIGURE 3.4 – *Format des messages d'observations du système opérant vers le système agent*

Toutes les communications entre BS et AS transitent par OS. Ces communications perçues comme des observations ponctuelles ou continues sont déclenchées par le système agent. Sur la base des observations et notifications, les processus de mise à jour de l'état de croyance vont permettre de définir les connaissances de l'agent sur l'évolution des OS et BS.

3.2 Le système de décision

3.2.1 Caractérisation du cycle de décision à simuler

Comme l'indique la Figure 3.5, le cycle de décision de l'agent est constitué de trois phases durant lesquelles l'agent observe, planifie et exécute son plan courant. En observant les événements issus du système opérant, il met à jour ses croyances sur l'état des systèmes opérant et biophysique. Ces connaissances, fonction du temps, sont exploitées afin de construire le plan courant de l'agent. Le plan résultant est exécuté. Ce cycle de décision se retrouve dans de nombreux travaux en IA, notamment pour les agents BDI (Belief Desire Intention Rao et Georgeff (1995)) . Nous notons cependant deux points particuliers de la phase de planification. D'une part, la planification des problèmes qui nous concerne porte sur des échelles temporelles et spatiales très différentes. Par exemple, le choix des rotations se fait sur plusieurs années et impacte l'ensemble de l'exploitation agricole. À l'inverse, les décisions de gestion de chantiers se font sur quelques jours et impactent essentiellement quelques îlots fonctionnels. D'autre part, les niveaux d'abstraction des plans obtenus sont très variables selon le type de problème de décision considéré. Par exemple, le plan issu de la décision d'assolement d'une année donnée n'est pas directement exécutable dans le système opérant. Ce plan doit être affiné. Nous distinguons deux classes de tâches qui sont : *tâches opérationnelles* et les *tâches de planification*.

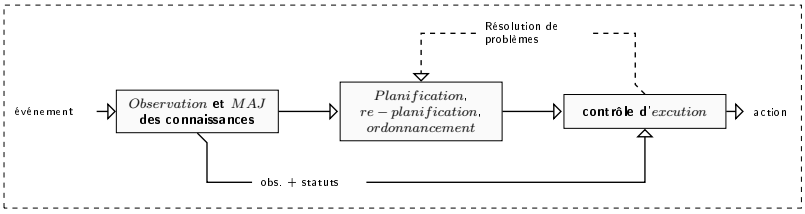


FIGURE 3.5 – *Cycle de décision à simuler*

3.2.1.a Tâches opérationnelles et tâches de planification

Les tâches *opérationnelles* sont des tâches directement exécutables dont l'effet direct est de déclencher un processus opérationnel du système opérant. Ces tâches modifient par conséquent l'état des systèmes biophysique et opérant.

Les tâches de *planification* sont des tâches abstraites dont l'effet direct est de déclencher la résolution d'un problème afin de modifier l'état interne de l'agent. Les tâches opérationnelles et de planification peuvent être combinées afin de construire un plan.

3.2.1.b Boucle de résolution de problèmes

Comme l'indique la Figure 3.5, les phases de planification et d'exécution peuvent être entrelacées (boucle de *résolution de problème*). De manière générale en planification continue (Ambros-Ingerson et Steel, 1988; Haigh et Veloso, 1998; Lemai, 2004), l'entrelacement des phases d'exécution et de planification se fait à condition que (i) le contrôleur d'exécution relève l'impossibilité (précondition insatisfaite / atteinte d'une date limite) d'exécution d'une tâche ou (ii) l'impossibilité d'atteindre un but.

À ces deux situations d'échecs susceptibles d'entraîner une replanification, nous ajoutons des conditions d'expansion de plan. En effet, le plan à exécuter étant constitué de tâches de planification, l'exécution de ces dernières entraîne une résolution de problème qui, par conséquent, modifie le plan courant de l'agent.

3.2.2 Définition de l'agent

Définition 3.1 *L'agent est défini par une tuple* $\langle \mathcal{B}, \mathcal{A}, \mathcal{P}, \mathcal{L}, \pi, \mathcal{Z} \rangle$ *où :*

- $\mathcal{B}$ *représente l'état de croyance de l'agent,*
- $\mathcal{A}$ *représente l'ensemble de tâches opérationnelles dont dispose l'agent,*
- $\mathcal{P}$ *représente l'ensemble de tâches de planification dont dispose l'agent,*
- $\mathcal{L}$ *est l'ensemble des plans partiels.* $\mathcal{L}$ *modélise la bibliothèque de plans constituant les connaissances du domaine de l'agent,*
- π *est un graphe orienté de tâches* $\mathcal{A}$, $\mathcal{P}$ *qui représente le plan courant de l'agent,*
- $\mathcal{Z}$ *est un ensemble d'algorithmes déclenchés par* $\mathcal{P}$ *et qui permettent d'étendre ou de réduire* π*.*

L'état de croyance $\mathcal{B}$ décrit l'état du monde du point de vue de l'agent. Il modélise la manière dont l'agent se représente son environnement ainsi que lui-même. Dans cette définition, on différencie les états de croyance et les connaissances nécessaires à l'agent pour atteindre ses objectifs. Ses connaissances sont représentées via la bibliothèque de plans $\mathcal{L}$. Les tâches opérationnelles $\mathcal{A}$ sont des tâches duratives soumises à des contraintes temporelles et de ressources et exécutables sur l'environnement. Les tâches de planification $\mathcal{P}$ sont instantanées et soumises à des contraintes temporelles. Le plan courant π est un graphe orienté acyclique de

tâches à exécuter. L'ensemble des algorithmes $\mathcal{Z}$ modélise les modes de résolutions disponibles pour la prise en compte des différents problèmes de décision en cours de simulation. Nous décrivons plus en détail dans les sections suivantes chacun des éléments du modèle de l'agent. Pour ce faire nous introduisons dans la section suivante l'architecture globale de l'agent.

3.2.3 L'architecture Safihr

3.2.3.a Description des composants

Dans cette thèse, nous proposons de voir l'agent comme un système hiérarchique dynamique et distribué explicitement en interaction avec le système opérant. L'interaction avec le système biophysique étant entièrement implicite. Nous proposons pour ce faire l'architecture Safihr (**S***imulation-based* **A***rchitecture* **F***or* **I***nterleaving* **H***eterogeneous decisions in* **R***eal world problems*) sur laquelle se base le système de décision.

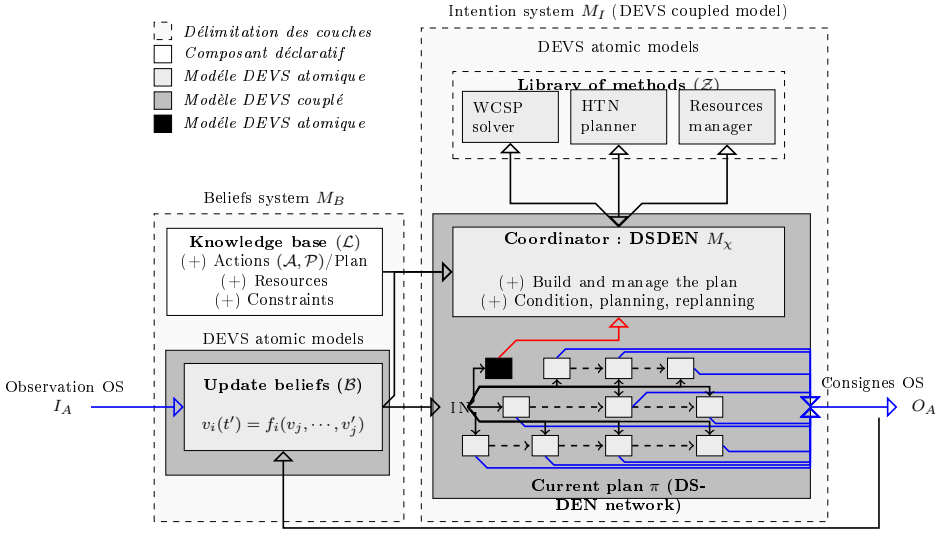


FIGURE 3.6 – *Architecture du système de décision* Safihr

Comme l'indique la Figure 3.6, les composants définissent des modèles DEVS où des entités déclaratives permettant d'initialiser ces modèles. Ces composants sont :

- ▷ *Base de connaissance :* ce composant modélise les connaissances statiques de l'agent. C'est un composant déclaratif qui contient des informations structurelles de l'environnement de l'agent. Ces informations sont relatives aux éléments tels que la structure de l'exploitation agricole, l'historique des parcelles, les capacités de ressources, les cultures productibles etc. D'autre part, ce composant contient les connaissances spécifiques à la planification. Il s'agit notamment des tâches opérationnelles $\mathcal{A}$ et des tâches de planification $\mathcal{P}$, les conditions d'activation des tâches, des plans partiels $\mathcal{L}$ décrivant les itinéraires techniques. Enfin, la base de connaissance contient les objectifs, les contraintes et les préférences de l'agent.
- ▷ *Gestion des états de croyance $\mathcal{B}$:* ce composant modélise la dynamique des états de croyance de l'agent. Contrairement aux données statiques contenues dans la base de connaissance, ce système intègre l'ensemble des connaissances permettant de mettre à jour l'état de l'agent.
- ▷ *Le plan courant π :* il est représenté par des séquences de tâches instanciées et partiellement ordonnées. Chaque séquence est associée à une cible du sys-

tème biophysique (par exemple, une parcelle). Les liens entre les tâches d'une séquence représentent les contraintes de précédences.

▷ *Bibliothèques d'algorithmes $\mathcal{Z}$:* elle représente l'ensemble des capacités dont dispose l'agent pour la construction et la modification de son plan. La bibliothèque d'algorithmes contient un ensemble d'algorithmes associés à chacun des problèmes de décisions spécifiques. Ces algorithmes permettent de planifier des tâches à différents niveaux d'abstraction. Ce composant contient autant de méthodes de résolution que de problèmes de décisions susceptibles de se poser durant la simulation.

▷ *Coordinateur centralisé :* ensemble des règles définissant l'ordre dans lequel les processus de décision doivent être activés. Le coordinateur centralisé se charge de la coordination de chacun des composants. Il intègre des mécanismes permettant de détecter les situations d'échec et de mettre des priorités dans leurs prises en compte.

3.2.3.b Aperçu du fonctionnement global

SAFIHR (Figure 3.6) est une architecture basée sur la version parallèle de l'extension DEVS à structure dynamique (Barros, 1997) (présentée dans la section 6.1.2.a). L'agent agriculteur est représenté comme un réseau de modèles DEVS parallèles. Ce réseau est nommé $DSDEN_A$. Les entrées I_A modélisent les fonctions d'observation tandis que les sorties O_A modélisent les tâches opérationnelles de l'agent. Le système reçoit en entrée des informations en provenance du système opérant. Ces informations caractérisent d'une part l'avancement des processus opérationnels (cf. Figure 3.3) et d'autre part l'état des processus biophysiques tel que la dynamique du sol, le stade de maturité des plantes etc. (cf. Figure 3.4). En sortie, le système envoie, via O_A, des consignes d'exécution de tâches (par exemple, la consigne « démarrer semis »).

Nous divisons l'architecture interne de SAFIHR en deux sous systèmes DEVS. D'un coté, le modèle atomique DEVS *système de croyance* - M_B et de l'autre, le modèle DEVS couplé *système intentionnel* - M_I. Ces deux sous systèmes sont des influenceurs l'un de l'autre et s'exécutent de manière complètement parallèle. Les connexions internes allant de M_B vers M_I sont utilisées afin de transmettre au *système intentionnel* les croyances mise à jour du système. Les connexions internes allant de M_I vers M_B permettent de transmettre au *système de croyance* des informations sur les tâches opérationnelles qui viennent de démarrer.

Sur la base des événements reçus et de l'état total (s, e), le *système de croyance* est mis à jour par les fonctions de transition δ_{int} et δ_{ext}. Les valeurs des sorties $\lambda(s)$ sont celles de prédicats nécessaires au fonctionnement de M_I.

Le *système intentionnel* est un modèle couplé intégrant un ensemble de modèles atomiques DEVS délibératifs couplés à un réseau $\langle \chi, M_\chi \rangle$ dans l'optique de construire et de contrôler l'exécution du plan courant de l'agent. Dans le cas des applications qui nous intéressent, nous avons identifié trois types de modèles atomiques DEVS délibératifs : *WCSP Solver*, *HTN planner* et *Resources manager*.

Le modèle atomique DEVS *WCSP solver* (cf. Chapitre 4) est dédié à la planification stratégique de l'organisation spatio-temporelle de l'exploitation agricole. Déclenché par M_χ, il produit une séquence de cultures de taille $\mathcal{H}$ pour chaque parcelle élémentaire (cf. Définition 4.7). Considérant une année donnée, le coordinateur M_χ déclenche le modèle atomique *HTN planner* (cf. Chapitre 5 section 5.3) afin de construire un plan tactique sur une année. Ces plans annuels sont utilisés comme modes de production pour chaque couple parcelles élémentaires, cultures. Une partie de plan est envoyée au modèle atomique DEVS *Resources manager* (cf.

Chapitre 5 section 5.6) afin de construire un ordonnancement sur quelques jours. L'horizon de l'ordonnancement est déterminé en fonction de la structure du réseau temporel représentant le plan annuel de l'agent.

Nous décrivons dans la suite la formalisation DEVS du système de croyance. Cela nous permettra, dans les chapitres suivants, de nous focaliser essentiellement sur les modèles atomiques DEVS délibératifs.

3.3 Les états de croyance de l'agent $\mathcal{B}$: description et formalisation DEVS

3.3.1 Caractéristique des variables d'état

L'état de croyance décrit l'état du monde du point de vue de l'agent. Pour les applications agronomiques que nous étudions, l'état de croyance de l'agent contient la disponibilité des ressources, la dynamique de l'environnement (ex : caractéristiques du sol, du climat), des plantes (ex : stade de croissance).

Exemple 3.1 *Dans le modèle de décision* Moderato *Bergez et al. (2001), afin de conduire une culture de maïs dont le modèle est présenté dans Muchow et al. (1990), l'agent perçoit :*

- *la température journalière T°,*
- *la quantité de pluie journalière $rain(t)$,*
- *le stade de maturité $cropSt(t)$.*

*Le modèle considéré est sensible aux facteurs climatiques et aux tâches de l'agent qui sont le semis, l'irrigation et la récolte. La tâche de semis peut se faire entre le 5 avril et le 5 mai à condition que le taux d'humidité du sol soit supérieur à un seuil sw. L'irrigation peut se faire à partir du 15 juin en cas de sécheresse du sol (seuil de sécheresse autorisé dp) et de disponibilité de la ressource eau. Enfin, la récolte peut se faire au plus tard 30 jours après que la culture ait atteint son stade de maturité (*MATURE*). À cette condition s'ajoute celle relative au seuil d'humidité du sol. Sur la base de ces informations, l'agent maintient quatre variables d'état relatives à l'humidité du sol $shumide(t)$ (cf. équation 3.1), le niveau de sécheresse du sol $pSeche(t)$ (cf. équation 3.3), la maturité de la plante $mature(t)$ (cf. équation 3.2), la réserve en eau $qEau(t)$ (cf. équation 3.4). En plus de ces variables, il dispose d'une constante lsd et d'une variable lhd qui définissent respectivement les dates de semis et de récolte au plus tard quelque soit les conditions du sol.*

$$shumide(t) \leftarrow \sum_{d=(t-3)}^{t} rain_d \quad (3.1) \qquad pSeche(t) \leftarrow \sum_{d=(t-5)}^{t} rain_d \quad (3.3)$$

$$mature(t) \leftarrow cropSt(t) \quad (3.2) \qquad qEau(t+1) \leftarrow qEau(t) - dose \quad (3.4)$$

t représente la date du jour, $dose$ une quantité fixe d'eau à chaque irrigation.

Cet exemple relativement simple de modèle de décision nous permet de montrer que les variables d'état de l'agent peuvent être des entiers (ex : $mature(t)$), des réels (ex : $shumide(t)$, $pSeche(t)$). La fonction de mise à jour de ces variables peut se réduire à une affectation simple comme dans le cas du stade de maturité de la plante (cf. équation 3.2) ou plus complexe, comme dans le cas du taux l'humidité où la valeur dépend d'humidité du $shumide(t)$ à t dépend de son historique sur les trois derniers jours. L'exemple 3.1 permet également d'illustrer le caractère multidimensionnel des objets manipulés par l'agent. En effet, dans cet exemple, la

parcelle est caractérisée par son couvert (la culture de maïs), le stade de cette culture, le taux d'humidité et le niveau de sécheresse du sol. Chacune de ces variables prennent leurs valeurs dans des domaines parfois différents. Cette caractéristique est commune à l'ensemble des objets manipulés par l'agent. Bien que la ressource en eau présentée ici soit uni-dimensionnelle, il n'est pas rare d'en avoir qui soit plus complexe à caractériser. Par exemple, une ressource réutilisable peut posséder un ensemble de facteurs. Un tracteur est défini comme une ressource réutilisable, ses caractéristiques sont sa capacité, sa gamme de vitesse (ex : lente ou rapide), sa consommation

3.3.2 Représentation des variables d'état de croyance

Nous représentons l'état de croyance $\mathcal{B}$ par un ensemble de variables et un ensemble d'objets décrits par ces variables.

Définition 3.2 (Objet) *Les* objets *permettent de définir les éléments appartenant à l'environnement de l'agent. Ils déterminent les caractéristiques fixes et l'évolution dans le temps de la valeur des variables définissant un élément de l'environnement.*

Un objet permet de définir des éléments de différentes natures tels que : des ressources *matérielles* et *humaines*, des *parcelles*, des *variables climatiques* etc.

Définition 3.3 (État de croyance) *L'ensemble des états de croyance $\mathcal{B}$ est représenté par un tuple $\langle V, \mathcal{R} \rangle$ où :*

- *V est un ensemble de variables numériques v_i décrivant l'évolution temporelle d'un objet. Les variables v_i appartiennent à un domaine discret ou continu D_i. Si $|D_i| = 1$ alors on en déduit que v_i est une constante. Chaque variable est associée à une fonction de correspondance f_i qui associe à un ensemble de variables V' une valeur $v_i(t) \in D_i$ (cf. l'équation 3.5).*

$$\forall v_i \in V, v_i(t') = f_i(\underbrace{v_j, \cdots, v'_j}_{V'}) \tag{3.5}$$

- *$\mathcal{R}$ est un ensemble d'objets. Soient $D_1, \cdots, D_n$ l'ensemble des domaines des variables V. Soit $\underset{i\in[1,n]}{\times} D_i$ le* produit cartésien *de n domaines. $\underset{i\in[1,n]}{\times} D_i$ est l'ensemble des n-uplets $\langle v_1, \cdots, v_n \rangle$ tel que $v_i \in D_i$ et que toutes les combinaisons possibles des valeurs de variables soient exprimées. $\underset{i\in[1,n]}{\times} D_i$ est un objet états de croyance $\mathcal{R}_\mathcal{B}$ qui représentent l'ensemble des états possibles pour l'agent. Cet ensemble peut être infini dès qu'une variable est continue.*

Considérant l'objet état de croyance $\mathcal{R}_\mathcal{B}$, nous définissons deux types d'objet : *simple* et *composé*. Un *objet simple* r^s sur les domaines D_i, D_j est une projection de $\mathcal{R}_\mathcal{B}$ sur les variables i et j. Les objets simples permettent de décrire certains objets de l'environnement de l'agent tels qu'une ressource, une parcelle etc. Dans la suite du manuscrit, on notera $\mathcal{R}_\mathcal{B}[D_i \times D_j]$ la projection de $\mathcal{R}_\mathcal{B}$ sur les domaines D_i et D_j. Les *objets composés* r^c sont quant à eux des agrégations d'objets simples ou composés auxquels s'appliquent un ou plusieurs opérateurs de composition. Ces opérateurs sont décrits dans la section suivante. Les objets composés permettent de définir des éléments complexes de l'environnement de l'agent tel qu'un parcellaire d'exploitation agricole, un ensemble de parcelles (ex : îlots structurels, blocs fonctionnels), un ensemble de ressources etc.

3.3.3 Les opérateurs sur les relations

Afin de définir les objets composés, nous définissons trois opérateurs de composition qui sont : l'*union disjointe* $\biguplus$, l'*intersection* $\bigcap$ et le *produit cartésien* $\times$.

▷ *Union disjointe :* notée $\biguplus$, l'union disjointe $\biguplus_{i=0}^{n} r_i$ de n objets est un objet composé r^c constitué de l'ensemble des n-uplets de chacune des objets r_i. L'opérateur d'union disjointe conserve l'origine des objets de départ permettant ainsi de palier à la perte d'informations induite par l'opérateur d'union $\cup$. Pour ce faire, $\biguplus$ rajoute des identifiants uniques aux objets de départ de manière à construire des objets de la forme (α_i, r_i) tels que $\forall\alpha_i, \forall\alpha_j, \alpha_i \neq \alpha_j$. Ainsi, soient deux objets r_i, r_j et deux identifiants $0, 1$:

$$r^c = r_i \biguplus r_j \Leftrightarrow \{(0, x)|x \in r_i\} \cup \{(1, y)|y \in r_j\} \tag{3.6}$$

L'union disjointe, telle que définit, permet de modéliser une disjonction entre des objets. On peut généraliser l'union disjointe de deux objets et définir l'union disjointe de n objets par :

$$r^c = \biguplus_{i=0}^{n-1} r_i \Leftrightarrow \bigcup_{i=0}^{n-1} \{(i, x)|x \in r_i\} \tag{3.7}$$

▷ *Produit cartésien :* noté $\times$, le produit $\times_{i=0}^{n} r_i$ de n objets est un objet r^c constitué de l'ensemble des n-uplets des n objets.

$$r^c = \times_{i=0}^{n-1} r_i \Leftrightarrow \{(x_0, \cdots, x_{n-1})|(x_0 \in r_0) \wedge \cdots \wedge (x_{n-1} \in r_{n-1})\} \tag{3.8}$$

Le produit cartésien des objets permet de modéliser une conjonction d'objets.

▷ *Intersection :* notée $\bigcap$, l'intersection $\bigcap_{i=0}^{n} r_i$ de n objets est un objet r^c constitué des n-uplets appartenant à la fois à chacun des objets r_i. L'intersection de deux objets permet de modéliser la réduction de domaines appartenant aux objets de départ.

Exemple 3.2 *Considérons l'exploitation virtuelle présentée à dans la section 1.6. Cette exploitation est constituée de quinze parcelles p_1, p_2, ..., p_15 de surfaces 12 ha. L'exploitation est divisée en quatre blocs fonctionnels $b_1 = (p_1, \ldots, p_4)$, $b_2 = (p_5, p_6)$, $b_3 = (p_7, \ldots, p_{10})$ et $b_4 = (p_{11}, \ldots, p_{15})$. L'exploitation dispose*[1] *de deux tracteurs, deux ouvriers et d'un quota annuel d'eau pour l'irrigation. Quatre cultures sont réalisables sur les parcelles. Chacune des cultures a trois stades de croissance qui sont : la* levée*, la* floraison *et la* maturité*. Admettons que pour contrôler la croissance des plantes, l'agent dispose de trois tâches (semis, irrigation, récolte). Les variables climatiques perçues par l'agent sont la température moyenne T°_{moy} de la journée et la quantité de pluie de la journée rain.*

1. Dans cet exemple, nous ignorons volontairement les autres ressources de l'exploitation

L'exemple 3.2 peut être représenté par les variables suivantes :

$$V = \left\{ \begin{array}{lcll} eau(t) & \in & D_0 & = & \mathbb{R}_0^+ \\ nbTracteur(t) & \in & D_1 & = & [0-2] \\ vitesse(t) & \in & D_2 & = & \{9,12,15,21\} \\ nbOuvrier(t) & \in & D_3 & = & [0-2] \\ surface(t) & \in & D_4 & = & \{12\} \\ stade(t) & \in & D_5 & = & [0-2] \\ culture(t) & \in & D_6 & = & [0-3] \\ T^{\circ}{}_{moy}(t) & \in & D_7 & = & \mathbb{R} \\ rain(t) & \in & D_8 & = & \mathbb{R}_0^+ \end{array} \right\}$$

La variable $eau(t)$ définit la disponibilité de la ressource en eau. Les variables $nbTracteur(t)$, $vitesse(t)$ définissent respectivement la disponibilité et la vitesse en hectare par jour de la ressource tracteur. $nbOuvrier(t)$ défini la disponibilité de la ressource ouvrière. Les variables $surface(t)$, $culture(t)$ et $stade(t)$ indiquent respectivement la superficie d'une parcelle, son couvert et le stade de développement du couvert. $T^{\circ}{}_{moy}(t)$ et $rain(t)$ quant à elles sont les variables climatiques ci-dessus décrites. Ces variables permettent de définir des objets simples portant sur les ressources et les parcelles. Le parcellaire de l'exploitation et les blocs fonctionnels sont définis par une union jointe, respectivement sur la totalité et des parties des objets simples parcelles $r^s_{p_i}$. Les ressources nécessaires à l'exécution des tâches de semis et de récolte sont représentées par le produit des objets $r^s_{ouvrier} \times r_{tracteur}$. Celle de l'irrigation est représentée r_{eau}.

$$\mathcal{R} = \left\{ \begin{array}{lcl} r^s_{eau} & = & \mathcal{R}_{\mathcal{B}}[D_0] \\ r^s_{tracteur} & = & \mathcal{R}_{\mathcal{B}}[D_1 \times D_2] \\ r^s_{ouvrier} & = & \mathcal{R}_{\mathcal{B}}[D_3] \\ r^s_{p_i} & = & \mathcal{R}_{\mathcal{B}}[D_4 \times D_5 \times D_6 \times D_7 \times D_8] \\ & & \\ r^c_{parcelle} & = & \uplus r_{p_i} \\ r^c_{b_1} & = & r_{p_1} \uplus r_{p_2} \uplus r_{p_3} \uplus r_{p_4} \\ r^c_{b_2} & = & r_{p_5} \uplus r_{p_6} \\ r^c_{b_3} & = & r_{p_7} \uplus r_{p_8} \uplus r_{p_9} \uplus r_{p_{10}} \\ r^c_{b_4} & = & r_{p_{11}} \uplus r_{p_{12}} \uplus r_{p_{13}} \uplus r_{p_{14}} \uplus r_{p_{15}} \\ r^s_{rsemis} = r^s_{rrecolte} & = & r^s_{ouvrier} \times r_{tracteur} \\ r^c_{rirrigation} & = & r_{eau} \end{array} \right\}$$

Cet exemple illustre une manière d'exploiter l'union disjointe et le produit cartésien pour modéliser une disjonction et une conjonction d'objets. De manière générale, les opérateurs présentés ci-dessus permettent de définir la structure des objets manipulés par l'agent. Nous définissons dans la section suivante les fonctions de correspondance permettant d'exploiter les objets.

3.3.4 Fonctions de correspondance sur les objets

Nous définissons deux classes de fonctions de correspondance sur les objets : les fonctions de *mise à jour* des variables et les *prédicats* sur les objets.

▷ *Fonction de mise à jour :* la fonction de mise à jour $\mathcal{F}$ d'un objet est définie par la *composée* des fonctions de correspondance f_i des variables $V' = \{v_i\}$ qui la définissent. Ces fonctions sont fortement dépendantes des modèles considérés. Par ailleurs, pour des éléments spécifiques de l'exploitation tels que les ressources, nous proposons dans la section 5.6 une caractérisation plus précise des fonctions de mise à jour.

▷ *Prédicat :* la valeur de vérité des prédicats $\mathcal{P}r = (P_1, \cdots, P_m)$ portant sur un objet dépend des variables $V' = \{v_i\}$ définissant l'objet concerné. Ces prédicats donnent des informations sur l'état des systèmes biophysique et opérant. Dans le premier cas, il s'agit des croyances sur l'état du sol et des plantes. Par exemple, le prédicat « portance de la parcelle p » dépend de la valeur cumulée des quantités de pluie sur les trois derniers jours. Dans le deuxième cas, il s'agit des croyances sur l'exécution des tâches opérationnelles (ex : fin, échec). Dans les deux cas, la valeur des prédicats se base essentiellement sur les variables « mise à jour » sans se préoccuper de la façon dont cette mise à jour est faite.

3.3.5 Formalisation DEVS du système $\mathcal{B}$

Pour chacun des objets simples, on associe un modèle atomique DEVS dans le composant $\mathcal{B}$ de l'architecture SAFIHR (cf. Figure 3.6). Ainsi, les éléments de l'environnement de l'agent (ex : parcelle) sont représentés par des modèles DEVS atomiques $M_r = \langle I_r, O_r, S_r, \delta_{int}, \delta_{ext}, \delta_{con}, \tau, \lambda \rangle$. Comme l'indique la Figure 3.7, chacun de ces modèles comporte deux ports d'entrées ($I_r = \langle I_O, I_N \rangle$). Le port I_O est associé aux messages d'observation du système opérant (cf. section 3.1.5.b) tandis que le port I_N est associé aux messages de notification (cf. section 3.1.5.b) de ce même système. Les sorties $O_r = \langle O_1, \cdots, O_m \rangle$ du système sont associées aux prédicats $\mathcal{P}r = (P_1, \cdots, P_m)$ portant sur un objet.

FIGURE 3.7 – *Interface d'une modèle DEVS définissant une relation*

3.3.5.a Observation et mise à jour de l'état de croyance

Ces systèmes DEVS sont entièrement réactifs. Lorsqu'un événement externe est reçu du système opérant, l'état S_r est mis à jour par les fonctions de correspondance $\mathcal{F}$ de l'objet. Cette mise à jour est réalisée en fonction de l'état courant des variables, des événements reçus sur les ports I_r et de la date courante.

L'état $S_r = \{V_r, \overrightarrow{\alpha}\}$ du modèle contient un ensemble V_r de variables v_i et un vecteur $\overrightarrow{\alpha} = (\alpha_1, \cdots, \alpha_m)$ avec $\alpha_k \in \{0, 1\}$. Chaque élément α_k est associé à un port de sortie O_k et est égal à 1 si les deux conditions suivantes sont réunies.

▷ *Condition 1 - perturbation :* au moins une des variables permettant de calculer les valeurs de vérité des prédicats P_k a été mise à jour à la date courante.

▷ *Condition 2 - traitement des bags :* l'ensemble des bags d'événement associés à la date courante a été traité.

Ainsi :

$$\begin{aligned}
\delta_{ext}(S_r, e, I_r) &: S_r = (v_i = f_i(v_i, v_{\{j\}}, e, I_{O,N}), \alpha_k = 1) \\
\delta_{int}(\alpha_k = 1) &: S_r = (\alpha_k = 0) \\
\delta_{con}(S_r, I_{O,N}) &= \delta_{ext}(\delta_{int}(S_r), 0, I_{O,N})) \\
\tau(\overrightarrow{\alpha} = \overrightarrow{0}) &= \infty \\
\tau(\overrightarrow{\alpha} \neq \overrightarrow{0}) &= 0 \\
\lambda(\ \alpha_k = 1) &: O_k \leftarrow P_k
\end{aligned}$$

En cas d'événements externes sur I_O et I_N, les fonctions de correspondance f_i des variables impactées par le message sont appelées. Une fois l'ensemble des bags traités, les variables α_k impactées sont affectées à 1. Une sortie est réalisée pour l'ensemble des prédicats P_k pour lesquels $\alpha_k = 1$. Ces sorties sont envoyées au *système intentionnel*.

3.4 Résumé du chapitre

Nous avons introduit dans ce chapitre, l'architecture modulaire nommée SAFIHR. Dans SAFIHR, l'agent est perçu comme un système hiérarchique dynamique et distribué en interaction avec le système opérant. Nous avons également présenté la représentation de l'état de croyance de l'agent. Enfin, nous avons montré comment un tel système de croyance peut être formalisé en utilisant la cadre DEVS.

Planification d'allocation des cultures : approche basée sur les WCSP

4

Sommaire

3.1 L'architecture des systèmes de production agricole 74
3.1.1 Vue d'ensemble 74
3.1.2 Système biophysique 74
3.1.3 Système opérant 74
3.1.4 Système agent 75
3.1.5 Interaction et message entre systèmes 75
3.2 Le système de décision 77
3.2.1 Caractérisation du cycle de décision à simuler 77
3.2.2 Définition de l'agent 78
3.2.3 L'architecture Safihr 79
3.3 Les états de croyance de l'agent $\mathcal{B}$: description et formalisation DEVS 81
3.3.1 Caractéristique des variables d'état 81
3.3.2 Représentation des variables d'état de croyance 82
3.3.3 Les opérateurs sur les relations 83
3.3.4 Fonctions de correspondance sur les objets 85
3.3.5 Formalisation DEVS du système $\mathcal{B}$ 85
3.4 Résumé du chapitre 86

Dans ce chapitre nous présentons une approche originale permettant de résoudre le problème d'allocation de culture. Nous avons décrit (cf. section 1.2) l'allocation de culture comme étant un problème de planification stratégique. Elle consiste à assigner sur un horizon de plusieurs années et de manière explicite, des cultures aux parcelles. Pour ce faire, nous introduisons le formalisme WCSP et les contraintes globales que nous exploitons. Nous décrivons la formulation du problème dans le cadre WCSP. Ensuite nous présentons la formalisation des contraintes et les mécanismes permettant d'évaluer (i) les superficies allouées aux cultures, (ii) l'utilité des allocations. Nous finissons ce chapitre en appliquant notre approche sur le cas d'étude présenté à la section 1.6.

4.1 État de l'art sur les contraintes globales

4.1.1 Les réseaux de contraintes pondérées

4.1.1.a Choix du cadre des réseaux de contraintes pondérées

Le réseaux de contraintes (Constraint Satisfaction Problem - CSP, cf. Définition 2.1) ne permettent pas de prendre en compte facilement les préférences. Nous nous basons ici sur les réseaux de contraintes pondérées (*Weighted CSP* - WCSP (Meseguer et al., 2006)) qui nous semblent plus appropriés pour la résolution des problèmes d'optimisation.

Le choix du cadre des réseaux de contraintes pondérées, pour la modélisation du problème de décision stratégique, a été motivé par la présence de contraintes dures et des préférences chez l'agriculteur (cf. sections 1.2.2.c et 1.6.4.a). Il a été démontré que certaines techniques de consistances locales relaxées sont particulièrement intéressantes (voir les expérimentations de de Givry et al. (2005)) pour des fonctions de coûts binaires. De plus, les récents travaux de Lee et Leung (2012) (cf. Théorème 4 et Section 5.2.2 de l'article) soulignent également la performance des techniques de consistances locales relaxées pour des fonctions de coûts globales telles que Soft-gcc (cf. section 4.1.4).

Pour ces différentes raisons, nous avons choisi de nous baser sur le cadre WCSP pour la résolution du problème de décision stratégique. Ces travaux ont été précédemment proposés dans Akplogan et al. (2011, 2012a).

4.1.1.b Rappels

Nous avons introduit, dans la section 2.1.2.b, les grandes lignes du cadre WCSP. Rappelons que ce cadre est une extension qui ajoute aux CSP une structure de valuation permettant de définir une structure algébrique caractérisant les coûts associés à certaines combinaisons de valeurs.

D'après la définition 2.6, un WCSP est défini par un triplet $\langle \mathcal{X}, \mathcal{D}, \mathcal{W} \rangle$ avec :
$\mathcal{X} = \{x_1, \cdots, x_n\}$ un ensemble fini de variables,
$\mathcal{D} = \{\mathcal{D}_1, \cdots, \mathcal{D}_n\}$ un ensemble fini de domaines de variables tel que chaque variable $x_i \in \mathcal{X}$ soit associée à un domaine fini de valeur $\mathcal{D}_i \in \mathcal{D}$
$\mathcal{W} = \{\mathcal{W}_{\mathrm{S}_1}, \cdots, \mathcal{W}_{\mathrm{S}_e}\}$ un ensemble de fonctions de coûts. Soit $l[\mathrm{S}_i]$ l'ensemble des combinaisons de valeurs sur la portée S_i. Chaque fonction de coûts $\mathcal{W}_{\mathrm{S}_i} \in \mathcal{W}$ est définie par $\mathcal{W}_{\mathrm{S}_i} : l(\mathrm{S}_i) \rightarrow [0, m]$ avec $m \in [1, \cdots, +\infty]$.

La solution d'un WCSP est une affection complète $\mathsf{A} \in l[\mathcal{X}]$ de coût $\mathsf{cost}(\mathsf{A})$ telle que :

$$\mathsf{cost}(\mathsf{A}) \quad = \quad \min_{\mathsf{A} \in l[\mathcal{X}]} \left(\sum_{\mathcal{W}_{\mathrm{S}_i} \in \mathcal{W}} \mathcal{W}_{\mathrm{S}_i}(\mathsf{A}[\mathrm{S}_i]) \right)$$

avec $\mathsf{A}[\mathrm{S}_i]$ la projection d'une affectation de valeurs sur le sous ensemble de variables S_i.

4.1.1.c Contraintes globales

Une contrainte globale est une contrainte dont la portée peut être de taille variable. Autrement dit, sans aucune redéfinition de la contrainte, cette dernière peut porter

sur différents nombre de variables. Ainsi, plutôt que de définir une conjonction de contraintes, les contraintes dites globales, réunissent un ensemble de propositions en une seule contrainte.

Dans les sections suivantes, nous présentons trois contraintes globales (Regular, Gcc, Soft-gcc, Same) particulièrement intéressantes pour la modélisation du problème de décision stratégique.

4.1.2 La contrainte globale Regular

4.1.2.a Le concept d'automate à états

Un automate à états M est un graphe orienté qui définit un langage régulier $\mathcal{L}$ et qui permet de déterminer un mot appartenant au langage.

Définition 4.1 (Automate à états finis déterministe) *Un automate à états finis déterministe (Deterministic Finite Automaton (*DFA*)) M est défini par un tuple $\langle S, \Sigma, \delta, q_0, F\rangle$ où :*

- ▷ S *: est un ensemble fini d'états,*
- ▷ Σ *: est un alphabet,*
- ▷ $\delta : S \times \Sigma \to S$ *: est un ensemble fini de transitions*
- ▷ $q_0 \in S$ *: un état initial,*
- ▷ $F \subseteq S$ *: est un ensemble d'états terminaux.*

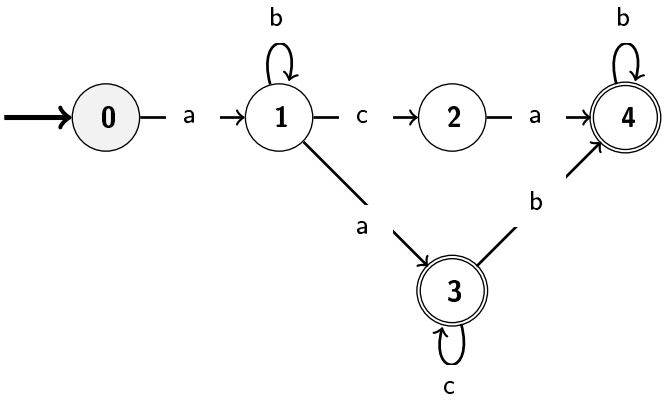


Figure 4.1 – *Exemple d'automate à états finis déterministe*

Par exemple, l'automate décrit par la Figure 4.1 comporte 5 états. L'état $\{0\}$ est un état initial. Les états $\{3, 4\}$ sont des états terminaux. L'alphabet $(\{a, b, c\})$ et les transitions de l'automate permettent de définir les expressions régulières de la forme $ab^*(ca|ac^*b)b^*$. Les mots *"aacc"*, *"abca"*, *"aaccbb"* appartiennent tous à ce langage.

4.1.2.b Définition de la contrainte globale Regular

Définition 4.2 (Contrainte globale Regular$(\mathcal{X}, M)$ (Pesant, 2004)) *Soient $M = \langle S, \Sigma, \delta, s_0, F\rangle$ un automate à états finis déterministe et $\mathcal{X} = \{x_0, \cdots, x_n\}$ une séquence de variables pour laquelle chaque variable appartient à un domaine fini noté $\mathcal{D}_0, \cdots, \mathcal{D}_n$ avec $\mathcal{D}_i \in \Sigma$. La contrainte* Regular$(\mathcal{X}, M)$ *est une contrainte globale portant sur la séquence $\mathcal{X}$. Elle impose que les affectations de la séquence de variables respectent le langage régulier $\mathcal{L}(M)$.*

La contrainte globale REGULAR permet de modéliser toutes sortes de problèmes dans lesquelles les affectations d'une séquence de variables doivent respecter une certaine structure.

Exemple 4.1 *Dans Métivier et al. (2009), les auteurs utilisent des contraintes* REGULAR *pour la modélisation des problèmes d'affectations d'infirmières dans les hôpitaux. Dans ce cas, chaque variable x_i représente une infirmière un jour donné. Le domaine $\mathcal{D}_i$ de la variable représente les différentes équipes (équipe du matin, équipe du soir, équipe de nuit, équipe de repos) auxquelles l'infirmière peut être affectée un jour donné. L'automate représente les règles de composition des équipes. Il prend en compte la législation du travail, les besoins de l'hôpital etc. Un exemple de règle est :* une infirmière ne peut pas enchaîner une équipe du matin après avoir travaillé dans une en équipe de nuit.

4.1.2.c Construction du graphe en couches

Pesant (2004) représente une contrainte REGULAR par un graphe en couches $G = (V, E)$ avec V et E respectivement l'ensemble des nœuds et des arcs du graphe. Considérons une séquence de variables $\mathcal{X} = \{x_0, \cdots, x_n\}$. Le graphe en couches $G = (V, E)$ associé à un automate M, est constitué de $n+1$ couches. Chacune des couches $V_0, \cdots, V_{n+1}$ est constituée de $|S|$ nœuds avec S le nombre d'états de l'automate. Le nœud s_k^i représente l'état $s_k \in S$ dans la couche i. Deux nœuds sources s et cible t sont également rajoutés. Ainsi le graphe en couches est défini comme suit.

$$
\begin{aligned}
V &= V_0 \cup V_1 \cup \cdots \cup V_{n+1} \cup \{s, t\} \text{ avec } V_i = \{s_k^i | s_k \in S\} \forall i \in \{0, 1, \cdots, n\} \\
E &= E_s \cup E_0 \cup \cdots \cup E_{n+1} \cup E_t \\
&\quad E_i = \{(s \to s_0^0)\} \\
&\quad E_t = \{(s_k^i \to s_l^{i+1}) | \delta(s_k, v) = s_l, v \in \mathcal{D}_i\} \forall i \in \{0, 1, \cdots, n\} \\
&\quad E_s = \{(s_k^n \to t) | s_k \in F\}
\end{aligned}
$$

Les transitions $\delta(s_k, v) = s_l$ sont représentées dans le graphe en couches en ajoutant des arcs labellisés avec v qui s_k^i à s_l^{i+1}. En plus de ces arcs, on ajoute les arcs $(s \to s_0^0)$ et des états terminaux de la dernière vers t.

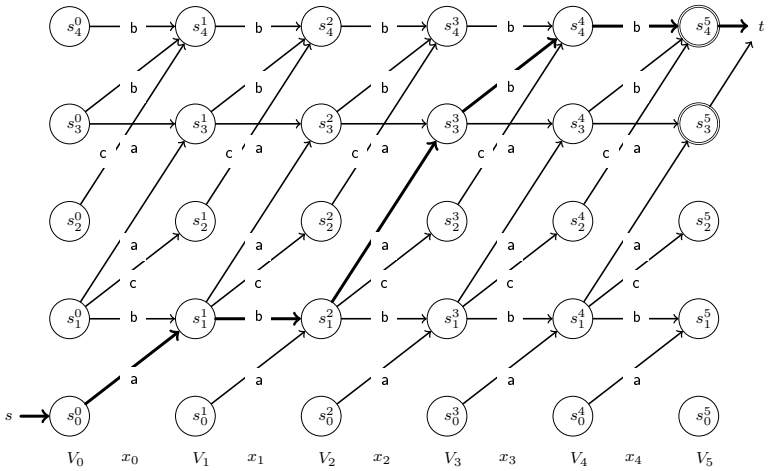


FIGURE 4.2 – *Graphe en couches pour l'automate de la Figure 4.1*

Comme l'indique la Figure 4.2, le graphe en couches représenté est celui associé à l'automate de la Figure 4.1. On y retrouve les couches V_i. Ce graphe en couches est représenté pour une séquence de cinq variables x_0, x_1, x_2, x_3, x_4.

4.1.2.d Consistance d'arcs

Une solution d'une contrainte Regular$(\mathcal{X}, M)$ correspond à un chemin de s à t dans le graphe en couches. Par exemple, dans le graphe en couches de la Figure 4.2, l'affectation $\{(x_0 = a), (x_1 = b), (x_2 = a), (x_3 = b), (x_4 = b)\}$ est une solution.

Par construction, certains arcs du graphe en couches sont obsolètes c'est-a-dire qu'il n'existe pas de chemin de s à t passant par ces arcs. C'est le cas par exemple de la couche V_0 où en dehors des arcs sortant de l'état initial (s_0^0 dans l'exemple), tous les autres arcs sont inatteignables. Le graphe en couches peut donc être filtré afin d'obtenir un graphe en couches réduit.

Le Filtrage se fait en deux phases : *ascendante* et *descendante*. Dans la phase ascendante, l'algorithme (disponible dans Pesant (2004)) de construction du graphe en couches réduit, élimine sur toutes les couches tous les arcs qui ne sont pas accessibles en partant de l'état s. Ensuite, la phase descendante élimine sur toutes les couches, en partant de la couche V_{n+1}, les arcs arrivant vers des états inaccessibles, c'est-à-dire tous les arcs qui ne sont pas connectés à t.

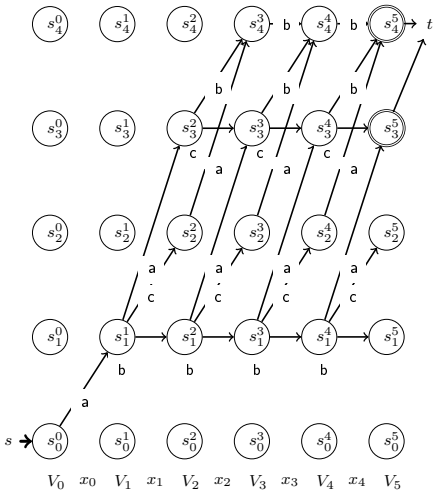


Figure 4.3 – *Graphe en couches pour l'automate de la Figure 4.1 après la phase ascendante*

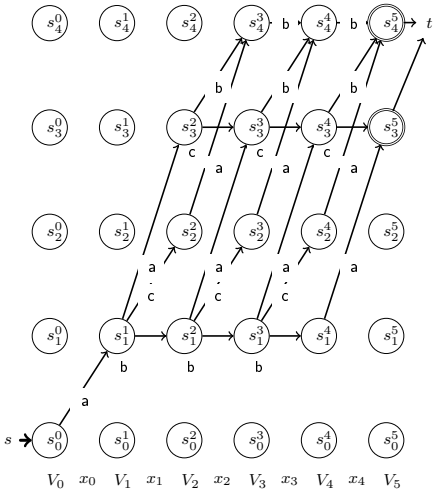


Figure 4.4 – *Graphe en couches pour l'automate de la Figure 4.1 après la phase descendante*

Une fois le graphe en couches réduit construit, le test de cohérence de la contrainte Regular$(\mathcal{X}, M)$ est valide si et seulement si il existe au moins un arc c'est-à-dire $E \neq \emptyset$.

4.1.2.e Consistance de domaines

Comme l'indique l'algorithme 2, la consistance de domaine peut être réalisée en parcourant le graphe en couches. Pour chaque variable $x_i \in \mathcal{X}$, les valeurs $v \in \mathcal{D}_i$ sont retirées du domaine s'il n'existe aucun arc labellisé v reliant deux couches successives i et $i+1$.

Dans notre exemple, si on considère que $\mathcal{D}_0, \mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3$ et $\mathcal{D}_4$ sont respectivement les domaines des variables x_0, x_1, x_2, x_3 et x_4, après exécution de l'algorithme 2 on obtiendra $\mathcal{D}_0 = \{a\}$, $\mathcal{D}_1 = \{a\}$, $\mathcal{D}_2 = \{a, b, c\}$, $\mathcal{D}_3 = \{a, b, c\}$, $\mathcal{D}_4 = \{a, b, c\}$

Algorithme 2 : Filtrage des valeurs sur le graphe en couches réduit

```
1 début
2   pour x_i ∈ 𝒳 faire
3     pour v ∈ 𝒟_i faire
4       si ∄(s_k^i → s_l^{i+1}) | δ(s_k, v) = s_l alors
5         𝒟_i → 𝒟_i \ {v}
6 fin
```

4.1.3 La contrainte globale de cardinalité Gcc

Les contraintes globales Gcc sont généralement représentées par des réseaux de transport dans lesquels circulent des flots de valeurs. Avant d'aborder la description des Gcc nous introduisons ces deux concepts dans la section suivante.

4.1.3.a Quelques rappels sur les notions de réseau de transport et de flot

Nous décrivons dans cette section quelques concepts issus de la théorie des flots (cf. Schrijver (2003) pour plus de détails) et permettant de mieux comprendre la suite du manuscrit.

Soit $G = (V, E)$ un réseau de transport définit par un graphe orienté. V et E sont respectivement l'ensemble des nœuds et des arcs du réseau. Un réseau de transport contient deux nœuds particuliers $s \in V$ et $p \in V$ respectivement nommés nœud source et nœud puits.

Définition 4.3 (Flot) *Un flot de s à p dans un réseau de transport $G = (V, E)$ est une fonction $f : E \to \mathbb{N}^+$ telle que :*

- ▷ *: $\forall (u \to v) \in E$, $f(u \to v) \geq 0$,*
- ▷ *: $\forall v \in V \backslash \{s, t\}$ la propriété de conservation des flots est respectée c'est-à-dire la quantité de flots entrants ($f(\delta^{in}(v))$ est égale à la quantité de flots sortants ($\delta^{out}(v)$) avec :*
  - *$\delta^{in}(v)$ est l'ensemble des arcs entrants du nœud v,*
  - *$\delta^{out}(v)$ est l'ensemble des arcs sortants du nœud v.*

La *valeur* f du flot de s à p est égale à la quantité de flot entrant dans p.

Flot réalisable : chaque arc $(u \to v)$ du réseau est associé à une fonction de *demande* $d : E \to \mathbb{N}^+$ et une fonction de *capacité* $c : E \to \mathbb{N}^+$. La fonction de demande $d(u \to v)$ correspond à la quantité minimale de flots qui doit passer par l'arc $(u \to v)$. La fonction de capacité $c(u \to v)$ correspond à la quantité maximale de flot passant par l'arc $c(u \to v)$. Un flow f est dit *réalisable* si pour tous les arcs $(u \to v) \in E$, $d(u \to v) \leq f(u \to v) \leq c(u \to v)$.

Soit $\mathsf{w}(u \to v)$ le poids associé à un arc $(u \to v) \in E$.

Définition 4.4 (Coût d'un flot) *Le coût (encore appelé poids) noté* $\mathsf{weight}(f)$ *d'un flot f est définit par :* $\mathsf{weight}(f) = \sum_{(u \to v) \in E} \mathsf{w}(u \to v) f(u \to v)$.

Un flot réalisable est dit de *coût minimum* si et seulement si il minimise le coût du flot pour tous les flots de même valeur. Pour obtenir un flot réalisable de coût minimum, l'algorithme de Ford & Fulkerson (Ford et Fulkerson, 1956) peut être utilisé.

Capacité résiduelle : la capacité c_f résiduelle d'un arc $(u \to v)$ représente la quantité de flot pouvant encore passer par cet arc. Les capacités c_f sont définies comme suit :

$$\begin{aligned} c_f(u \to v) &= c(u \to v) - f(u \to v) \ \textit{si l'arc} \ (u \to v) \in E \\ c_f(v \to u) &= f(u \to v) \ \textit{si l'arc} \ (u \to v) \in E \\ c_f(u \to v) &= 0 \ \textit{sinon} \end{aligned}$$

Précisons que dans le cas où un poids $\mathsf{w}(u \to v)$ est associé à un arc $(u \to v)$, on diminue non seulement la capacité de l'arc $(u \to v)$ de $f(u \to v)$, mais le coût $\mathsf{w}(v \to u)$ de l'arc inverse $(v \to u)$ est égal à $-\mathsf{w}(u \to v)$.

Graphe résiduel : le réseau résiduel $G_f = (X, E_f)$ d'un flot f est un graphe obtenu à partir du graphe d'origine. Il définit les mises à jour et les transformations subies par le graphe G après le passage du flot f. Dans le réseau résiduel, on retire tous les arcs dont la capacité résiduelle est nulle. $E_f = \{(u \to v) | (u \to v) \in E, c(u \to v) > 0\}$.

L'exemple de la Figure 4.5 illustre, en partant d'un réseau de transport, et pour un flot de $f = 7$, la mise à jour des capacités résiduelles et le réseau résiduel résultant.

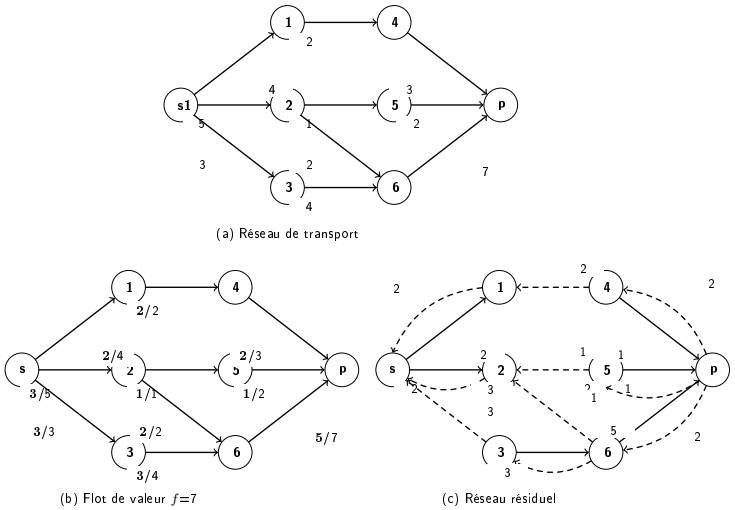


FIGURE 4.5 – *Exemples de réseau de transport (a), un flot de valeur 7 (b) et un réseau résiduel (c)*

4.1.3.b Description de la contrainte globale de cardinalité Gcc

Une contrainte globale de cardinalité GCC sur une portée $\mathcal{X} = \{x_0, \cdots, x_n\}$ permet de spécifier pour chaque valeur v appartenant à l'union $\bigcup \mathcal{D}_i$ des domaines des variables de la portée, une borne inférieure et une borne supérieure correspondant au nombre d'occurrences de la valeur v_k dans l'affectation $\mathsf{A}[\mathcal{X}]$.

Définition 4.5 (Contrainte globale de cardinalité $\textsc{Gcc}(\mathcal{X}, lb, ub)$ (Régin, 1996)) *Soient $\mathcal{X} = \{x_0, \cdots, x_n\}$ une portée, $\textsc{Gcc}(\mathcal{X}, lb, ub)$ une contrainte globale de cardinalité, $v_k \in \bigcup \mathcal{D}_i$, $lb(v_k)$ et $ub(v_k)$ respectivement la borne inférieure et la borne supérieure d'occurrence de v_k. La contrainte globale de cardinalité $\textsc{Gcc}(\mathcal{X}, [lb(v_0), \cdots, lb(v_m)], [ub(v_0), \cdots, ub(v_m)])$ admet une solution si et seulement si :*

$$\forall v_k \in \bigcup \mathcal{D}_i, \ lb(v_k) \leq |\{x_i \in \mathcal{X} | x_i = v_k\}| \leq ub(v_k)$$

La contrainte globale Gcc permet de modéliser un grand nombre de problèmes. Par exemple, une grille de sodoku 9×9 peut être modélisée [1] par un ensemble de contraintes Gcc . Les variables correspondent aux cellules de la grille tandis que les domaines sont les chiffres de 1 à 9. On définit pour cela 27 Gcc d'arité 9, dont 9 portent sur les grilles 3×3, 9 autres portent sur les cellules de chaque ligne et enfin 9 portent sur les cellules de chaque colonne. Dans ce cas particulier, les bornes supérieures $ub(v_k)$ et les bornes inférieures $lb(v_k)$ sont égales et valent 1.

Si nous reprenons l'exemple du problème d'affectations d'infirmières dans les hôpitaux (cf. exemple 4.1). Deux contraintes d'équipes ont été utilisées par Métivier et al. (2009). Elles stipulent que :

1. *du lundi au vendredi, les équipes du matin, du soir et de nuit sont respectivement composées de 2, 2, et 1 infirmières,*
2. *durant un week-end, les équipes du matin, du soir et de nuit sont toutes composées d'une seule infirmière.*

Considérons que les variables représentent une infirmière, un jour donné, et que le domaine $\mathcal{D}_i$ représente les différentes équipes. Les contraintes d'équipes sont représentées par des contraintes globales Gcc. Dans ces conditions, le nombre d'infirmières affectées aux équipes du matin, du soir et de la nuit est minoré et majoré par les valeurs stipulées dans la contrainte d'équipes. Par ailleurs, en l'absence de contraintes sur l'équipe de repos, la borne inférieure $lb(repos)$ sera de 0 et la borne supérieure $ub(repos)$ sera égale au nombre maximum d'infirmières.

4.1.3.c Représentation de la contrainte

Régin (1996) propose de représenter une contrainte $\textsc{Gcc}(\mathcal{X}, lb, ub)$ par un réseau de transport $G = (V, E)$. Ce réseau de transport est constitué du graphe de valeurs $G_t = (\mathcal{X} \cup Y, E)$ de la contrainte $\textsc{Gcc}(\mathcal{X}, lb, ub)$ auquel on rajoute les nœuds source s et puits p. Ainsi, l'ensemble des nœuds $V = \mathcal{X} \cup Y$ avec $\mathcal{X}$ l'ensemble des variables de x_i et Y l'ensemble des valeurs appartenant à l'union des domaines des variables x_i.

Le nœud s est connecté aux nœuds $x_i \in \mathcal{X}$ et $\mathcal{X} \subset V$ (les variables appartenant à la portée de la contrainte) du graphe de valeurs. Chaque arc $(s \to x_i) \in E$ du réseau de transport est associé à une capacité et à une demande de 1. Cela permet de modéliser le fait qu'une variable ne puisse prendre qu'une et une seule valeur. Les arcs du graphe de valeurs sont tous orientés des variables vers les valeurs. La demande sur ces arcs est de 0 tandis que la capacité est de 1. Cela permet de modéliser le fait qu'une variable x_i ne doit prendre qu'une et une seule valeur parmis celles de son domaine. Enfin, tous les nœuds $v_k \in Y$ représentant les valeurs de variables dans le graphe de valeurs, sont connectés au nœud puits p du réseau de transport. Pour chacun des arcs $(v_k \to p) \in E$, la demande est égale à la borne inférieure $lb(v_k)$ et la capacité est égale à la borne supérieure $ub(v_k)$. Ainsi le graphe G est défini comme suit.

$$
\begin{aligned}
V &= \{s\} \cup \mathcal{X} \cup Y \cup \{p\} \\
E &= E_s \cup E_{\mathcal{X}} \cup E_p \quad \text{avec} \quad E_s = \{(s \to x_i) | x_i \in \mathcal{X}\} \\
&\qquad\qquad\qquad\quad E_{\mathcal{X}} = \{(x_i \to v_k) | x_i \in \mathcal{X}, v_k \in Y\} \\
&\qquad\qquad\qquad\quad E_p = \{(v_k \to p) | v_k \in Y\}
\end{aligned}
$$

1. Généralement sur ce problème, une contrainte particulier appelée Alldiff est plus adaptée

La demande d et la capacité c d'un arc e sont définies comme suit.

$$\begin{array}{rl} \text{si } e \in E_s & d(e) = 1 \text{ et } c(e) = 1 \\ \text{si } e \in E_{\mathcal{X}} & d(e) = 0 \text{ et } c(e) = 1 \\ \text{si } e \in E_p & d(e) = lb(v_k) \text{ et } c(e) = ub(v_k) \end{array}$$

Exemple 4.2 *Considérons un ensemble de quatre variables* $\mathcal{X} = \{x_0, x_1, x_2, x_3\}$. *Les domaines de variables sont :* $\mathcal{D}_0 = \mathcal{D}_1 = \{0, 2\}$, $= \mathcal{D}_2 = \{0, 1\}$, $= \mathcal{D}_3 = \{1, 2\}$. *On définit* Gcc$(\{x_0, x_1, x_2, x_3\}, [0, 1, 0], [4, 2, 1])$

La Figure 4.6 représente le réseau de transport associé à la contrainte globale Gcc$(\{x_0, x_1, x_2, x_3\}, [0, 1, 0], [4, 2, 1])$.

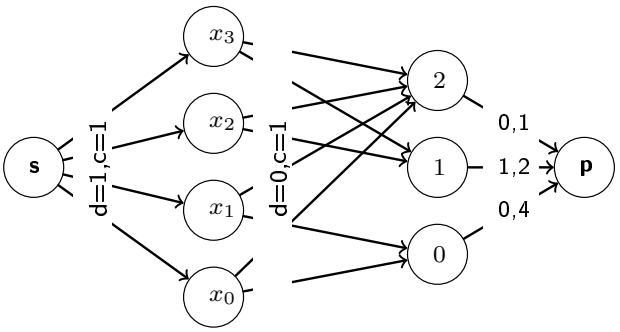


FIGURE 4.6 – *Exemple d'un réseau de transport représentant une contrainte* Gcc

4.1.3.d Consistance d'un Gcc

Comme l'indique l'Algorithme 3, tester la consistance d'une contrainte Gcc revient à rechercher l'existence d'un flot réalisable dans le réseau de transport associé (Régin, 1996).

Algorithme 3 : Test de consistance de la contrainte Gcc$(\mathcal{X}, lb, ub)$

```
1 début
2 |  G ← reseau2Transport(X, lb, ub)
3 |  flot ← calculerFlot(G)
4 |  si flot est réalisable alors Retour true
5 |  sinon Retour false
6 fin
```

Une valeur v_k d'une variable x_i est considérée comme étant inconsistante avec la contrainte Gcc si et seulement si :

- la valeur du flot $f(x_i \to v_k)$ vaut 0,
- x_i et v_k appartiennent à des composantes fortement connexes [2] différentes du graphe résiduel associé au réseau de transport de la contrainte Gcc.

Dans la section suivante, nous présentons une version relaxée de la contrainte Gcc.

4.1.4 La contrainte globale de cardinalité relaxée Soft-gcc

Très souvent, les problèmes du monde réel sont sur-contraints c'est-à-dire qu'ils ne possèdent pas de solutions. Dans ces conditions, l'objectif est de satisfaire toutes

2. Dans un graphe orienté, une composante fortement connexe, est un sous-ensemble maximal de nœuds tel que pour tout couple de nœuds x, y, il existe un chemin de x à y et de y à x.

les contraintes dures et de minimiser l'agrégation des coûts induit par la violation des préférences. Dans ce cas, on parle de relaxation de contraintes. Le but de cette relaxation n'est pas de satisfaire toutes les contraintes mais de les satisfaire au mieux.

Pour une contrainte globale, il existe plusieurs sémantiques de violation. Dans ce manuscrit, nous ne présenterons que la sémantique de violation orientée sur les variables. Le lecteur peut se référer à Hoeve et al. (2006) pour plus de détails sur ce point.

4.1.4.a La violation orientée sur les variables μ_{var}

La sémantique de violation orientée variables μ_{var}, pour une contrainte globale, permet de déterminer le nombre de variables pour lesquelles les valeurs doivent être ré-instancier afin de satisfaire la contrainte globale dure associée.

Pour ce faire, on définit une *fonction d'excès overflow* et une *fonction de manque underflow*.

$$overflow(\mathcal{X}, v_k) = max(|\{x_i | x_i = v_k\}| - ub(v_k), 0) \tag{4.1}$$
$$underflow(\mathcal{X}, v_k) = max(lb(v_k) - |\{x_i | x_i = v_k\}|, 0) \tag{4.2}$$

où lb et ub sont respectivement la borne inférieure et la borne supérieure de chacune des valeurs.

Comme l'indique l'équation 4.1, la *fonction d'excès*, mesure le nombre de variables pour lesquelles il faudra **dé-instancier** la valeur v_k afin de satisfaire la borne supérieure $ub(v_k)$. L'équation 4.2 indique la *fonction de manque*. Cette dernière mesure le nombre de variables pour lesquelles il faudra **ré-instancier** la valeur à v_k afin de satisfaire la borne inférieure $lb(v_k)$.

Pour une contrainte Gcc telle que $\sum_{v_k \in \bigcup \mathcal{D}_i} lb(v_k) \leq |\mathcal{X}| \leq \sum_{v_k \in \bigcup \mathcal{D}_i} ub(v_k)$, la mesure de violation μ_{var} est exprimée par :

$$\mu_{var}(\mathcal{X}) = max\left(\sum_{v_k \in \bigcup \mathcal{D}_i} overflow(\mathcal{X}, v_k), \sum_{v_k \in \bigcup \mathcal{D}_i} underflow(\mathcal{X}, v_k) \right) \tag{4.3}$$

Ainsi, cette sémantique de violation permet de mesurer la somme des écarts aux bornes lb et ub pour chacune des valeurs $v_k \in \bigcup \mathcal{D}_i$.

4.1.4.b Contrainte globale Soft-gcc

La contrainte globale Soft-gcc$(\mathcal{X}, lb, ub, \mu_{var}, z)$ est une version « relaxée » de la contrainte Gcc$(\mathcal{X}, lb, ub)$ avec μ_{var}, la mesure de violation orientée variable et z une variable de coût associée à un domaine fini D_z.

Pour représenter cette mesure de violation, on adapte le réseau de transport $G = (V, E)$ de la contrainte Gcc$(\mathcal{X}, lb, ub)$ (cf. section 4.1.3.c). Cette adaptation consiste à rajouter un ensemble d'arcs de violation E_p à G tel que :

$$E_p \quad = \quad \{(v_k \rightarrow v_l) | v_k, v_l \in \bigcup \mathcal{D}_i, k \neq l\} \tag{4.4}$$

À chaque couple de valeurs (v_k, v_l), on associe un arc de violation $(v_k \rightarrow v_l) \in E_p$. Cet arc permet de modéliser la ré-instanciation à v_l d'une variable initialement

assignée v_k. Pour chacun de ces arcs, la demande d, la capacité c et le poids w sont définis comme suit :

$$\text{si } e \in E_p \qquad d(e) = 0, c(e) = n \text{ et } \mathsf{w}(e) = 1$$
$$\text{avec } n \text{ le nombre total de variable.}$$

Ainsi, la quantité de flot passant par un arc de violation $(v_k \rightarrow v_l)$ correspond au nombre de variables ayant initialement pour valeur v_k et qui devront être ré-instanciées. La fonction de coût associée à chaque contrainte Soft-gcc$(\mathcal{X}, lb, ub, \mu_{var}, z)$ est $W = \mu_{var}(\mathcal{X}) \times z$.

4.1.4.c Consistance d'un Soft-gcc

Tester la consistance d'une contrainte Soft-gcc$(\mathcal{X}, lb, ub, \mu_{var}, z)$ revient à rechercher l'existence d'un flot réalisable de valeurs n dans le réseau de transport associé de poids $\mathsf{weight}(f)$ inférieur ou égal à $max(D_z)$. Pour plus de détail sur l'algorithme de relaxation le lecteur peut se référer à Hoeve et al. (2006).

4.1.5 La contrainte globale Same

La contrainte globale Same (Beldiceanu et al., 2004) est une contrainte définie sur deux séquences de variables $\mathcal{X}$ et $\mathcal{X}'$. Cette contrainte impose que l'une des séquences de variables soit une permutation de l'autre. Autrement dit, l'une des séquences de variables utilise les mêmes valeurs que celles utilisées dans l'autre.

Définition 4.6 (Contrainte globale Same$(\mathcal{X}, \mathcal{X}')$ ()) *Soient $\mathcal{X} = \{x_0, \cdots, x_n\}$ et $\mathcal{X}' = \{x'_0, \cdots, x'_n\}$ deux séquences de variables de tailles identiques $|\mathcal{X}| = |\mathcal{X}'|$, dont les domaines sont respectivement $\mathcal{D}_0, \ldots, \mathcal{D}_n$ et $\mathcal{D}'_0, \ldots, \mathcal{D}'_n$. Notons $\bigcup_{i=0}^{n} v_i$ et $\bigcup_{i=0}^{n} v'_i$ deux multi-ensembles représentant les affectations de valeurs aux variables $\mathcal{X}$ et $\mathcal{X}'$. La contrainte globale* Same$(\mathcal{X}, \mathcal{X}')$ *admet une solution si et seulement si il existe une affectation telle que :*

$$\text{Same}(\mathcal{X}, \mathcal{X}') = \left\{ (v_0, \ldots, v_n, v'_0, \ldots, v'_n) | v_i \in \mathcal{D}_i, v'_i \in \mathcal{D}'_i \wedge \bigcup_{i=1}^{n} v_i = \bigcup_{i=0}^{n} v'_i \right\}$$

L'égalité $\bigcup_{i=1}^{n} v_i = \bigcup_{i=0}^{n} v'_i$ de la définition 4.6 stipule que dans une solution de Same$(\mathcal{X}, \mathcal{X}')$, le nombre d'occurrences de chaque valeur v_i doit être strictement identique aussi bien pour $\mathcal{X}$ que pour $\mathcal{X}'$.

4.1.5.a Représentation de la contrainte

L'approche générale de représentation d'une contrainte Same est proche de celle de la contrainte Gcc. Beldiceanu et al. (2004) proposent de représenter une contrainte Same$(\mathcal{X}, \mathcal{X}')$ par un réseau de transport $G = (V, E)$. A la différence de Gcc, le réseau de transport est constitué d'un graphe de valeurs $G_t = (\mathcal{X} \cup \mathcal{X}' \cup Y, E)$ bipartie auquel on rajoute les nœuds source s et puits p. Ainsi, l'ensemble des nœuds $V = \mathcal{X} \cup \mathcal{X}' \cup Y$ avec $\mathcal{X}$ et $\mathcal{X}'$ les séquences de variables $\{x_i\}$ et $\{x'_i\}$. L'ensemble des valeurs Y appartenant à $\mathcal{D}_\mathcal{X} \cap \mathcal{D}'_\mathcal{X}$, où $\mathcal{D}_\mathcal{X}$ et $\mathcal{D}'_\mathcal{X}$ désignent respectivement l'union des domaines de variables x_i et x'_i.

Le graphe G est défini comme suit :

$$\begin{aligned} V &= \{s,p\} \cup \mathcal{X} \cup \mathcal{X}' \cup Y \\ E &= E_s \cup E_{\mathcal{X}} \cup E_Y \cup E_p \text{ avec} \\ & E_s = \{(s \to x_i) | x_i \in \mathcal{X}\} \\ & E_{\mathcal{X}} = \{(x_i \to v_k) | x_i \in \mathcal{X}, v_k \in \mathcal{D}_{\mathcal{X}} \cap Y\} \\ & E_Y = \{(v_k \to x'_i) | x'_i \in \mathcal{X}', v_k \in \mathcal{D}_{\mathcal{X}'} \cap Y\} \\ & E_p = \{(x'_i \to p) | x'_i \in \mathcal{X}'\} \end{aligned}$$

La demande d et la capacité c d'un arc e sont définies comme suit.

$$\begin{aligned} \text{si } e \in E_s \quad & d(e) = 1 \text{ et } c(e) = 1 \\ \text{si } e \in E_{\mathcal{X}} \quad & d(e) = 0 \text{ et } c(e) = 1 \\ \text{si } e \in E_Y \quad & d(e) = 0 \text{ et } c(e) = 1 \\ \text{si } e \in E_p \quad & d(e) = n \text{ et } c(e) = n \text{ avec } n = |\mathcal{X}| = |\mathcal{X}'| \end{aligned}$$

De cette manière on garantit qu'une unité de flot passera par chacun des nœuds $\mathcal{X} \cup \mathcal{X}'$. Cela implique que ces variables ne seront affectées qu'à une et une seule valeur. La propriété de conservation des flot permettra d'assurer que le nombre d'occurrence de chaque valeur dans $\mathcal{X}$ est identique au nombre d'occurrence de la même valeur dans $\mathcal{X}$.

Exemple 4.3 *Considérons deux séquences de quatre variables* $\mathcal{X} = \{x_0, x_1, x_2, x_3\}$ *et* $\mathcal{X}' = \{x'_0, x'_1, x'_2, x'_3\}$.

Les domaines de ces variables sont définis par : $\mathcal{D}_0 = \mathcal{D}_1 = \{0, 2\}$, $= \mathcal{D}_2 = \{1, 2\}$, $= \mathcal{D}_3 = \{0, 1, 2\}$, $\mathcal{D}'_0 = \mathcal{D}'_2 = \{0, 1, 2\}$, $= \mathcal{D}_1 = \{1, 2\}$, $= \mathcal{D}_3 = \{0, 2\}$. *On défini* Same$(\{x_0, x_1, x_2, x_3, x'_0, x'_1, x'_2, x'_3\})$

La Figure 4.7 illustre la représentation du réseau de transport représentant la contrainte Same de l'exemple 4.3.

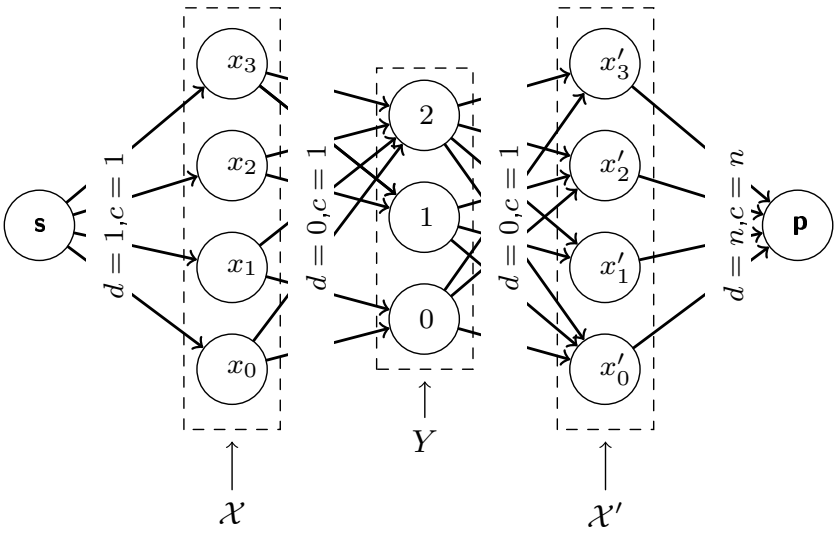


Figure 4.7 – *Example d'un réseau de transport représentant une contrainte* Same

4.1.5.b Consistance d'un Same

De la même manière que pour la contrainte Gcc, tester la consistance d'une contrainte Same revient à rechercher l'existence d'un flot réalisable dans le réseau

de transport associé (Beldiceanu et al., 2004).

Dans la suite, nous décrivons les contraintes prises en compte pour la résolution du problème de décision stratégique. Cette description nous permettra d'aborder dans la section suivante la formalisation du problème dans le cadre WCSP.

4.2 Description détaillée des contraintes pour la planification stratégique

4.2.1 Caractéristiques et hypothèses de notre approche de résolution

4.2.1.a Rappel sur les niveaux d'organisation d'une exploitation

Comme l'indique la Figure 4.8 et conformément à la description présentée dans le Chapitre 1, l'allocation de cultures dépend des niveaux d'organisations structurels et fonctionnels de l'exploitation. Ces niveaux d'organisation intègrent : l'exploitation globale, les îlots structurels, les zones cultivables, les blocs fonctionnels, les parcelles (cf. chapitre 1 pour plus de détails). Cependant, la taille de parcelle change chaque année. Il est donc nécessaire d'introduire le concept de parcelle élémentaire qui est défini comme suit :

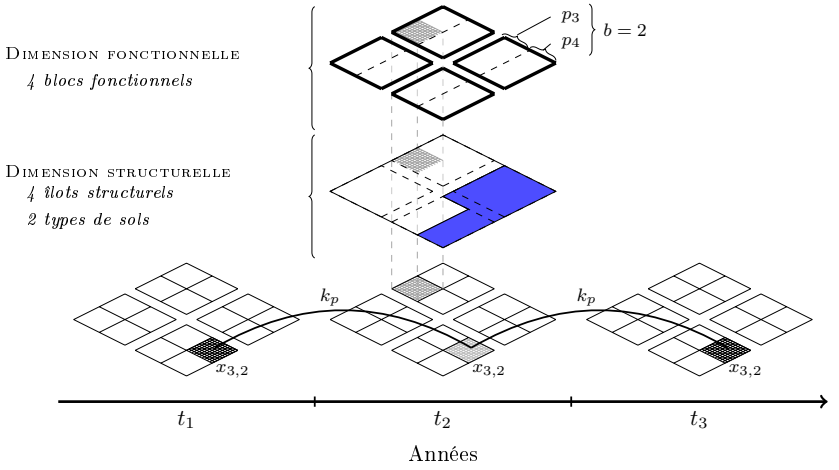


Figure 4.8 – *Représentation schématique des concepts spatio-temporels du problème de décision (t_i : année, b : bloc, p_j : parcelle, $x_{b,i}$: parcelle élémentaire, k_p : effet précédent)*

Définition 4.7 (Parcelle élémentaire) *Une parcelle élémentaire est une entité spatiale homogène, indivisible, ayant les mêmes historiques et les mêmes propriétés biophysiques.*

Ainsi, soient un ensemble de cultures et un ensemble de parcelles élémentaires $x_{b,i}$ où i est l'identifiant de la parcelle élémentaire du bloc fonctionnel b. Nous considérons que l'allocation de culture est réalisée sur un horizon fixe $\mathcal{H}$ durant lequel les cultures sont affectées aux parcelles élémentaires $x_{b,i}$.

Proposition 4.1 (Considération explicite de deux unités de gestion) *Sous l'hypothèse que la structure des blocs fonctionnels reste fixe durant tout l'horizon de planification, nous proposons de ne considérer que les* parcelles élémentaires *et les* blocs fonctionnels*, pour la formulation du problème.*

L'exploitation agricole, les îlots structurels et les parcelles sont pris en compte

via les contraintes topologiques qu'ils imposent au niveau de l'exploitation. Du fait de la dynamique temporelle du concept de parcelle, celle-ci est utilisée comme un élément d'évaluation de la qualité d'une allocation. La délimitation des parcelles sera ainsi adaptée chaque année dans l'optique de respecter l'équilibre spatial inhérent aux proportions annuelles des cultures à produire.

4.2.1.b Hypothèse d'homogénéité surfacique des parcelles élémentaires

Considérant l'hypothèse 4.1, et la définition 4.7, nous supposons dans la suite de ce manuscrit que les exploitations considérées peuvent être échantillonnées en parcelles élémentaires de tailles équivalentes. Cela suppose qu'il est possible de trouver une taille minimale des parcelles élémentaires pour lesquelles toutes les entités spatiales issues d'un redécoupage ont les mêmes historiques et les mêmes propriétés bio-physiques.

Hypothèse 4.1 (Échantillonnage de l'exploitation) *Soient b, b' deux blocs fonctionnels et i, i' deux parcelles élémentaires appartenant respectivement au bloc b et b'. Les superficies des parcelles élémentaires i et i' sont considérées comme étant équivalentes en fonction de la taille de l'exploitation.*

4.2.2 Description des contraintes

Nous avons présenté les contraintes de l'agriculteur dans les sections 1.2.2.c et 1.6.4.a. Pour faciliter la modélisation, nous nous basons sur les travaux existants afin de mieux caractériser ces différentes contraintes. Cette caractérisation nous servira de base pour la définition des classes de contraintes à prendre en compte.

4.2.2.a Classification des contraintes et préférences

Le choix des contraintes et préférences modélisées est basé sur une récente revue de la littérature, réalisée par Dury et al. (2011). Nous préfixons respectivement les contraintes dures et les préférences par les symboles h («) hard » ») et s (« soft »). Les contraintes retenues (cf. Table 4.1) peuvent être classées suivant deux axes (*espace* et *temps* - cf. *colonne 1* de la Table 4.1) en fonction de l'impact qu'elles ont sur l'organisation spatiale ou temporelle de l'exploitation agricole et selon les facteurs (*colonne 2*) et caractéristiques (*colonne 3*) qu'elles permettent de prendre en compte.

Les contraintes considérées permettent d'appréhender les trois facteurs principaux d'une allocation de culture. Ces facteurs sont relatifs aux exigences Agronomiques, Organisationnels et aux Objectifs de production. Suivant ces facteurs, différentes contraintes sont définies. Ces contraintes vont permettre de caractériser la prise en compte des *unités de gestions* (parcelles, îlots structurels, blocs fonctionnels, exploitation), des *capacités de ressources* et des *règles de succession* des cultures. Chacune des contraintes est définie soit de manière dure, soit de manière souple (préférence).

TABLE 4.1 – *Description des contraintes et préférences prises en compte*

	Facteurs pris en compte	**Caractéristiques**	**Noms des contraintes et préférences**	**Impact spatial**	**Type des contraintes**
Espace	AGRONOMIQUE	Unité de gestion	*h-SCC* - Zone cultivable	Parcelle élémentaire	*Dure*
	ORGANISATIONNEL		*h-EQU* - Interdépendance entre parcelles élémentaires	Parcelle	*Dure*
			s-TOP - Topologie	Ilots structurels	*Préférence*
	OBJECTIF		*s-SBC* - Proportion annuelle par culture	Exploitation	*Préférence*
	ORGANISATIONNEL	Ressources	*h-RSC* - Capacité de ressources	Exploitation	*Dure*
Temps	AGRONOMIQUE	Règles de succession	*h-HST* - Historique	Parcelle élémentaire	*Dure*
			h-TSC - Délai de retour/Itération	Bloc fonctionnel	*Dure*
			h-CCS - Rotation culturale	Bloc fonctionnel	*Dure*
			s-CSQ - Effet précédent	Parcelle élémentaire	*Préférence*
		Unité de gestion	*h-SCA* - Collection de cultures par bloc	Bloc fonctionnel	*Dure*
	OBJECTIF		*s-TBC* - Proportion de culture par parcelle élémentaire	Parcelle élémentaire	*Préférence*

4.2.2.b Contraintes et préférences agronomiques

▷ *Zone cultivable - h-SCC :* cette contrainte dure définit si une culture donnée peut être produite sur une parcelle élémentaire. Les raisons qui justifient la compatibilité spatiale de la culture peuvent être très variables. Les contraintes de zone cultivable peuvent être dues au type de sol (ex : une culture donnée ne peut être produite que sur certains types de sol), à la disponibilité de ressources spécifiques à la production de la culture (ex : une culture irriguée telle que le maïs ne peut être produite que sur des parcelles élémentaires ayant un accès à une ressource en eau). La contrainte de zone cultivable peut être associée sans aucune restriction à chacune des parcelles élémentaires de l'exploitation.

▷ *Historique des parcelles élémentaires - h-HST :* cette contrainte dure définit pour chaque parcelle élémentaire, l'occupation du sol durant les précédentes années. L'objectif de la contrainte d'historique des parcelles élémentaires est de prendre en compte les assolements passés qui contraignent les allocations possibles dans le futur.

▷ *Délai de retour et itération de cultures - h-TSC :* cette contrainte dure définit le délai de retour et les itérations (cf. Définition 1.6) de cultures pour chaque couple parcelle élémentaire/culture. Cette contrainte doit permettre de respecter les délais de retour en prenant en compte les valeurs d'historique tout en autorisant les éventuelles violations de ce délai de retour dans le passé. La contrainte de délai de retour et itération est associée sans aucune restriction et distinction à chacune des parcelles élémentaires d'un bloc fonctionnel.

▷ *Rotation culturale - h-CCS :* cette contrainte dure est définit pour chacune des parcelles élémentaires d'un bloc fonctionnel pour lequel une rotation culturale est imposée (cf. Définition 1.8). Ainsi, la contrainte de rotation culturale permet de concevoir des séquences indéfiniment répétables sans enfreindre la contrainte de délai de retour.

▷ *Effet précédent - s-CSQ :* cette contrainte est une préférence de l'agriculteur qui permet de favoriser une séquence de cultures par rapport à une autre. La contrainte d'effet précédent (cf. Définition 1.5) porte sur une succession de deux cultures. Elle agrège l'effet d'une culture sur la suivante en prenant en compte différentes caractéristiques telles que la dégradation de la structure du sol, les maladies, les résidus de pesticide, la quantité d'azote et les mauvaises herbes. Cette contrainte est associée sans aucune restriction et distinction à chacune des parcelles élémentaires de l'exploitation.

▷ *Collection de cultures par bloc - h-SCA :* considérant que dans un bloc fonctionnel (cf. Définition 1.9) la séquence de cultures doit être applicable sur toutes les parcelles élémentaires, la contrainte de collection de cultures par bloc fonctionnels est une contrainte dure qui définit les séquences admissibles sur un bloc fonctionnel. Ainsi, sur l'horizon de planification, à l'exception de l'historique, le sous-ensemble de cultures individuellement assigné aux parcelles élémentaires d'un bloc fonctionnel doit être strictement le même.

4.2.2.c Contraintes et préférences organisationnelles

▷ *Interdépendance entre parcelles élémentaires - h-EQU :* cette contrainte dure définit de manière statique la nature de l'allocation attendue pour une parcelle. La contrainte d'interdépendance entre des parcelles élémentaires permet de préciser si des découpages spatiaux sont interdits sur une parcelle.

Les raisons de cette interdiction sont essentiellement liées à l'organisation du travail souhaitée par l'agriculteur. Une raison possible pourrait être l'éloignement de la parcelle par rapport au siège de l'exploitation.

▷ *Topologie de l'exploitation - s-TOP :* cette contrainte est une préférence de l'agriculteur. Elle favorise les organisations spatiales de l'exploitation pour lesquelles les parcelles élémentaires affectées à une même culture sont groupées. La motivation pour ce regroupement spatial est liée à la conduite journalière. Ainsi, il est préférable de grouper les cultures au sein d'un îlot structurel pour limiter la perte de temps induit aux déplacements entre les parcelles élémentaires durant la conduite journalière de la culture. Les préférences sur la topologie de l'exploitation sont associées à chaque îlot structurel.

▷ *Capacité de ressources - h-RSC :* l'exploitation agricole dispose de quantités fixes de ressources consommables nécessaires à la production des cultures. Il s'agit ici de prendre en compte des ressources telles que le volume d'eau destiné à l'irrigation sur un bloc fonctionnel, la ressource humaine à l'échelle de l'exploitation etc. La contrainte de capacité de ressources est une contrainte dure qui détermine pour chaque ressource considérée la limite de consommation à respecter en accumulant les quantités nécessaires à la production des cultures affectées à chaque parcelle élémentaire. Cette contrainte est associée à un ou plusieurs blocs fonctionnels.

4.2.2.d Contraintes et préférences liées aux objectifs

▷ *Proportion annuelle par culture - s-SBC :* cette contrainte est une préférence de l'agriculteur qui définit pour chaque culture considérée une tranche annuelle de la proportion de l'exploitation associée à la culture considérée. La détermination de cette proportion peut résulter d'un contrat sur les cultures considérées, d'une production nécessaire au fonctionnement de l'exploitation (ex : alimentation du bétail) etc. En conséquence, pour chacune des cultures considérées la somme des superficies des parcelles élémentaires doit être comprise dans un intervalle de proportions admissibles.

▷ *Proportion de cultures par parcelle élémentaire - s-TBC :* cette contrainte est une préférence de l'agriculteur qui définit pour chaque culture considérée une tranche de proportion à réaliser sur chaque parcelle élémentaire tout au long de l'horizon de planification à l'exception de l'historique. Il s'agit en particulier de respecter certaines proportions afin de ne pas affecter la fertilité des sols. En conséquence, pour chaque paire culture/parcelle élémentaire considérée, la somme sur l'horizon de planification, à l'exception de l'historique, des superficies des parcelles élémentaires doit être comprise dans un intervalle de proportions admissibles.

4.2.3 Variables et domaines du problème d'allocation de cultures

Le problème d'allocation de cultures est défini sur un horizon fini $\mathcal{H}$ par un ensemble de parcelles élémentaires et de cultures. Nous définissons le WCSP (cf. section 2.1.2) associé dans les sous sections suivantes.

4.2.3.a Les variables $\mathcal{X}$ du WCSP

L'ensemble $\mathcal{X}$ est constitué des variables $x_{b,i}^t \in \mathcal{X}$. Chaque variable $x_{b,i}^t$ définit l'occupation de la parcelle élémentaire i du bloc fonctionnel b à la date t avec

$i \in [1, \mathcal{N}_b]$, $b \in [1, \mathcal{B}]$ et $t \in [1, \mathcal{H}]$. Considérons $[1, h]$ et $[h+1, \mathcal{H}]$ respectivement les années d'historique et celles du futur. Chaque parcelle élémentaire i du bloc b sera représentée par $\mathcal{H}$ variables correspondant à l'occupation de la parcelle élémentaire à chaque instant.

4.2.3.b Le domaine $\mathcal{D}$ des variables

Les domaines $D_{b,i}$ des variables $x^t_{b,i}$ est l'ensemble des cultures possibles sur toutes les parcelles élémentaires i des blocs fonctionnels b. Ainsi, toutes les variables sont associées à un seul et même domaine.

4.2.3.c Les fonctions de coût $\mathcal{W}$

Les fonctions de coûts $\mathcal{W}$ sont définies sur la base des contraintes présentées dans le tableau 4.1. Nous avons choisi pour la suite, de regrouper ces contraintes en fonction des facteurs qu'elles permettent de modéliser. Les contraintes relatives aux facteurs agronomiques sont abordées dans la section 4.3. Ensuite, les contraintes organisationnelles sont présentées dans la section 4.4. Enfin, les contraintes liées aux objectifs de production de l'agriculteur sont formalisées dans la section 4.5. Ces contraintes sont formalisées en utilisant cinq différents types de contraintes dures et préférences qui sont :

- des *fonctions de coût tabulaires* dont l'arité maximale dépend de la parcelle ayant le plus grand nombre de voisins.
- des contraintes globales same,
- des contraintes globales regular,
- des contraintes globales de cardinalité gcc,
- des contraintes globales de cardinalité relaxée soft-gcc.

4.3 Formalisation des contraintes agronomiques

4.3.1 Contraintes de zone cultivable - h-SCC

La contrainte de zone cultivable h-SCC détermine si une culture est réalisable ou pas sur une parcelle élémentaire. Elle est définie par une réduction de domaine pour les variables du futur $x^t_{b,i}$. Ainsi, $\forall t \in [h+1, \mathcal{H}]$, $\forall b \in \mathcal{B}$, $\forall i \in \mathcal{N}_b$, on associe à chaque couple variable $x^t_{b,i}$ et culture $a \in D_{b,i}$, une fonction de coût unaire associée $W^{SCC}_{x^t_{b,i}}$ telle que :

$$\forall a \in D_{b,i}, W^{SCC}_{x^t_{b,i}}(a) \quad = \quad \begin{cases} \infty & \textbf{si } a \textit{ est interdit sur la} \\ & \textit{parcelle élémentaire } i \textit{ du bloc } b \\ 0 & \textbf{sinon} \end{cases} \tag{4.5}$$

4.3.2 Historique des parcelles élémentaires - h-HST

Chacune des parcelles élémentaires est associée à h valeurs d'historique $\mathsf{historic}(x^t_{b,i})$ avec, $\mathsf{historic}(x^t_{b,i})$ la valeur de la variable d'historique $x^t_{b,i}$ de la parcelle élémentaire i du bloc fonctionnel b à la date t. Toutes les autres valeurs à l'exception des valeurs $\mathsf{historic}(x^t_{b,i})$ sont ainsi considérées comme interdites. En conséquence une allocation n'est consistante que si pour chacune des variables décrivant l'historique d'une parcelle élémentaire, l'affectation de $x^t_{b,i}$ ($t \in [1, h]$) coïncide avec la valeur $\mathsf{historic}(x^t_{b,i})$.

Ainsi, $\forall b \in \mathcal{B}$, $\forall i \in \mathcal{N}_b$, $\forall t \in [1, h]$, soit $W^{HST}_{x^t_{b,i}}$ une fonction de coût unaire associée à une variable d'historique de parcelles élémentaires.

$$\forall a \in D_{b,i}, W^{HST}_{x^t_{b,i}}(a) \quad = \quad \begin{cases} 0 & \text{si } a = \mathsf{historic}(x^t_{b,i}) \\ \infty & sinon \end{cases} \tag{4.6}$$

4.3.3 Délai de retour - h-TSC

4.3.3.a Modélisation

Pour modéliser le délai de retour (h-TSC) nous proposons de voir les séquences de cultures sur une parcelle élémentaire comme des *mots*. Pour construire ces mots, il nous faut un *alphabet* et un *langage*. L'alphabet utilisé est représenté par l'ensemble de cultures. Le langage « délai de retour » ne possède qu'une seule règle (règle de succession des cultures) qui stipule que pour un mot donné, deux instances consécutives d'un même élément a de l'alphabet (culture) doivent être séparées d'au moins $dr(a) - 1$ éléments de l'alphabet, différents de a, ($dr(a)$ étant le délai de retour de a.

Sachant que les mots pris en compte sont de tailles fixes, nous pouvons représenter h-TSC en utilisant des contraintes globales regular proposées par Pesant (2004) (cf. section 4.1.2). Les variables sur lesquelles portent la contrainte regular représentent la succession temporelle de $\mathcal{H}$ variables décrivant la parcelle élémentaire i du bloc b sur tout l'horizon de planification. Ainsi, $\forall t \in [1, \mathcal{H}]$, $\forall b \in \mathcal{B}$, $\forall i \in \mathcal{N}_b$, $\forall a \in D_{b,i}$, on associe un automate à états finis déterministe (DFA) $M^a_{b,i}$ à la contrainte regular. Soient $\mathcal{L}(M^a_{b,i})$ le langage défini par l'automate $M^a_{b,i}$ et $S_{b,i}$ la succession temporelle de $\mathcal{H}$ variables décrivant une parcelle élémentaire. Une solution de $\mathsf{regular}(S_{b,i}, M^a_{b,i})$ est une affectation $A[S_{b,i}]$ telle que $A[S_{b,i}] \in \mathcal{L}(M^a_{b,i})$.

En pratique, la règle de succession définissant le langage $\mathcal{L}(M^a_{b,i})$ pourrait ne pas être respectée sur la séquence de variables décrivant l'historique. Autrement dit, l'automate $M^a_{b,i}$ doit pouvoir accepter la violation du délai de retour sur les variables de l'historique tout en l'imposant sur les variables du futur.

Nous définissons pour chaque paire parcelle élémentaire i/culture a, une fonction de coût $W^{TSC^a}_{S_{b,i}}$ d'arité $\mathcal{H}$ telle que :

$$\begin{aligned} &\forall b \in \mathcal{B}, \forall i \in \mathcal{N}_b, \forall a \in D_{b,i} \\ &\qquad W^{TSC^a}_{S_{b,i}} \quad = \quad \mathsf{regular}(x^1_{b,i}, \cdots, x^t_{b,i}, \cdots, x^{\mathcal{H}}_{b,i}, M^a_{b,i}) \end{aligned} \tag{4.7}$$

4.3.3.b Construction de l'automate

L'algorithme 4 présente la construction de l'automate pour une culture a en fonction du délai de retour $dr(a)$, de la taille de l'historique h, et du domaine $D_{b,i}$ des variables définissant la parcelle élémentaire i du bloc b. Le nombre d'états de l'automate (ligne 3 de l'algorithme 4) est une fonction du délai de retour ($dr(a)$) et de l'historique. On construit dans un premier temps les états d'un automate dans lequel les nœuds sont numérotés de 0 à nbstate-1. L'état initial de l'automate est 0. Les états terminaux sont tous ceux qui sont compris entre $[h, h + dr(a) - 1]$.

Dans la première partie (lignes 7 à 9) de l'algorithme, on ajoute toutes les transitions de l'historique. Entre deux états consécutifs on rajoute entre $|D_{b,i}| - 1$ et $|D_{b,i}|$ arcs en fonction de son impact sur la satisfaction du délai de retour.

Dans la deuxième partie (lines 10 à 16) on construit sur les $dr(a)$ suivant l'historique un automate imposant les délais de retour. À l'exception de la première,

toutes les transitions entre deux états successifs sont labellisées par toutes les cultures a exceptées.

Enfin dans la troisième partie de l'algorithme (lignes 17 à 26) on ajoute les arcs permettant de mettre en relation les occurrences de la culture a dans la phase d'historique et le contraintes liées à son apparition dans la suite de la séquence.

Algorithme 4 : Construction du langage $\mathcal{L}(M^a_{b,i})$ d'une séquence de culture

```
1  début
       input  : ⟨ a,D_{b,i}, h ⟩
2      si h ≥ dr(a) -1 alors m = dr(a)-1 sinon m = h
3      nbstate = dr(a) + h + m(m−1)/2
4      T = dr(a) + h- 1
5      Init(M^a_{b,i}, nbstate)  /* initialiser M^a_{b,i} avec nbstate états */
6      si dr(a) > 1 alors
7          ∀ i ∈ [0,h [, ∀ a' ∈ D_{b,i}/ a' ≠ a ajouterArc(M^a_{b,i}, i → i+1, a')
8          si h > dr(a) alors ∀ i ∈ [0,h- dr(a)+1[ ajouterArc(M^a_{b,i}, i →
           i+1, c)
9          si h > 0 alors ajouterArc(M^a_{b,i}, h-1 → h+2, c)
10         pour i = h to T-1 faire
11             si i = h alors
12                 ∀ a' ∈ D_{b,i}/ a' ≠ a ajouterArc(M^a_{b,i}, i → i, a')
13                 ajouterArc(M^a_{b,i}, h → h +1, a)
14             sinon
15                 ∀ a' ∈ D_{b,i}/ a' ≠ a ajouterArc(M^a_{b,i}, i → i+1, a')
16         ∀ a' ∈ D_{b,i}/ a' ≠ a ajouterArc(M^a_{b,i}, T → h, a')
17         pour k = 2 to m faire
18             ajouterArc(M^a_{b,i}, h- k → T +1, a)
19             pour i = 1 to k-1 faire
20                 si i ≠ k-1 alors
21                     ∀ a' ∈ D_{b,i}/ a' ≠ a ajouterArc(M^a_{b,i}, T +1 → T
                       +i+1, a')
22                     ajouterArc(M^a_{b,i}, T +i → T-(k-2)+i, a)
23                 sinon
24                     ∀ a' ∈ D_{b,i}/ a' ≠ a ajouterArc(M^a_{b,i}, T + i → h + k,
                       a')
25                     ajouterArc(M^a_{b,i}, T + i → h + 1, a)
26             T = T + k +1
27     retourner M^a_{b,i}
28 fin
```

4.3.3.c Exemple du colza d'hiver

Considérons l'exploitation virtuelle présentée dans la section 1.6. Cette exploitation respecte bien l'hypothèse d'homogénéité surfacique (cf. Hypothèse 4.1). La parcelle p_5 du bloc fonctionnel 2 peut être considérée comme élémentaire du point de vue de la définition 4.7. Comme l'indique le tableau 1.3 nous disposons pour cette parcelle

élémentaire des cinq dernières années d'historiques. Sachant que le délai de retour du colza d'hiver (CH) est de 3 ans, on définit une contrainte $\mathsf{regular}(S_{b,i}, M^{CH}_{b,i})$ où $b = 2$ et i l'indice de la parcelle p_5 dans le bloc 2.

L'automate non déterministe à états finis $M^{CH}_{b,i}$ est décrit par la Figure 4.9. L'état initial de l'automate est 0 (nœud gris) tandis que les états terminaux sont $5, 6$ et 7 (nœud double). Les arcs sont labellisés avec les éléments de l'alphabet c'est-à-dire les cultures. Le label a défini l'ensemble des valeurs de $D_{b,i}$. La notation $\overline{CH}$ correspond à $D_{b,i} \setminus \{CH\}$. Comme l'indique le DFA, les états d'historiques contraignent ceux du futur via les arcs de $3 \to 8$ et $4 \to 6$. L'automate autorise tous les mots ne respectant pas le délai de retour dans l'historique (nœuds en pointillés).

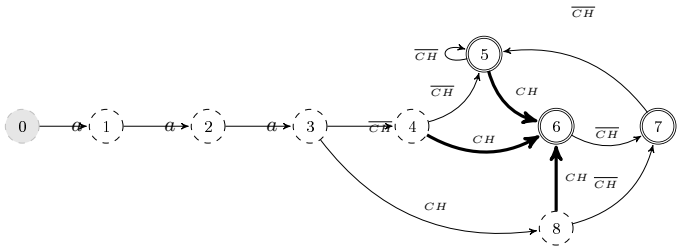


FIGURE 4.9 – *DFA décrivant le langage associé au délai de retour de la culture CH avec ($dr(CH) = 3$ et $h = 5$). a désigne toutes les valeurs de $D_{b,i}$. La notation $\overline{CH}$ correspond à $D_{b,i} \setminus \{CH\}$. Le langage associé accepte toutes les séquences sur $\mathcal{H}$ pour lesquelles le délai de retour est respecté pour l'ensemble des variables du futur (ex : CH-OP-CH-OP-CH-BH-OP-CH-BH).*

4.3.4 Rotation culturale - h-CCS

4.3.4.a Modélisation

Pour modéliser une rotation (cf. Définition 1.8), nous combinons la contrainte de délai de retour h-TSC (section 4.3.3) avec une contrainte de cyclicité (h-CCS). Cette contrainte est également définie par un ensemble de contraintes globales $\mathsf{regular}(S_{b,i}, M^{a}_{b,i})$ avec $S_{b,i}$ la séquence de $\mathcal{H} - h$ variables décrivant les variables du futur de la parcelle élémentaire i du bloc B. La contrainte h-CCS impose que la séquence des valeurs de $\mathsf{A}[S_{b,i}]$ appartienne au langage régulier $\mathcal{L}$ reconnu par l'automate $M^{a}_{b,i}$.

Nous définissons pour chaque paire parcelle élémentaire i/culture a, une fonction de coût $W^{CCS^a}_{S_{b,i}}$ d'arité $\mathcal{H} - h$ telle que :

$$\begin{aligned} &\forall b \in \mathcal{B}, \forall i \in \mathcal{N}_b, \forall a \in D_{b,i} \\ &\qquad W^{CCS^a}_{S_{b,i}} \quad = \quad \mathsf{regular}(x^{h+1}_{b,i}, \cdots, x^{\mathcal{H}}_{b,i}, M^{a}_{b,i}) \end{aligned} \tag{4.8}$$

4.3.4.b Construction de l'automate

Considérant une parcelle élémentaire, l'algorithme 5 permet de construire le langage pour une culture a en fonction du délai de retour $dr(a)$ et du domaine $D_{b,i}$. Le nombre d'états $\mathsf{nbstate}$ de l'automate (ligne 2 de l'algorithme 5) est égal à $dr(a)^2 + dr(a) + 1$. Ces états sont numérotés de 0 à $\mathsf{nbstate}$-1. L'état initial de l'automate est 0 tandis que les états terminaux sont tous ceux qui sont multiples de $dr(a)$ à l'exception de l'état 0.

L'automate écrivant le langage est constitué de $dr(a) + 1$ couches (ligne 4). Chacune des couches est constituée d'un automate (lignes 5 à 19) représentant

le délai de retour. La différence entre les automates se réfère à la position de la transition labellisée par la culture a.

Algorithme 5 : Construction du langage $\mathcal{L}(M^a_{b,i})$ d'une rotation

```
1  début
       input  : ⟨ a,D_{b,i} ⟩
2      nbstate = dr(a)^2 + dr(a) + 1
3      Init(M^a_{b,i}, nbstate)  /* initialiser M^a_{b,i} avec nbstate états */
       // Itération sur chaque couche i
4      pour i = 1 to dr(a)+1 faire
           // Construction de l'automate i
5          pour j = 1 to dr(a) faire
6              cur ← (i-1)dr(a) + j
7              si j ≠ 1 alors prec ← cur-1 sinon prec ← 0
               /* Ajout des transitions entre états en décalant
                  progressivement l'arc labellisé avec a          */
8              si i = j alors ajouterArc(M^a_{b,i}, prec → cur, a)
9              sinon ∀ a' ∈ D_{b,i}/ a' ≠ a ajouterArc(M^a_{b,i}, prec → cur, a')
               /* Traitement de l'état final de la couche i       */
10             si (cur % dr(a)) = 0 alors
11                 pour a' ∈ D_{b,i} faire
12                     si a' ≠ a alors
13                         ajouterArc(M^a_{b,i}, prec → cur, a')
14                         si i ≠ 1 and i ≤ dr(a) alors
15                             ajouterArc(M^a_{b,i}, cur → dr(a)*(i − 1) + 1, a')
16                     sinon si i=1 alors
17                         ajouterArc(M^a_{b,i}, cur → dr(a)*(i − 1) + 1, a)
18                     sinon si i=dr(a)+1 alors
19                         ajouterArc(M^a_{b,i}, cur → dr(a), a)
20     retourner M^a_{b,i}
21 fin
```

4.3.4.c Exemple du colza d'hiver

Considérons la parcelle p_5 du bloc fonctionnel 2 (exploitation virtuelle de la section 1.6). On définit la contrainte de rotation $W^{CCS^a}_{S_{b,i}}$ sur les variables du futur de la parcelle considérée. L'automate non déterministe à états finis présenté à la Figure 4.10 décrit le langage pour la culture CH dont le délai de retour est de 3 ans. L'état initial est 0 tandis que les état terminaux sont $3, 6, 9$ et 12.

La notation $\overline{CH}$ correspond à $D_{b,i} \setminus \{CH\}$. Dans ce langage, on retrouve bien les quatre couches (1 à 3), (4 à 6) à (7 à 9) et (10 à 12). Chacune d'elle impose que la séquence de valeurs $\{x^{h+1}_{b,i}, \cdots, x^{\mathcal{H}}_{b,i}\}$ appartienne au langage défini par la Figure 4.10. Les arcs en gras indiquent les transitions liées à une réalisation de CH.

4.3.5 Collection de cultures par bloc - h-SCA

En nous basant sur la définition 1.9, la cohérence d'un bloc fonctionnel dépend de deux principales conditions. D'une part, la même collection de cultures doit être

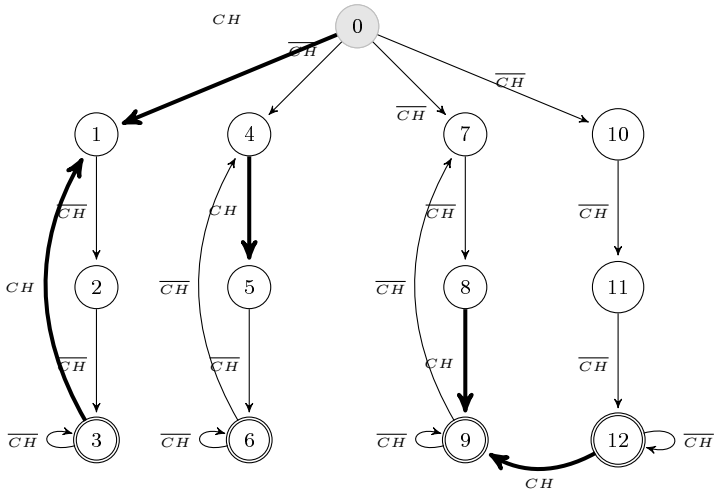


FIGURE 4.10 – *DFA décrivant le langage associé au délai de retour du colza d'hiver (CH) en mode rotation ($dr(CH) = 3$)*

produite sur toutes les parcelles élémentaires du bloc fonctionnel. D'autre part, la séquence de cultures doit être la même quelque soit la parcelle élémentaire du bloc considéré. Nous présentons dans les sous sections suivantes la modélisation de ces conditions.

4.3.5.a Modélisation

Pour toutes les parcelles élémentaires, le nombre de variables du futur est strictement identique. Considérons un bloc fonctionnel b. Par définition, les séquences de variables du futur $x^t_{b,i}$ (avec $t \in [h+1, \mathcal{H}]$) associées à chaque parcelle élémentaire i doivent toutes être assignées à une seule et même collection de cultures.

Nous pouvons modéliser la contrainte de collection de cultures par bloc en utilisant la contrainte globale same (cf. section 4.1.5). Cette contrainte se défini sur deux séquences de variables et impose que l'une des séquences de variables soit une permutation de l'autre.

Pour chaque bloc fonctionnel, on choisit une parcelle élémentaire de *référence* i. $\forall (i,j) \in \mathcal{N}_b \times \mathcal{N}_b$ (avec $i \neq j$) on définit ensuite une fonction de coût W^{SCA}_S d'arité $2 * (\mathcal{H} - h)$. La portée S de la fonction est déterminée par chaque couple de séquences de variables qui définissent $x^t_{b,i}$ et $x^t_{b,j}$ ($i \neq j$). L'ensemble des variables S sur lesquelles portent la contrainte est $\{x^{h+1}_{b,i}, \cdots, x^{\mathcal{H}}_{b,i}, x^{h+1}_{b,j}, \cdots, x^{\mathcal{H}}_{b,j}\}$. Soient $\mathsf{A}[x^{h+1}_{b,i}, \cdots, x^{\mathcal{H}}_{b,i}]$ et $\mathsf{A}[x^{h+1}_{b,j}, \cdots, x^{\mathcal{H}}_{b,j}]$ les deux sous affectations des variables de S. La contrainte W^{SCA}_S impose que $\mathsf{A}[x^{h+1}_{b,i}, \cdots, x^{\mathcal{H}}_{b,i}]$ soit une permutation de $\mathsf{A}[x^{h+1}_{b,j}, \cdots, x^{\mathcal{H}}_{b,j}]$.

$$W^{SCA}_S = \mathsf{same}(\underbrace{x^{h+1}_{b,i}, \cdots, x^{\mathcal{H}}_{b,i}}_{i}, \underbrace{x^{h+1}_{b,j}, \cdots, x^{\mathcal{H}}_{b,j}}_{j}) \tag{4.9}$$

4.3.5.b Illustration de la contrainte h-SCA

Illustrons la contrainte h-SCA en considérant deux parcelles élémentaires $x_{b,0}$ et $x_{b,1}$ du bloc fonctionnel b. Quatre cultures a_0, a_1, a_2 et a_3 sont productibles sur ce bloc. Chacune des parcelles élémentaires i est décrite par quatre variables du futur $x^{h+1}_{b,i}, x^{h+2}_{b,i}, x^{h+3}_{b,i}, x^{h+4}_{b,i}$. Considérons qu'en raison du type de sol, la culture a_0 ne puisse être produite sur la parcelle élémentaire 1. Le graphe biparti de la Figure 4.11

illustre cet énoncé. Les nœuds circulaires et carrés sont respectivement les séquences de variables et les cultures. Les arcs représentent les domaines des variables $x_{b,i}^t$. La collection de cultures productibles sur b revient à réduire le domaine des variables décrivant la parcelle élémentaire 0 en supprimant les arcs en pointillés puis à tester la consistance d'arc de la contrainte SAME suivant la technique présentée dans la section 4.1.5.

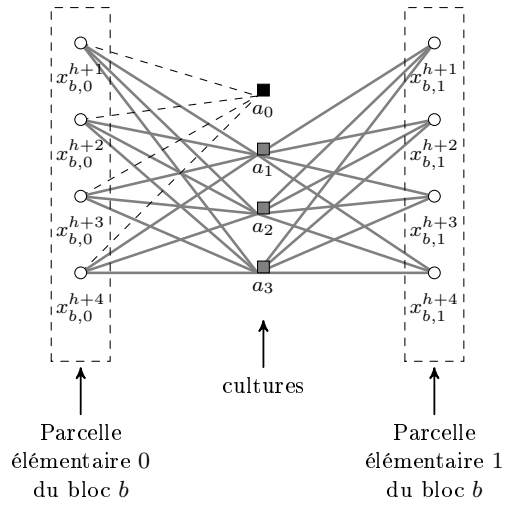


FIGURE 4.11 – *Représentation sous forme de graphes bipartis des collections de cultures par bloc fonctionnel*

4.3.6 Qualité des séquences de cultures : effet précédent - s-CSQ

L'indicateur agronomique k_p proposé par Leteinturier et al. (2006) permet d'agréger la variation de l'état du sol pour une succession culturale en prenant en compte la culture produite, la dégradation de la structure du sol, les maladies, les résidus de pesticide, d'azote et les mauvaises herbes. Cet indicateur représenté par une matrice de coût d'une succession culturale, permet de mesurer l'utilité d'une succession culturale par rapport à une autre. Plus le k_p est faible, plus l'enchaînement des cultures est meilleur. Ainsi, le but est de trouver les séquences de cultures qui optimisent la somme des k_p et qui quantifient la qualité d'une succession.

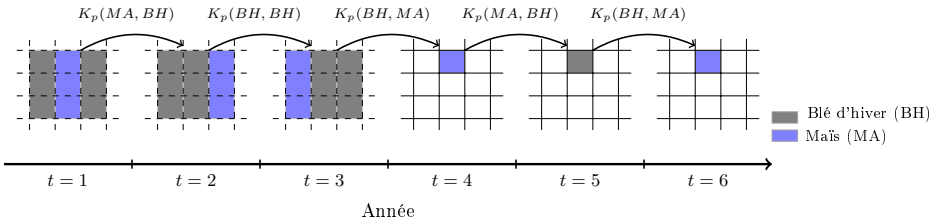


FIGURE 4.12 – *Qualité des successions culturales*

Par exemple, connaissant les allocations des trois années d'historiques ($\{1, 2, 3\}$ Figure 4.12) on cherche à trouver les affectations de cultures aux parcelles élémentaires sur les trois années du futur ($\{4, 5, 6\}$) qui, minimisent $\sum_{t=1}^{5} k_p(\mathsf{A}[x_{b,i}^t], \mathsf{A}[x_{b,i}^{t+1}])$ où $\mathsf{A} \in D_{\mathcal{X}}$ est une affectation complète et $\mathsf{A}[x_{b,i}^t]$ et $\mathsf{A}[x_{b,i}^{t+1}]$ la sous affectation de A respectivement sur les variables $x_{b,i}^t$ et $x_{b,i}^{t+1}$. Cet exemple montre la relation entre les successions culturales et le coût lié à la variation de l'état du sol. En se basant sur ces informations on peut estimer l'utilité d'une séquence de culture sur l'horizon de planification.

Ainsi, $\forall t \in [1, \mathcal{H}]$, $\forall b \in \mathcal{B}$, $\forall i \in \mathcal{N}_b$, soit $W^{CSQ}_{x^t_{b,i}, x^{t+1}_{b,i}}$ la fonction de coût binaire associée à la contrainte d'effet précédent s-CSQ. Nous définissons une fonction $K_p(a, a')$ qui retourne le coût k_p de la succession culturale a' précédé de a sur la parcelle élémentaire i.

$$\forall a \in D_{b,i}, \forall a' \in D_{b,i}, W^{CSQ}_{x^t_{b,i}, x^{t+1}_{b,i}}(a, a') \quad = \quad K_p(a, a') \tag{4.10}$$

4.4 Formulation des contraintes organisationnelles

4.4.1 Interdépendance entre parcelles élémentaires - h-EQU

La contrainte d'interdépendance entre des parcelles élémentaires (h-EQU) définit les sous-ensembles de parcelles pour lesquels la même valeur doit être assignée à chaque instant. Ainsi, $\forall t \in [h+1, \mathcal{H}]$, $\forall b \in \mathcal{B}$, pour tous les couples de parcelles élémentaires $(i, j) \in \mathcal{N}_b \times \mathcal{N}_b$ qui doivent être gérés de la manière, on définit une contrainte d'égalité $W^{EQU}_{x^t_{b,i}, x^t_{b,j}}$ sur les variables $x^t_{b,i}$ et $x^t_{b,j}$.

$$\forall a \in D_{b,i}, \forall a' \in D_{b,j}, W^{EQU}_{x^t_{b,i}, x^t_{b,j}}(a, a') \quad = \quad \begin{cases} 0 & si\ a = a' \\ \infty & sinon \end{cases} \tag{4.11}$$

4.4.2 Préférence topologique - s-TOP

Considérant un îlot structurel (cf. Définition 1.2), la contrainte de préférences topologiques vise à favoriser les assignations pour lesquelles les parcelles élémentaires ayant les mêmes cultures sont groupées. La Figure 4.13 montre la dynamique spatiale sur trois ans d'un îlot structurel sur lequel deux cultures sont produites (maïs et blé d'hiver) ; le reste de l'îlot étant en jachère. Cet îlot structurel est représenté par une grille dans laquelle les cellules sont les parcelles élémentaires. On observe dans ce cas un regroupement parfait des parcelles élémentaires en maïs et en blé d'hiver. En pratique, cette organisation n'est généralement pas réalisable pour de nombreuses raisons dues aux facteurs biophysiques ou aux contraintes agronomiques. Il s'agira dans ces conditions de privilégier les affectations qui optimisent les contours des parcelles (cf. Définition 1.1) c'est-à-dire qui maximisent la superficie des parcelles au sein de l'îlot structurel.

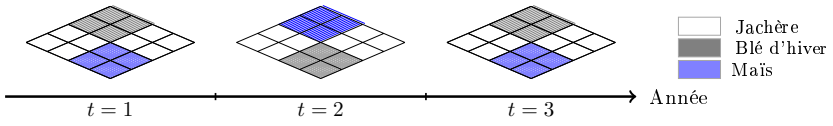


Figure 4.13 – *Exemple de préférence topologique sur un ilot structurel*

En exploitant la discrétisation de l'exploitation agricole nous pouvons assurer une localisation optimale du voisinage des parcelles élémentaires ainsi que l'unicité de la détection du contour des parcelles élémentaires.

Ainsi, $\forall t \in [1, \mathcal{H}]$, $\forall b \in \mathcal{B}$, $\forall i \in \mathcal{N}_b$, soit W^{TOP}_S une fonction de coût d'arité n associée à la préférence topologique s-TOP. Nous définissons une fonction de voisinage $\mathsf{neighbor}(i)$ qui associe à toutes les parcelles élémentaires i, les parcelles élémentaires $j \in \mathcal{N}_b$ appartenant au même îlot structurel que i et qui ont une frontière commune avec ce dernier. Par exemple, considérant l'îlot structurel de la Figure 4.13, $\mathsf{neighbor}(i)$ peut être défini par un voisinage de von Neumann dans lequel les quatre plus proches voisins sont considérés. La portée S de W^{TOP}_S serait dans ce cas $\{x^t_{b,i}, x^t_{b,n}, x^t_{b,s}, x^t_{b,e}, x^t_{b,o}\}$ où les parcelles élémentaires n, s, e, o sont les

quatre voisins respectivement au nord, sud, est et ouest de i. Ainsi, $\forall a \in D_{b,i}, \forall a_n \in D_{b,n}, \forall a_s \in D_{b,s}, \forall a_e \in D_{b,e}, \forall a_o \in D_{b,o}$

$$W_S^{TOP}(a, a_n, a_s, a_e, a_o) = \begin{cases} 0 & \text{si } a = a_n = a_s = a_e = a_o \\ \delta_1 & \text{sinon} \end{cases} \quad (4.12)$$

En fonction de la localisation de i dans l'ilot structurel, l'arité de W_S^{TOP} pourrait se réduire à 4 ou 3.

4.4.3 Capacité de ressources - h-RSC

4.4.3.a Hypothèses

Chaque parcelle élémentaire consomme une quantité fixe de ressources en fonction de ses exigences qualitatives (type de culture) ou numériques (superficie de la parcelle élémentaire, dose d'irrigation). Par exemple, contrairement au blé d'hiver, le maïs est une culture irriguée. La gestion des ressources est un problème de séquencement et de comptage des quantités allouées. L'approche de résolution classique est basée sur la recherche du plus court chemin sous contrainte de ressources Irnich et Desaulniers (2005). Considérant les hypothèses 4.2 et 4.1, nous suggérons de réduire le problème d'allocation de ressources à un problème de comptage.

Hypothèse 4.2 *: Les ressources sont supposées consommables et systématiquement renouvelables chaque année sans aucune fonction de production (ex : un quota annuel d'eau pour l'irrigation).*

Cette hypothèse est très proche de la réalité car les agriculteurs disposent généralement d'un quota annuel d'eau. Il en est de même pour le nombre d'heures de travail annuel disponible en fonction de la législation ou des choix de l'agriculteur. Considérant l'hypothèse 4.1, nous supposons également que l'échantillonnage de l'exploitation est fait de telle sorte que les parcelles élémentaires soient toutes homogènes.

4.4.3.b Modélisation

Sous ces hypothèses, l'allocation annuelle des ressources est perçue comme un problème de comptage à chaque instant $t \in [h+1, \mathcal{H}]$. Ainsi, connaissant la capacité annuelle des ressources, nous définissons pour chaque instant $t \in [h+1, \mathcal{H}]$ une borne supérieure et une borne inférieure du nombre de variables $x_{i,b}^t$ qui peuvent être affectées à une culture donnée et ceci en fonction des exigences qualitatives et numériques de l'exploitation.

La contrainte de capacité de ressource h-RSC est modélisée par une contrainte globale de cardinalité gcc Régin (1996) portant sur les affectations de cultures aux parcelles élémentaires.

$\forall t \in [h+1, \mathcal{H}]$, soit $W_{S_b^t}^{RSC}$ une contrainte globale d'arité $|\mathcal{N}_b|$ associée aux capacités de ressources.

Sachant $S_b^t = (x_{b,1}^t, \cdots, x_{b,|\mathcal{N}_b|}^t)$, la contrainte globale de cardinalité (gcc) spécifie, pour chaque valeur $a \in \bigcup D_{b,i}$, une borne inférieure $lb(a)$ et une borne supérieure $ub(a)$ correspondant au nombre d'occurrences de la valeur a dans l'affectation $A[S_b^t]$.

$$W_{S_b^t}^{RSC} = \mathsf{gcc}(S_b^t, lb, ub) \quad (4.13)$$

admet une solution s'il existe une affectation de S_b^t telle que :

$$\forall a \in \bigcup D_{b,i},\ lb(a) \leq |\{x_{b,i}^t \in S_b^t | x_{b,i}^t = a\}| \leq ub(a) \tag{4.14}$$

4.4.3.c Illustration de la contrainte h-RSC

Considérons un bloc $b = 1$ de 48 ha d'une exploitation disposant d'un quota annuel de $6000m^3$ d'eau. Supposons un échantillonnage de l'espace en parcelles élémentaires de 1.5 ha. Sachant qu'il faut $165m^3$ d'eau par hectare de maïs, les superficies minimum et maximum de maïs sur ce bloc sont 0 et 36.36 ha. Ainsi, sur le bloc $b = 1$, la borne inférieure du nombre d'occurrences de maïs est $lb(MA) = \lfloor \frac{0\ ha}{1.5\ ha} \rfloor = 0$. La borne supérieure $ub(MA) = \lfloor \frac{36.36\ ha}{1.5\ ha} \rfloor = 24$.

4.5 Les objectifs de production

Les objectifs de production de l'agriculteur sont perçus comme des préférences. Ces préférences sont relatives aux proportions annuelle (s-SBC) et pluri-annuelle des cultures (s-TBC).

4.5.1 Proportion annuelle par culture - s-SBC

Pour modéliser la proportion annuelle par culture (s-SBC), nous partons des préférences explicites définies dans le problème d'allocation. Il s'agit notamment des superficies de cultures que l'agriculteur projette de réaliser par an. En nous basant sur l'hypothèse d'homogénéité surfacique des parcelles élémentaires, nous abordons également la proportion annuelle des cultures comme un problème de comptage avec une possibilité de relaxation dans le cas où les problèmes sont sur-contraints.

Les proportions annuelles (s-SBC) sont donc définies par des contraintes globales de cardinalités relaxées (soft-gcc). Elles autorisent une violation du minorant et du majorant de la contrainte gcc associée. Nous utilisons la mesure de violation orientée variables μ_{var} Hoeve et al. (2006) qui se base sur le nombre minimum de variables dont les valeurs doivent être changées afin de satisfaire la contrainte gcc associée. Si $\sum_{a \in \bigcup D_{b,i}} lb(a) \leq |S| \leq \sum_{a \in \bigcup D_{b,i}} ub(a)$, la mesure de violation μ est exprimée par :

$$\mu_{var}(S) = max\left(\sum_{a \in \bigcup D_{b,i}} overflow(S,a), \sum_{a \in \bigcup D_{b,i}} underflow(S,a) \right) \tag{4.15}$$

La fonction de coût associée à chaque contrainte soft-gcc(S, lb, ub, δ) est $W = \mu(S) \times \delta$. Sur la base de cette définition le contrainte s-SBC est formalisée comme ci-dessous.

Ainsi, $\forall t \in [h+1, \mathcal{H}]$, $\forall b \in \mathcal{B}' \subseteq \mathcal{B}$. Soit $W_{S_b^t}^{SBC}$ une contrainte soft-gcc d'arité $|\mathcal{B}'|$ associée au bloc b à la date t. La portée $S_b^t = \{x_{b,i}^t | i \in \mathcal{N}_b\}$.

$$W_{S_b^t}^{SBC} \quad = \quad \text{soft-gcc}(S_b^t, lb, ub, \mu_{var}, z) \tag{4.16}$$

avec z la variable de coût associée à un domaine fini D_z.

4.5.2 Proportion pluri-annuelle des cultures par parcelle élémentaire - s-TBC

Pour modéliser la proportion pluri-annuelle des cultures par parcelle élémentaire (s-TBC), nous exploitons exactement la même modélisation que celle utilisée pour les proportions annuelles (s-SBC). Ainsi, $\forall b \in \mathcal{B}' \subseteq \mathcal{B}$, $\forall i \in \mathcal{N}_b$. Soit $W^{TBC}_{S_{b,i}}$ une contrainte soft-gcc d'arité $(\mathcal{H}-h)$ associée à chaque parcelle élémentaire i la portée $S_{b,i} = \{x^{h+1}_{b,i}, \cdots, x^{\mathcal{H}}_{b,i}\}$. À l'exception de la portée et de la variable de coût z, $W^{TBC}_{S_{b,i}}$ est définie exactement comme $W^{SBC}_{S^t_b}$.

4.5.2.a Illustration de la contrainte s-TBC

Considérons un bloc $b = 1$ de 48 ha d'une exploitation. Supposons un échantillonnage de l'espace en parcelles élémentaires de 1.5 ha. Considérons la proposition suivante : « la proportion de colza d'hiver (CH) produite sur chaque parcelle doit être comprise entre $[1.5, 3]$ ha » . Dans ce cas, pour chaque séquence de variables $\{x^{h+1}_{b,i}, \cdots, x^{\mathcal{H}}_{b,i}\}$ qui définit une parcelle élémentaire, la borne inférieure du nombre d'occurrences de colza d'hiver est $lb(CH) = \lfloor \frac{1.5\ ha}{1.5\ ha} \rfloor = 1$. La borne supérieure est $ub(CH) = \lfloor \frac{3\ ha}{1.5\ ha} \rfloor = 2$.

4.6 Application

Pour illustrer notre formalisation, nous nous basons sur l'exploitation virtuelle de la section 1.6. Le profil de décision de l'agriculteur considéré est celui défini dans la section 1.6.4.a.

4.6.1 Description des instances du problème d'allocation de cultures

Nous avons conduit nos expérimentations en utilisant quatre instances de l'exploitation virtuelle décrite à la Figure 1.12. Chaque instance correspondant à un échantillonnage spécifique de l'exploitation en parcelles élémentaires. Nous accroissons le nombre de parcelles élémentaires de 15 à 120. Pour ce faire, nous partons de l'exploitation initiale et découpons successivement les parcelles de 12ha en 2, 4 puis 8 plus petites parcelles élémentaires. Cet échantillonnage est réalisé de manière à montrer les performances de l'approche sur différentes tailles de problèmes.

4.6.1.a Instance B[1-4]-LU15

Pour cette instance (cf. Figure 4.14), le nombre de parcelles élémentaires est 15. Les parcelles du problème initial sont supposées élémentaires et leur taille est de 12ha. Le nombre $\mathcal{N}_b$ de parcelles élémentaires par bloc est de :

- 4 pour les blocs 1 et 3,
- 2 pour le bloc 2,
- 5 pour le bloc 4.

Avec ses 15 parcelles élémentaires, le problème ainsi considéré est de petite taille.

4.6.1.b Instance B[1-4]-LU30

Pour cette instance (cf. Figure 4.15), chacune des parcelles élémentaires de l'instance B[1-4]-LU15 a été divisée en deux. On obtient ainsi 30 parcelles élémentaires de 6ha. Le nombre $\mathcal{N}_b$ de parcelles élémentaires par bloc est de :

- 8 pour les blocs 1 et 3,

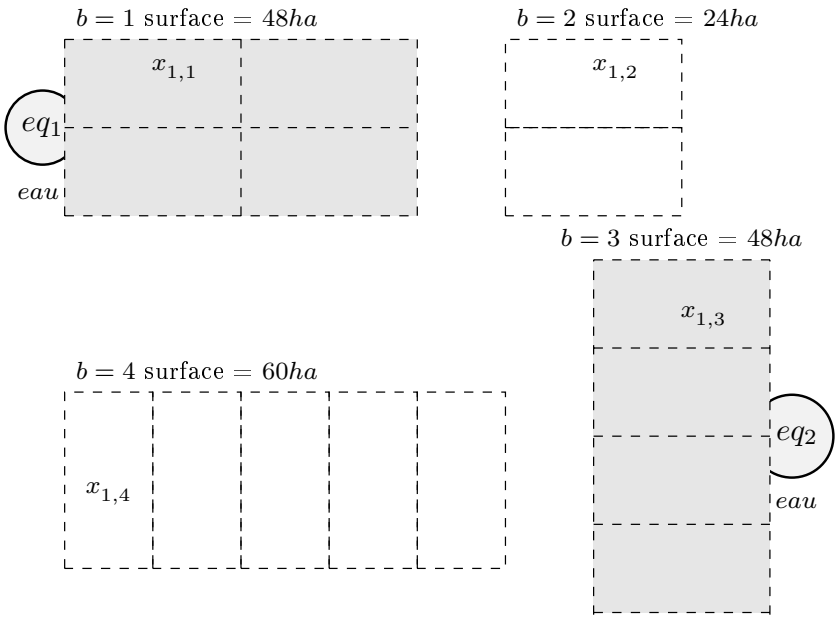


FIGURE 4.14 – *Exploitation virtuelle échantillonnée en 15 parcelles élémentaires de 12ha*

– 4 pour le bloc 2,
– 10 pour le bloc 4.

De la même manière que le précédent, avec 30 parcelles élémentaires, ce problème est de petite taille.

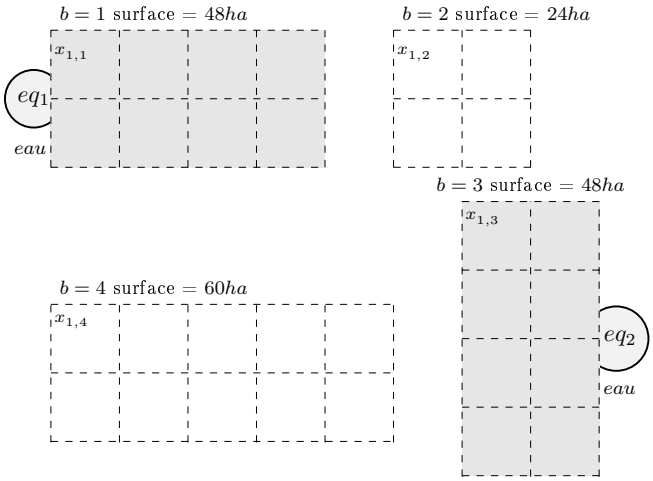


FIGURE 4.15 – *Exploitation virtuelle échantillonnée en 15 parcelles élémentaires de 6ha*

4.6.1.c Instance B[1-4]-LU60

À l'instar du précédent, cette instance (cf. Figure 4.16) est obtenue en divisant les parcelles élémentaires de l'instance B[1-4]-LU15 en quatre. On obtient ainsi 60 parcelles élémentaires de 3ha. Le nombre $\mathcal{N}_b$ de parcelles élémentaires par bloc est de :

– 16 pour les blocs 1 et 3,
– 8 pour le bloc 2,
– 20 pour le bloc 4.

Avec 60 parcelles élémentaires, ce problème est de taille moyenne taille.

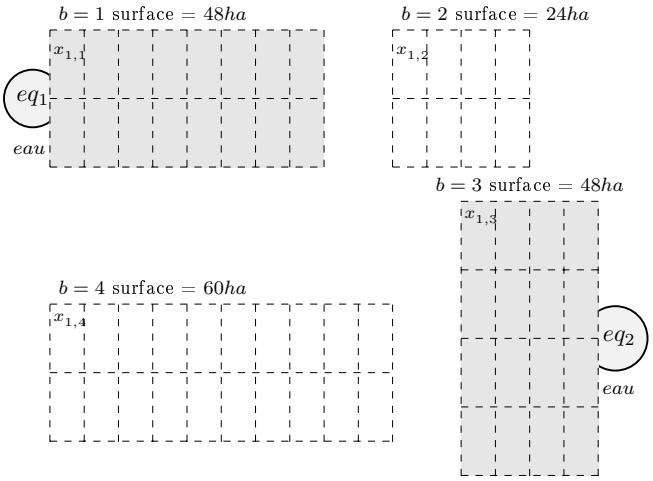


FIGURE 4.16 – *Exploitation virtuelle échantillonnée en 15 parcelles élémentaires de 3ha*

4.6.1.d Instance B[1-4]-LU120

De la même manière que les précédents, cette instance (cf. Figure 4.17) de 120 parcelles élémentaires de 12ha est obtenue en divisant les parcelles élémentaires de l'instance B[1-4]-LU15 en huit. Le nombre $\mathcal{N}_b$ de parcelles élémentaires par bloc est de :

- 32 pour les blocs 1 et 3,
- 16 pour le bloc 2,
- 40 pour le bloc 4.

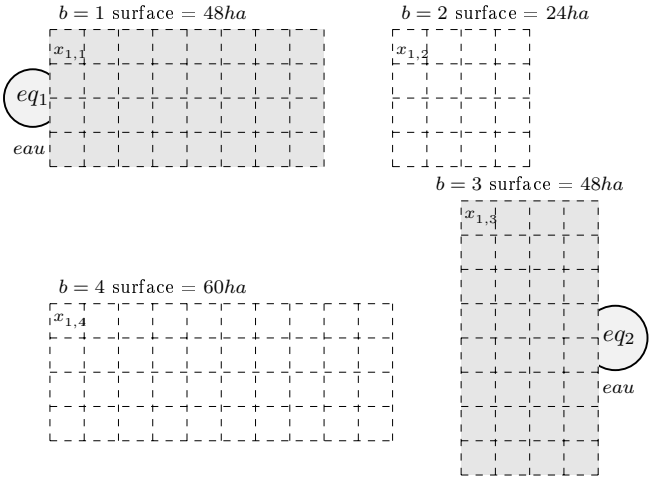


FIGURE 4.17 – *Exploitation virtuelle échantillonnée en 15 parcelles élémentaires de 12ha*

Avec 120 parcelles élémentaires, ce problème est de taille relativement grande.

4.6.1.e Horizon de planification

L'horizon de planification est de 9 ans pour toutes les instances. En associant les cinq premières années d'historique présentées dans le tableau 1.3 nous disposons de 4 années du futur pour la décision stratégique. Cette taille d'horizon de planification est assez confortable si nous considérons le délai de retour des cultures (tableau 1.2) qui est de 2 ans pour le blé d'hiver et le maïs puis de 3 ans pour l'orge de printemps et le colza d'hiver.

4.6.1.f Hiérarchisation des contraintes

En nous basant sur la stratégie d'allocation de l'agriculteur décrite à la section 1.6.4.a, nous savons que pour les préférences :

- ▷ *s-CSQ :* il préfère les séquences de cultures qui minimisent les effets précédents k_p (cf. tableau 4.2),
- ▷ *s-SBC :* il s'impose de respecter ses quotas de production annuelle (contrainte s-SBC et coût δ_2),
- ▷ *s-TOP :* il cherche à grouper au maximum ses îlots fonctionnels afin de réduire les déplacements liés à la réalisation des chantiers (contrainte s-TOP et coût δ_1),
- ▷ *s-TBC :* que les séquences de cultures admissibles sur les blocs fonctionnels 2 et 4 doivent contenir au moins une occurrence de colza (contrainte s-TBC et coût δ_3).

Le tableau 4.2 récapitule les effets précédents de cultures.

TABLE 4.2 – *Effets précédents : valeurs inspirées de Leteinturier et al. (2006)*

	Cultures précédentes			
	BH	OP	MA	CH
BH	4	1	1	0
OP	2	3	1	0
MA	0	0	3	0
CH	0	0	0	4

Sur la base de cette stratégie, nous suggérons de définir des coûts k_p, δ_1, δ_2 et δ_3 tels que $\sum k_p > \sum \delta_2 > \sum \delta_1 > \sum \delta_3$. Cela permettra de pondérer les coûts de manière à les hiérarchiser suivant la stratégie de l'agriculteur.

Pour toutes les expérimentations, les coûts associés aux contraintes topologiques (s-TOP), aux proportions annuelles (s-SBC) et aux proportions pluriannuelles des cultures par parcelle élémentaire (s-TBC) sont respectivement de $\delta_1 = 2$, $\delta_2 = 100$ and $\delta_3 = 10$. Cette pondération permet de hiérarchiser les points 2, 3 et 4 de la stratégie d'allocation. Afin de privilégier les séquences de culture qui minimisent les effets précédents, nous introduisons un facteur $\delta_4 = 10$. Pour tous les effets précédents, k_p (Tableau 4.2), nous calculons une nouvelle valeur de $k_p = \delta_4 * k_p$.

Afin de mettre une priorité sur les effets précédents, nous proposons de multiplier tous les k_p du tableau 4.2 par un coefficient $\delta_4 = 10$.

4.6.1.g Les instances étudiées

D'après l'énoncé du problème, certaines préférences ne portent que sur un et un seul bloc. Il s'agit notamment de la production de maïs qui doit être réalisée sur les blocs 1 et 3 de manière à avoir entre $[24, 48]$ *ha* et $[12, 24]$ *ha* respectivement sur les blocs fonctionnels 1 et 3.

D'autres préférences se rapportent à la superficie globale sur l'exploitation. Il s'agit dans ce cas de la superficie de maïs sur l'ensemble de l'exploitation qui doit être comprise entre $[40, 72]$ *ha*. De la même manière, la production annuelle de blé d'hiver sur l'ensemble de l'exploitation est comprise entre $[70, 100]$ *ha*.

Pour nos expérimentations, nous allons procéder en deux étapes. Dans un premier temps nous supprimerons les préférences qui portent sur plusieurs blocs fonctionnels. Cela nous permet de montrer les performances de la résolution de l'allocation sur les blocs fonctionnels pris indépendamment. Dans ces conditions, les instances associées au bloc 1 seront nommées B1-LU4, B1-LU8, B1-LU16, B1-LU32 respectivement pour les échantillonnages de ce bloc en 4, 8, 16 et 32 parcelles élémentaires. Il en est de même pour les autres blocs fonctionnels.

Dans un second temps, nous rajoutons les préférences qui portent sur l'ensemble de l'exploitation. Les instances résultantes seront nommées B1[1-4]-LU15(*), B[1-4]-LU30(*), B[1-4]-LU60(*) et B[1-4]-LU120(*). Les blocs sont dans ce cas tous interdépendants. Toutes les instances ci-dessus décrites sont disponibles dans le *benchmark*, test de performance, *cost function*[3].

Pour chacune des instances, le nombre de contraintes est approximativement égal à $\frac{5}{2} \times \mathcal{N} \times \mathcal{H} \pm 30$, où $\mathcal{N}$ est le nombre de parcelles élémentaires, et $\mathcal{H}$ l'horizon de planification.

4.6.2 Analyse des résultats

4.6.2.a Protocole expérimental

Pour la résolution d'un problème d'allocation de culture, nous utilisons l'algorithme *Depth-First Branch and Bound* (DFBB, Algorithme 1) implémenté dans le solver **Toulbar2**[4] (version 0.9.1). Le paramétrage utilisé est celui par défaut. Les colonnes $|\mathcal{X}|$ et $|\mathcal{W}|$ du tableau 4.3 montrent le nombre de variables ($\mathcal{N} \times \mathcal{H}$) et de contraintes pour chaque instance.

Les résultats présentés dans le tableau 4.3 sont obtenus avec un processeur Intel(R) Xeon(R) de 2.27GHz. Le temps de calcul mentionné est en seconde. Il indique la durée totale pour trouver et prouver l'optimalité (colonne « Temps » de « Une sol. optimale ») en commençant avec un majorant initial relativement bon (colonne UB). Ce majorant influence considérablement la performance. Dans notre cas le choix du majorant initial est empirique. Sur la base de l'optimum trouvé (colonne Opt.) pour la recherche d'une solution, nous recherchons toutes les solutions (colonne « Toutes les sol. optimales ») avec un majorant initial fixé à l'optimum plus un. Les colonnes « Nœuds » et « BT » représentent respectivement le nombre de nœuds parcourus et le nombre de *backtrack* réalisés pour trouver le ou les solutions optimales.

Pour les instances avec les blocs indépendants, les meilleures solutions sont obtenues en moins d'une minute à l'exception de B1-LU32. Dans ce cas, les solutions optimales sont obtenues et prouvées pour toutes les instances. La différence entre le temps de calcul pour la recherche d'une ou de toutes les solutions est principalement liée au majorant initial. Les résultats obtenus en introduisant les interdépendances entre les blocs sont dans l'ensemble acceptables comparativement à la taille des problèmes. En effet, l'arité maximum de certaines contraintes gcc et soft-gcc est égale au nombre total de parcelles élémentaires soit 120 dans l'instance la plus importante. Cela peut expliquer la raison pour laquelle l'instance B[1-4]-LU120(*) n'a pas été résolue au bout de 48 heures.

3. http://www.costfunction.org/benchmark?task=browseAnonymous&idb=33
4. http://mulcyber.toulouse.inra.fr/projects/toulbar2

TABLE 4.3 – *Performance pour la recherche d'une solution optimale et de toutes les solutions optimales avec DFBB*

Instances	$\|\mathcal{X}\|$	UB	$\|\mathcal{W}\|$	Opt.	Une solution optimal (DFBB)			Toutes les solution optimales (DFBB)			
					Temps(s)	Nœuds	BT	Temps(s)	Nœuds	BT	Nb.Sol
B1-LU4	36	1000	91	92	0.39	17	10	0.08	8	4	5
B1-LU8	72	2000	175	184	2.96	94	49	0.21	32	16	17
B1-LU16	144	4000	343	368	21.47	413	209	2.64	256	512	257
B1-LU32	288	6000	679	640	228	285	147	6.19	38	19	17
B2-LU2	18	1000	47	38	0.08	2	2	0.06	2	1	1
B2-LU4	36	2000	95	116	0.22	8	4	0.22	8	4	1
B2-LU8	72	4000	191	392	4.19	6	5	0.36	2	1	1
B2-LU16	144	6000	383	752	7.9	10	9	0.78	2	1	1
B3-LU4	36	1000	99	328	0.3	14	7	0.29	16	8	2
B3-LU8	72	2000	199	656	0.64	14	7	0.6	16	8	2
B3-LU16	144	4000	367	1312	1.51	18	9	1.37	16	8	2
B3-LU32	288	6000	703	2592	4.1	20	10	3.79	18	9	2
B4-LU5	45	1000	119	46	0.53	4	4	0.08	0	0	1
B4-LU10	90	2000	239	192	11.64	5	4	0.57	0	0	1
B4-LU20	180	4000	479	752	12.32	12	10	0.73	0	0	1
B4-LU40	360	6000	959	1504	39.33	23	19	1.97	2	1	1
B[1-4]-LU15(*)	135	2000	465	1824	21.02	257	131	7.87	96	48	2
B[1-4]-LU30(*)	270	4000	922	3660	323.02	1029	521	155.9	498	249	12
B[1-4]-LU60(*)	540	8000	1804	7492	2412.97	1297	658	3697.23	2228	1114	136
B[1-4]-LU120(*)	1080	16000	3568	-	-	-	-	-	-	-	-

De manière générale, les solutions obtenues sont en pratique de très bonnes qualités. Nous présentons ici l'analyse de quelques une d'entre-elles. Plus particulièrement, notre analyse portera sur les instances pour lesquelles les blocs sont interdépendants. L'ensemble des solutions est disponible à l'adresse http://www.akplogan-mahuna.com/assolement/index.php. Nous en présentons quelques une dans l'annexe A.3.

4.6.3 Analyse des solutions trouvées pour l'instance B1[1-4]-LU15(*)

4.6.3.a Proportion annuelle des cultures pour l'instance B1[1-4]-LU15(*)

Considérons les deux solutions optimales trouvées pour l'instance B1[1-4]-LU15(*). La Figure 4.18 permet de visualiser pour chacune des solutions, le nombre de parcelles affectées à une culture donnée. Les valeurs d'historique ne sont pas représentées. Sachant que pour cette instance, la superficie des parcelles élémentaires est de 12 *ha*, on déduit implicitement les proportions annuelles des cultures.

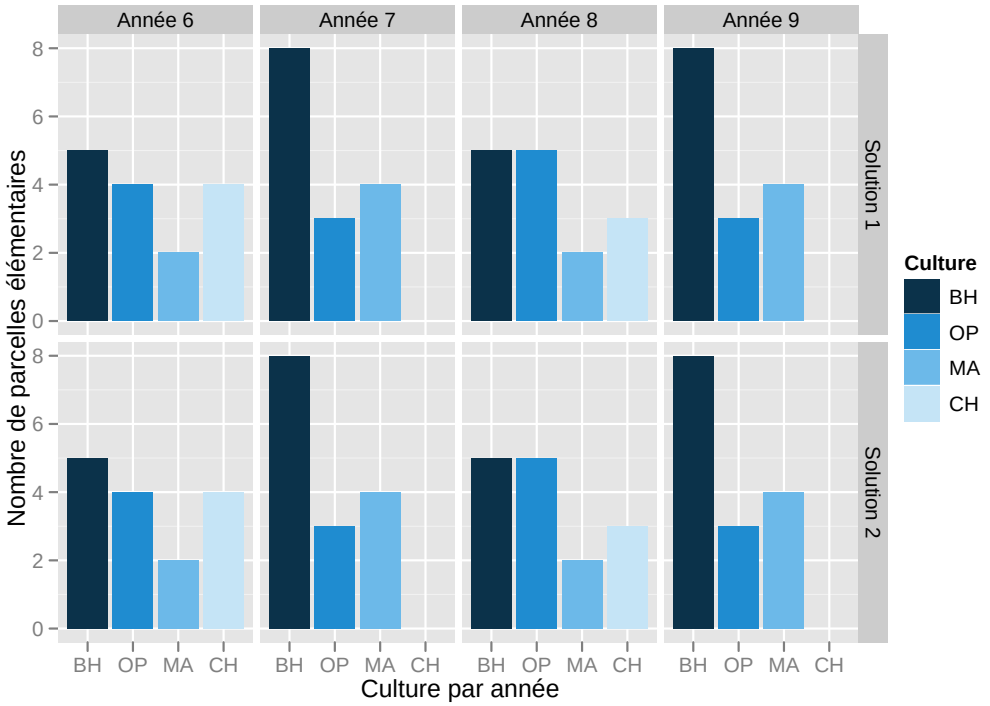


FIGURE 4.18 – *Proportion annuelle des cultures pour les différentes années.*

Sur le graphique, on observe qu'en considérant une année donnée, les proportions de cultures pour les deux solutions sont équivalentes. Par exemple, pour l'année 6, quelque soit la solution, les proportions de cultures sont de 60 *ha*, 48 *ha*, 24 *ha* et 48 *ha* respectivement pour le blé d'hiver, l'orge de printemps, le maïs et le colza d'hiver. Cela montre que pour l'agriculteur, les solutions obtenues sont équivalentes en terme d'assolement et de proportion de cultures. L'ensemble des valeurs des proportions de cultures est présenté en Annexe A.3 dans le Tableau A.1.

Nous pouvons également souligner que la préférence de l'agriculteur relative à la proposition annuelle des maïs n'est pas respectée pour les années 6 et 8. Cela est dû aux valeurs d'historique sur le bloc 3 (cf. Tableau 1.3). En effet, les pratiques historiques de l'agriculteur consistaient à réaliser une monoculture de maïs sur le

bloc 3. Les contraintes de délais de retour ne permettent donc pas de réaliser du maïs sur le bloc 3 pour les années 6 et 8. De plus, les contraintes de capacités de ressources combinées aux objectifs de production du blé ne permettent pas de concentrer la production du maïs sur le bloc 1.

Ces solutions montrent très bien que le problème est sur-contraint, et que les préférences de l'agriculteur ne peuvent pas toutes être satisfaites.

4.6.3.b Allocation des cultures

La Figure 4.19 montre pour chaque solution les séquences de culture sur certaines parcelles élémentaires représentatives qui sont : 1, 4 (bloc 1), 5 (bloc 2), 7, 9 (bloc 3) et 11, 13 (bloc 4). Il est important de noter la nature explicite de l'allocation. Contrairement à la majorité des travaux existants, chaque solution représente bien un plan stratégique spatialement et temporellement explicite.

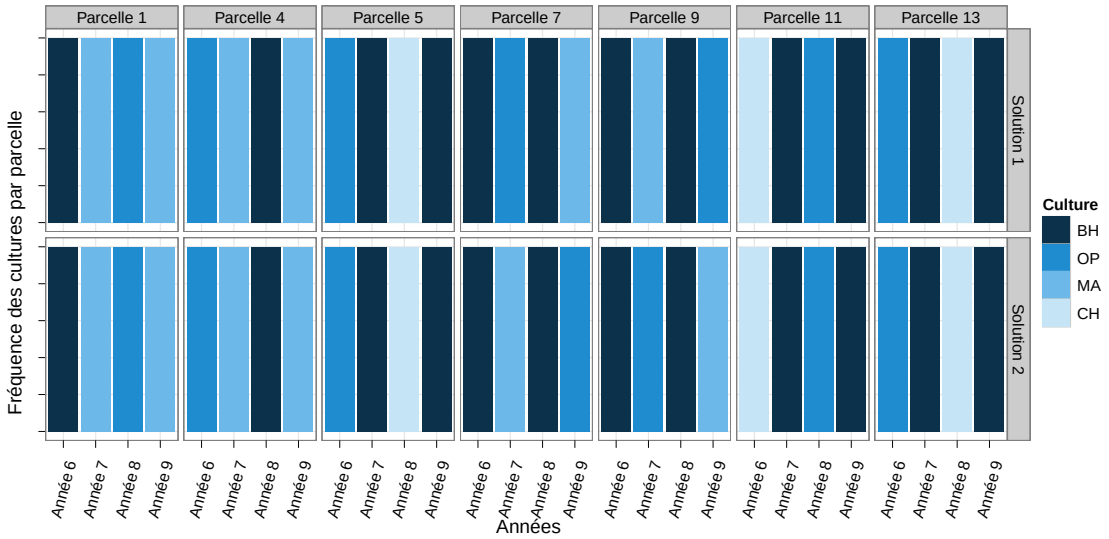


FIGURE 4.19 – *Allocation des cultures pour l'instance B1[1-4]-LU15(*)*

Le graphique nous permet d'observer que les délais de retour sont respectés. Par exemple, sur la parcelle élémentaire 1 on note une séquence *blé d'hiver* → *maïs* → *orge de printemps* → *maïs*. À l'instar de toutes les solutions, la séquence obtenue est bien une rotation fixe cyclique à taille fixe (cf. Définition 1.8) au sens de Castellazzi et al. (2008) (cf. Figure 1.3-a). Par ailleurs, si l'on combine les rotations fixes sur la parcelle élémentaire 7, on peut construire une rotation flexible cyclique à taille fixe (cf. Figure 1.3-b).

La différence entre les solutions de la Figure 4.19 repose sur les allocations de cultures sur le bloc 3 (parcelles élémentaires 7 et 9). Pour les années 7 et 9, l'orge de printemps (OP) peut être substitué à du maïs (MA).

Notons également qu'il n'existe aucune symétrie entre les solutions. Cela résulte du fait que les parcelles élémentaires ont des types de sol et des valeurs d'historiques différentes.

4.6.4 Analyse des solutions trouvées pour l'instance B1[1-4]-LU30(*)

4.6.4.a Proportion annuelle des cultures B1[1-4]-LU30(*)

De façon analogue à la section précédente, nous considérons l'ensemble des solutions optimales trouvé pour l'instance B1[1-4]-LU30(*). Le Tableau 4.4 montre pour

chaque solution et pour chacune des années, les proportions de culture. Pour cette instance, la superficie des parcelles élémentaires est de 6 *ha*. Les proportions de cultures pour les années 6 et 8 sont équivalentes pour toutes les solutions. Pour les deux autres années on note quelques différences. Le variation est globalement de 6 *ha* attribués soit à l'orge de printemps soit au blé d'hiver.

De la même manière et pour les mêmes raisons la préférence de l'agriculteur relative à la proposition annuelle des maïs n'est pas respectée pour les années 6 et 8.

TABLE 4.4 – *Proportions de cultures des solutions de l'instance B1234-LU30-ALL(*)*

	Année 6				Année 7				Année 8				Année 9			
	BH	OP	MA	CH	BH	OP	MA	CH	BH	OP	MA	CH	BH	OP	MA	CH
1	66	48	24	42	96	30	48	6	66	54	24	36	78	48	48	0
2	66	48	24	42	96	30	48	6	66	54	24	36	78	48	48	0
3	66	48	24	42	90	36	48	6	66	54	24	36	84	42	48	0
4	66	48	24	42	96	30	48	6	66	54	24	36	78	48	48	0
5	66	48	24	42	96	30	48	6	66	54	24	36	78	48	48	0
6	66	48	24	42	90	36	48	6	66	54	24	36	84	42	48	0
7	66	48	24	42	96	30	48	6	66	54	24	36	78	48	48	0
8	66	48	24	42	96	30	48	6	66	54	24	36	78	48	48	0
9	66	48	24	42	90	36	48	6	66	54	24	36	84	42	48	0
10	66	48	24	42	96	30	48	6	66	54	24	36	78	48	48	0
11	66	48	24	42	96	30	48	6	66	54	24	36	78	48	48	0
12	66	48	24	42	90	36	48	6	66	54	24	36	84	42	48	0

Fin des solutions de l'instance

4.6.4.b Allocation des cultures

La Figure 4.20 montre les allocations spatiales et temporelles sur les blocs. Le graphique présente pour chaque solution les séquences de cultures sur certaines parcelles élémentaires représentatives : 1, 8 (bloc 1), 9 (bloc 2), 13, 17 (bloc 3) et 21, 25 (bloc 4). Dans ce cas aussi, les délais de retour sont respectés.

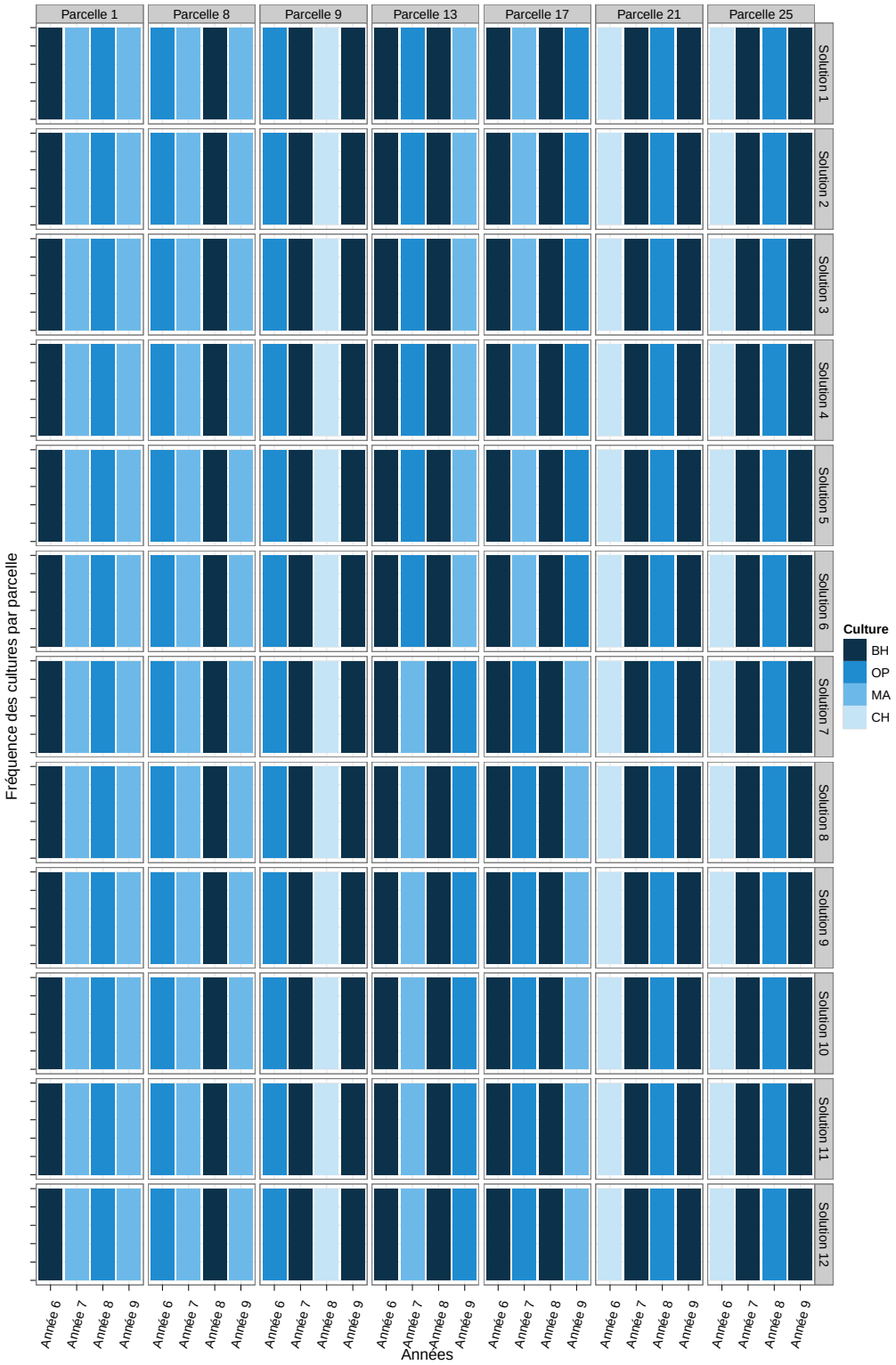


FIGURE 4.20 – *Allocation des cultures pour l'instance B1[1-4]-LU30(*)*

4.6.5 Analyse des solutions trouvées pour l'instance B1[1-4]-LU60(*)

Comparée aux deux précédentes, l'instance B1[1-4]-LU60(*), présente beaucoup plus de solutions (136 solutions). Le Tableau A.3 de l'Annexe A.3 montre la proportion des cultures pour chacune des solutions. Nous nous proposons d'analyser ces solutions de manière plus globale. Dans un premier temps, nous décrivons la nature des séquences de cultures. Ensuite nous observons la distribution des proportions annuelles de cultures sur l'ensemble des cultures. Enfin, nous visualisons une des 136 solutions afin de montrer la nature d'un plan stratégique.

4.6.5.a Analyse des séquences de cultures par bloc fonctionnel

Pour l'ensemble des solutions optimales, nous considérons ici, certaines parcelles élémentaires représentatives qui sont :

▷ *bloc 1 :* 1, 8, 9, 16,

▷ *bloc 2 :* 17, 20, 21, 24,

▷ *bloc 3 :* 25, 32, 33, 40,

▷ *bloc 4 :* 41, 43, 47, 60.

Pour chacune de ces parcelles élémentaires nous visualisons (Figures 4.21, 4.23, 4.22 et 4.24) à chaque année, les différentes options en terme d'occupation du sol.

Par exemple, en considérant les blocs 1 et 3 (Figures 4.21 et 4.22), on note l'absence du colza d'hiver dans les séquences de cultures. Ceci est dû au type de sol du bloc. Sur le bloc 1, les séquences de culture sur les parcelles élémentaires proposent une flexibilité entre le blé d'hiver et l'orge de printemps. Le nombre de solutions flexibles est relativement faible pour les parcelles élémentaires 1,8 et 16. A ce stade, il est important de souligner qu'en substituant le blé d'hiver par l'orge de printemps, l'agriculteur fera implicitement le choix de laisser le sol nu durant l'hiver (et inversement). Ce type de plan stratégique pourrait nécessiter, durant l'hiver, des opérations agricoles de maintien de la structure du sol afin de la rendre réceptive à la culture de printemps.

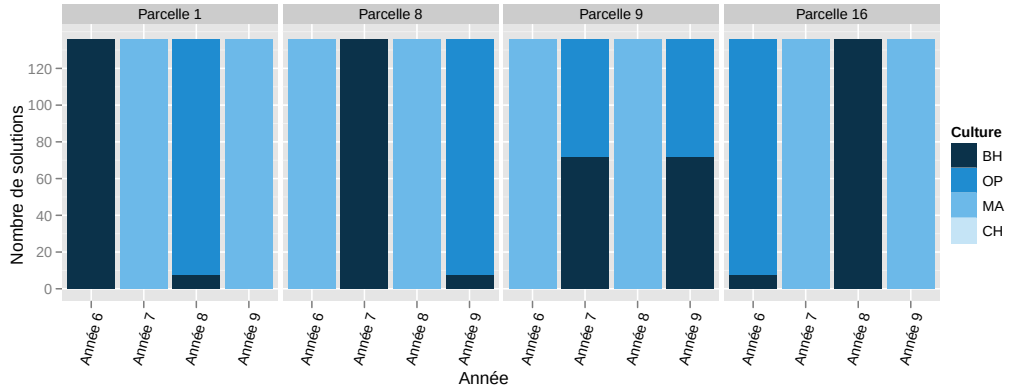


FIGURE 4.21 – *Bloc 1 : séquences de cultures sur les parcelles élémentaires* 1, 8, 9 *et* 16

En considérant le bloc 3, la flexibilité porte plutôt sur le maïs et l'orge de printemps. Ces deux cultures sont semées au printemps. On peut donc envisager de les substituer en cas d'échec de semis.

Pour les blocs 2 et 4 (Figures 4.23 et 4.24), nous soulignons l'absence du maïs dans les séquences de cultures. Ceci est dû à la disponibilité des ressources en eau sur ces blocs. Le colza est réalisé une fois sur toutes les parcelles élémentaires. On en déduit que la préférence de proportion pluri-annuelle des cultures par parcelle élémentaire (cf. section 1.6.4.a) a été bien respectée.

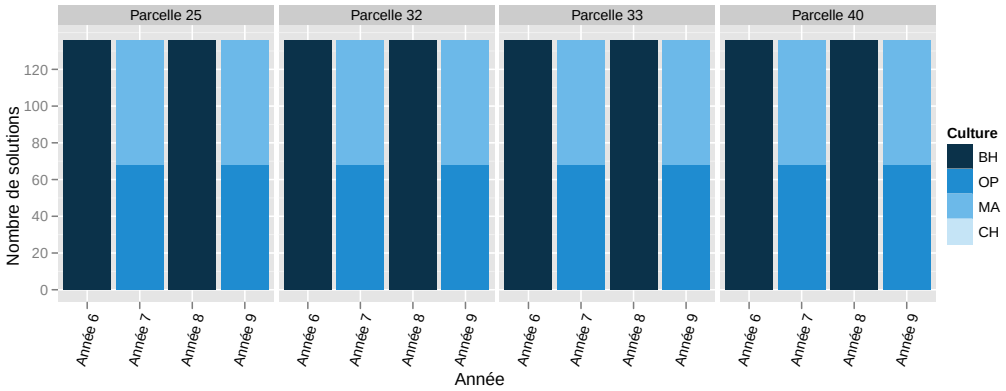


FIGURE 4.22 – *Bloc 3 : séquences de cultures sur les parcelles élémentaires* 25, 32, 33 *et* 40

En considérant le bloc 2, nous pouvons souligner qu'il existe plusieurs options d'allocation pour les parcelles élémentaires 21 et 24.

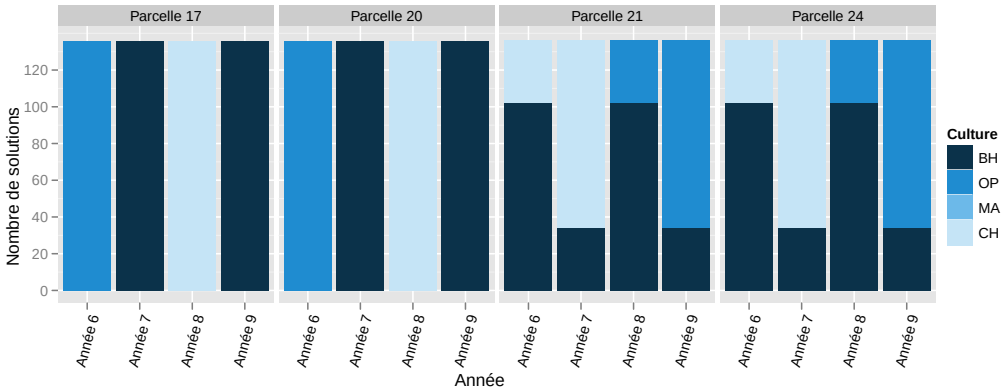


FIGURE 4.23 – *Bloc 2 : séquences de cultures sur les parcelles élémentaires* 17, 20, 21 *et* 24

Contrairement au bloc 2, il n'existe aucune flexibilité pour les séquences de cultures du bloc 4.

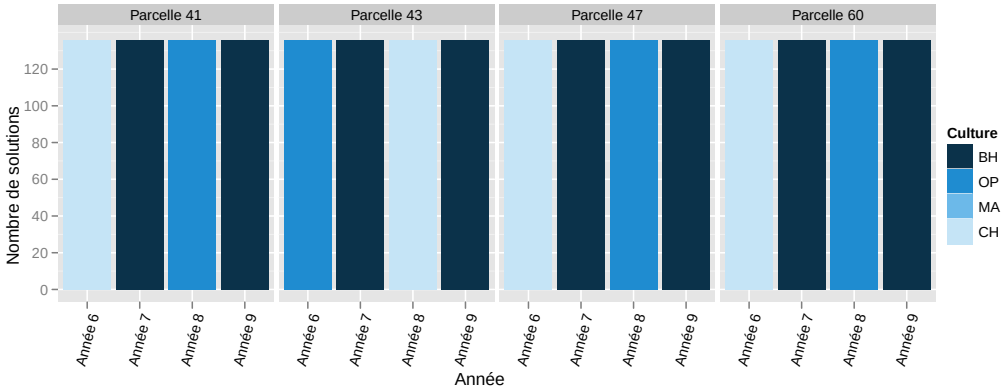


FIGURE 4.24 – *Bloc 4 : séquences de cultures sur les parcelles élémentaires* 41, 43, 47 *et* 60

4.6.5.b Distribution des proportions annuelles de cultures

Les proportions annuelles des cultures sont variables en fonction des solutions. Afin de vérifier si ces proportions correspondent aux préférences de l'agriculteur (cf. section 1.6.4.a), nous visualisons leur distribution sur l'ensemble des solutions.

Rappelons que la production annuelle de maïs doit être comprise entre $[40, 72]$ *ha* sur l'ensemble de l'exploitation. Celle du blé d'hiver doit être comprise entre $[70, 100]$ *ha*.

Considérant une culture, les Figures 4.25, 4.26, 4.27 et 4.28 présentent le nombre de solutions en fonction des proportions annuelles.

Pour le blé d'hiver (Figure 4.25), quelque soit la solution, les proportions sont strictement respectées pour les années 7 et 9. En considérant les années 6 et 8, bien qu'il existe des solutions qui respectent la préférence, la majorité d'entre elles propose 69 *ha* plutôt que 70 *ha* au minimum. Sachant que pour cette instance, la superficie des parcelles élémentaires est de 3 *ha*, nous pensons qu'un échantillonnage plus fin de l'espace aurait pu permettre de satisfaire entièrement la préférence.

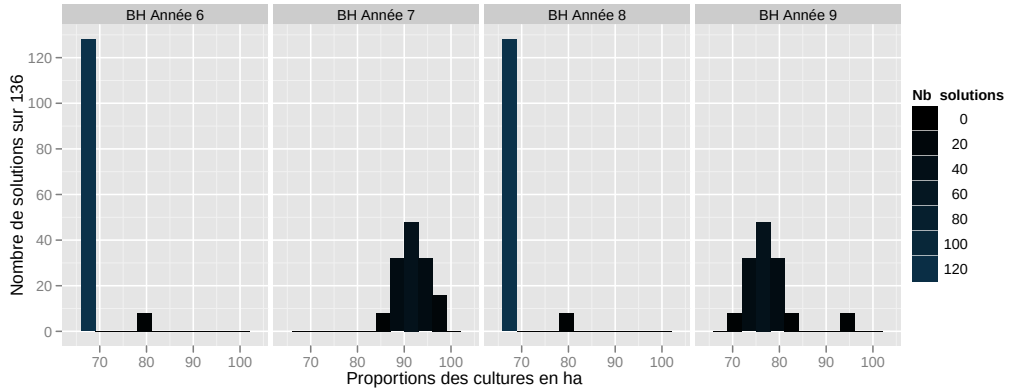


FIGURE 4.25 – *Distribution des proportions annuelles de BH sur l'ensemble de solutions*

La préférence de l'agriculteur relative à la proposition annuelle du maïs n'est pas respectée pour les années 6 et 8. Encore une fois, ceci est dû aux valeurs d'historiques sur le bloc 3

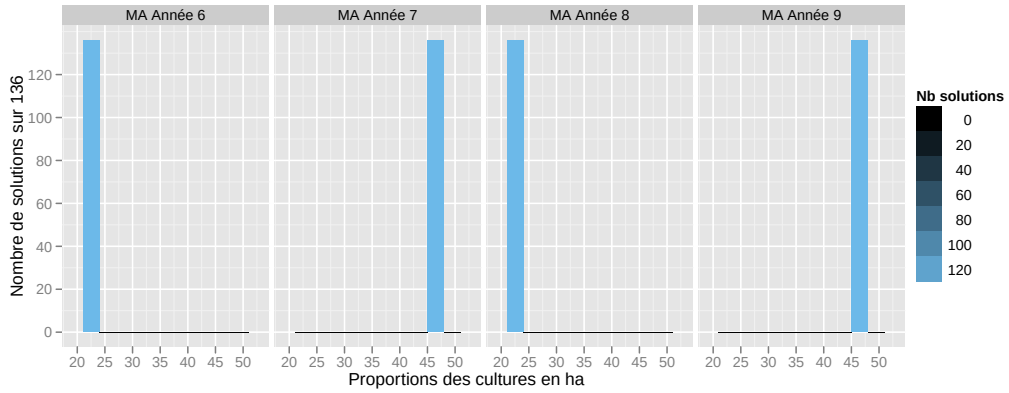


FIGURE 4.26 – *Distribution des proportions annuelles de MA sur l'ensemble de solutions*

Pour les deux autres cultures, il n'y avait aucune contrainte de proportion annuelle.

4.6.5.c Visualisation d'une solution

Afin de montrer la nature d'un plan stratégique, nous sommes partis de quatre de 136 solutions optimales de l'instance B1[1-4]-LU60(*). Nous en présentons quelques unes de plus dans l'annexe A.3. Comme l'indique la Figure 4.29 les plans stratégiques sont spatialement et temporellement explicite.

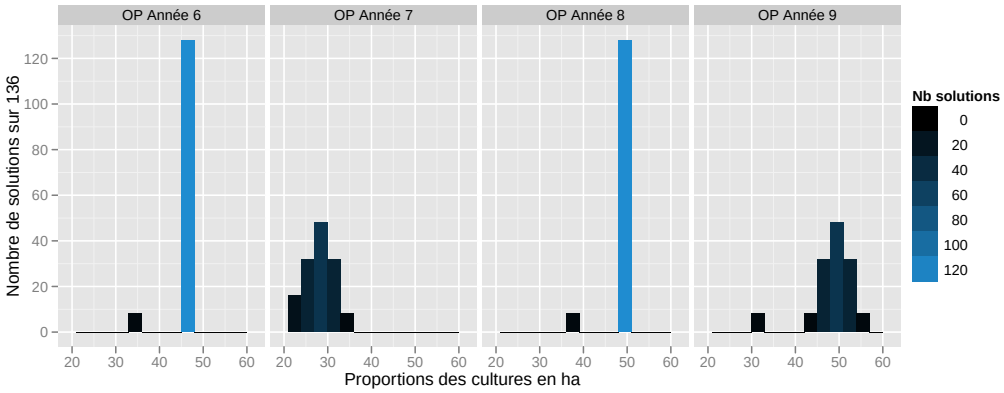


FIGURE 4.27 – *Distribution des proportions annuelles de OP sur l'ensemble de solutions*

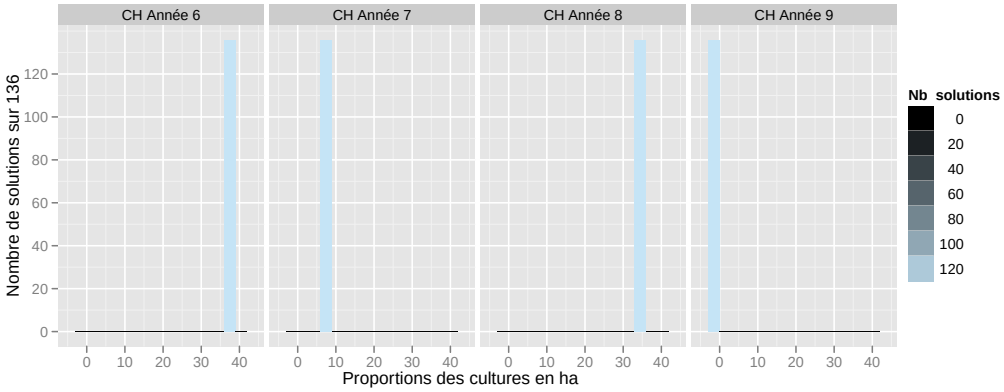


FIGURE 4.28 – *Distribution des proportions annuelles de CH sur l'ensemble de solutions*

Au sein d'un même bloc, les cultures sont spatialement groupées [5]. On en déduit que les préférences topologiques sont bien prises en compte. Il en est de même pour les délais de retour des cultures.

La dynamique des assolements successifs permet de voir que notre approche est capable de prendre en compte les fusions et redécoupages de parcelles.

De plus, en observant très attentivement cette solution, on remarque que les mêmes sous-ensembles de cultures sont produits sur toutes les parcelles élémentaires d'un même bloc. En conséquence, les séquences de culture au sein des blocs définissent bien un système de culture.

4.7 Conclusion et discussion

Dans ce chapitre nous avons proposé une nouvelle approche de résolution du problème d'allocation de culture basée sur les CSP pondérés. Contrairement à la majorité des approches existantes, notre proposition est spatialement explicite et intègre les deux dimensions (spatiale et temporelle) du problème. Nous avons décrit la manière dont les contraintes dures et les préférences d'un agriculteur peuvent être abordées sous la forme d'une optimisation de fonction objective globale. Malgré la complexité intrinsèque des approches CSP les résultats obtenus montrent qu'avec le solver *Toulbar2*, des solutions peuvent être trouvées en temps raisonnable pour des problèmes de petites et moyennes tailles.

5. Bien que dans la plupart des solutions, la préférence topologique est respectée, il existe cependant des solutions pour lesquelles nous avons observé des violations de cette préférence (cf. annexe A.3). Ceci s'explique par l'agrégation de l'ensemble des coûts.

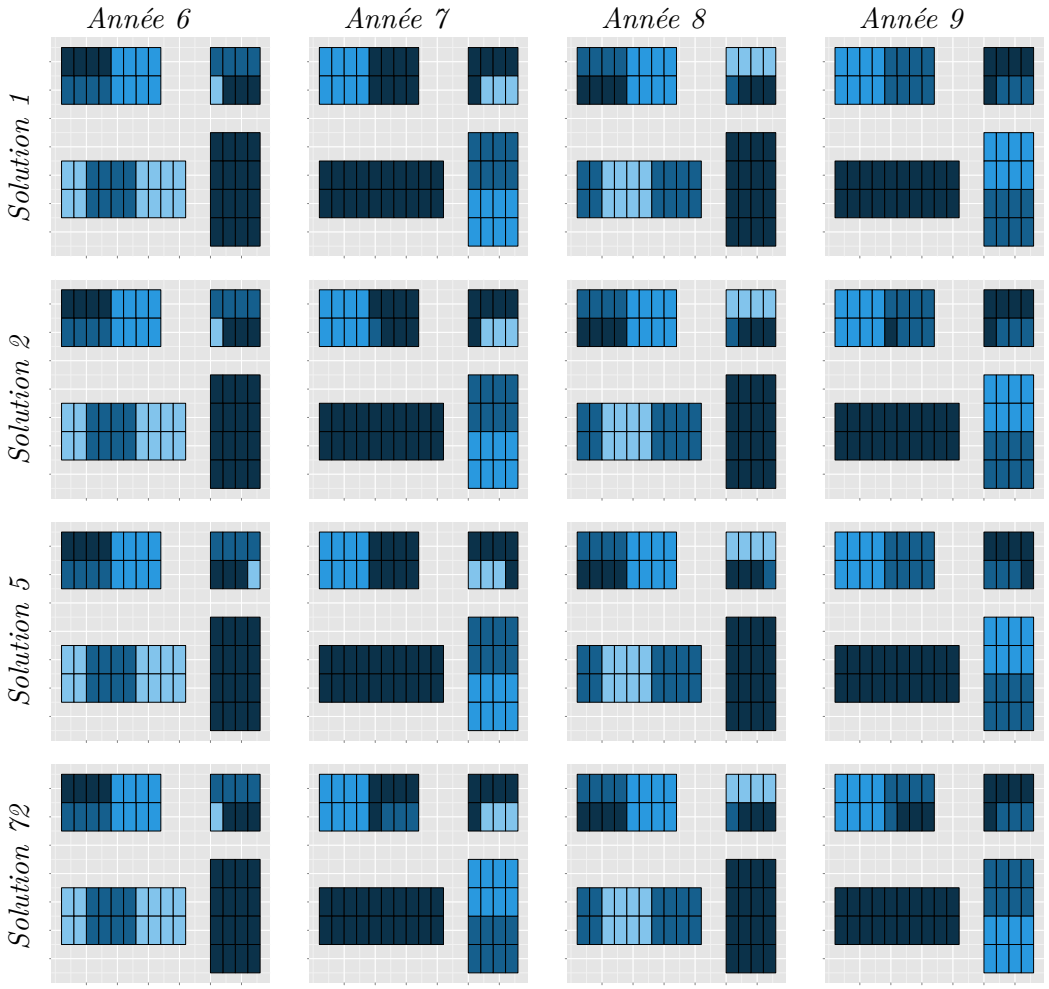


FIGURE 4.29 – *Illustration de 4/136 plans stratégiques, solutions de l'instance B[1-4]-LU60-ALL(*) :* ■ *Blè d'hivers,* ■ *Orge de printemps,* ■ *Maïs,* ■ *Colza d'hivers*

Notre proposition ouvre une nouvelle piste de recherche qui permet d'élargir les techniques de résolutions du problème d'allocation de cultures. Elle illustre, par la même occasion, la possibilité d'aborder ce problème de manière globale. A terme, cette approche, pourrait conduire à des outils opérationnels, capables d'être mis à disposition d'utilisateurs finaux tels que des agriculteurs, des agronomes etc.

Pour atteindre ce stade de maturité de la méthode proposée, certaines études doivent être menées afin d'enrichir les connaissances prises en compte d'une part puis d'améliorer les performances de notre approche d'autre part. Ainsi, deux grandes pistes s'ouvrent.

Enrichissement des connaissances représentées

Échantillonnage de l'espace : l'homogénéité surfacique des parcelles élémentaires est l'une des grandes hypothèses de notre proposition. En soit, la notion de parcelles élémentaires est essentielle pour déterminer la granularité des découpages et fusions des parcelles. Toutefois, nous sommes persuadés qu'en utilisant d'autres types de contraintes globales telles que les CUMULATIVE (Beldiceanu et Contejean, 1994), nous pourrions nous affranchir de l'hypothèse d'homogénéité surfacique. De plus, lever cette hypothèse a pour conséquence directe la réduction du nombre de parcelles élémentaires et la diminution de la taille du problème.

Monocultures partielles : nous n'avons pas explicitement montré dans ce chapitre, comment la formalisation de la notion de délai de retour peut être généralisée à celle des cultures répétitives [6]. Pour ces classes de cultures, l'algorithme de construction des automates à états finis déterministe représentant le langage des délai de retour devra être amélioré.

Amélioration des performances de l'approche

Résoudre les problèmes de grandes tailles en temps raisonnable est crucial. Nous savons que dans notre formalisation, les contraintes globales sont les plus coûteuses. Plusieurs pistes sont envisageables afin de palier à cette limite.

Fusion des automates associés aux contraintes Regular : dans la formalisation que nous proposons, le nombre total de contraintes REGULAR est de l'ordre de $\mathcal{N} \times |\mathcal{D}|$, avec $\mathcal{N}$ le nombre total de parcelles et $|\mathcal{D}|$ le taille des domaines des variables. On pourrait envisager une fusion des automates par parcelles élémentaires. Cela réduira à $\mathcal{N}$ le nombre de contraintes. Cependant, étant donné la taille de nos automates, cette piste n'est pas envisageable car le gain en nombre d'automates ne compense pas la complexité de résolution d'automates de plus grandes tailles. Il faudra donc construire des automates plus compacts. Pour y arriver, l'utilisation des contraintes globales de type WEIGHTEDREGULAR (Katsirelos et al., 2011) nous semble plus appropriée.

Décomposition des contraintes globales : une autre piste d'amélioration possible serait de proposer une formulation du problème qui décompose les contraintes globales en contraintes plus simples. Des travaux récents illustrent la pertinence de cette approche. Par exemple, dans Allouche et al. (2012), les auteurs montrent qu'on obtient de meilleures performances en décomposant les contraintes REGULAR en contraintes ternaires.

6. Certaines cultures peuvent être en monocultures partielles c'est-à-dire qu'elles imposent des itérations successives de la culture avant l'application du délai de retour

Planification des opérations agricoles et gestion des ressources

5

Sommaire

4.1 État de l'art sur les contraintes globales 88
4.1.1 Les réseaux de contraintes pondérées 88
4.1.2 La contrainte globale Regular 89
4.1.3 La contrainte globale de cardinalité Gcc 92
4.1.4 La contrainte globale de cardinalité relaxée Soft-gcc 95
4.1.5 La contrainte globale Same 97
4.2 Description détaillée des contraintes pour la planification stratégique 99
4.2.1 Caractéristiques et hypothèses de notre approche de résolution . . 99
4.2.2 Description des contraintes 100
4.2.3 Variables et domaines du problème d'allocation de cultures 103
4.3 Formalisation des contraintes agronomiques 104
4.3.1 Contraintes de zone cultivable - h-SCC 104
4.3.2 Historique des parcelles élémentaires - h-HST 104
4.3.3 Délai de retour - h-TSC 105
4.3.4 Rotation culturale - h-CCS 107
4.3.5 Collection de cultures par bloc - h-SCA 108
4.3.6 Qualité des séquences de cultures : effet précédent - s-CSQ 110
4.4 Formulation des contraintes organisationnelles 111
4.4.1 Interdépendance entre parcelles élémentaires - h-EQU 111
4.4.2 Préférence topologique - s-TOP 111
4.4.3 Capacité de ressources - h-RSC 112
4.5 Les objectifs de production 113
4.5.1 Proportion annuelle par culture - s-SBC 113
4.5.2 Proportion pluri-annuelle des cultures par parcelle élémentaire - s-TBC 114
4.6 Application 114
4.6.1 Description des instances du problème d'allocation de cultures . . . 114
4.6.2 Analyse des résultats 118
4.6.3 Analyse des solutions trouvées pour l'instance B1[1-4]-LU15(*) . . 120
4.6.4 Analyse des solutions trouvées pour l'instance B1[1-4]-LU30(*) . . 121
4.6.5 Analyse des solutions trouvées pour l'instance B1[1-4]-LU60(*) . . 124

4.7 Conclusion et discussion 127

Dans ce chapitre nous présentons nos approches de résolution des problèmes de planification tactique et opérationnelle. La première proposition relative à la planification tactique apporte une réponse au problème d'affectation des itinéraires techniques (ITKs) aux cultures allouées lors de la phase de planification stratégique. La seconde proposition (planification opérationnelle) quant-à-elle consiste d'une part à déterminer les ressources à affecter aux tâches pour assurer leur réalisation et, d'autre par à définir l'organisation de ces tâches dans le temps. Pour ce faire, nous rappelons au début de ce chapitre, les interdépendances entre les deux problèmes de décision (tactique et opérationnelle). Dans un premier temps, nous introduisons en ensuite quelques travaux de planification hiérarchique temporelle qui servent de bases à notre proposition pour la planification tactique. Nous décrivons ensuite les algorithmes d'affectation des ITKs et de propagation des contraintes temporelles. Dans un second temps nous présentons l'algorithme de planification opérationnelle. Les expérimentations réalisées pour chacune de nos propositions sont également présentées dans ce chapitre.

5.1 Résolution des problèmes de décisions tactique et opérationnelle

5.1.1 Rappel du problème de décision tactique

5.1.1.a De la décision stratégique à la décision tactique

La décision tactique est un problème de planification intra-annuelle qui consiste à choisir le mode de conduite adéquat pour chacun des couples culture/parcelle. Pour ce faire, les résultats de la planification stratégique, que nous notons Π^{str}, sont exploités afin de décider annuellement la façon de conduire Π^{tac} (cf. Figure 5.1). Cela revient à choisir pour chacune des parcelles, l'enchaînement des opérations agricoles (cf. section 1.3.2.b) qui permettra d'atteindre les objectifs de production de l'agriculteur. Chaque opération agricole est soumise à un ensemble de contraintes temporelles (durée d'exécution, période d'activation et période de fermeture) et de ressources.

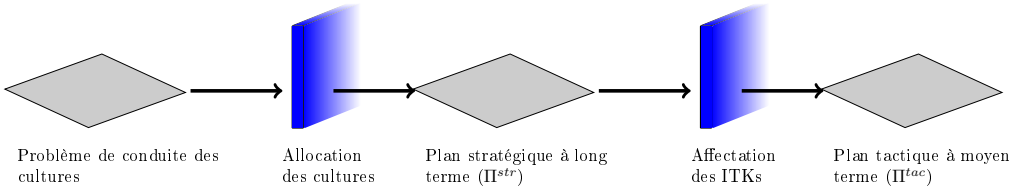


Figure 5.1 – *Principe du lien entre la planification stratégique et la planification tactique.*

Les plans d'actions résultants de la planification tactique, définissent les modes de conduite des cultures allouées durant la phase de planification stratégique. Les différents modes de conduite, encore appelés *itinéraires techniques* (ITK cf. Définition 1.10), sont prédéfinis. Il en existe plusieurs pour chaque culture. Par exemple, dans notre cas d'étude (cf. section 1.6), l'agriculteur dispose, pour chaque culture, de trois modes de conduite possibles. Chacun d'eux permet d'atteindre un objectif spécifique qui peut être de maximiser le rendement, de minimiser les intrants ou de minimiser la charge de travail.

5.1.1.b Le problème de décision tactique et la mesure de l'utilité des ITK

Pour un plan stratégique donné, la décision tactique consiste à associer un itinéraire technique à chaque couple culture/parcelle de manière à atteindre les objectifs de production de l'agriculteur. Le problème de décision associé est un problème de planification d'actions duratives (opérations agricoles) dans l'espace des plans partiels (ITK).

Dans la section 1.3.3 nous avons mentionné que l'utilité d'un ITK de culture par rapport aux autres dépend :

- de l'ensemble des parcelles d'un même bloc fonctionnel, affectées à la même culture,
- de la disponibilité des ressources,
- des effets attendus de l'ITK sur les objectifs de production de l'exploitation (ex : rendement, charge de travail).

En effet, l'affectation d'un ITK à une parcelle d'un bloc fonctionnel donné, conditionne sa réalisation sur l'ensemble des parcelles du bloc fonctionnel affecté à la même culture. Autrement dit, il n'existe dans un bloc fonctionnel, qu'un et un seul itinéraire technique pour chaque sous ensemble de parcelles affecté à la même culture (cf. Définition 1.9).

5.1.1.c Horizon de la planification tactique

Les plans recherchés portent sur un horizon de quelques mois voir une année au plus. Cette taille d'horizon correspond à la durée de production [1] des cultures.

5.1.1.d Nature des plans tactiques recherchés

Les ITKs solutions sont des plans partiellement instanciés, temporellement consistants, réalisables en fonction des capacités de ressources et dont l'exécution nécessite une allocation explicite de ressources. En effet, une instanciation sur une année entière des dates d'exécution et des affectations de ressources d'une opération agricole ne pourraient prendre en compte l'incertitude intrinsèque de la conduite des systèmes de culture (ex : le climat).

5.1.2 Rappel du problème de décision opérationnelle

5.1.2.a De la décision tactique à la décision opérationnelle

Sur la base des plans tactiques Π^{tac} (itinéraire technique affectés aux cultures) la décision opérationnelle consiste à assigner des ressources aux différentes opérations agricoles de manière à respecter les contraintes temporelles et de ressources (cf. Figure 5.2). Ainsi, cette phase de décision s'attache à l'ordonnancement Π^{ope}, c'est-à-dire, au positionnement dans le temps (date de début) et dans l'espace (affectation aux ressources) des tâches en fonction des moyens de production de l'exploitation.

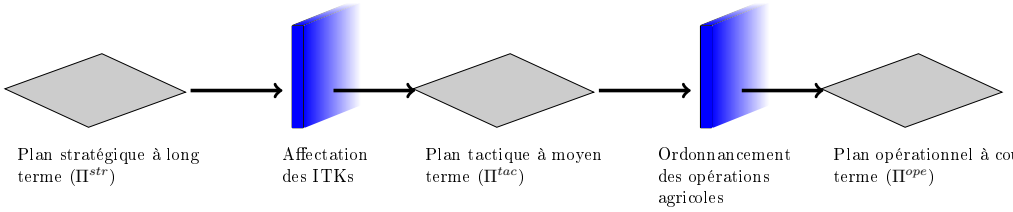


FIGURE 5.2 – *Principe du lien entre la planification tactique et la planification opérationnelle*

5.1.2.b Mesure de l'utilité du plan opérationnel

Plusieurs ordonnancements sont possibles pour l'organisation du travail.Il faut donc pouvoir en choisir un qui respecte les préférences d'organisation du travail de l'agriculteur. Dans notre cas, nous cherchons à minimiser l'étendue temporelle de l'ordonnancement (*makespan*). En d'autre terme, considérant l'horizon de l'ordonnancement, il s'agit d'allouer des ressources aux tâches à exécuter de manière à minimiser la durée entre la date de début de la première opération agricole et la date de fin de la dernière opération agricole. Notons cependant que cette thèse n'apporte aucune preuve d'optimalité des ordonnancements obtenus.

5.1.2.c Horizon de la planification opérationnelle

L'ordonnancement recherché porte sur un horizon de quelques jours. Cette taille d'horizon correspond à la durée sur laquelle l'agriculteur souhaite anticiper son organisation du travail sur l'ensemble de l'exploitation.

1. Généralement, il s'agit de la période allant du travail du sol avant le semis à la préparation du sol pour la culture suivante.

5.1.3 Choix des approches de résolution

Pour choisir les méthodes de résolution de ces deux problèmes, nous partons du lien entre la décision tactique Π^{tac} et la décision opérationnelle Π^{ope} (cf. Figure 5.2). La première, nous permet de répondre à la question : *Que fait t-on pour produire les cultures allouées dans la phase stratégique ?* Cela revient à déterminer les tâches à entreprendre pour atteindre les objectifs de production de l'agriculteur. La seconde nous permet de répondre à la question *Quand et comment réaliser ce qui a été prévu de faire ?* Cela revient à déterminer quelles ressources affecter aux tâches pour assurer leur réalisation et comment organiser ces tâches dans le temps.

Pour répondre à la première question nous utilisons des techniques de planification notamment celles liées à la planification hiérarchique (HTN). Quant à la deuxième question, elle sera abordée en utilisant des techniques inspirées de la recherche opérationnelle notamment les algorithmes de coloration de graphe.

5.2 État de l'art sur la planification hiérarchique temporelle

Nous avons indiqué dans la section 2.1.3.c l'intérêt que nous portons aux approches de planifications hiérarchiques afin d'aborder le problème de décision tactique. Contrairement aux approches de planification classique, le but de la planification hiérarchique est d'exécuter un plan. Nous pouvons justifier ce choix en soulignant la capacité des planificateurs hiérarchiques à prendre en compte les connaissances expertes spécifiques au domaine. En ce qui nous concerne, il s'agit notamment de l'existence des itinéraires techniques. En effet, l'agriculteur ne réinvente pas à chaque fois la manière de conduire ses cultures à partir d'un objectif final qui serait de produire une culture. Il dispose, via ses itinéraires techniques, de connaissances suffisantes sur les différentes manières de résoudre le problème de conduite de culture.

L'autre avantage de cette approche est relatif au fait qu'elle est plus efficace en ligne car elle réduit l'espace de recherche. En contrepartie, elle nécessite l'utilisation d'heuristiques de décomposition. Dans notre cas, l'heuristique de décomposition doit intégrer la notion « d'exécutabilité » d'une décomposition au vue des autres décompositions choisies. Cela résulte du fait que chaque décomposition est consommatrice de ressources. Or, les disponibilités de ressources sont définies au niveau global de l'exploitation.

Dans cette section 5.2, nous présentons les approches de planification hiérarchique et la prise en compte des contraintes temporelles. Nous décrivons plus particulièrement, dans la section 5.2.2.c, les travaux de Castillo et al. (2005) autour de la prise en compte des contraintes temporelles dans les réseaux de tâches hiérarchiques.

5.2.1 Les réseaux de tâches hiérarchiques

Dans un réseau de tâches hiérarchiques (*Hierarchical Task Network* HTN) on retrouve des *tâches primitives*, des *tâches composées* et des *méthodes* de décomposition.

5.2.1.a Les tâches primitives et opérateurs

Les tâches primitives correspondent à des actions dans la planification de type STRIPS. Une tâche primitive est définie par l'application d'un opérateur de tran-

sition d'état. Ces opérateurs définissent, via des prédicats, les préconditions et les effets d'une tâche. Les préconditions d'un opérateur représentent les conditions de mise en œuvre de la tâche. Autrement dit, elles définissent les états dans lesquels la tâche est applicable. Les effets d'un opérateur définissent les propriétés du monde après l'exécution de la tâche.

En conséquence, si une tâche est définie par un opérateur qui transforme un état courant s en un état s', les effets de la tâche sont représentés par des prédicats rendus vrais (eff^+) et les prédicats rendus faux (eff^-) afin de passer de s à s'.

Définition 5.1 (Opérateur) *Un opérateur o est défini par*

$$o \quad = \quad (name(o), prec(o), eff(o))$$

où :

- ▷ *$name(o)$: représente le nom de l'opérateur. Il est défini par $a(h_1, h_2, \ldots, h_n)$ où a est un symbole identifiant l'opérateur et $h_1, h_2, \ldots, h_n$ représentent les paramètres de l'opérateur.*
- ▷ *$prec(o)$: définit les préconditions de l'opérateur,*
- ▷ *$eff(o)$: définit les effets de l'opérateur.*

Définition 5.2 (Tâche primitive) *Une tâche primitive est une instance d'opérateur. Si la tâche a est exécutable dans l'état s alors $prec(a) \subseteq s$. Le résultat de l'exécution de a est l'état $s' = \delta(s, a) = (s - eff_a^-) \cup eff_a^+$ avec δ la fonction de transition de s vers s'.*

Par exemple, une opération agricole de semis peut être considérée comme une tâche primitive.

5.2.1.b Les tâches composées et méthodes de décomposition

Une tâche *composée* est associée à des *méthodes* qui permettent la décomposition de la tâche en un réseau de sous-tâches (cf. Définition 5.3). Elles décrivent des plans ordonnés (partiellement ou totalement) de tâches en y ajoutant les conditions d'exécution de ces dernières.

Définition 5.3 (Réseau de tâches) *Un réseau de tâches est un graphe acyclique défini par un tuple $\langle \mathcal{A}, \mathcal{C} \rangle$ où :*

- ▷ *$\mathcal{A}$: est un ensemble de nœuds représentant les tâches,*
- ▷ *$\mathcal{C}$: est un ensemble d'arcs représentant les contraintes d'ordre entre les tâches.*

Chaque contrainte $\mathcal{C}_i \in \mathcal{C}$ spécifie les conditions nécessaires pour les plans solutions du HTN. Généralement, il s'agit de :

- *contraintes d'ordre* (total ou partiel) qui décrivent les précédences entre les tâches.
- *conditions* qui décrivent les conditions d'exécution d'une tâche.

Définition 5.4 (Méthode Ghallab et al. (2004)) *Une méthode HTN m est définie par :*

$$m \quad = \quad (name(m), task(m), \mathcal{A}(m), \mathcal{C}(m))$$

- ▷ *$name(m)$: est une expression de la forme $n(h_1, h_2, \ldots, h_k)$ où n représente le nom de la méthode et $h_1, h_2, \ldots, h_k$ ses paramètres,*
- ▷ *$task(m)$: est une tâche composée,*
- ▷ *$(\mathcal{A}(m), \mathcal{C}(m))$: est un réseau de tâches (cf. Définition 5.3).*

Par exemple, si l'on considère une tâche composée « maïs » , qui consiste à réaliser la culture de maïs, chacun des itinéraires techniques ↓ *travail*, ↑ *rendement* et ↓ *intrant* (cf. section 1.6.3.c) décrit différentes méthodes pour le « maïs » . Ces méthodes permettent d'atteindre des objectifs spécifiques. Dans de Silva et Padgham (2004); de Silva et al. (2009), les auteurs ont établi un lien entre les méthodes HTN et les buts d'un agent BDI.

5.2.1.c Problème de planification hiérarchique

Un problème de planification hiérarchique peut s'exprimer par un tuple $\langle s_0, \pi_0, \mathcal{O}, \mathcal{M}\rangle$ où s_0 est l'état initial, π_0 le réseau de tâches initial, $\mathcal{O}$ l'ensemble des opérateurs et $\mathcal{M}$ l'ensemble des méthodes de décomposition.

5.2.1.d Algorithme de planification hiérarchique

La recherche des solutions se fait par une réduction des tâches composées en tâches primitives. Le principe de la planification hiérarchique consiste à rechercher les opérateurs et méthodes applicables afin de décomposer les tâches composées en un plan « solution » composé uniquement de tâches primitives. Cela revient à tester et à faire évoluer les variables des préconditions en partant de l'état courant s.

Pour ce faire, le planificateur reçoit en entrée un ensemble d'*opérateurs* et de *méthodes*. L'Algorithme 6 décompose récursivement, et en profondeur d'abord, des tâches composées jusqu'à l'obtention d'une séquence de tâches primitives. La condition d'arrêt de la récursion est atteinte dès que le réseau de tâches est vide.

On initialise l'algorithme avec l'état initial s_0, et un réseau des tâches π_0 donné en entrée du planificateur. Dans un premier temps, on teste si la liste des tâches est vide (ligne 2). Dans le cas où ce test est vrai, on déduit qu'une solution à été trouvée. Dans le cas contraire, l'algorithme tente de réaliser l'une des tâches a_u (ligne 3) n'ayant pas de prédécesseur.

Algorithme 6 : `Reduction-HTN(`s_0,π_0,$\mathcal{O}$,$\mathcal{M}$`)` Ghallab et al. (2004)

```
1  début
2      si π0 = ∅ alors retourner plan vide
3      au ← choisir une tâche u ∈ π0 telle que predecesseur(u) = ∅
4      restant ← π0 − {u}
5      si au est une tâche primitive alors
6          actif ← {(o, σ)|o ∈ O, o est applicable dans s0 ,
7                        σ une substitution telle que name(o) = σ(au)}
8          si actif = ∅ alors Retour echec
9          choisir(o, σ) ∈ actif
10         p ← Reduction-HTN(δ( s0 , o), σ(restant),O,M)
11         si p = echec alors Retour echec
12         sinon retourner append(p,o)
13     sinon
14         actif ← {(m, σ)|m ∈ M, m est applicable dans s,
15                       σ une substitution telle que name(m) = σ(au)}
16         si actif = ∅ alors Retour echec
17         choisir(m, σ) ∈ actif
18         choisir une décomposition d pour m
19         retourner Reduction-HTN(s0, d,O,M)
20 fin
```

Si la tâche a_u est une tâche primitive (lignes de 5 à 12), alors pour chaque opérateur o, l'algorithme teste la substitution $\sigma(a_u)$ en utilisant l'opérateur o. L'ensemble des substitutions possibles est stocké dans $actif$. L'algorithme échoue en l'absence de substitutions applicables. Dans le cas contraire, l'une des substitutions est sélectionnée. Pour la substitution considérée, l'algorithme déduit un nouvel état $\delta(s_0, o)$, résultant de l'application des effets de l'opérateur o. Ensuite, un appel récursif est réalisé sur le nouvel état.

Si la tâche a_u est une tâche composée (lignes de 13 à 19), la procédure cherche à appliquer une méthode m contenue dans l'ensemble des méthodes $\mathcal{M}$ du domaine. Puis l'algorithme teste, pour chaque décomposition d de la méthode, les préconditions de la même façon que pour une tâche primitive. Pour chaque substitution σ trouvée, il crée et remplace la tâche a_u par le réseau de tâches spécifié dans la décomposition d de la méthode m.

Dans la section suivante, nous introduisons un cadre pour la prise en compte du temps. Cette présentation nous permettra d'aborder ensuite la prise en compte des contraintes temporelles dans les réseaux de tâches hiérarchiques.

5.2.2 Propagation des contraintes temporelles en planification

Nous avons expliqué dans la section 2.1.1.c l'importance de la représentation du temps en planification. Nous avons également expliqué qu'en planification, la notion de temps est généralement définie de manière implicite via les liens de causalité entre les effets et les préconditions des tâches. La représentation explicite du temps peut être réalisée soit en utilisant les relations temporelles symboliques (*Algèbre des intervalles* Allen (1984)), soit en utilisant les relations temporelles absolues ou des restrictions portant sur la distance entre des instants (*Algèbre des instants* Vilain et al. (1986)). La première représentation permet un raisonnement qualitatif alors que la seconde est plus adaptée au raisonnement quantitatif sur le temps. Marc Vilain explique que l'algèbre des instants est une sous-classe de l'algèbre des intervalles d'Allen. Cette sous-classe est capable de représenter les relations temporelles sous la forme d'une conjonction de relations entre des instants continus appartenant à des intervalles d'instants. L'algèbre des instants utilise des relations primitives $\{=, <, >, \leq, \geq, \neq\}$. Les avantages de cette représentation sont notamment, la spécification des intervalles sur les dates de début et de fin des tâches, et la formulation des contraintes numériques liées aux décalages temporels (*timelag*) entre tâches.

À l'instar de la plupart des planificateurs temporels (IxTeT (Ghallab et Laruelle, 1994), Europa (Frank et Jónsson, 2003), VHPOP (Younes et Simmons, 2002)), nous avons choisi d'utiliser l'algèbre des instants pour la représentation du temps.

Cette section décrit les éléments théoriques des réseaux des contraintes temporelles (TCSP *Temporal Constraint Satisfaction Problem*) proposés par Dechter et al. (1991). Nous nous focalisons plus particulièrement sur la version simplifiée appelée STN (*Simple Temporal Networks*). Nous décrivons la formulation et la propagation des contraintes temporelles dans un STN.

5.2.2.a Problème de satisfaction de contraintes temporelles

Les réseaux de contraintes temporelles peuvent être utilisés pour la représentation de plans temporels. Ces derniers sont constitués d'un ensemble de tâches ordonnées (partiellement ou totalement) soumises à des relations temporelles :

– *absolues* portant sur les instants de débuts et de fin des tâches,

– *relatives* portant sur la durée des tâches et les restrictions liées aux distances entre les instants de débuts et de fin des tâches.

Chaque tâche du plan est associée à deux événements appelés *instants* ou *time-points*. Plus particulièrement, le début de chaque tâche a est associé à un instant de début s_a tandis que la fin est associée à un instant de fermeture f_a. Les instants de début et de fin sont séparés par une durée positive.

Les contraintes temporelles d'un plan sont représentées par un ensemble d'inégalités qui contraignent les instants de début et de fin des tâches.

Exemple 5.1 *Considérons le plan décrit par l'affirmation suivante. « L'irrigation du maïs est réalisée entre le 15 mai et le 15 novembre. Chaque irrigation a une durée d'un jour. Deux occurrences successives d'irrigation (irr1 et irr2) sont toujours séparées de 9 jours ». Cette affirmation peut être convertie en contraintes temporelles associées aux inégalités :*

$$\begin{aligned} 15\ mai \leq s_{irr1} \leq 15\ nov. \qquad & 15\ mai \leq f_{irr1} \leq 15\ nov. \\ 15\ mai \leq s_{irr2} \leq 15\ nov. \qquad & 15\ mai \leq f_{irr2} \leq 15\ nov. \\ 1 \leq f_{irr1} - s_{irr1} \leq 1 \qquad & 1 \leq f_{irr2} - s_{irr2} \leq 1 \\ 9 \leq \quad f_{irr1} - s_{irr2} \quad & \leq \infty \end{aligned}$$

De manière générale, l'ensemble des inégalités constitue un réseau de contraintes temporelles (*Temporal Constraint Satisfaction Problem*) Dechter et al. (1991).

Définition 5.5 (Problème de satisfaction de contraintes temporelles TCSP) *Un problème de satisfaction de contraintes temporelles est défini par un tuple $\langle \mathcal{X}, \mathcal{I}, \mathcal{C} \rangle$ où :*

- ▷ $\mathcal{X}$ *: est un ensemble $x_k \in \mathcal{X}$ de variables d'instants continus ou discrets associés à l'apparition d'événements,*
- ▷ $\mathcal{I}$ *: est un ensemble d'intervalles $\{I_0, \cdots I_n\} = \{[l_0, u_0], \cdots, [l_n, u_n]\}$*
- ▷ $\mathcal{C}$ *: est un ensemble de contraintes*
  - unaires disjonctives $\mathcal{C}_k$ $((l_0 \leq x_k \leq u_0) \vee (l_1 \leq x_k \leq u_1) \dots)$ *qui restreignent le domaine de x_k à des intervalles $\{I_m, \cdots I_{m'}\} = \{[l_m, u_m], \cdots, [l_{m'}, u_{m'}]\}$,*
  - binaires disjonctives $\mathcal{C}_{kk'}$ $((l_0 \leq x_{k'} - x_k \leq u_0) \vee (l_1 \leq x_j - x_i \leq u_1) \dots)$ *qui restreignent la distances entre les instants x_k et $x_{k'}$ à des intervalles $\{I_m, \cdots I_{m'}\} = \{[l_m, u_m], \cdots, [l_{m'}, u_{m'}]\}$.*

La solution d'un TCSP est une assignation complète A des variables x_k qui satisfait l'ensemble des contraintes $\mathcal{C}_k$ et $\mathcal{C}_{kk'}$. L'ensemble des valeurs possibles d'une variable x_k est appelé le *domaine minimal*. La résolution d'un TSCP est un problème NP difficile.

Dans notre cas, nous utilisons une version simplifiée des réseaux de contraintes temporelles. Cette dernière permet de représenter des réseaux pour lesquels les contraintes sont toutes non-disjonctives.

5.2.2.b Réseaux de contraintes temporelles simples

Les réseaux de contraintes temporelles simples STN (*Simple Temporal Networks*) sont des cas particuliers de TCSP. Ils sont considérés comme simples, en raison de l'absence de contraintes disjonctives. En d'autres termes, les contraintes n'autorisent qu'un seul intervalle entre chaque paire d'instants. Les STN sont largement utilisés en planification/ordonnancement car ils permettent de réaliser dynamiquement et de manière rapide des tests de consistances temporelles. De plus, l'expressivité des STN permet de représenter plusieurs problèmes issus du monde réel.

Plus formellement, un STN est défini comme ci-dessous.

Définition 5.6 (Simple Temporal Networks STN (Dechter et al., 1991)) *Un STN est définit par un tuple* $\langle \mathcal{X}, \mathcal{I}, \mathcal{C} \rangle$ *où :*

- $\mathcal{X} = \{x_0, \cdots, x_k, \cdots, x_N\}$ *: est un ensemble de variables d'instants,*
- $\mathcal{I}$ *: est un ensemble d'intervalles* $\{I_0, \cdots I_n\} = \{[l_0, u_0], \cdots, [l_n, u_n]\}$
- $\mathcal{C} = \{\mathcal{C}_{kk'}\}$ *: est un ensemble de contraintes binaires portant sur les variables* $\mathcal{X}$.

Chaque contrainte $\mathcal{C}_{kk'}$ *entre deux instants* x_k *et* $x_{k'}$*, contraint les bornes inférieure et supérieure de l'intervalle* $[l_{kk'}, u_{kk'}]$ *de sorte que* $l_{kk'} \leq x_{k'} - x_k \leq u_{kk'}$*. Cette inégalité peut être aussi exprimée par la paire d'inégalités* $x_k - x_{k'} \leq -l_{kk'}$ *et* $x_{k'} - x_k \leq u_{kk'}$*. Ainsi, la forme générale des* $\mathcal{C}_{kk'}$ *est* $x_{k'} - x_k \leq \delta$ *avec* $\delta \in \mathbb{R}$.

La solution d'un STN est une assignation complète A des variables x_i qui satisfait l'ensemble des contraintes $\mathcal{C}_{kk'}$.

STN et représentation de plans temporels Afin de trouver, pour chaque tâche, les dates de début et de fin possibles, un plan temporel peut être représenté sous la forme d'un STN. Dans ce cas, les variables d'instants x_k du STN correspondent aux instants de début s_a et de fin f_a des tâches a du plan. x_0 est une variable particulière qui représente l'instant de référence de toutes les autres variables. Cette représentation est identique à celle que l'on retrouve dans les travaux tel que Williams et al. (2001b).

Représentation graphique d'un STN Un STN $\langle \mathcal{X}, \mathcal{I}, \mathcal{C} \rangle$ peut être représenté par un graphe orienté $G = (X, E)$ avec X l'ensemble des nœuds décrivant les variables $\mathcal{X}$ du STN, et E l'ensemble des arcs décrivant les relations entre les variables. Ainsi, chaque arc $(x_k \rightarrow x_{k'})$ représente les contraintes sur les bornes inférieure $l_{kk'}$ et supérieure $u_{kk'}$.

Exemple 5.2 *Soient deux tâches* labour *a et* Semis *b. Le labour dure sept jours et doit être réalisé dans l'intervalle* $[1^{er}, 30^{ième}]$ *jour du mois d'octobre. Le semis dure cinq jours et doit être réalisé dans l'intervalle* $[5^{ième}, 20^{ième}]$ *jour du même mois. Le labour précède le semis. De plus, ces deux tâches doivent être espacées de six à sept jours.*

La Figure 5.3 montre une représentation graphique du STN associé à l'exemple 5.2. Ce STN contient 5 variables d'instants et 7 relations représentées par les arcs. L'instant de référence x_0 est affecté au $30^{ième}$ jour du mois de septembre. Considérons l'arc $(x_0 \rightarrow s_a)$, l'intervalle $[1, \infty]$ indique la distance entre l'instant de référence et les dates de début au plus tôt (1) et au plus tard (∞ car cette contrainte n'est pas spécifiée par le problème). Ainsi, l'arc $(x_0 \rightarrow s_a)$ impose une borne inférieure de 1 jour $(x_0 - s_a \leq -1)$ et une borne supérieure indéfinie $(s_a - x_0 \leq \infty)$.

Graphe de distance d'un STN Un STN peut être représenté de manière équivalente par un *graphe de distance* (Dechter et al., 1991). Cette dernière représentation est plus intéressante pour la propagation de contraintes temporelles. Contrairement à la représentation précédente, les arcs d'un graphe de distance ne définissent qu'une seule contrainte.

Définition 5.7 (Graphe de distance) *Le graphe de distance d'un STN* $\langle \mathcal{X}, \mathcal{I}, \mathcal{C} \rangle$ *est un graphe orienté et pondéré* $G_d = (X, E_d)$*. Les nœuds* X *du graphe représentent*

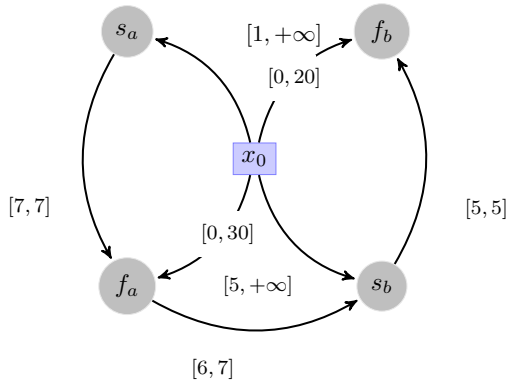


FIGURE 5.3 – *Réseau de contraintes temporelles simples STN.*

les variables $\mathcal{X}$ du STN. Les arcs orientés du graphe correspondent aux contraintes temporelles $\mathcal{C}$.

$$X = \mathcal{X} \quad et \quad E_d = \{(x_k \rightarrow x_{k'}, \delta) : (x_{k'} - x_k \leq \delta) \in \mathcal{C}_{kk'}\}$$

La valeur δ de l'arc correspond au poids (ou distance) de ce dernier.

Afin d'éviter toute confusion, soulignons que les arcs du graphe G décrivent des relations entre les variables d'instants, tandis que les arcs du graphe de distance G_d symbolisent des distances entre les variables d'instants. Pour transformer un STN en un graphe de distance, il suffit de remplacer les relations du graphe G en une paire d'arcs dirigés. Comme l'indique la Figure 5.3-a, chaque relation $(x_k \rightarrow x_{k'})$ du STN est remplacée par un arc de borne supérieure $(x_k \rightarrow x_{k'})$ de poids $\delta = u_{kk'}$ et un arc de borne inférieure $(x_{k'} \rightarrow x_k)$ de poids $\delta = -l_{kk'}$. La Figure 5.3-b présente le graphe de distance associé au STN de la Figure 5.3.

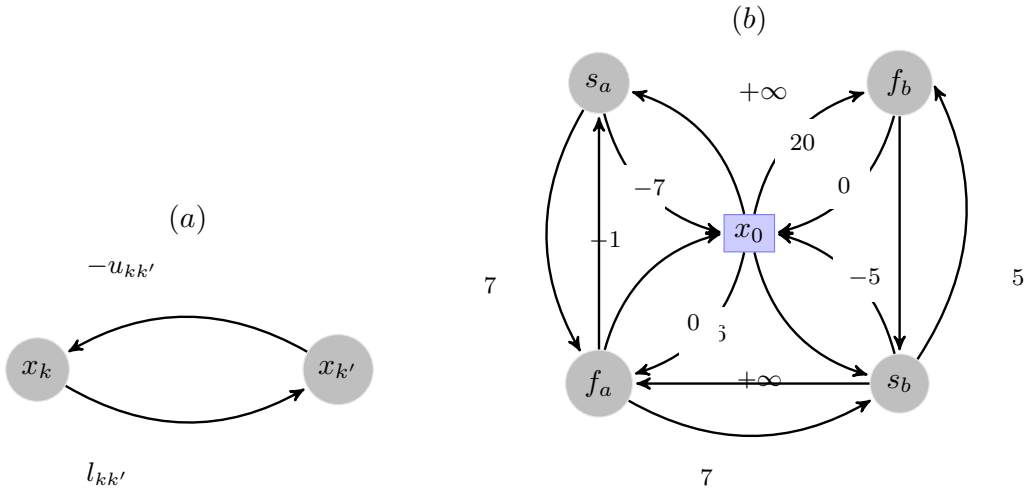


FIGURE 5.4 – *(a) Transformation des relations d'un STN en contraintes dans le graphe de distance, (b) Graphe de distance associé au STN de la Figure 5.3.*

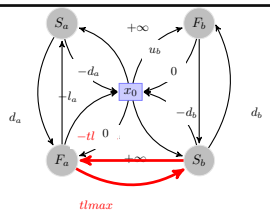
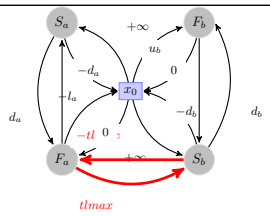
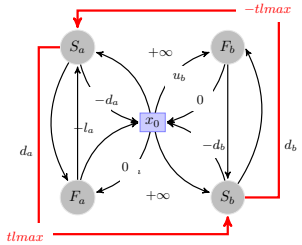
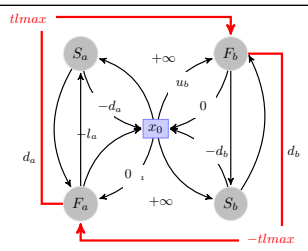
En fonction de la valeur des bornes inférieure et supérieure des relations d'un STN, leurs reformulations dans le graphe de distance G_d permettent de décrire des contraintes :

1. de *bornes inférieure et supérieure* $(u_{kk'} > l_{kk'})$ dont la reformulation traduit une précédence de x_k par rapport à $x_{k'}$,

2. d'*inflexibilité* ($u_{kk'} = l_{kk'}$) dont la reformulation traduit l'absence de flexibilité sur la distance entre les instants x_k et $x_{k'}$. Le cas particulier où $u_{kk'} = l_{kk'} = 0$ traduit une *simultanéité* des événements liés aux instants x_k et $x_{k'}$.

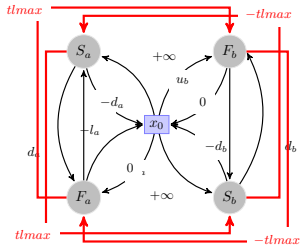
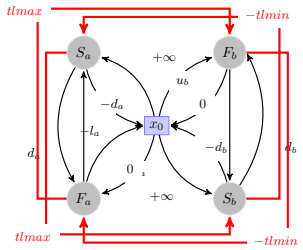
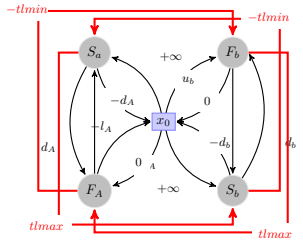
Le tableau 5.1 présente des exemples de reformulation des relations primitives d'Allen en graphe de distance. Pour cela, considérons deux tâches a (s'exécutant entre $[l_a, u_a]$) et b (s'exécutant entre $[l_b, u_b]$) qui durent respectivement d_a et d_b. En fonction des types de relations, la distance $tl_{a,b}$ (« timelag ») entre le début (ou la fin) de a et le début (ou la fin) de b est comprise entre $[tlmin, tlmax]$.

TABLE 5.1 – *Reformulation des relations primitives d'Allen entre deux tâches a et b en graphe de distance.*

RELATIONS TEMPORELLES	CONTRAINTES	⇒	TRADUCTION EN GRAPHE DE DISTANCE
a **before** b	A ——F⋯⋯→ B ⋯⋯S——→ $(S_a < F_a) \wedge (F_a < S_b) \wedge (S_b < F_b)$	⇒	
a **meet** b	A ——F⋯⋯→ B ⋯⋯S——→ $(S_a < F_a) \wedge (F_a = S_b) \wedge (S_b < F_b)$	⇒	 $tlmin = tlmax$
a **start** b	A ⋯S——→ B ⋯S——→ $(S_a < F_a) \wedge (S_a = S_b) \wedge (S_b < F_b)$	⇒	 $tlmin = tlmax$
a **finish** b	A ——F⋯⋯→ B ——F⋯⋯→ $(S_a < F_a) \wedge (F_a = F_b) \wedge (S_b < F_b)$	⇒	 $tlmin = tlmax$

suite sur la page suivante ...

TABLE 5.1 – *Reformulation des relations primitives d'Allen entre deux tâches a et b en graphe de distance.*

RELATIONS TEMPORELLES	CONTRAINTES	⇒	TRADUCTION EN GRAPHE DE DISTANCE
a **equal** b	A ···▬··· (S, F) B ···▬··· (S, F) $(S_a = S_b) \wedge (S_b < F_a) \wedge (F_a = F_b)$	⇒	 $tlmin = tlmax$
a **overlap** b	A ···▬····· (S, F) B ···▬··· (S, F) $(S_a < S_b) \wedge (S_b < F_a) \wedge (F_a < F_b)$	⇒	
a **during** b	A ···▬····· (S, F) B ▬··· (S, F) $(S_a < F_a) \wedge (S_a > S_b) \wedge (F_a < F_b)$	⇒	

Propagation des contraintes temporelles Le graphe de distance offre un moyen de déduire les contraintes implicites d'un STN en combinant les inégalités. Ces contraintes implicites sont obtenues par une recherche de l'ensemble des plus courts chemins du graphe de distance. Or, trouver l'ensemble des plus courts chemins c'est trouver le domaine minimal de chacune des variables. Cela revient à tester la consistance temporelle du STN.

La consistance temporelle d'un STN peut être réalisée en temps polynomial en utilisant des algorithmes de types *Bellman-Ford Single-Source Shortest-Path* (Cormen et al., 1990; Dechter et al., 1991). En outre, d'autres algorithmes très connus tel que *Floyd-Warshall All-Pairs Shortest-Path* (Papadimitriou et Steiglitz (1982); Dechter et al. (1991)) ou l'algorithme de cohérence d'arcs PC-2 (Dechter, 2003) peuvent être utilisés afin de trouver le domaine minimal de chacune des variables.

Nous présentons ci-dessous un exemple d'algorithme (*Floyd-Warshall All-Pairs Shortest-Path* cf. Algorithme 7) de propagation des contraintes temporelles. La complexité de cet algorithme est en $O(|X^3|)$. Nous utilisons *Floyd-Warshall All-Pairs Shortest-Path* dans notre algorithme de planification tactique présenté dans la section 5.4.3.

L'application de l'algorithme 7 sur le graphe de distance de la Figure 5.3-

Algorithme 7 : Floyd-Warshall All-Pairs Shortest-Path

```
1 début
2 |  pour i ← 1 to n faire d_ii ← 0 ;
3 |  pour i,j ← 1 to n faire  d_ij ← a_ij ;
4 |  pour k ← 1 to n faire
5 |  |  pour i,j ← 1 to n faire  d_ij ← min{d_ij, d_ik + d_kj} ;
6 fin
```

b permet d'obtenir le domaine minimal de chaque variable. Ces domaines sont représentés dans les cellules du tableau 5.2.

TABLE 5.2 – *Réseau minimal résultant de l'application de l'algorithme 7 sur le graphe de distance de la Figure 5.3-b*

	x_0	s_a	f_a	s_b	s_b
x_0	[0 , 0]	[1 , 2]	[8 , 9]	[14 , 15]	[19 , 20]
s_a		[0 , 0]	[7 , 7]	[13 , 14]	[18 , 19]
f_a			[0 , 0]	[6 , 7]	[11 , 12]
s_b				[0 , 0]	[5 , 5]
f_b					[0 , 0]

Les lignes du tableau 5.2 indiquent les distances entre une variable d'instant donnée et chacune des variables d'instants du réseau. On peut donc voir que la distance entre l'instant de référence x_0 et le début s_a de la tâche a (respectivement la fin f_a) doit être comprise dans l'intervalle $[1, 2]$ (respectivement $[8, 9]$). Les lignes s_a, f_a, s_b et f_b indique les contraintes implicites entre les variables d'instants du réseau.

Cet exemple permet d'illustrer au passage la possibilité de construire grâce aux STN des plans temporellement flexibles.

Dans la section suivante, nous introduisons les travaux de Castillo et al. (2005) autour de la prise en compte des contraintes temporelles dans les HTN.

5.2.2.c Contraintes temporelles et planification hiérarchique

Bien qu'ayant été utilisés pour de nombreuses applications du monde réel, les planificateurs HTN classiques apportent très peu de réponses à la prise en compte du temps. Or, étant polynomiaux, les algorithmes de propagations de contraintes temporelles définis pour les STN peuvent être exploités afin de compenser cette limite des planificateurs HTN classiques. Le planificateur SIADEX (Castillo et al., 2005, 2006) a été développé dans cette optique. Les auteurs proposent la première version d'un planificateur temporel hiérarchique capable de supporter une représentation très riche des connaissances relatives à la notion de temps. Il s'agit notamment des dates de début et de fin, des relations temporelles d'enchaînements et de synchronisations.

En se basant sur les réseaux de contraintes temporelles simples (STN), SIADEX introduit les mécanismes d'extraction et de propagation des contraintes. Les STN sont donc utilisés pour définir les réseaux de tâches associés aux tâches composées. Chaque contrainte temporelle dans SIADEX exprime soit les dates de débuts, de fins, soit les relations temporelles entre les tâches d'un réseau.

Pour représenter le problème de planification, SIADEX se base sur la représentation PDDL 2.2 (Edelkamp et al., 2004). Cette représentation permet d'exprimer facilement les STN.

Dans SIADEX, la propagation des contraintes temporelles est réalisée par une version modifiée de l'algorithme de cohérence de chemin **Path Consistency** (PC-2 Dechter (2003)).

5.2.3 Prise en compte de contraintes de ressources

5.2.3.a Familles de ressources dans une exploitation agricole

De manière générale, les ressources sont des données physiques dont les disponibilités conditionnent la réalisation des tâches. Dans la section 2.1.1.d, nous avons présenté les deux grandes familles de ressources (**consommables** et **réutilisables**).

Dans le cas de conduites de systèmes de culture à l'échelle de l'exploitation, on retrouve ces deux familles de ressources. Pour les ressources consommables il s'agit généralement des quotas d'eau disponibles pour l'irrigation. Les ressources réutilisables sont couramment des ressources humaines et matérielles (ex : ouvriers, tracteurs, cultivateurs).

5.2.3.b Contraintes de ressources

Une contrainte de ressources exprime la manière dont une tâche a affecte la capacité d'une ressource r. Une tâche peut affecter plusieurs ressources.

Les contraintes de ressources sont définies par un tuple $\langle q(r,a), [s_a, f_a]\rangle$ où $q(r,a)$ est une variable de décision définissant la quantité de la ressource r consommée (si $q(r,a) \leq 0$) ou produite (si $q(r,a) \geq 0$) par a sur un intervalle de temps $[s_a, f_a]$ durant lequel la tâche a affecte la disponibilité de la ressource r. Notons cependant que dans notre cas, les changements des capacités de ressources au démarrage (date $t = s_a$) et à la fin (date $t = f_a$) d'une tâche sont instantanés. Les consommations et les productions continues ne sont donc pas considérées dans cette thèse.

Par exemple, $\langle q(tracteur2, semisMA) = -1, [s_{semis}, f_{semis}]\rangle$ est une contrainte de ressources spécifiant que la tâche $semisMA$ (semis de maïs) consomme une unité du « *tracteur2* » entre s_{semis} et f_{semis}. La disponibilité de *tracteur2* sera alors décrémentée de 1 à la date s_{semis} puis incrémentée de 1 à la date f_{semis}.

5.2.3.c Interdépendance entre planification et ordonnancement

La gestion des ressources implique des mécanismes d'ordonnancement c'est-à-dire des mécanismes permettant pouvoir manipuler l'ordre des tâches et leurs allocations de ressources.

Les problématiques de planification et d'ordonnancement sont traitées en profondeur aussi bien en intelligence artificielle qu'en recherche opérationnelle. De manière générale dans les problèmes nécessitant des phases de planification et d'ordonnancement, ces deux processus doivent s'exécuter successivement. Cela n'empêche pas que des planificateurs fassent également de l'ordonnancement [2].

Il existe des interactions évidentes entre la planification et l'ordonnancement. Dans le cas où plusieurs plans permettent d'atteindre les objectifs de la planification, des travaux proposent d'intégrer des notions spécifiques sur les ressources

2. Le contraire est aussi valable. Certains systèmes d'ordonnancement font aussi de la planification.

au sein d'un planificateur (Wilkins, 1984, 1988). D'autres incorporent des raisonnements sur la construction de plans au sein des systèmes d'ordonnancement (Currie et Tate, 1991; Tate et al., 1994; Drabble et al., 1994). Enfin, certains proposent des approches consistant à intégrer les processus de planification et d'ordonnancement puis de mettre en œuvre des mécanismes permettant de coordonner les décisions résultantes de chacun d'eux (Laborie et Ghallab, 1995; Lemai, 2004).

Dans notre cas, nous nous intéresserons plus spécifiquement à la coordination des systèmes de planification et d'ordonnancement. Cette approche semble être plus appropriée pour la mise en œuvre de notre architecture et notamment pour entrelacer de manière continue les phases de planifications tactiques et opérationnelles. Toutefois, notre système de planification tactique doit intégrer des mécanismes permettant de raisonner de manière globale sur l'utilisation des ressources en fonction des itinéraires techniques choisis.

Dans la suite de cette section, nous présentons (section 5.2.3.d) très brièvement quelques systèmes de planification intégrant la problématique de gestion des ressources. Nous nous focalisons ensuite (section 5.2.3.e) sur les problèmes de plus court chemin sous contraintes de ressources que nous utiliserons pour résoudre le problème d'affectation d'itinéraire technique aux cultures.

5.2.3.d Gestion des ressources en planification : quelques travaux en IA

Généralement dans les planificateurs, les mécanismes d'ordonnancement sont directement intégrés dans la planification.

SIPE (Wilkins, 1984) est le premier système de planification à intégrer la notion de ressources. Il représente les ressources consommables et se base sur cette connaissance pour élaguer les branches sur-consommatrices de ressources de l'arbre de recherche. SIPE n'intègre pas une représentation explicite des contraintes permettant d'éviter les conflits [3]. Il utilise cependant un ordonnancement des conflits d'accès aux ressources afin d'intégrer la notion de ressources non-partageables [4].

Dans O-Plan (Currie et Tate, 1991) la prise en compte des contraintes de ressources est réalisée en définissant des profils optimistes et pessimistes d'utilisation des ressources (Drabble et al., 1994). Cette approche lui permet de détecter et de résoudre les conflits.

Dans le planificateur IxTeT, la détection des conflits est réalisée en utilisant un algorithme de recherche de cliques dans un graphe (Laborie et Ghallab, 1995). Cette approche de gestion des ressources a été étendue par Lemai (2004) pour prendre en compte des consommations ou des productions variables.

Il existe de nombreuses autres approches permettant de raisonner sur les ressources. À ce jour, la plupart d'entre elles se base sur la propagation de contraintes dans l'espace de plans ou d'actions (Laborie, 2003)

5.2.3.e Problème de plus court chemin sous contrainte de ressources

Nous nous intéressons au problème de plus court chemin sous contrainte de ressources car nous l'utilisons comme heuristique de décomposition pour la planification tactique.

3. Un conflit de ressource peut être défini comme un ensemble d'intentions de consommation de ressources pour lesquelles il existe des éventualités de recouvrement temporel conduisant à une sur-consommation de la quantité de ressources disponibles.

4. Voir la section 2.1.1.d pour la définition des ressources non-partageables.

Problèmes de plus court chemin Les problèmes de plus court chemin entre deux nœuds d'un graphe pondéré sont des problèmes largement étudiés en recherche opérationnelle. Il existe pour cette classe de problèmes des algorithmes polynomiaux. Dans le cas de poids positifs les algorithmes de types Dijkstra sont généralement intéressants. La consommation des ressources disponibles en quantités limitées le long du chemin rend le problème NP-difficile. Cette classe de problème est connue sous le nom de problème de plus court chemin sous contrainte de ressources. (*Shortest Path Problem with Resource Constraints - SPPRC* Irnich et Desaulniers (2005)).

Problème de plus court chemin sous contraintes de ressources SPPRC Le SPPRC est un problème d'optimisation combinatoire très classique généralement utilisé dans le domaine du transport. Comme son nom l'indique, le SPPRC est une extension du problème de recherche d'un plus court chemin dans un graphe pondéré. Sa spécificité repose sur le fait que des contraintes de ressources interdisent certains chemins. Pour chaque ressource, on définit :

- une *fonction d'extension de ressource* pour chaque arc du graphe. Cette fonction indique l'évolution de la consommation de la ressource lorsqu'un arc est emprunté (Irnich et Desaulniers, 2005) ;
- une *contrainte de ressource* pour chaque nœud du graphe. Cette contrainte indique une valeur maximale sur la consommation de la ressource permettant d'atteindre le nœud.

Il existe des algorithmes de programmation dynamique efficaces permettant de résoudre les SPPRC.

Exemple 5.3 *Considérons la graphe représenté par la Figure 5.5. Il représente un SPPRC entre s et p. Dans ce problème, on considère une ressource. Moyennant un coût, chaque arc consomme une certaine quantité de cette ressource. Le cumul de la quantité consommée est indiqué près des nœuds.*

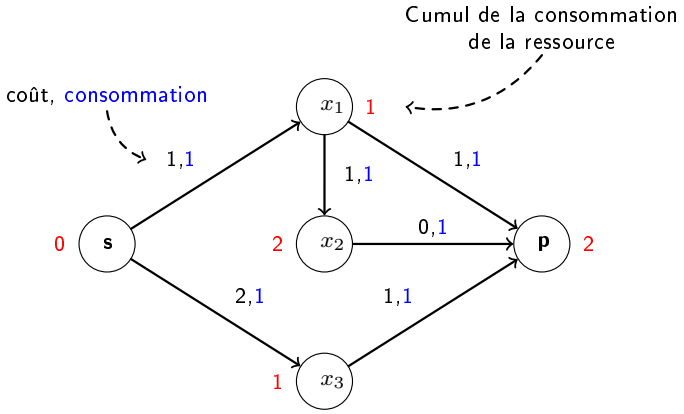


FIGURE 5.5 – *Exemple d'un SPPRC avec une ressource.*

Dans la section suivante, nous présentons la modélisation des SPPRC en nous focalisant spécifiquement sur les SPPRC élémentaires (ESPPRC) dans lesquels le chemin recherché ne passe que une et une seule fois par les nœuds du chemin.

Modélisation des ESPPRC Soit $G(X, E, R)$ un graphe multivalué. On note $x_i \in X$ $(i \in \{s, p\} \cup \{1, \ldots, N\})$ les nœuds du graphe. Les nœuds s et p sont respectivement la source et le puits. Chaque arc $(x_i \rightarrow x_j)$ est associé à une consommation $q(r, (x_i \rightarrow x_j))$ de la ressource $r \in \{0, \ldots, K\}$. Par définition, la ressource 0 est

associée au coût à payer pour traverser l'arc. En dehors de la ressource 0, les contraintes portant sur les autres ressources sont associées aux nœuds du graphe. On notera $Q_i(r)$ la consommation cumulée de la ressource r le long du chemin allant de la source s au nœud x_i. La consommation de la ressource r sur le nœud x_i est bornée par $[l_r^i, u_r^i]$, où l_r^i indique le seuil de consommation de la ressource r. Ainsi, au nœud i on met à jour $Q_i(r)$ suivant l'équation 5.1.

$$Q_i(r) \quad = \quad max(Q_i(r), l_r^i) \tag{5.1}$$

L'équation 5.2 permet de trouver un chemin de coût minimal de s à p parmi les chemins réalisables sur l'ensemble des ressources. La variable de décision y_{ij} est fixée à 1 si l'arc (x_i, x_j) appartient à une solution et 0 sinon (équation 5.4).

$$min \sum_{(x_i \to x_j) \in E} q(0, (x_i \to x_j)) y_{ij} \tag{5.2}$$

$$y_{ij} \in \{0, 1\} \ \forall (x_i \to x_j) \in E \tag{5.3}$$

Afin d'assurer la continuité du chemin allant de s à p on pose un ensemble de contraintes décrites par l'inégalité 5.8.

$$\sum_{(x_i \to x_j) \in E} y_{ij} - \sum_{(x_j \to x_i) \in E} y_{ji} = 0 \qquad \forall \ j \in \{1, \ldots, N\}, \tag{5.4}$$

Tous les chemins devant être issus de la source s, on pose des contraintes relatives à l'existence d'un chemin unique issu de s. Cette contrainte est exprimée par l'équation 5.5.

$$\sum_{(x_0 \to x_j) \in E} y_{0j} \leq 1, \tag{5.5}$$

Pour garantir que les chemins réalisables ne passent qu'une et une seule fois par les nœuds constituant le chemin, on pose un ensemble de contraintes d'élémentarités des chemins. Ces contraintes sont décrites par l'inégalité 5.6.

$$\sum_{(x_i \to x_j) \in E} y_{ij} \leq 1 \qquad \forall \ i \in \{s, p\} \cup \{1, \ldots, N\}, \tag{5.6}$$

Les contraintes décrites par l'équation 5.7 permettent de contrôler la consommation cumulée sur chaque ressource et d'assurer le seuil de consommation de la ressource r.

$$l_r^i \leq Q_i(r) \leq u_r^i \qquad \forall \ i \in \{s, p\} \cup \{1, \ldots, N\}; \forall \ r = 1, \ldots, K, \tag{5.7}$$

Les contraintes 5.8 assure la continuité du flot de consommation des ressources. On fait ici l'hypothèse que toutes les ressources se cumulent par incrémentation des valeurs associées à la ressource et ceci dans la limite du seuil autorisé.

$$y_{ij}(Q_i(r) + q(r, (x_i \to x_j)) - Q_j(r)) \quad \leq \quad 0 \qquad \forall (x_i \to x_j) \in E; \forall \ r = 1, \ldots, K, \tag{5.8}$$

Résolution des ESPPRC par programmation dynamique La résolution d'un ESPPRC peut être réalisée en utilisant l'algorithme de programmation dynamique 8. La complexité de l'algorithme peut être pseudo-polynomiale si la consommation des ressources prend des valeurs entières et que le nombre de valeurs prises est borné et indépendant de la taille des données. Dans le cas contraire, sa complexité dépend des ressources considérées.

Le principe de cet algorithme est de partir du nœud source s (ligne 2) et de construire itérativement des chemins vers le nœud destination p. Cette construction est réalisée par extension de chemins partiels contenus dans $Chemins$. Les chemins partiels ont pour origine la source s et pour extrémité terminale un nœud quelconque du graphe (ligne 16). L'extension d'un chemin partiel ch se fait par le calcul des chemins partiels correspondant à ch auquel un arc sortant de l'extrémité terminale de ch est ajouté. Les nouveaux chemins partiels étendus ch' sont rejetés s'ils violent au moins une contrainte (ligne 10). Sinon chaque chemin partiel est comparé avec ceux ayant la même extrémité terminale. Ensuite, les chemins partiels dominés sont éliminés. Il y a dominance entre deux chemins partiels si un des deux domine l'autre pour chacune des ressource prise individuellement. On admet qu'après extension vers le sommet i, la consommation $(Q_i(r))$ d'une ressource est ajustée à sa borne inférieure (l_i^r) si elle est inférieure. On note $ref_r(ch, (x_i \to x_j))$ la valeur de la fonction d'extension de la ressource r du chemin partiel ch par l'arc $(x_i \to x_j)$. Soit $Q_{ch}(r)$ la consommation d'un chemin partiel ch sur une ressource r. L'expression décrite par l'équation 5.9 donne les règles d'extension des ressources.

Algorithme 8 : `Resoudre-ESPPRC(G,` $Chemins$`,`$CheminDExtrem_i$`)`

```
1  début
2      Initialisation : Chemins ← {s} /* chemin partiel réduit à s */
3                       etiq = 0
4      tant que Chemins ≠ ∅ faire
5          choisir ch ∈ Chemins
6          etiq = etiq + 1
7          domine = faux
8          pour chaque chemin partiel ch' ∈ CheminDExtrem_etch faire
9              si Q_ch(r) ≤ Q_ch'(r), ∀r ∈ {0,...,K} alors
10                 CheminDExtrem_etiq = CheminDExtrem_etiq \ {ch'}
11             sinon
12                 si Q_ch'(r) ≤ Q_ch(r), ∀r ∈ {0,...,K} alors
13                     domine = vrai
14                     stop
15         si domine = faux alors
16             CheminDExtrem_etiq = CheminDExtrem_etiq ∪ {ch}
17             pour chaque arc (x_extrenite(ch) → x_i) faire
18                 ch' = l'extension de ch par (x_extrenite(ch) → x_i)
19                 si ch' est valide alors
20                     Chemins = Chemins ∪ {ch'}
21     retourner CheminDExtrem_p
22 fin
```

$$ref_r(ch, (x_i \to x_j)) \quad = \quad max\left\{Q_{ch}(r) + q(r, (x_i \to x_j)), l_r^j\right\} \tag{5.9}$$

La Figure 5.6 montre l'exécution de l'algorithme 8 sur un graphe représenté par la Figure 5.5. Les chemins partiels *ch* sont identifiés par leur nom et leur niveau de consommation des ressources.

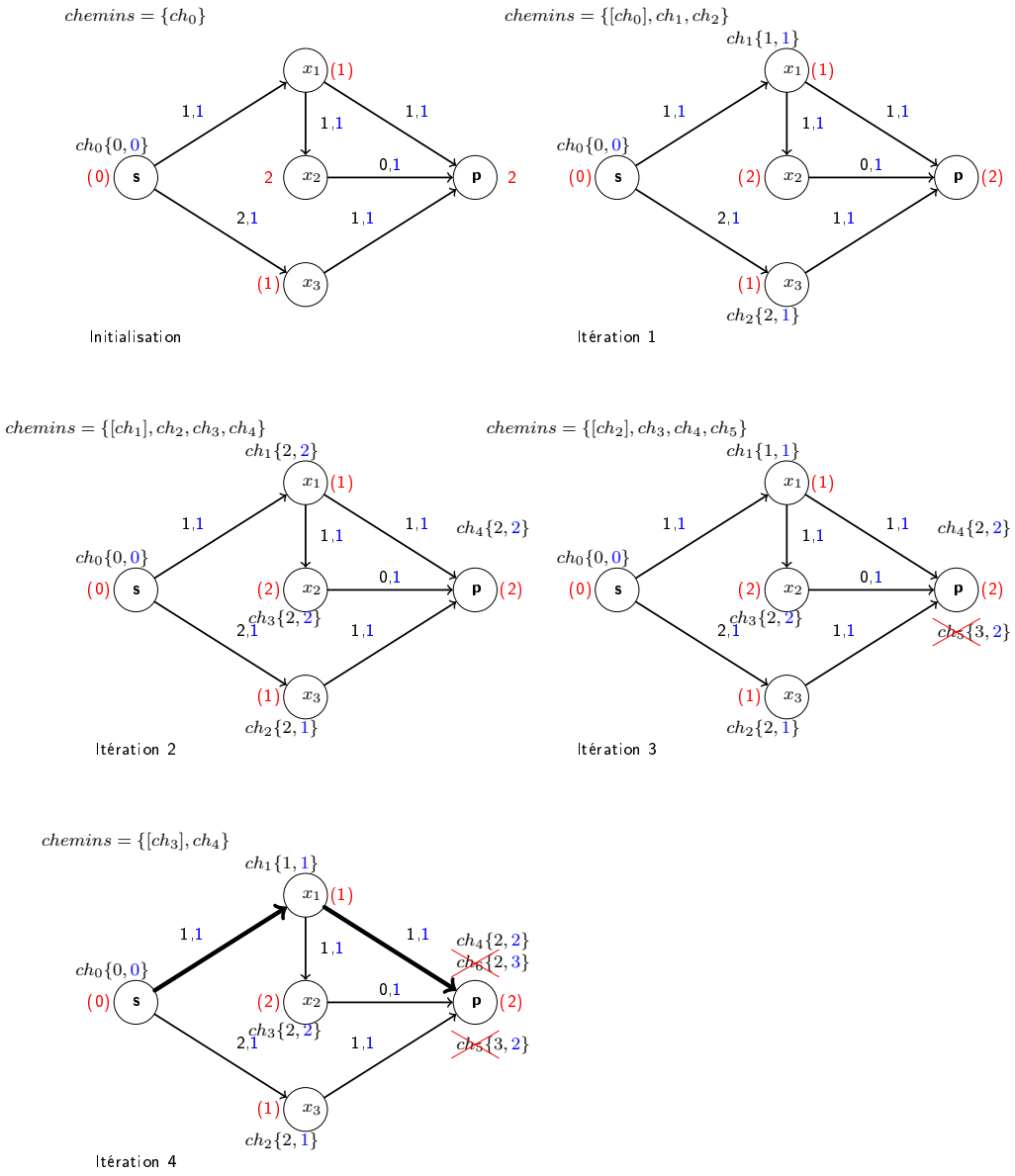


FIGURE 5.6 – *Application de l'algorithme 8 sur le graphe de la Figure 5.5.*

On cherche à minimiser le coût des chemins partant de s tout en respectant les contraintes de ressources à chaque nœud. Ces niveaux de consommation des ressources sont indiqués entre parenthèses à côté du nœud. À chaque itération, on étend le chemin partiel entre croquets dans $Chemin$. Le chemin partiel ch_5 issu de l'extension du chemin ch_2 est supprimé car il est dominé par $ch4$ en raison de son coût qui est de 3. Le chemin partiel ch_6 est quant à lui supprimé car il consomme 3 unités sur la ressource qui est limitée à 2 au sommet p

5.2.4 Bilan

Dans cette section, nous avons présenté les techniques de planification hiérarchique (HTN). Néanmoins, l'expressivité des réseaux de tâches hiérarchiques est très critiquable dans la mesure où ils ne représentent que des séquences partiellement ordonnées. Nous avons souligné le fait que, combinés avec les réseaux de contraintes temporelles simples (STN), les réseaux de tâches hiérarchiques peuvent être utilisés pour représenter et résoudre le problème de planification tactique.

Nous avons ensuite présenté une approche pour la prise en compte des ressources. Nous nous sommes focalisés sur une classe de problèmes classiques, en recherche opérationnelle, capables de gérer les ressources. Ces problèmes sont : les problèmes de plus court chemin sous contraintes de ressources. Cette approche nous intéresse et nous l'utilisons comme heuristique de décomposition dans les HTN.

5.3 Représentation du domaine de planification tactique

5.3.1 Prise en compte des effets

L'exécution d'une opération agricole a un effet *direct* instantané qui porte sur la consommation et la production des ressources. Par ailleurs, l'effet *attendu* sur les processus biophysiques est soumis à un délai. Ces effets *attendus* ne surviennent ni au début ni à la fin de l'exécution de l'opération agricole. Ils peuvent être le fruit de plusieurs opérations agricoles jointes. En conséquence, nous considérons dans le cadre de cette thèse que les opérations agricoles ont des effets indéterminés qui ne peuvent donc pas être explicitement représentés par une opération agricole.

Par ailleurs, les itinéraires techniques peuvent être caractérisés par leurs effets globaux attendus c'est-à-dire les effets cumulés de chacune des opérations du plan. Ces effets globaux attendus sont d'une certaine manière les sous buts à atteindre.

5.3.2 Représentation des itinéraires techniques

5.3.2.a Inspiration agronomique issu du système DIESE

Les itinéraires techniques, peuvent être représentés comme un réseau de tâches liées par des relations temporelles d'enchaînements ou de synchronisations. Cette représentation est celle utilisée dans DIESE (Martin-Clouaire et Rellier, 2009). Ce dernier représente les plans en utilisant des conjonctions ou disjonctions de :

- relations temporelles qualitatives *before, meet, overlap, co-start, co-end*,
- relations symboliques *iterate, optional*.

Les itinéraires techniques ainsi représentés peuvent être de simples séquences de tâches, des itérations sur des tâches voir des plans conditionnels. Dans DIESE, un opérateur de composition peut être appliqué aux tâches pour obtenir des tâches abstraites. Cette logique d'abstraction des tâches dans DIESE permet de construire des réseaux de tâches. Toutefois, la représentation de DIESE comporte deux points faibles. Le premier est qu'il n'existe pas d'heuristique formel pour la réduction des tâches abstraites. La seconde limite est relative à l'impossibilité de raisonner sur les effets d'une réduction. Ces deux aspects sont cependant primordiaux pour la décision tactique telle que nous la voyons dans cette thèse.

5.3.2.b ITK comme un réseau de tâches hiérarchiques

Nous proposons de représenter la « *réalisation d'une culture* » par une tâche composée (cf. section 5.2.1.b) dans un réseau de tâches hiérarchiques (cf. section 5.2.1).

Les itinéraires techniques seront alors modélisés par les méthodes (cf. Définition 5.4) des tâches composées. Ces itinéraires techniques serviront à réduire la « *réalisation d'une culture* » en un réseau d'opérations agricoles. Les opérations agricoles seront définies par des tâches primitives (cf. Définition 5.2) du réseau. Les avantages majeurs de cette représentation hiérarchique reposent sur :

- sa capacité à exploiter facilement la structure des connaissances expertes de l'agriculteur en terme de modes de production des cultures,
- la possibilité de développer des approches plus formelles pour la réduction des tâches composées,
- la possibilité de développer et de raisonner sur les effets d'une réduction.

5.3.2.c Représentation des opérations agricoles

Pour décrire le domaine de planification, nous avons choisi d'utiliser le langage PDDL [5] (cf. section 2.1.1.e). Les opérations agricoles sont représentées comme des « *actions duratives* » de PDDL. La BNF [6] ci-dessous, indique la manière de représenter chaque opération agricole (cf. Annexe A.4 pour la BNF complète).

```
(<durative-action-def>  ::= (:durative-action  <da-symbol>
                               :parameters (<typed list (variable)>)
                               <da-def body>
)
```

```
<da-symbol>                ::= <name>
<da-def body>              ::= :duration <duration-constraint>
                               :condition <da-GD>
                               :effect <da-effect>
```

Dans une tâche durative de PDDL, :duration indique la durée d'exécution attendue de la tâche. Les conditions d'exécution (:condition) définissent à la fois les conditions d'ouverture de fermeture. La description native en PDDL des conditions est réduite à des conjonctions d'expression temporelles définies comme ci-dessous.

```
<da-GD>                ::= () | <timed-GD> | (and <timed-GD>+)
<timed-GD>             ::= (at <time-specifier> <GD>)
                             |(over <interval> <GD>)
<time-specifier>       ::= start | end
<interval>             ::= all
```

Par exemple, les représentations suivantes sont celles des tâches de semis, de la première fertilisation et de l'irrigation de maïs, conformément aux connaissances de l'agriculteur définies dans la section 1.6.3.c.

```
(:durative-action Sowing-MA
 :parameters(?p - Plot ?cdate - Date ?tractor2 - Tractor2 ?worker - Worker
      ?seeddrill - Seeddrill)
 ;; Durée de l'action
 :duration (= ?duration (* (area ?p) (speed-sowing ?tractor2)) )
 :condition (and
     ;; Prédicat indiquant si le semis est réalisable
     (RuleSowing)
     ;; Contrainte temporelle d'ouverture de l'action : à partir du 10/04
    (at start (and (>= (date) 99  ) (<= (date)  +oo  )))
    ;; Contrainte temporelle de fermeture de l'action : avant le 10/05
     (at end (and (>= (date) -oo ) (<= (date)  129  )))
   )
```

5. Planning Domain Definition Language, Mcdermott et al. (1998); Fox et Long (2003)
6. Backus Normal Form

```
  :effect (and
        ;; Consommation des ressources: un ouvrier, un tracteur et un
            semoir
        (at start (decrease ?tractor2 1))
        (at start (decrease ?worker 1))
        (at start (decrease ?seeddrill 1))
        ;; Production des ressources: un ouvrier, un tracteur et un semoir
        (at end (increase ?tractor2 1))
        (at end (increase ?worker) 1)
        (at end (increase ?seeddrill 1))
    )
)
```

Listing 5.1 – *Opération de semis du maïs*

```
(:durative-action Fertilization-MA1
 :parameters(?p - Plot ?cdate - Date ?tractor2 - Tractor2 ?worker - Worker
      ?spray - Spray)
 :duration (= ?duration (* (area ?p) (speed-fertilization ?tractor2)) )
 :condition (and
     (RuleFertilization)
     (at start (and (>= (date) -oo ) (<= (date)  +oo )))
     (at end (and (>= (date) -oo ) (<= (date)  +oo )))
    )
  :effect (and (at start (decrease ?tractor2 1))
        (at start (decrease ?worker 1))
        (at start (decrease ?spray 1))
        (at end (increase ?tractor2 1))
        (at end (increase ?worker 1))
        (at end (increase ?spray 1))
      )
)
```

Listing 5.2 – *Opération de fertilisation du maïs*

```
(:durative-action Irrigation-MA
 :parameters(?p - Plot ?cdate - Date ?waterEquipment - WaterEquipment ?
     water - Water)
 :duration (= ?duration (* (area ?p) (speed-irrigation ?waterEquipment)) )
 :condition (and
     (RuleIrrigation)
     (at start (and (>= (date) 134 ) (<= (date)  +oo )))
     (at end (and (>= (date) -oo ) (<= (date)  313 )))
    )
  :effect (and (at start (decrease ?waterEquipment 1))
        (at start (decrease ?water 16))
        (at end (increase ?waterEquipment 1))
      )
)
```

Listing 5.3 – *Opération d'irrigation du maïs*

5.3.2.d Représentation des itinéraires techniques

Chaque itinéraire technique est défini par un plan temporel décrivant la manière de produire une culture pour les effets globaux attendus donnés. Ainsi, connaissant une culture définie par une tâche composée, ses différents modes de production sont représentés par des réseaux de tâches. Contrairement aux opérations agricoles représentées par des « *actions duratives* », PDDL 2.1 natif ne représente pas le plan

dans la description du domaine de planification. Les planificateurs hiérarchiques tels que SHOP (Nau et al., 1999), SHOP2 (Nau et al., 2003) et SIADEX (Castillo et al., 2005) utilisent une description spécifique pour représenter les méthodes de décomposition des tâches composées.

Dans notre cas, nous nous inspirons des représentations proposées dans Castillo et al. (2005); González-Ferrer et al. (2011). Ainsi, la BNF ci-dessous, indique la manière de représenter chaque tâche composée de réalisation de culture (cf. Annexe A.4 pour la BNF complète). Le mot clé « :task » indique une tâche composée de réalisation d'une culture. « :method » indique la description d'un ITK. Le réseau de tâches associé à la méthode est défini par « :tasks ». Ce réseau de tâches décrit les relations temporelles entre les tâches. Dans cette représentation, les effets globaux (ou les sous buts) sont associés aux labels des méthodes.

```
(<compound-task-def> ::=      (:task <ct-symbol>
                                :parameters (<typed list (variable)>)
                                <ct-def body>
)
```

```
<ct-symbol>                  ::= <name>
<ct-def body>                ::= <desc-method>*
<desc-method>                ::= (:method <m-symol> <m-def body>)
```

```
<m-symbol>                   ::= <name>
<m-def body>                 ::= :condition <m-GD>
                                 :tasks (<task-def>*)

<task-def>                   ::= <taskname>|
                                   | <def-constraint>

<def-constraint>             ::= (<constraint> <taskname>)*
<constraint>                 ::= (<binary-comp>
                                     <time-specifier>
                                     <interv-specifier>)
<taskname>                   ::= <da-symbol> | <ct-symbol>
<binary-comp>                ::= > | < | = | >= | <=
<time-specifier>             ::= ?start | ?end
<const-head>                 ::= start | end
<interv-specifier>           ::= (<number> <number>)
```

Par exemple, considérant la description de la section 1.6.3.c, présentant les modes de production du maïs, on note trois ITK différents pour cette culture. La représentation suivante décrit la tâche composée associée. On y retrouve les trois modes de production qui sont : minimiser le travail, minimiser les intrants et maximiser le rendement.

```
(:task MA
 :parameters(?parcelle - Parcelle)
(:method minWorkingDays ;; Décomposition pour l'ITK ↓travail
:condition ()
:tasks (
    ((> ?start (end Sowing-MA) (3 +oo)) Weeding-MA_1)
    ((> ?start (end Weeding-MA_1) (0 +oo)) Fertilization-MA1)
    ((> ?start (end Fertilization-MA1) (0 +oo)) Weeding-MA_2)
    ((> ?start (end Sowing-MA) (15 +oo)) Weeding-MA_2)
    ((> ?start (end Sowing-MA) (15 +oo)) Fertilization-MA1)
    ((> ?start (end Weeding-MA_2) (0 +oo)) Fertilization-MA2)
    ((> ?start (end Weeding-MA_2) (0 +oo)) Irrigation-MA)
    ((> ?start (end Irrigation-MA) (9 +oo)) Irrigation-MA)
```

```
    ((> ?start (end Irrigation-MA) (3 +oo)) Harvesting-MA)
    ((> ?start (end Fertilization-MA2) (0 +oo)) Harvesting-MA)
))

(:method minChemicalInput ;; Décomposition pour l'ITK ↓intrant
:condition ()
:tasks (
    ((> ?start (end Plowing-MA) (0 +oo)) Sowing-MA)
    ((> ?start (end Sowing-MA) (3 +oo)) Weeding-MA_1)
    ((> ?start (end Weeding-MA_1) (0 +oo)) Weeding-MA_2)
    ((> ?start (end Sowing-MA) (15 +oo)) Weeding-MA_2)
    ((> ?start (end Weeding-MA_2) (0 +oo)) Fertilization-MA22)
    ((> ?start (end Weeding-MA_2) (0 +oo)) Irrigation-MA)
    ((> ?start (end Irrigation-MA) (9 +oo)) Irrigation-MA)
    ((> ?start (end Irrigation-MA) (3 +oo)) Harvesting-MA)
    ((> ?start (end Fertilization-MA22) (0 +oo)) Harvesting-MA)
    ((> ?start (end Harvesting-MA) (0 +oo)) Harrowing-MA)
))

(:method maxYield ;; Décomposition pour l'ITK ↑rendement
:condition ()
:tasks (
    ((> ?start (end Plowing-MA) (0 +oo)) Sowing-MA)
    ((> ?start (end Sowing-MA) (3 +oo)) Weeding-MA_1)
    ((> ?start (end Weeding-MA_1) (0 +oo)) Fertilization-MA1)
    ((> ?start (end Fertilization-MA1) (0 +oo)) Weeding-MA_2)
    ((> ?start (end Sowing-MA) (15 +oo)) Weeding-MA_2)
    ((> ?start (end Sowing-MA) (15 +oo)) Fertilization-MA1)
    ((> ?start (end Weeding-MA_2) (0 +oo)) Fertilization-MA2)
    ((> ?start (end Weeding-MA_2) (0 +oo)) Irrigation-MA)
    ((> ?start (end Irrigation-MA) (9 +oo)) Irrigation-MA)
    ((> ?start (end Irrigation-MA) (3 +oo)) Harvesting-MA)
    ((> ?start (end Fertilization-MA2) (0 +oo)) Harvesting-MA)
    ((> ?start (end Harvesting-MA) (0 +oo)) Harrowing-MA)
)))
```

Listing 5.4 – *Tâche composée représentant le maïs. (cf. section 1.6.3.c, Figure 1.15)*

Dans cette représentation, la contrainte ci-dessous exprime le fait que le démarrage du premier désherbage de maïs (Weeding-MA_1) succède à la fin du semis (Sowing-MA) dans un délai de 3 jours minimum.

```
((> ?start (end Sowing-MA) (3 +oo)) Weeding-MA_1)
```

La contrainte ci-dessous exprime une répétition de l'irrigation. Chaque instance d'irrigation étant séparée de 9 jours minimum.

```
((> ?start (end Irrigation-MA) (9 +oo)) Irrigation-MA)
```

5.3.2.e Représentation du réseau de tâches

Partant de la description du domaine de planification, nous construisons un ensemble de réseaux de tâches définissant les méthodes du HTN. Chaque méthode est associée à un réseau de contraintes temporelles simples (STN). Chaque tâche primitive représentant une opération agricole est définie par deux instants de début S_a et de fin F_a (avec a l'opération agricole). La durée entre les opérations est représentée par un arc entre les instants S_a et F_a. Les relations entre deux tâches primitives a et b sont représentées par des arcs entre les instants S_a, F_a, S_b et F_b.

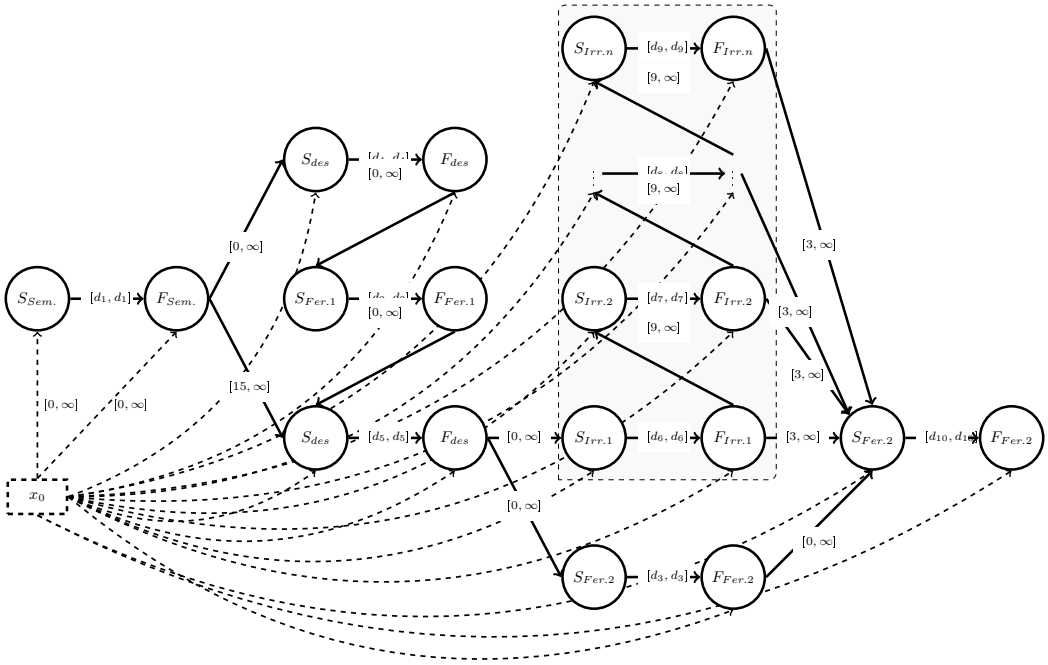


FIGURE 5.7 – *Graphe représentant le réseau de contraintes temporelles simples (STN) de l'ITK ↓ travail du maïs*

La Figure 5.7 indique la représentation graphique du réseau de tâches associé à l'ITK « ↓ *travail* » du maïs.

Chaque nœud du graphe indique soit l'instant de démarrage soit l'instant de fermeture d'une tâche. Le nœud x_0 représente l'instant de référence du STN. En fonction du type de nœud S_a ou F_a, les arcs $(x_0 \rightarrow S_a)$ (respectivement $(x_0 \rightarrow F_a)$) définissent les dates de début au plus tôt et au plus tard (respectivement dates de fin au plus tôt et au plus tard).

5.3.2.f Heuristique d'expansion des tâches répétitives

Pour construire le graphe correspondant au réseau de tâches, nous procédons avant tout à un prétraitement durant lequel toutes les tâches répétitives sont développées en plusieurs instances élémentaires de tâches primitives. Par exemple, la contrainte de répétition de l'irrigation est développée en une séquence de $NB_{it}(irrigation)$ irrigations, deux à deux liées par des contraintes de précédences simples.

L'heuristique que nous utilisons pour l'expansion des tâches répétitives n'est valable que pour les tâches primitives. Il peut être facilement étendu aux tâches composées. Cet heuristique dépend :

1. *de la durée de la tâche répétitive* : calculée en fonction de la taille des parcelles élémentaires [7] et de la vitesse d'exécution de la tâche,
2. *du «* timelag *» entre deux instances successives de la tâche répétitive* : déterminé par le « *timelag* » minimum ($tlmin_{a,a}$),
3. *de l'étendu temporel de la répétition* : calculé en fonction des dates de début (S_a) et de fin (S_a) de la tâche répétitive puis de la date de début (S_b) d'une tâche primitive de la décomposition, contrainte par a,

7. Nous gardons ici l'hypothèse d'homogénéité surfaciques (cf. Hypothèse 4.1) des parcelles élémentaires

4. *du cumul des consommations des ressources consommables de la décomposition* : calculé en fonction des quantités de ressources consommables requises [8] pour l'exécution de l'ensemble des tâches de la décomposition,
5. *des contraintes de précédence de la tâche répétitive* : déterminées en fonction des contraintes temporelles de la décomposition.

Ainsi, le nombre de répétition $NB_{it}(a)$ de la tâche primitive a est donné par l'équation 5.10.

$$NB_{it}(a) = min\left(\underbrace{\left(\frac{min(F_a, S_b) - S_a}{duree(a) + tlmin_{a,a}}\right)}_{\text{aspect lié au temps}}, \underbrace{min\left(\frac{Q^d_{max}(r_k) - \sum_{c \in \mathcal{O}\setminus\{a\}} q(r_k, c)}{q(r_k, a)}\right)}_{\text{aspect lié aux ressources}}\right) \quad (5.10)$$

avec a la tâche répétitive, $duree(a)$ la durée de la tâche répétitive a, b et c des tâches primitives de la décomposition, r_k une ressource, $Q^d_{max}(r_k)$ la quantité maximale de la ressource r_k requise pour exécuter la décomposition et $q(r_k, c)$ (respectivement $q(r_k, a)$) la quantité de la ressource r_k requise pour l'exécution de la tâche primitive c (respectivement a).

Pour déterminer la tâche b par rapport à laquelle l'étendu temporel de la répétition est calculé, nous considérons deux cas de figure.

Cas 1 - a précède b :

$$b = \underset{c \in \ successeur(a)}{\text{argmin}} \ S_c$$

Cas 2 - a n'a pas de prédécesseur :

$$b = \underset{c \in \ \mathcal{O}\setminus\{a\}}{\text{argmax}} \ S_c$$

Les nœuds encadrés en gris dans la Figure 5.7 illustrent les contraintes entre les tâches résultantes du développement d'une contrainte de répétition de tâches.

Sur la base des éléments présentés dans cette section, nous pouvons représenter l'ensemble de modes de conduite des cultures. Dans la section suivante, nous présentons notre proposition pour l'affection des itinéraires technique.

5.4 Affectation des ITKs et propagation des contraintes temporelles

5.4.1 Définition des buts de la planification tactique

Les itinéraires techniques déterminent les effets globaux d'un mode de conduite donné. Ces effets définissent d'une certaine manière les sous buts atteignables en utilisant un itinéraire technique (une décomposition). Ainsi, le but de la planification tactique est de choisir les ITKs de manière à satisfaire les contraintes liées au choix du mode de conduite et les préférences de conduite de l'agriculteur. Ces contraintes et préférences sont définies dans les sous-sections suivantes.

8. Nous faisons ici l'hypothèse qu'il existe (ou que nous pouvons déduire) des connaissances relatives aux quantités maximales des ressources consommables nécessaires à l'exécution de la décomposition. Par exemple, dans notre cas d'étude, nous savons que $165m^3$ d'eau par hectare sont nécessaires à la production du maïs

5.4.1.a Contraintes liées au choix du mode de conduite

Le choix des décompositions est soumis aux trois contraintes suivantes :

1. l'affectation d'un ITK à une parcelle est admissible si l'ITK affecté est réalisable sur l'ensemble des parcelles du bloc fonctionnel affectées à la même culture. Cette contrainte est la conséquence des Définitions 1.11 et 1.12. Satisfaire cette contrainte, revient à choisir les décompositions de sorte que, les îlots fonctionnels d'une année donnée respectent la sémantique des systèmes de culture.
2. l'ensemble des ITK affectés doit être compatible avec les contraintes de ressources de l'exploitation.
3. chaque ITK affecté doit satisfaire les contraintes temporelles et de précédences sur l'ensemble des opérations agricoles de la décomposition.

5.4.1.b Préférences de conduite de l'agriculteur

Généralement, l'agriculteur dispose d'un ensemble de préférences de décomposition en fonction de ses objectifs de production. Par exemple, dans le cas de l'exploitation virtuelle, il préfère des conduites intensives (maximisation du rendement) pour les cultures principales de l'exploitation que sont le blé et le maïs (cf. section 1.6.4.b).

Nous proposons un heuristique de décomposition qui est une fonction globale permettant d'expliciter les préférences et les interdépendances (liées aux ressources) entre différents modes de conduites des cultures. Cet heuristique est présenté dans la section suivante.

5.4.2 Heuristique de décomposition du réseau de tâches

5.4.2.a Construction des îlots fonctionnels

D'après la Définition 1.11, pour une année donnée, les parcelles de l'exploitation agricole sur lesquelles une même culture est produite, en utilisant un même itinéraire technique, définit un îlot fonctionnel.

Soit $\mathcal{X} = \{x^t_{b,i}\}$ l'ensemble des variables représentant les parcelles élémentaires. Chaque variable $x^t_{b,i}$ définit l'occupation de la parcelle élémentaire i du bloc fonctionnel b à la date t avec $i \in [1, \mathcal{N}_b]$, $b \in [1, \mathcal{B}]$ et $t \in [1, \mathcal{H}]$. Soit $Dom = \bigcup D_{b,i} = \{c_1, \ldots, c_m\}$ l'union des domaines des variables $\mathcal{X}$.

On note $Is^t_{b,c}$ l'ensemble des parcelles élémentaires du bloc b affectées à la culture c à la date t. $Is^t_{b,c}$ représente un *potentiel*[9] îlot fonctionnel de l'exploitation agricole à la date t. Dans la suite du manuscrit, on parlera, par abus de langage, d'îlot fonctionnel. Ainsi,

$$x^t_{b,i} \in Is^t_{b,c} \quad ssi \quad x^t_{b,i} = c \ avec \ t \in [h+1, \mathcal{H}] \tag{5.11}$$

On note $Is^t = \{(idisl, Is^t_{b,a})\}$ l'ensemble des îlots fonctionnels de l'exploitation à la date t. $idisl$ représente l'identifiant de l'îlot fonctionnel. Ainsi, considérant à

9. En toute rigueur, l'îlot fonctionnel est défini sur l'ensemble des blocs b. Cependant, pour déterminer les îlots fonctionnels nous savons par définition que les parcelles élémentaires d'un bloc affecté à une même culture doivent être associées à un et un seul itinéraire technique.

la date t une assignation complète A [10] des cultures aux parcelles, l'algorithme 10 retourne l'ensemble des îlots fonctionnels Is^t

Algorithme 10 : `ConstruireIlotsFonctionnels(A,t)`

```
1 début
2   pour b ← 1 to |B| faire
        // initialiser à un vecteur vide
3       pour a ← 1 to |Dom| faire Is^t_{b,c} ← ()
4   pour b ← 1 to |B| faire
5       pour i ← 1 to N_b faire
6           Ajouter i à l'îlot fonctionnel Is^t_{b,c} si x^t_{b,i} = a
7   retourner Is^t = {(id, Is^t_{b,c})}
8 fin
```

5.4.2.b Consommation de ressources par itinéraire technique

Le réseau de tâches $\langle \mathcal{A}, \mathcal{C} \rangle$ décrivant un itinéraire technique est constitué d'un ensemble de tâches $\mathcal{A}$ et d'un ensemble de relations temporelles $\mathcal{C}$ entre les tâches. Soit $R = \{r_1, \ldots, r_k, \ldots, r_K\}$ l'ensemble des ressources de l'exploitation. La consommation des ressources d'un ITK correspond à l'accumulation de la consommation de la ressource r_k le long de l'itinéraire technique. Soit $Q_{itk,i}(r_k)$ l'accumulation de la consommation de la ressource r_k pour le réseau de tâches représentant l'ITK itk appliqué à la parcelle élémentaire i.

$$Q_{itk,i}(r_k) = \sum_{a \in \mathcal{A}} q(r_k, a) \qquad \forall\ r_k \in R \tag{5.12}$$

avec $q(r_k, a)$ la quantité de r_k requise pour l'exécution de la tâche a. La valeur de la quantité $q(r_k, a)$ est calculée conformément à l'équation 5.13.

$$q(r_k, a) = \begin{cases} q(r_k, a) & \textit{si consommable} \\ nbu(r_k) \times duree(a) = \left\lceil nbu(r_k) \times \frac{taille(x^t_{b,i}) \times DTJ)}{vitesse(a)} \right\rceil & \textit{si réutilisable} \end{cases} \tag{5.13}$$

où $duree(a)$ est la durée de la tâche a, $nbu(r_k)$ le nombre d'unités de la ressource r_k consommé par a, $taille(x^t_{b,i})$ la taille de la parcelle élémentaire sur laquelle la tâche est réalisée, DTJ le nombre d'heures travaillé dans la journée (8h/j, 10h/j, etc.) et $vitesse(a)$ la vitesse d'exécution de la tâche. Dans notre cas d'étude par exemple, cette vitesse est spécifiée dans la colonne *Vitesse (ha/j)* du tableau 1.5.

Notons qu'à ce stade, le calcul de la consommation des ressources réutilisables est basé sur le nombre d'heures durant lequel la tâche mobilise un nombre d'unités donné de la ressource.

5.4.2.c Consommation de ressources par îlot fonctionnel

Considérant une ressource r_k, pour un îlot fonctionnel donné, il existe autant d'options de consommation de r_k que d'itinéraires techniques pour conduire la culture associée à l'îlot fonctionnel. On note $Q^{idisl}_{itk}(r_k)$ la consommation de la ressource r_k

10. un assolement résultant des solutions de la planification stratégique

sur un îlot fonctionnel $Is^t_{b,c}$, qui esr identifié par *idisl*, en appliquant l'itinéraire technique *itk*. La quantité de r_k consommée est donnée par l'équation 5.14. Elle correspond au cumul des consommations de ressource r_k par l'ITK *itk* appliqué à chacune des parcelles élémentaires i de l'îlot fonctionnel.

$$Q^{idisl}_{itk}(r_k) = \sum_{i \in Is^t_{b,c}} Q_{itk,i}(r_k) \qquad \forall\ r_k \in R \tag{5.14}$$

5.4.2.d Notre approche pour l'affectation des ITKs

Pour résoudre le sous problème d'affectation des ITKs aux parcelles élémentaires sous contraintes de ressources, nous établissons un raisonnement sur les îlots fonctionnels. Cela revient à affecter des ITKs aux îlots fonctionnels.

Proposition 5.1 (Heuristique de décomposition) *Le problème d'affectation des ITKs aux îlots fonctionnels est équivalent à un problème de recherche de plus court chemin sous contraintes de ressources (ESPPRC cf. section 5.2.3.e). Chaque nœud du graphe multivalué représente soit un îlot fonctionnel, soit un itinéraire technique. Chaque arc est étiqueté par un vecteur de consommations des ressources r_k sur l'îlot fonctionnel idisl, en appliquant l'itinéraire technique itk. La ressource r_0 est associée au coût à payer pour traverser l'arc. Elle modélise les préférences en terme de modes de conduites décrites dans les sections 1.6.4.b et 5.4.1.b. L'objectif est de trouver un chemin de coût minimal de s à p parmi les chemins réalisables sur l'ensemble des ressources.*

5.4.2.e Construction du graphe multivalué $G = (X, E, R)$

Soit $G = (X, E, R)$ un graphe multivalué représentant l'ESPPRC.

Les nœuds de G Hormis la source s et le puits p, on distingue deux types de nœuds dans X tels que $X = \{idisl\} \cup \{itk^{idisl}_{num}\} \cup \{s, p\}$. $\{idisl\}$ représente l'ensemble des identifiants *idisl* des îlots fonctionnels $Is^t = \{(idisl, Is^t_{b,c})\}$ de l'exploitation agricole à la date t. $\{itk^{idisl}_{num}\}$ correspond à l'ITK *num* appliqué sur les parcelles élémentaires de l'îlot fonctionnel *idisl*. Cet ensemble représente implicitement l'union disjointe [11] des itinéraires techniques possibles pour chaque culture c associée à un îlot fonctionnel $Is^t_{b,c}$.

Les consommations de ressources associées aux nœuds du graphe sont bornées par les capacités maximales des ressources ($\{Q_{max}(r_k)\}$). Autrement dit, il n'y a pas, dans notre cas, des limites de consommation spécifiques aux nœuds du graphe.

Les arcs de G Chaque arc $e = (idisl \rightarrow itk^{idisl}_{num})$ est associé au coût *cost* et à la consommation $Q^{idisl}_{itk}(r_k)$ de chaque ressource r_k ($k \in \{1, \ldots, K\}$) sur l'îlot *idisl*. Le coût *cost* est par définition associé à la ressource r_0. Les arcs $e = (itk^{idisl}_{num} \rightarrow idisl + 1)$ sont des arcs fictifs dont le but est d'assurer la continuité du chemin de s à p. Chacun de ces arcs est associé à un coût nul et à une consommation de ressources nulle.

Formellement le graphe G sera défini comme suit

11. Rappel : l'opérateur d'union disjointe $\uplus$ rajoute des identifiants uniques (*idisl* dans notre cas) aux ensembles de départ (ensemble des ITKs d'une culture c) de manière à construire un ensemble d'arrivée de la forme $(idisl, itk_{num})$. Cet ensemble d'arrivée est noté $\{itk^{idisl}_{num}\}$ dans notre cas

$$
\begin{aligned}
X &= \{idisl\} \cup \{itk_{num}^{idisl}\} \cup \{s, p\} \quad (5.15)\\
&\quad \text{avec } idisl \in \{1, \dots |Is^t|\},\ num \in \mathbb{N}\\
R &= \{0, \dots, K\} \quad (5.16)\\
E &= E_{idisl} \cup E_{itk} \cup E_p \quad (5.17)\\
&\quad E_{idisl} = \{(idisl \rightarrow itk_{num}^{idisl})\}\\
&\quad E_{itk} = \{(itk_{num}^{idisl} \rightarrow idisl + 1)\}\\
&\quad E_p = \{(itk_{num}^{|Is^t|} \rightarrow p)\}
\end{aligned}
$$

La Figure 5.8 montre un exemple de problème d'affectation d'ITKs. Dans le problème, chacune des cultures associée aux îlots fonctionnels admet trois itinéraires techniques possibles.

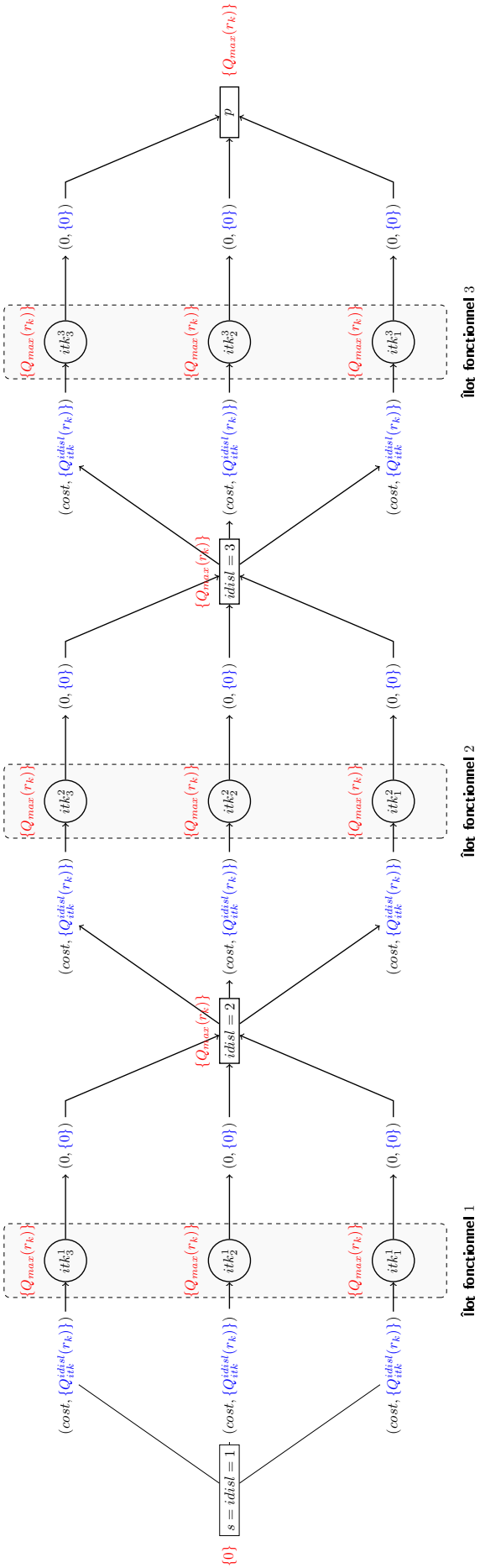


FIGURE 5.8 – *Graphe multivalué représentant l'ESPPRC à résoudre pour l'affectation des ITKs aux îlots fonctionnels*

5.4.2.f Fonction de dominance

Pour comparer les chemins partiels du graphe G ayant la même extrémité terminale, la fonction de dominance que nous utilisons dépend essentiellement du coût du chemin partiel. Pour deux chemins donnés, la fonction de dominance consiste à comparer le cumul des coûts le long du chemin. Un chemin en domine un second si son coût est strictement inférieur à celui du second.

5.4.2.g Fonction d'extension de ressources

Soit $ref_{r_k}(ch, e)$ la fonction d'extension de la ressource r_k du chemin partiel ch par l'arc $e \in E$. Soit $Q_{ch}(r_k)$ la consommation d'un chemin partiel ch sur une ressource r_k. L'expression décrite par l'équation 5.18 donne les règles d'extension des ressources.

$$ref_{r_k}(ch, e) = \left\{ Q_{ch}(r_k) + Q^{idisl}_{itk}(r_k) \right\} \qquad \forall\ k \in \{0, \ldots, K\} \tag{5.18}$$

5.4.2.h Résolution du ESPPRC

Pour résoudre le sous problème d'affectation des ITKs, nous utilisons l'algorithme de programmation dynamique présente dans la section 5.2.3.e (cf. Algorithme 8). Pour l'implémentation, nous nous sommes inspirés de la version proposée dans la bibliothèque de graphe BGL (*Boost Graph Library*[12] Siek et al. (2002)).

Nous avons expliqué, dans cette section, notre proposition pour l'affectation des ITK aux parcelles élémentaires. Dans la section suivante nous présentons dans sa globalité notre approche de planification tactique.

5.4.3 Algorithme de planification tactique

L'algorithme 11 représente le principe de fonctionnement de notre système de planification tactique. Il permet de construire pour chaque parcelle élémentaire un plan tactique. Cet algorithme prend en entrée :

- les plans Π^{str} issus de la planification stratégique,
- les plans tactiques Π^{tac} en cours d'exécution. La notation $\Pi^{tac}(x^t_{b,i})$ exprime le plan tactique associé à la parcelle i du bloc fonctionnel b à la date t.
- les préférences d'affectation (cf. sections 1.6.4.b et 5.4.1.b) des ITKs, exprimées sous la forme d'une fonction de coût tabulaire. La notation $cost[itk]$ indique le coût lié à l'ITK itk
- les capacités maximales de ressources disponibles sur l'ensemble de l'exploitation.

Le fonctionnement de l'algorithme se déroule en deux phases. La première phase (lignes 3 à 17) est destinée à l'affectation des itinéraires techniques aux îlots fonctionnels de l'exploitation agricole. La seconde phase (lignes 18 à 22) est quant à elle destinée à la propagation des contraintes temporelles pour chaque parcelle élémentaire.

Dans la phase d'affectation des itinéraires techniques, on procède à la construction des îlots fonctionnels (ligne 3) en appliquant l'algorithme 10. Ensuite, le graphe multivalué G définissant l'ESPPRC est construit (lignes 5 à 16). Pour ce faire, nous procédons à l'expansion (ligne 11) de la tâche composée associée à

12. http://www.boost.org/doc/libs/

Algorithme 11 : `PlanificationTactique`(Π^{str}, Π^{tac}, $pref$, Q_{max})

Entrées : Π^{str} - *plans stratégiques courant pour chacune des parcelles* ; Π^{tac} - *plans tactique courant pour chacune des parcelles* ; $pref$ - *préférence d'affectation des ITK exprimées sous la forme de coûts* ; Q_{max} - *capacités maximales des ressources*

début

Initialisation : $\pi_{develop} \leftarrow ()$: *structure contenant la mise à plat de la hiéarchie des ITKs* ; G : *un graphe multivalué* ; G_d *un graphe de distances* ; $lstitks \leftarrow ()$: *structure contenant l'affectation des ITKs à chaque parcelle élémenataire*

`// Construction et affectation des ITKs aux îlots fonctionnels`

Is^t = `ConstruireIlotsFonctionnels(`Π^{str},t`)`

pour chaque $(idisl, Is^t_{b,c}) \in Is^t$ **faire**

`AjouterNoeud(`G, $idisl$,Q_{max}`)`

`AjouterNoeud(`G, $idisl+1$,Q_{max}`)` `/*` $idisl+1 = p$ `si` $idisl = |Is^t|$ `*/`

pour chaque *itk de la culture* c **faire**

`AjouterNoeud(`G, itk^{idisl}_{num}, Q_{max}`)`

`/* Mise à plat de la hiérarchie de l'itk */`

si $\pi_{develop}$ *[itk]*$=()$ **alors**

Choisir la tâche composée a telle que $name(a) = c$

$\pi_{develop}$ $[itk]$ = `Expansion-HTN(`a,itk,$\mathcal{O}$,$\mathcal{M}$`)`

`/* Ressources consommées par îlot fonctionnel */`

pour $k \leftarrow 0$ **to** K **faire**

si $k = 0$ **alors** $Q^{idisl}_{itk}(r_k) = pref[itk]$

sinon $Q^{idisl}_{itk}(r_k) = \sum_{i \in Is^t_{b,c}} Q_{itk,i}(r_k)$

`AjouterArc(`G, $(idisl \rightarrow itk^{idisl}_{num})$, $\{Q^{idisl}_{itk}(r_k)\}$`)`

`AjouterArc(`G, $(itk^{idisl}_{num} \rightarrow idisl+1)$, $\{0\}$`)`

$lstitks$ = `ChoisirUneAffectation(Resoudre-ESPPRC(`G, $\emptyset, \emptyset$`))`

`// Propagation des contraintes temporelles pour chaque parcelle élémentaire`

pour chaque $(idisl, Is^t_{b,c}) \in Is^t$ **faire**

pour chaque $x^t_{b,i} \in Is^t_{b,c}$ **faire**

$\Pi^{tac}(x^t_{b,i})$ = `AjouterA(`Π^{tac}, $\pi_{develop}$ *[*$lstitks$ *[*$x^t_{b,i}$ *]]*`)`

Construire le graphe de distance G_d *associé à* $\Pi^{tac}(x^t_{b,i})$

`MiseAJour(`$\Pi^{tac}(x^t_{b,i})$, `FloydWarshallAllPSP(`G_d`))`

retourner Π^{tac}

fin

la culture c définissant chaque îlot fonctionnel $Is^t_{b,c}$. L'expansion est réalisée en utilisant une version simplifiée (Algorithme 12) de l'algorithme de planification hiérarchique. Cette version développe la tâche composée en suivant la méthode m associée à l'itinéraire technique. Les consommations de ressources (ligne 12 à 14) sur les arcs du graphe G sont calculées sur la base de la méthode développée dans les sections 5.4.2.b et 5.4.2.c. Enfin l'ESPPRC est résolu et une affectation des ITKs aux parcelles élémentaires des ilots fonctionnels est stockée dans *lstitks*.

Algorithme 12 : `Expansion-HTN(`π_0,m,$\mathcal{O}$,$\mathcal{M}$`)`

Entrées : π_0 - *le plan courant* ; m - *méthode recommandée pour la décomposition* ; $\mathcal{O}$ - *ensemble des opérateurs* ; $\mathcal{M}$ - *ensemble des méthodes*

```
1  début
2      si π0 = ∅ alors retourner plan vide
3      au ← choisir une tâche u ∈ π0 telle que predecesseur(u) = ∅
4      restant ← π0 − {u}
5      si au est une tâche primitive alors
6          actif ← {(o, σ)|o ∈ O ,
7                        σ une substitution telle que name(o) = σ(au)}
8          si actif = ∅ alors Retour echec
9          choisir(o, σ) ∈ actif
10         p ← Expansion-HTN(σ(restant),m, O,M)
11         si p = echec alors Retour echec
12         sinon retourner AjouterA(p,o)
13     sinon
14         si ∃ une décomposition d pour la méthode m alors
15             choisir la décomposition d pour m
16         sinon
17             choisir de manière non-déterministe une décomposition d
               pour une m′ ≠ m
18         retourner Expansion-HTN(d,m′, O,M)
19 fin
```

Notons, $\pi_{develop}[lstitks[x^t_{b,i}]]$, le plan issu de l'expansion de la tâche composée associée de la culture c de la parcelle élémentaire $x^t_{b,i}$ en suivant la méthode associée à l'itinéraire technique $lstitks[x^t_{b,i}]$. Dans la seconde phase, pour chacune des parcelles élémentaires $x^t_{b,i}$, $\pi_{develop}[lstitks[x^t_{b,i}]]$ est ajouté au plan tactique courant de la parcelle (ligne 20). Le graphe de distances G_d est ensuite construit en suivant la méthode décrite dans la section 5.3.2.e (ligne 21). L'algorithme procède enfin à la propagation des contraintes temporelle en utilisant l'algorithme *all-pairs-shortest-path* de Floyd-Warshall (Algorithme 7).

5.5 Application de l'approche de planification tactique

Dans cette section nous présentons les expérimentations que nous avons menées afin de tester les algorithmes présentés pour la planification tactique. Pour ce faire, nous divisons la section en deux parties. La première partie présente le contexte de nos expérimentations. La seconde partie quant à elle présente et analyse les résultats obtenus en utilisant notre méthode de planification tactique.

5.5.1 Contexte des expérimentations

5.5.1.a Description des plans stratégiques utilisés pour la planification tactique

Les expérimentations on été menées en utilisant les instances B1[1-4]-LU15(*), B[1-4]-LU30(*) et B[1-4]-LU60(*) (cf. section 4.6.1). Ces instances correspondent à différents échantillonnages (15, 30 et 60 parcelles élémentaires) du problème initial dans lesquels les blocs b_1, b_2, b_3 et b_4 sont tous interdépendants [13]

Pour chacune de ces instances, nous prenons comme plan stratégique la première solution optimale. La Figure 5.9 présente l'occupation du sol pour chacune des quatre années de l'horizon de planification stratégique.

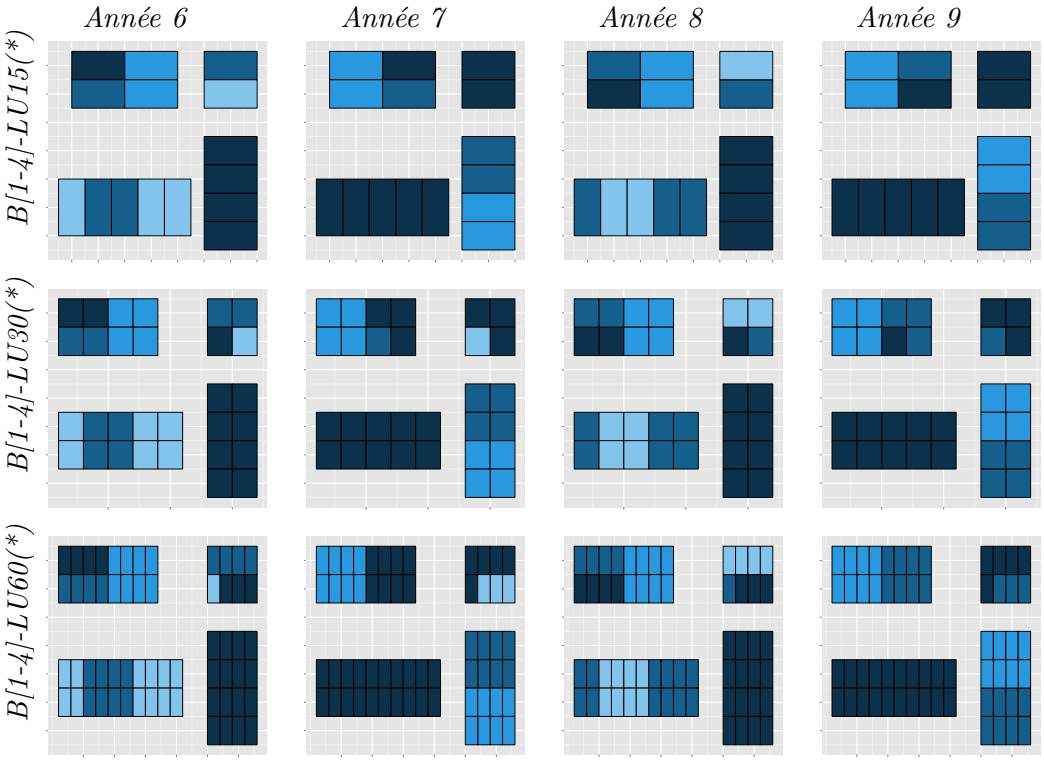


FIGURE 5.9 – *Plans stratégiques :* ■ *Blé d'hiver,* ■ *Orge de printemps,* ■ *Maïs,* ■ *Colza d'hiver*

5.5.1.b Les coûts associés aux itinéraires techniques

Pour déterminer les coûts associés à chaque itinéraire technique, nous nous basons sur la description des objectifs tactiques de production, présentée dans la section 1.6.4.b. Les cellules du Tableau 5.3 indiquent le coût affecté aux ITK en fonction de la culture considérée.

5.5.1.c Description des capacités de ressources pour l'affectation des ITK

Ressources de l'exploitation virtuelle Conformément à la description de la section 1.6, les Tableaux 5.5 et 5.6 résument respectivement les caractéristiques des ressources réutilisables et consommables de l'exploitation. La ligne *Capacité*

13. L'interdépendance des blocs résulte de la prise en compte des préférences qui portent sur l'ensemble de l'exploitation.

TABLE 5.3 – *Fonction de coûts tabulaire associée aux itinéraires techniques*

Itinéraires techniques	↑ *rendement*	↓ *intrant*	↓ *travail*
Blé d'hivers (BH)	0	1	3
Orge de printemps (OP)	2	0	1
Maïs (MA)	0	1	2
Colza d'hivers (CH)	2	0	1

indique le nombre d'unités de chaque ressource. La ligne *Disponibilités* indique la disponibilité annuelle de chaque ressource. Pour les ressources réutilisables (cf. Tableau 5.5), cette disponibilité est exprimée en heure et sur la base de 8 heures de travail par jour, 5 jours par semaine et 52 semaines par an.

TABLE 5.4 – *Les ressources consommables de l'exploitation virtuelle*

Noms	Quota Eau 1	Quota Eau 2
Capacités	6000 m^3	4000 m^3
Disponibilités	6000 m^3	4000 m^3

TABLE 5.5 – *Les ressources réutilisables de l'exploitation virtuelle*

Noms	Ouvriers	Tracteur1	Tracteur2	Charrue	Semoir	Pulvérisateur	Cultivateur	Moissonneuse	Équip.irri. 1	Équip.irri. 2
Capacités	2	1	1	1	1	1	1	1	1	1
Disponibilités	4160 *h*	2080 *h*	2080 *h*	2080 *h*	2080 *h*	2080 *h*	2080 *h*	2080 *h*	2080 *h*	2080 *h*

TABLE 5.6 – *Les ressources consommables de l'exploitation virtuelle*

Noms	Quota Eau 1	Quota Eau 2
Capacités	6000 m^3	4000 m^3
Disponibilités	6000 m^3	4000 m^3

Vitesse de réalisation des tâches Le Tableau 5.7 résume pour chaque tâche, la vitesse d'exécution (en $hectare/jour$) décrite dans de la section 1.6.

TABLE 5.7 – *Vitesse de travail*

Tâches	Labour	Semis	Désherbage	Fertilisation	Irrigation	Récolte	Déchaumage
Vitesse d'exécution (ha/j)	9	12	21	21	21	15	12

5.5.1.d Protocole expérimental

Pour la résolution d'un problème d'affectation des itinéraires techniques aux îlots fonctionnels, nous nous sommes inspirés de la version de l'Algorithme de résolution des ESPPRC disponible dans la bibliothèque de graphe BGL (*Boost Graph Library*[14] Siek et al. (2002)). La fonction de dominance et les fonctions d'extension de ressources sont celles décrites respectivement dans les sections 5.4.2.f et 5.4.2.g

5.5.2 Analyse des résultats

5.5.2.a Performances de l'algorithme d'affectation des ITKs

Les résultats présentés dans le tableau 5.8 sont obtenus avec un processeur Intel(R) Xeon(R) de 2.27GHz. Le temps de calcul mentionné est en milliseconde. Il indique la durée (ligne « Temps ») pour trouver les chemins réalisables de coût minimum. Les lignes « Année » et « Nb Îlots » indiquent l'année considérée et le nombre d'îlots fonctionnels sur l'ensemble de l'exploitation.

TABLE 5.8 – *Performance pour la recherche des solutions*

Instance	B[1-4]-LU15(*)				B[1-4]-LU30(*)				B[1-4]-LU60(*)			
Année	6	7	8	9	6	7	8	9	6	7	8	9
Nb Îlots	8	7	8	7	9	8	9	8	9	7	9	7
Temps (ms)	**1.6**	**1.4**	**1.5**	**1.35**	**1.75**	**1.5**	**1.7**	**1.4**	**1.8**	**1.3**	**1.65**	**1.28**

Comme l'indique le Tableau 5.8, le temps nécessaire pour trouver les affectations d'ITK avoisine la milliseconde. Pour les différentes instances que nous considérons, le temps de recherche des chemins réalisables est, dans le pire des cas, égal à 1.8 ms.

Ces résultats montrent qu'en terme de complexité temporelle, notre méthode d'affectation des itinéraires peut être exploitée dans un système de planification continue.

5.5.2.b Analyse des ITKs affectés

Considérons, les plans stratégiques issus de la résolution de l'instance B[1-4]-LU15(*). La Figure 5.10 montre les affectations des itinéraires techniques aux îlots fonctionnels . Pour chaque année, le graphique présente en abscisse la culture et l'ITK associés à chaque îlot fonctionnel. En ordonnée, nous représentons le nombre de parcelles élémentaires contenues dans l'îlot fonctionnel concerné.

Sur le graphique, on observe que les objectifs tactiques de production sont tous respectés. En effet, pour les deux cultures principales (Blé d'hivers et Maïs), les itinéraires techniques appliqués aux blocs fonctionnels, sont ceux qui maximisent

14. http://www.boost.org/doc/libs/

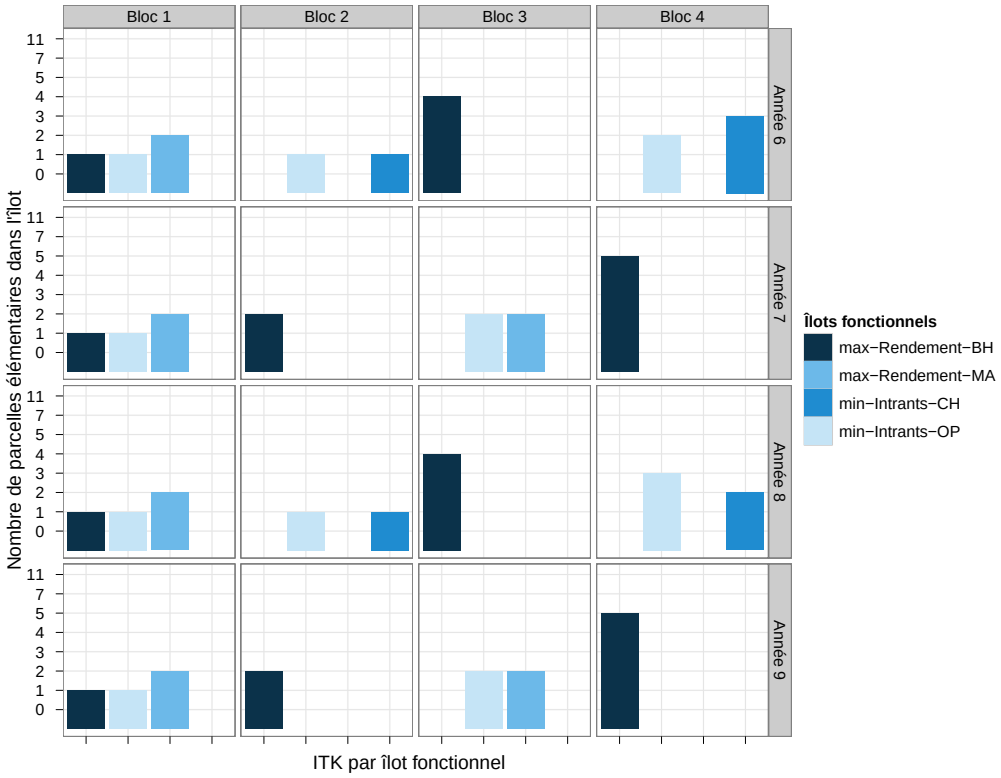


FIGURE 5.10 – *Modes de conduite des cultures par îlots fonctionnels : B[1-4]-LU15(*)*

l'espérance de rendement. Par ailleurs, ceux appliqués pour l'orge de printemps et le colza d'hiver correspondent aux ITK qui réduisent les intrants tout en garantissant un rendement acceptable. Ainsi, les ITK affectés sont ceux préconisés par l'agriculteur.

Une explication analogue à celle présentée pour l'instant B[1-4]-LU15(*) pourrait être menée afin de décrire l'affectation des ITKs pour les instances B[1-4]-LU30(*) est B[1-4]-LU60(*). Pour ces deux dernières, nous présentons les affectations d'ITK respectivement sur les graphiques 5.11 et 5.12.

5.5.2.c Analyses des consommations de ressources par année

Dans cette section nous observons pour chaque année l'occupation/la consommation de chacune des ressources de l'exploitation. Cette analyse permet de montrer les déterminants de l'affectation des ITK qui sont présentés dans la section précédente.

Les graphiques présentés sur la figure 5.13 montrent les cumuls annuels (année 6, 7, 8 et 9), pour chacune des instances, de l'utilisation des ressources sur l'ensemble de l'exploitation agricole. Sur ces graphiques, les mentions h (pour l'heure) ou $m3$ (pour le mètre cube) en face des noms de ressources, indiquent l'unité utilisée pour le calcul de l'occupation/la consommation de la ressource. Nous utilisons la couleur « bleu » afin de différentier la capacité limite de la ressource par rapport à l'occupation/la consommation de cette dernière.

Ainsi, nous pouvons souligner que pour l'ensemble des ressources, les cumuls annuels sont bien en dessous des capacités limites. En dehors de la ressource en eau nous observons que les occupations annuelles des autres ressources de l'exploitation sont largement en dessous des capacités limites. Ceci soulève une question essentielle relative à la disponibilité réelle des ressources. En effet, en considérant

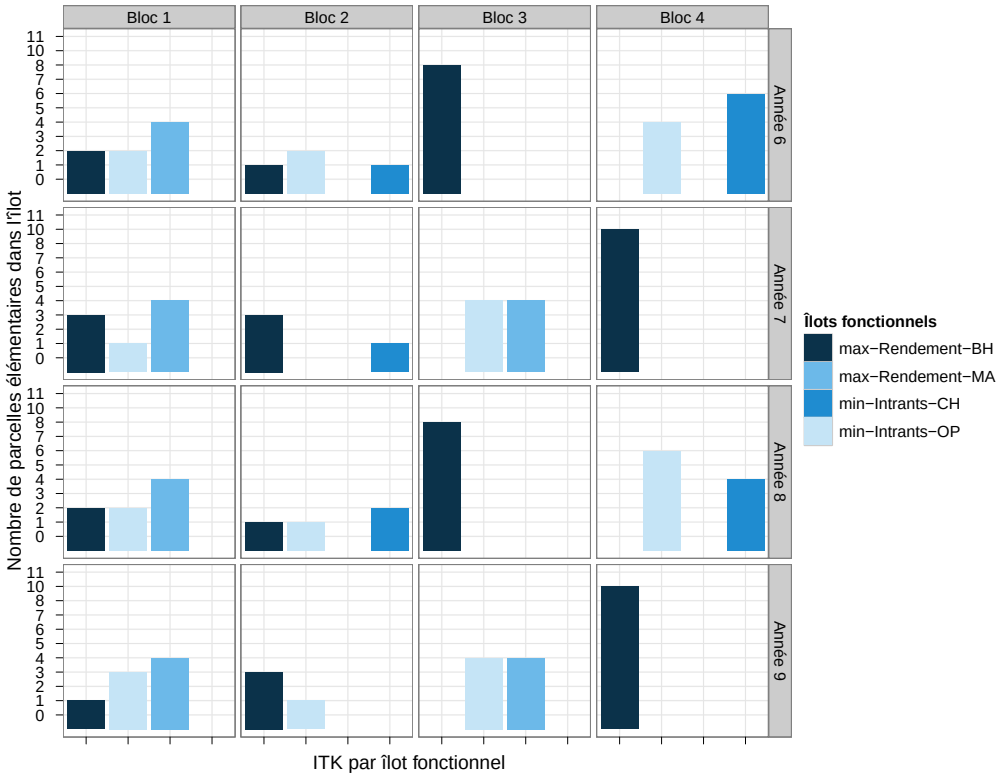


FIGURE 5.11 – *Modes de conduite des cultures par îlots fonctionnels : B[1-4]-LU30(*)*

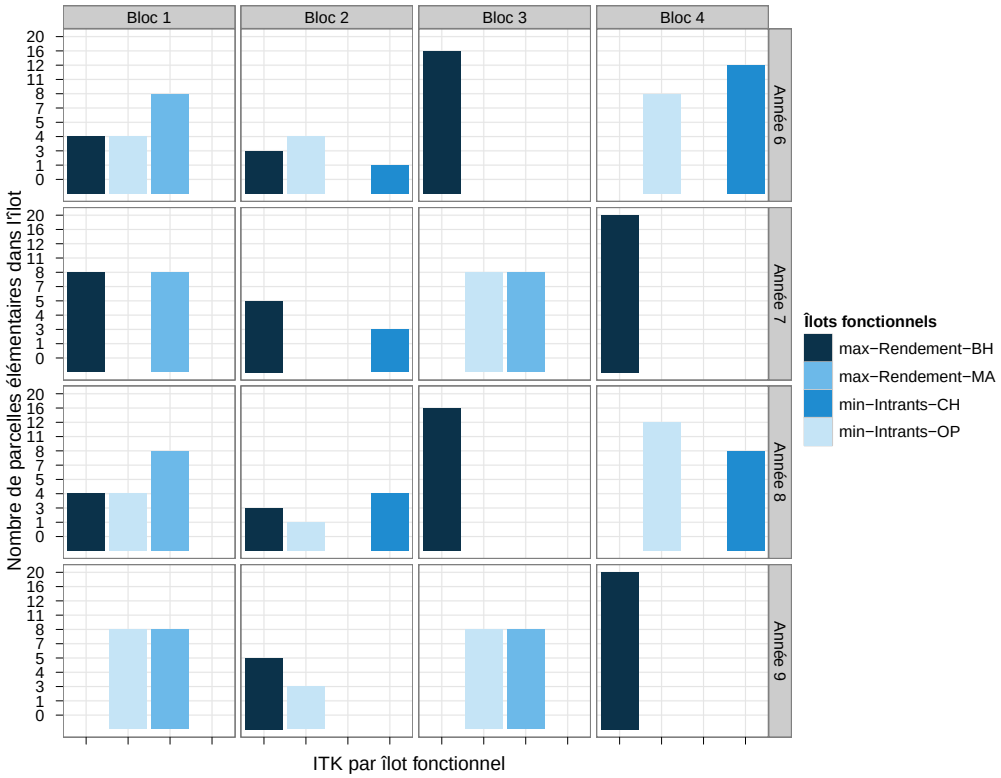
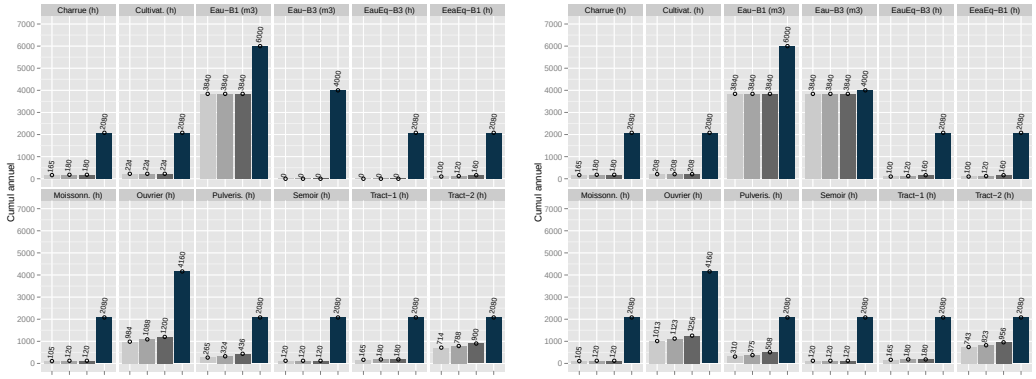


FIGURE 5.12 – *Modes de conduite des cultures par îlots fonctionnels : B[1-4]-LU60(*)*

Année 6 *Année 7*



Année 8 *Année 9*

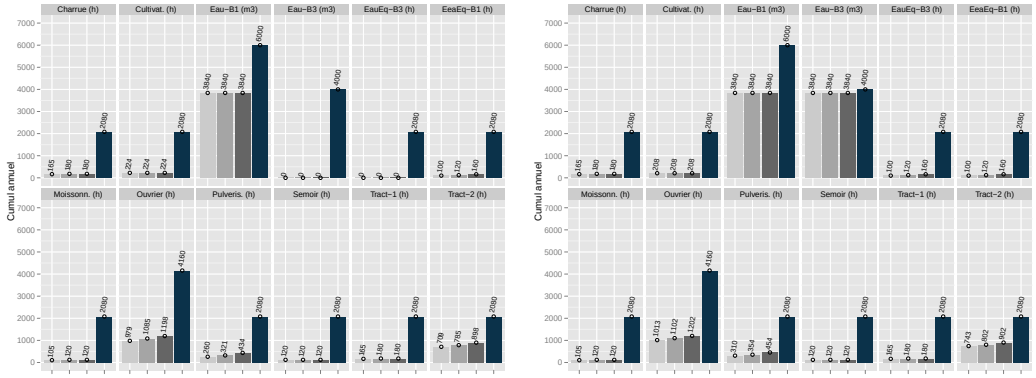


FIGURE 5.13 – *Consommation de ressources pour les années 6, 7, 8 et 9 :* ■ *B[1-4]-LU15(*),* ■ *B[1-4]-LU30(*),* ■ *B[1-4]-LU60(*),* ■ *Capacités limites des ressources*

2080 heures [15] de disponibilité annuelle par ressource élémentaire, notre estimation des capacités limites est excessivement optimiste. Tous les jours d'une année ne sont pas disponibles pour la réalisation des opérations agricoles. La disponibilité des jours dépend du climat voire de la nature du sol. Par exemple, pour éviter des effets de tassement du sol, un tracteur ne peut pas se rendre sur une parcelle s'il pleut voire même quelques jours après la pluie.

Partant de ce point, nous avons réalisé de expérimentation afin d'observer l'effet de l'estimation des jours disponibles sur les affectations des ITK.

5.5.2.d Influence des capacités limites de ressources sur les affectations des ITK

Afin de tester l'influence des jours disponibles sur les affectations de ITK, nous avons considéré trois cas de figures dans lesquels les capacités limites des ressources réutilisables sont estimées à 50%, 45% et 35% de celles présentées dans le Tableau 5.5.

À 50% de jours disponibles, nous avons observé que les affectations restent inchangées. Sous cette considération, l'agriculteur peut conduire ces parcelles en utilisant :

- des ITK qui maximisent l'espérance de rendement pour le blé et le maïs,
- des ITK qui réduisent les intrants tout en garantissant un rendement acceptable pour l'orge de printemps et le colza d'hivers.

À 45% de jours disponibles, les instances B[1-4]-LU15(*) et B[1-4]-LU30(*) offre des résultats identiques aux précédents. Cependant, pour l'instance B[1-4]-LU60(*) l'affectation obtenue est présentée sur le graphique 5.14. Nous observons, dans ce cas, que pour le bloc fonctionnel 4, à l'année 7, l'ITK maximisant l'espérance de rendement du blé a été remplacé par l'ITK qui réduit les intrants tout en garantissant un rendement acceptable. La ressource la plus discriminante ayant conduit à ce résultat est le tracteur 2 (cf. Figure 5.13 pour le cumul annuel de l'occupation avec le mode maximisation de l'espérance de rendement). Sur les 936 heures (45% de 2080 heures) disponibles pour ce tracteur, 900 heures seront utilisées pour des opérations agricoles.

Avec 35% de jours disponibles, les plans stratégiques des instances B[1-4]-LU30(*) et B[1-4]-LU60(*) sont irréalisables. Aucun ITK ne peut être appliqué car les capacités limites sont inférieures au besoin de l'exploitation. Par ailleurs, comme l'indique la Figure 5.15, il existe une affectation des ITK pour l'instance B[1-4]-LU15(*). Cette affectation propose, pour le bloc fonctionnel 4, et pour les années 7 et 9, de substituer l'ITK maximisant l'espérance de rendement du blé par celui qui réduit les intrants tout en garantissant un rendement acceptable. La ressource discriminante, dans ce cas, est à nouveau le tracteur 2.

Dans la section suivante, nous présentons les performances de l'algorithme de planification tactique.

5.5.2.e Performances de l'algorithme de planification tactique

Nous avons mesuré les temps nécessaires pour réaliser conjointement l'affectation des itinéraires techniques, le test de consistance et le calcul du domaine minimal des dates de débuts et de fins des chacune des tâches à réaliser.

Les résultats présentés dans le tableau 5.9 sont obtenu avec un processeur Intel(R) Xeon(R) de 2.27GHz. Le temps (ligne « Temps ») de calcul en milliseconde indique la durée pour affecter les ITK et calculer le domaine minimale.

15. 8 heures/jour 5 jours/semaine et 52 semaines/an

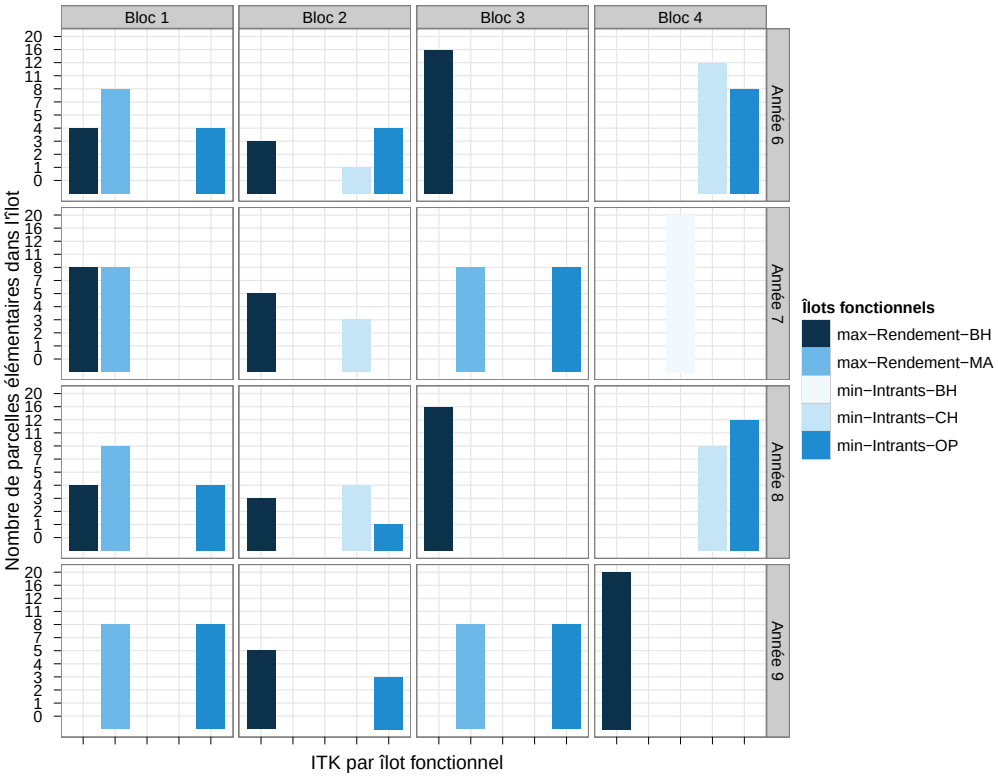


FIGURE 5.14 – *Modes de conduite des cultures par îlots fonctionnel pour 45% de jours disponibles : B[1-4]-LU60(*)*

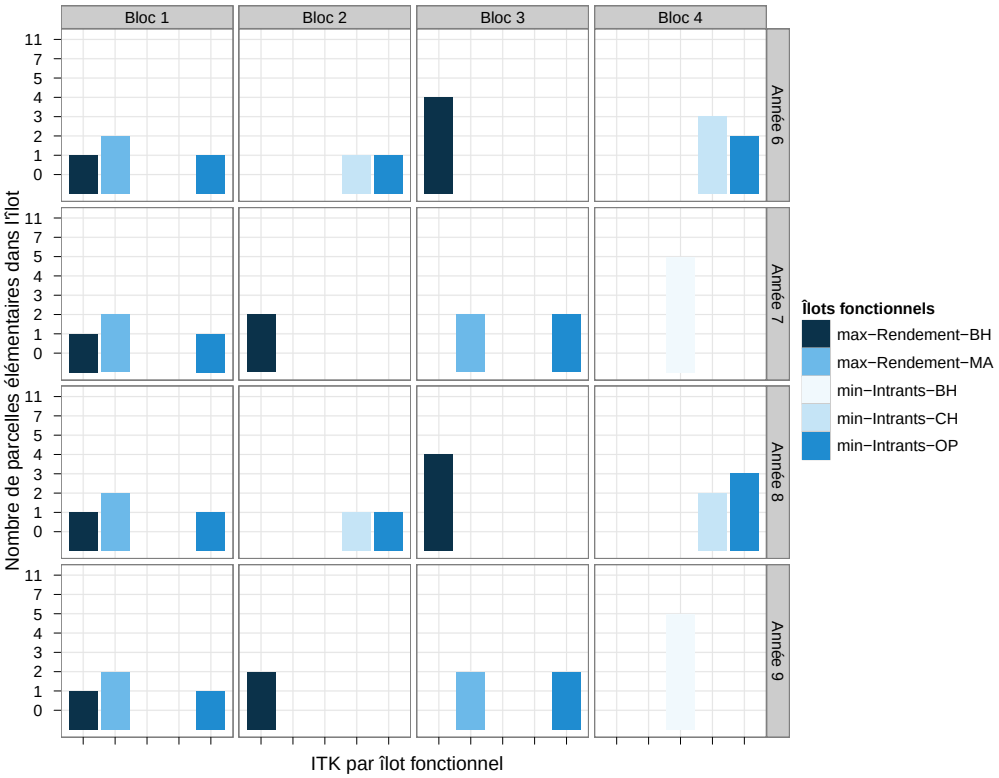


FIGURE 5.15 – *Modes de conduite des cultures par îlots fonctionnels et pour 35% de jours disponibles : B[1-4]-LU15(*)*

TABLE 5.9 – *Performances de l'algorithme de planification tactique*

Instance	B[1-4]-LU15(*)				B[1-4]-LU30(*)			
Année	6	7	8	9	6	7	8	9
Nb Îlots	8	7	8	7	9	8	9	8
Temps (ms)	**64.7**	**71**	**65.9**	**70.9**	**118.2**	**136.2**	**120.5**	**130.4**

Instance	B[1-4]-LU60(*)			
Année	6	7	8	9
Nb Îlots	9	7	9	7
Temps (ms)	**213**	**246.75**	**226.2**	**242.9**

Cette phase de planification produit pour chaque parcelle élémentaire, un plan qui correspond à l'ITK à appliquer pour conduire la culture déterminée dans la phase de planification stratégique. Nous présentons dans la section suivante, un exemple de plan tactique que nous obtenons.

5.5.2.f Exemple de plan tactique

Le graphique ci-dessous (Listing 5.5) montre le plan tactique obtenu pour la conduite du blé, en utilisant l'ITK qui maximise l'espérance de rendement, sur la parcelle 1 du bloc fonctionnel 1 de l'instance B[1-4]-LU15(*). Les contraintes exprimées pour cet ITK sont celles exprimées dans le tableau 1.4. Les relations temporelles entre les tâches sont celles exprimées par la Figure 1.13. Listing 5.5 indique la distance (timelag), en nombre de jours, entre la fin d'une tâche et le début de l'autre. Listing 5.6 indique le domaine minimal calculé pour chaque instant de début et de fin des tâches. es et ls indiquent respectivement les dates de début au plus tôt et au plus tard. ef et lf indique respectivement les dates de fin au plus tôt et au plus tard. Ces dates sont exprimée en jour Julien [16].

```
1  Plowing-BH----------[timelag 0,31]-----> Sowing-BH
2  Sowing-BH-----------[timelag 15,46]----> Weeding-BH1
3  Weeding-BH1---------[timelag 80,140]---> Weeding-BH2
4  Weeding-BH1---------[timelag 75,125]---> Fertilization-BH1
5  Fertilization-BH1---[timelag 15,53]----> Fertilization-BH2
6  Weeding-BH2---------[timelag 0,48]-----> Fertilization-BH2
7  Fertilization-BH2---[timelag 40,78]----> Fertilization-BH3
8  Fertilization-BH3---[timelag 45,93]----> Harvesting-BH
9  Harvesting-BH-------[timelag 0,42]-----> Harrowing-BH1
10 Harrowing-BH1-------[timelag 15,57]----> Harrowing-BH2
```

Listing 5.5 – *ITK du blé sur la parcelle élémentaire 1 du bloc fonctionnel 1*

```
1  Plowing-BH
2       start : [es = 2437571, ls = 2437602]
3           ;;[es = 28-09-1961, ls = 29-10-1961]
4       end : [ef = 2437572, lf = 2437603]
5           ;;[es = 29-09-1961, ls = 30-10-1961]
6  Sowing-BH
7       start : [ es = 2437572, ls = 2437603]
```

16. La datation en jour julien est un système de datation qui mesure le temps écoulé depuis 1er janvier 4713 av. J.-C à midi selon le calendrier julien proleptique.

```
8         end : [ ef = 2437573, lf = 2437604]
9     Weeding-BH1
10        start : [es = 2437588, ls = 2437619]
11        end : [ef = 2437589, lf = 2437620]
12    Weeding-BH2
13        start : [es = 2437700, ls = 2437729]
14        end : [ef = 2437701, lf = 2437730]
15    Fertilization-BH1
16        start : [es = 2437695, ls = 2437714]
17        end : [ef = 2437696, lf = 2437715]
18    Fertilization-BH2
19        start : [es = 2437730, ls = 2437749]
20        end : [ef = 2437731, lf = 2437750]
21    Fertilization-BH3
22        start : [es = 2437790, ls = 2437809]
23        end : [ef = 2437791, lf = 2437810]
24    Harvesting-BH
25        start : [es = 2437855, ls = 2437884]
26        end : [ef = 2437856, lf = 2437885]
27    Harrowing-BH
28        start : [es = 2437856, ls = 2437898]
29        end : [ef = 2437857, lf = 2437899]
30    Harrowing-BH
31        start : [es = 2437872, ls = 2437914]
32        end : [ef = 2437873, lf = 2437915]
```

Listing 5.6 – *Domaines minimaux des instants de début de chacune des tâches destinées à la conduite du blé sur la parcelle élémentaire 1 du bloc fonctionnel 1*

5.5.3 Bilan sur la planification tactique

Les trois sections précédentes (sections 5.5, 5.4 et 5.3) permettent de répondre à la question *Que fait t-on pour produire les cultures allouées dans la phase stratégique ?*. Dans les sections suivantes, nous allons répondre à la question *Quand et comment réaliser ce qui a été prévu de faire dans la phase tactique ?*. Nous présentons dans la section 5.6, notre réponse à cette dernière question.

5.6 Planification de la décision opérationnelle

Dans la phase de décision opérationnelle nous disposons pour chaque parcelle élémentaire d'un ensemble [17] de tâches à réaliser. Ces tâches résultent de la phase de planification tactique. Il s'agit alors d'ordonner des événements de début et de fin des tâches sous contraintes de temps et de ressources. Le problème d'optimisation sous-jacent consiste à trouver les affectations de ressources aux tâches de manière à minimiser l'étendu temporel de l'ordonnancement « *makespan* » . Ce problème est proche de celui des emplois du temps (« *University Course Timetabling Problem* »). Nous nous en inspirons afin de modéliser le problème de décision opérationnelle. Dans la littérature, plusieurs méthodes ont été décrites pour appréhender cette classe de problèmes. Le lecteur peut se référer à Burke et Petrovic (2002) pour plus de détails. Notons cependant que dans cette thèse, nous n'apportons aucune réponse liée à l'optimalité des ordonnancements obtenu.

17. Un ensemble de tâches potentiellement vide

5.6.1 Modélisation du problème de décision opérationnelle

5.6.1.a Modèle de programmation linéaire pour la décision opérationnelle

Le problème de décision opérationnelle peut être formalisé par un modèle de programmation linéaire en nombre entiers. Pour ce faire, le temps est divisé en fenêtres temporelle encore appelées « *slots* » temporels. Les slots temporels sont tous de tailles fixes et représentent une heure dans une journée de travail. Chaque jour est constitué d'un volume horaire fixe.

Le modèle que nous proposons est défini par un tuple $\langle T, \mathcal{A}, R\rangle$ où :

- $T = \{1, \ldots, \mathcal{H}\}$ *:* est un ensemble de « *slots* » temporels avec $\mathcal{H}$ l'horizon de l'ordonnancement ;
- $\mathcal{A} = \{\mathcal{A}_i\}$ *:* est un ensemble de tâches réalisables sur les parcelles i, dans la fenêtre temporelle correspondant à l'horizon $\mathcal{H}$ de l'ordonnancement. $i \in [1, \mathcal{N}_b]$ correspond à la parcelle élémentaire i du bloc b avec $\mathcal{N}_b$ le nombre de parcelles élémentaires du bloc b.
- $R = \{1, \ldots, k, \ldots K\}$ *:* est un ensemble de ressources.

Soit $x_{i,j}^{t,k} \in X$ un ensemble de variables de décision binaires, tel que $x_{i,j}^{t,k} = 1$ si dans le slot temporel t, la ressource r_k est affectée à la tâche j de la parcelle i ($x_{i,j}^{t,k} = 0$ sinon).

Chaque tâche nécessite pour son exécution une ou plusieurs ressources. On note, $R_{i,j}$ l'ensemble de ressources nécessaires à la réalisation de la tâche j de la parcelle i. Deux tâches en concurrence sur une ressource, ne seront réalisées dans une même fenêtre temporelle qu'à condition d'avoir la capacité de ressources nécessaire à l'exécution simultanée des deux. Les contraintes du problème sont définies comme suit :

Contraintes de parcelle : plusieurs tâches d'une parcelle ne peuvent pas s'exécuter dans un même slot temporel.

$$x_{i,j}^{t,k} + x_{i,j'}^{t,k} \leq 1 \qquad \begin{array}{l}\forall\, i \in [1, \mathcal{N}_b], \forall\, j, j' \in \mathcal{A}_i \times \mathcal{A}_i, \\ \forall\, t \in [1, \mathcal{H}], \forall\, k \in R\end{array} \tag{5.19}$$

Contraintes de ressources : dans un slot temporel, la somme des demandes de chaque ressource ne peut excéder la capacité disponible de la ressource.

$$\sum_{i \in [1, \mathcal{N}_b]} \sum_{j \in \mathcal{A}_i} q(r_k, j) x_{i,j}^{t,k} \leq Q(r_k) \qquad \forall\, t \in [1, \mathcal{H}], \forall\, k \in R \tag{5.20}$$

Contraintes de domaine : la date de début d'une tâche doit être supérieure ou égale à sa date de début au plus tôt es. La date de fin d'une tâche doit être inférieure à sa date de fin au plus tard lf.

$$\sum_{t=1}^{es_j} x_{i,j}^{t,k} = 0 \qquad \forall\, i \in [1, \mathcal{N}_b], \forall\, j \in \mathcal{A}_i, \forall\, k \in R \tag{5.21}$$

$$\sum_{t=lf_j}^{\mathcal{H}} x_{i,j}^{t,k} = 0 \qquad \forall\, i \in [1, \mathcal{N}_b], \forall\, j \in \mathcal{A}_i, \forall\, k \in R \tag{5.22}$$

Contraintes de durée : la durée d'affectation d'une ressource à une tâche doit être égale à celle de la tâche.

$$\sum_{t=1}^{\mathcal{H}} x_{i,j}^{t,k} = duree(j) \qquad \forall\, i \in [1, \mathcal{N}_b], \forall\, j \in \mathcal{A}_i, \forall\, k \in R \tag{5.23}$$

Contraintes de précédence : si une tâche j précède une tâche j' ($j \prec j'$) alors :

- la date de début de j doit être inférieure à la date de début de j'

$$s_{i,j}^{k} = \underset{t}{argmin}\ x_{ij}^{tk} \ /\ \{x_{i,j}^{t,k} | x_{i,j}^{t,k} = 1\} \qquad \forall\, k \in R \tag{5.24}$$

$$f_{i,j}^{k} = s_{i,j}^{k} + duree(j) \qquad \forall\, k \in R \tag{5.25}$$

$$s_{i,j}^{k} = s_{i,j}^{k'} \qquad \forall\, k, k' \in R \times R \tag{5.26}$$

$$f_{i,j}^{k} = f_{i,j}^{k'} \qquad \forall\, k, k' \in R \times R \tag{5.27}$$

  avec $s_{i,j}^{k}$ la date de début d'utilisation de la ressource r_k par la tâche j de la parcelle i et $f_{i,j}^{k}$ la date de fin d'utilisation de la ressource r_k par la tâche j de la parcelle i.
- la date de fin du précédent j doit être séparée d'au moins $tlmin_{jj'}$ et d'au plus $tlmax_{jj'}$ slot temporel du début du suivant j'.

$$\forall\, k \in R, \forall\, j, j' \in \mathcal{A}_i \times \mathcal{A}_i, j \prec j'$$

$$s_{i,j'}^{k} \geq f_{i,j}^{k} + tlmin_{j,j'} \tag{5.28}$$

$$\geq s_{i,j}^{k} + duree(j) + tlmin_{j,j'} \tag{5.29}$$

$$s_{i,j'}^{k} < f_{i,j}^{k} + tlmax_{j,j'} \tag{5.30}$$

$$< s_{i,j}^{k} + duree(j) + tlmax_{j,j'} \tag{5.31}$$

5.6.1.b Notre approche de résolution

La décision opérationnelle étant proche du problème d'emploi du temps, nous nous sommes inspiré d'une méthode de résolution généralement utilisée pour cette classe de problème. Selon de Werra (1985); Pinedo (2005), dans le cas particulier où toutes les tâches ont une durée de 1, un problème d'emploi du temps est structurellement équivalent à un problème de coloration de graphe. Il peut être modélisé par un graphe dans lequel chaque nœud représente une tâche et les conflits entre les tâches sont représentés par les arcs. Par exemple si deux tâches consomment une même ressource, le conflit entre ces tâches est représenté par un arc qui les relie. Les nœuds du graphe sont colorés de telle sorte que deux nœuds adjacents ne puissent être colorés par la même couleur. L'algorithme ordonne les tâches puis affecte séquentiellement les slots temporels de manière à éviter les conflits de ressources. Le problème d'optimisation consiste à colorer le graphe en utilisant un minimum de couleurs.

Pour la résolution du problème de planification opérationnel, nous reprenons l'idée de l'algorithme de coloration de graphe. Nous nous plaçons dans le cas où l'exécution d'une tâche ne peut être interrompue. Chaque slot temporel correspond à une couleur. Nous proposons trois adaptations majeures qui sont :

- chaque nœud du graphe correspond à la date de début d'une tâche.
- l'affectation d'une couleur à un nœud du graphe impose une consommation des ressources nécessaires tout au long des slots temporels correspondant à la durée d'exécution de la tâche.

– deux nœuds adjacents représentant deux tâches sur deux parcelles différentes peuvent être colorés par une même couleur dès lors que les contraintes temporelles et de ressources sont respectées.

Dans les sections suivantes, nous présentons le modèle du graphe et l'algorithme de résolution.

5.6.1.c Modèle du graphe

Soit $G(X, E)$ un graphe orienté tel que chaque nœud $x_{ij} \in X$ du graphe représente la date de début d'une tâche. Si deux tâches j, j' doivent être réalisées sur une même parcelle élémentaire i, elles ne peuvent être réalisées dans un même slot temporel. Dans ces conditions, les deux nœuds du graphe sont reliés par un arc dirigé $(x_{ij} \rightarrow x_{ij'})$ qui définit la précédence entre les tâches j et j' à réaliser sur la parcelle i. Si deux tâches j et j' des parcelles élémentaires i et i' (avec $i \neq i'$) sont en concurrence sur au moins une ressource, elles ne peuvent être réalisées dans un même slot temporel qu'à condition que le cumul des consommation de ressource n'excède pas la capacité disponible de la ressource. Dans ces conditions, les nœuds x_{ij} et $x_{i'j'}$ représentant les tâches j et j' des parcelles élémentaires i et i' (avec $i \neq i'$) sont reliés par deux arcs dirigés $(x_{ij} \rightarrow x_{i'j'})$ et $(x_{i'j'} \rightarrow x_{ij})$ définissent les conflits de ressources entre les tâches j et j'.

Exemple 5.4 *Dix tâches réparties sur trois parcelles élémentaires doivent être réalisées. Chacune de tâches est associée à l'une au moins de deux ressources* 1 *et* 2. *L'objectif est d'ordonnancer l'ensemble des dix tâches dans une fenêtre temporelle de 5 jours. Le tableau 5.10 décrit le problème. Les valeurs* 0 *ou* 1 *associées aux ressources décrivent si la ressource est indispensable ou non à l'exécution de la tâche.*

TABLE 5.10 – *Exemple de problème de planification opérationnelle*

tâche j	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
parcelle i	1	1	1	2	2	2	3	3	3	3
durée	-	-	-	-	-	-	-	-	-	-
ressource 1	1	1	1	0	1	0	1	1	1	1
ressource 2	0	1	1	1	0	1	0	1	0	1

La Figure 5.16-(a) représente graphiquement les séquences de tâches sur chacune des trois parcelles élémentaires. La Figure 5.16-(b) présente le modèle de graphe associé aux contraintes de précédence. La figure 5.16-(c) représente l'ajout des contraintes liées à la ressource 1. *Enfin, la figure 5.16-(d) présente l'ajout des contraintes liées à la ressource* 2.

5.6.2 Algorithme de résolution

Nous proposons l'algorithme 13 pour la résolution du problème de planification opérationnelle. L'algorithme reçoit en entrée le graphe associé au problème et les capacités de ressources. Le principe général de l'algorithme consiste à affecter à chaque nœud du graphe $G(X, E)$ une date t correspondant au slot temporel de démarrage de la tâche associée. Pour ce faire, les nœuds du graphe sont ordonnés suivant un ordre de découverte (ligne 2). L'heuristique de découverte que nous utilisons ordonne les nœuds par ordre décroissant de leur degré. Les nœuds de degrés maximums sont ceux associés aux tâches ayant un grand nombre de conflits de

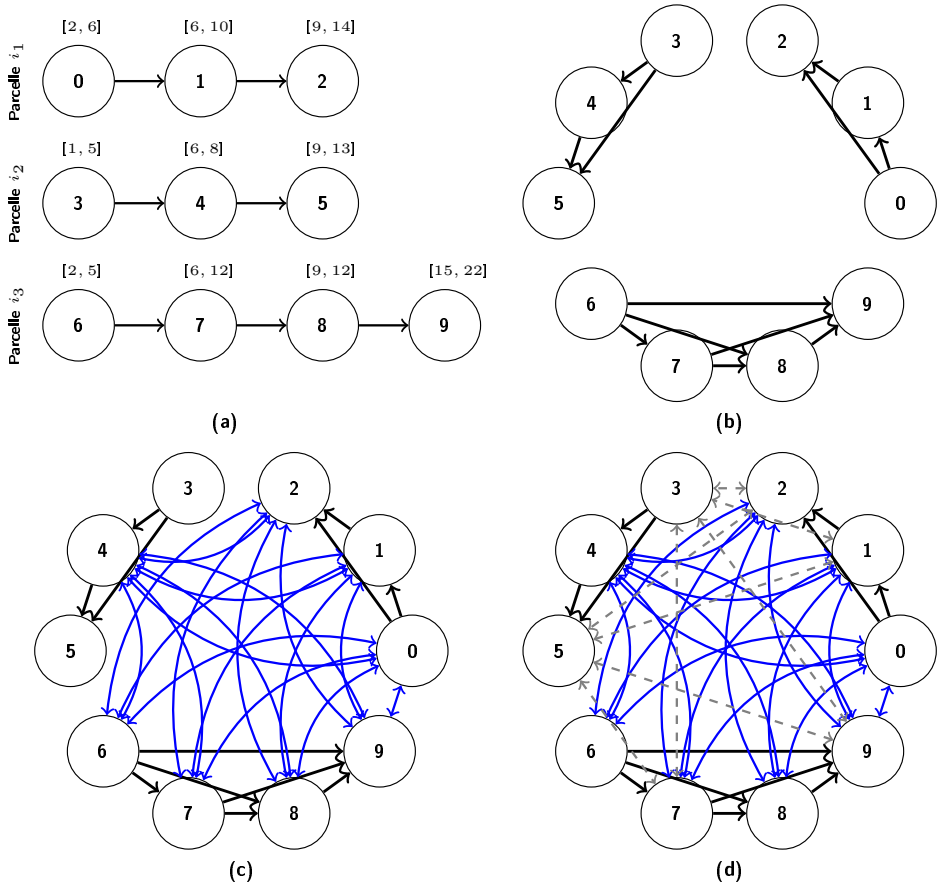


FIGURE 5.16 – *Exemple d'un modèle de graphe pour la planification opérationnelle*

ressources avec d'autres tâches. Ces tâches sont plus difficiles à prévoir et donc doivent être traitées en priorité. Dans le cas ou deux nœuds ont des degrés identiques, la priorité de découverte est laissée à celui dont la date de début au plus tôt, de la tâche correspondante, est la plus petite. Dans le cas où les dates de début au plut tôt sont identiques, la priorité de découverte est laissée au nœud associé à la tâche ayant la plus petite fenêtre d'exécution.

La ligne 4 permet d'initialiser les capacités de ressources à la date t_0. Pour chaque nœud i, la ligne 6 permet d'exclure les slots temporels $t \in [t_0, es_i[\cup]lf_i, \mathcal{H}]$ du domaine de démarrage de la tâche correspondant au nœud i.

Ensuite, l'algorithme découvre séquentiellement les nœuds suivant l'ordre de découverte définit. Chaque nœud est alors associé à un slot de démarrage de manière à respecter les contraintes de ressources le long de la durée d'exécution (ligne 10) de la tâche. Ensuite l'algorithme met à jour le domaine des successeurs en excluant les slots temporels correspondant à $[s_j, (s_j + duree(j) + tlmin_{j,j'})]$ avec $j \prec j'$ (ligne 15). Enfin, les lignes de 16 à 20 permettent de mettre à jour l'état des ressources tout au long de l'horizon de l'ordonnancement.

5.7 Application de l'approche de planification opérationnelle

Dans cette section nous présentons les expérimentations menées pour la phase de planification opérationnelle. Cette section est divisée en deux sous sections. La première introduit le contexte expérimental. La seconde quant à elle présente notre analyse des résultats obtenus.

Algorithme 13 : `Operationnal-Planner(`$G(X,\ E)$`,`rCap`)`

Entrées : $G(X,\ E)$ - *graphe* ; rCap - *capacités de ressources*

début

`// Heuristique de découverte des nœuds`

Ordonner et stocker les nœuds i dans decouverte *suivant trois niveaux de priorités :*

1. *les nœuds de plus grand degré d'abord*
2. *les dates de débuts au plus tôt* $\min(es_i)$
3. *les fenêtres de réalisation les plus petites* $\min(lf_i - es_i)$

`// Capacités de ressources initiales`

pour $k \leftarrow 1$ **to** K **faire**

  pour $t \leftarrow 1$ **to** $\mathcal{H}$ **faire** $\mathsf{Q}(t, r_k) = \mathsf{rCap}(r_k)$

`// Pour chaque nœud` i`, initialiser à vide la liste des dates d'exécution interdites`

pour $i \leftarrow 1$ **to** $\mid X \mid$ **faire**

  `AjouterA` (lstExclut(i), t') $\quad \forall t' \in [t_0, es_i[\cup]lf_i, \mathcal{H}]$

`// Affecter les dates de début suivant l'ordre de découverte des nœuds`

pour $i \leftarrow 1$ **to** $\mid X \mid$ **faire**

  courant $\leftarrow$ decouverte(i) `/* Choix du nœud à découvrir */`

  Associer le nœud sélectionné courant *à la date* debut *correspondant au slot temporel au plus tôt t.q*

$$\text{debut} \leftarrow \underset{t}{argmin} \begin{cases} \mathsf{Q}(t', r_k) \geq \mathsf{q}(\text{courant},\ r_k)\ \forall t' \in [t, t + \mathsf{dur}(\text{courant})] \\ t \notin \mathsf{lstExclut}(\text{courant}) \end{cases}$$

  si debut $\leq \mathcal{H}$ **alors**

    `// Choix de debut pour le nœud courant`

    datesDebut [courant] $\leftarrow$ debut

    `// Mise à jour de la liste des couleurs interdites pour les adjacents à courant`

    pour chaque (courant,suivant) / (courant $\prec$ suivant) $\wedge$ (suivant $\not\prec$ courant) **faire**

      $\forall t \in$ [debut , debut + dur(courant)+$\mathsf{tlmin}_{\text{courant,suivant}}$]

      `AjouterA` (lstExclut(courant), t)

    `// Mise à jour de l'état des ressources`

    pour chaque $r_k \in R$ **faire**

      si r_k *est une ressource consommable* **alors**

        $\mathsf{Q}(t, r_k) = \mathsf{Q}(t, r_k) - \mathsf{q}(\text{courant},\ r_k) \quad \forall\ t \geq$ debut

      sinon si r_k *est une ressource réutilisable* **alors**

        $\mathsf{Q}(t, r_k) = \mathsf{Q}(t, r_k) - \mathsf{q}(\text{courant},\ r_k)$

        $\forall$ [debut, debut + dur(courant)]

retourner datesDebut

fin

5.7.1 Contexte des expérimentations

5.7.1.a Rappel des plans tactiques utilisés pour la planification opérationnelle

Les expérimentations ont été menées en utilisant les instances B1[1-4]-LU15(*), B[1-4]-LU30(*) et B[1-4]-LU60(*) (cf. section 4.6.1). Pour ces instances, les plans tactiques considérés sont ceux représentés par les Figures 5.10, 5.11 et 5.12. L'horizon de planification opérationnel considéré est d'une semaine, allant du lundi au dimanche. Les samedis et dimanches sont des jours de repos. Aucune tâche ne peut être démarrée durant un week-end. Cependant, les tâches commencées en semaine peuvent s'étendre au week-end. La durée de travail journalière considérée est de 8 heures (de $8h$ à $16h$). Les interruptions de travail au sein de la journée ne sont pas prisent en compte.

5.7.1.b Capacités de ressources pour la planification opérationnelle

Sur la base de l'énoncé présenté dans la section 1.6, les capacités des ressources considérées sont celles précédemment décrites dans les Tableaux 5.5 et 5.6.

5.7.1.c Vitesse de réalisation des tâches

Le Tableau 5.11 rappel pour chaque tâche, la vitesse d'exécution (en $hectare/jour$) donnée par l'énoncé du problème (cf. section 1.6).

TABLE 5.11 – *Vitesse de travail*

Tâches	Labour	Semis	Désherbage	Fertilisation	Irrigation	Récolte	Déchaumage
Vitesse d'exécution (ha/j)	9	12	21	21	21	15	12

5.7.1.d Protocole expérimental

La résolution du problème de décision opérationnelle est réalisée en utilisant l'algorithme 13. Afin d'évaluer les performances de ce dernier, nous avons considéré les plans tactiques obtenus pour l'année 6. Ces plans tactiques correspondent à ceux de la première année après l'historique.

Chaque semaine de l'année considérée, des tâches doivent être réalisées conformément aux plans tactiques. Une planification opérationnelle est alors réalisée sur l'ensemble des tâches susceptibles d'être réalisé. Ces tâches sont ensuite exécutées. Ici, nous nous focalisons essentiellement sur la nature des plans opérationnels. Nous omettons volontairement les mécanismes liés au contrôle d'exécution de ces plans. Cette dimension est abordée dans le chapitre 6. La section suivante présente ainsi une analyse des plans opérationnels.

5.7.2 Analyse des résultats

5.7.2.a Performances de l'algorithme de planification opérationnelles

Pour chacune des planifications opérationnelles réalisées, nous avons stocké le nombre de nœuds du graphe puis le temps de recherche d'un plan opérationnel. La Fig-

ure 5.17 indique, pour un même nombre de nœuds du graphe, la durée moyenne de recherche d'un plan opérationnel. Cette durée n'intègre pas le temps de construction du graphe. Les résultats présentés par la figure sont obtenus avec un processeur Intel(R) Xeon(R) de 2.27GHz. Le temps de calcul mentionné est en milliseconde.

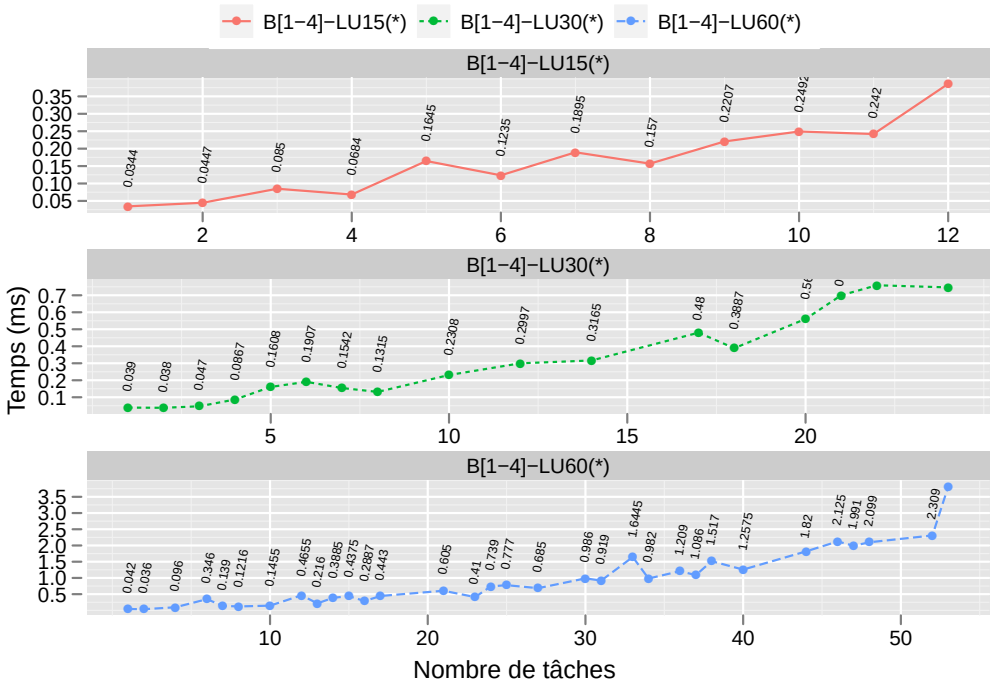


FIGURE 5.17 – *Performance pour la recherche des plans opérationnels*

Instance B1[1-4]-LU15(*) : pour un nombre de nœuds allant de 1 à 12, le temps moyen nécessaire pour trouver le plan opérationnel varie entre 0.04 à 0.38 millisecondes.

Instance B1[1-4]-LU30(*) : pour un nombre de nœuds allant de 1 à 24, le temps moyen nécessaire pour trouver le plan opérationnel varie entre 0.03 à 0.78 millisecondes.

Instance B1[1-4]-LU60(*) pour un nombre de nœuds allant de 1 à 53, le temps moyen nécessaire pour trouver le plan opérationnel varie entre 0.04 à 3.81 millisecondes.

D'après la figure, le temps de recherche d'un plan opérationnel n'est pas proportionnel au nombre de nœuds. Ceci peut s'expliquer par la densité des arcs du graphe. Ces résultats montrent qu'en terme de complexité temporelle, l'algorithme de planification opérationnelle peut être exploité dans un système de planification continue.

5.7.2.b Analyse des plans opérationnels

Considérant l'instance B[1-4]-LU15(*), la Figure 5.18 présente les plans opérationnels obtenus pour huit différentes semaines de l'année 1961. Nous avons choisi de nous focaliser sur ces semaines car elles correspondent aux périodes :

- de semis de l'orge de printemps,
- de semis du maïs,

– de récolte de l'orge de printemps,
– de semis du blé d'hiver,
– de récolte de maïs.

Pour ces mêmes semaines, les plans opérationnels obtenus pour les instances B[1-4]-LU30(*) et B1[1-4]-LU60(*) sont présentés en annexe A.5.

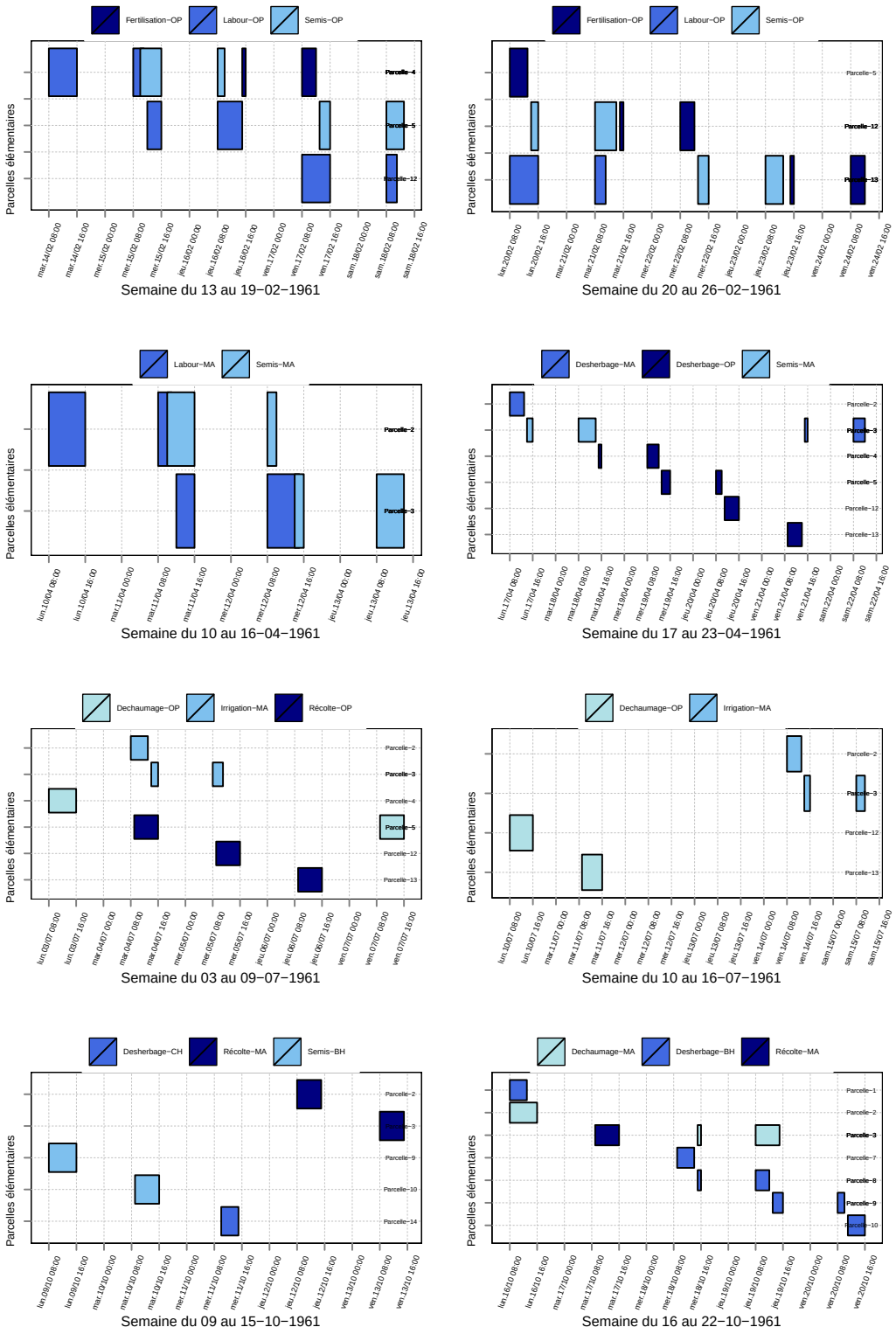


FIGURE 5.18 – *Instance B1[1-4]-LU15(*) : plan opérationnel obtenus pour 6 semaines de l'année 1961*

13 au 26 février Comme l'indique la Figure 5.18, les semaines du 13 au 19 février puis du 20 au 26 février correspondent aux période de semis de l'orge de printemps. La majorité des opérations agricoles de labours, de semis et de fertilisations ont été programmées sur ces deux semaines. Notons cependant qu'aucune opération n'a été prévue pour le lundi 13 février. Ceci résulte des dates de débuts aux plus tôt obtenues dans la phase de planification tactique.

10 au 23 avril Les opérations agricoles de labours et les semis du maïs ont été programmées pour les semaines du 10 au 16 avril. L'ITK utilisé pour le maïs impose un premier désherbage après le semis. Ces opérations de désherbages coïncident avec celle de l'orge de printemps. En conséquence, la semaine du 17 au 23 avril est principalement dédiée aux désherbages de l'orge de printemps et du maïs. Notons par ailleurs que l'opération de semis de maïs sur la parcelle élémentaire 3, précédemment prévue le mercredi 12 et le jeudi 13 a été reprogrammée le lundi 17 et mardi 18 avril. Cette nouvelle programmation du semis est une conséquence d'un échec lors de l'exécution de la première. Nous reviendrons (section 6.6.2) plus en détail sur les raisons de cette replanification.

3 au 16 juillet Les récoltes de l'orge de printemps commencent à la fin de la semaine précédent celle du 3 au 9 juillet. Elles se concentrent principalement sur cette dernière. Comme l'indique la figure, les récoltes de l'orge de printemps sont réalisées en parallèle avec l'irrigation des parcelles de maïs. Elles sont suivies d'une série de déchaumages d'orge de printemps qui s'étendent par ailleurs sur la semaine suivant celle du 3 au 9 juillet.

9 au 22 octobre Durant la première partie de la semaine du 9 au 15 octobre, les dernières parcelles de blé d'hiver ont été semées. La seconde partie de cette semaine est consacrée aux récoltes de maïs. Soulignons au passage que suite à un échec lors de l'exécution, l'opération de récolte de maïs sur la parcelle élémentaire 3, prévue pour le vendredi 13 a été reprogrammé au mardi de la semaine suivante (semaine du 16 au 22 octobre). Durant cette dernière semaine, les déchaumages de parcelles de maïs et les désherbages de blé ont été planifiés.

5.7.2.c Critiques des plans opérationnels

Les plans opérationnels présentés dans la section précédentes mettent en évidence l'incapacité de notre approche de résolution à prendre en compte la notion de chantier de semis ou de récolte. En effet, il aurait été souhaitable d'avoir un meilleur regroupement des tâches de semis et de récolte. Or, nous observons (semaines du 17 au 23 avril, du 03 au 09 juillet et du 16 au 22 octobre) que les opérations de semis et de récoltes sont parfois séparées par des opération de déchaumage et de désherbage. Cette limite des plans opérationnels est principalement liée à notre heuristique de découverte des nœuds du graphe (cf. section 5.6.2). Cet heuristique ne tient pas compte de la nature des tâches.

5.7.2.d Évolution de l'état des ressources

Afin d'illustrer l'évolution de l'état des ressources, les Figures 5.19 (semaine du 13 au 26 février) et 5.20 (semaine du 3 au 9 juillet) présentent pour les deux semaines considérées l'occupation des ressources.

Considérant la semaine du 13 au 26 février, nous pouvons observer que toutes les ressources sont inutilisé les lundi 13. Comme l'indique la Figure 5.17, aucune

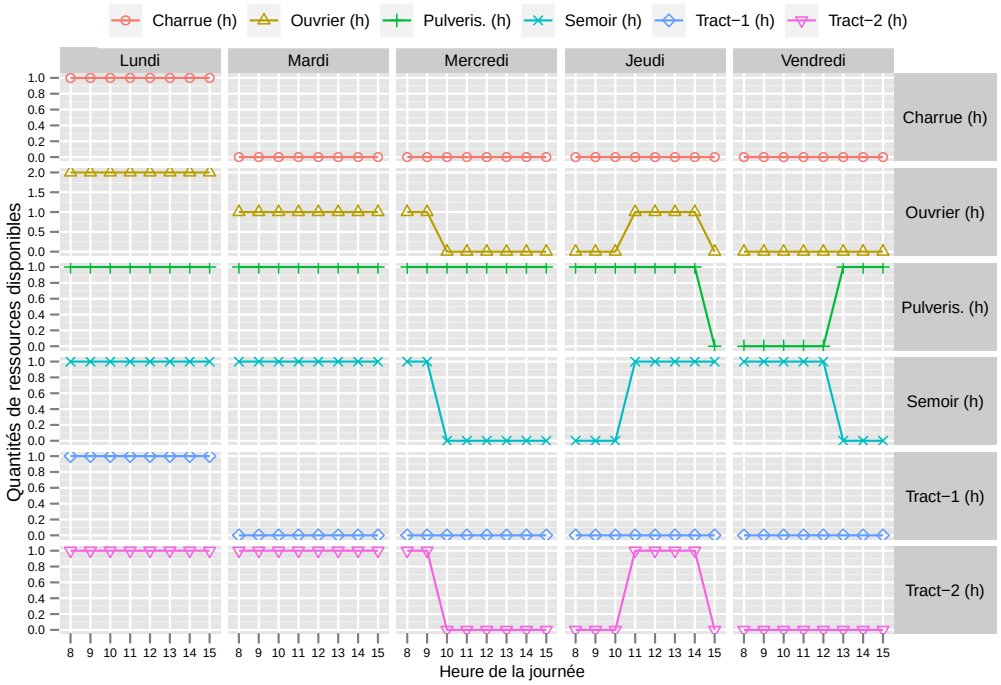


FIGURE 5.19 – *Évolution de l'état des ressources sur la semaine du 13 au 26 février*

tâche n'a été programmé pour ce jour. Pour le reste de la semaine, un ouvrier agricole avec le tracteur 1 et la charrue ont été affectés aux opération de labour. Le second ouvrier avec le tracteur 2 interviennent pour les opérations de semis et de fertilisation. Pour ce faire, le tracteur 2 est combiné avec un semoir ou un pulvérisateur.

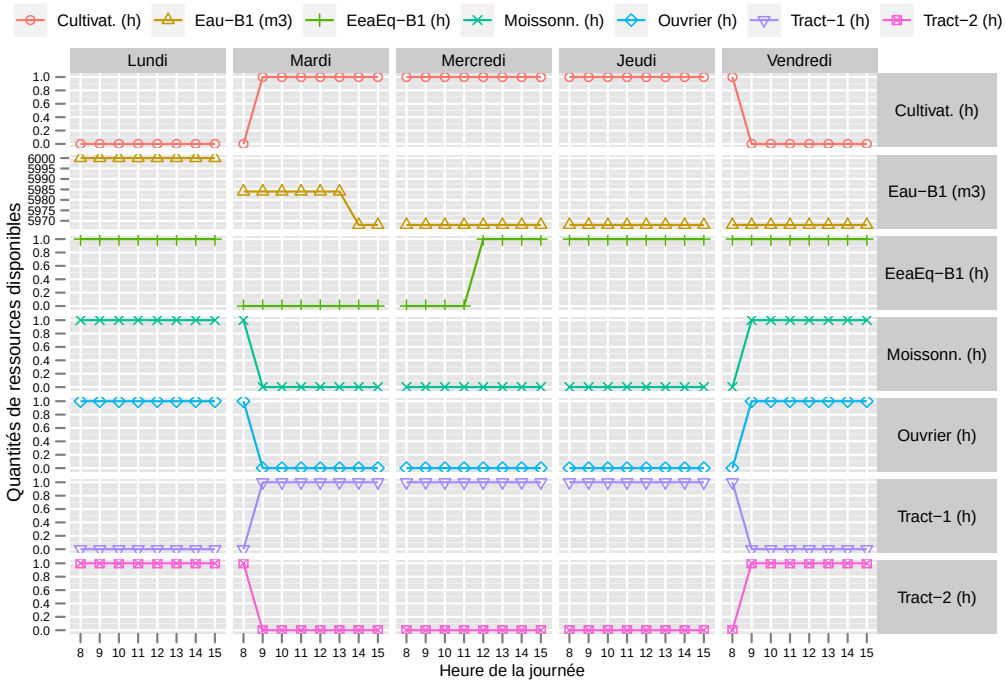


FIGURE 5.20 – *Évolution de l'état des ressources sur la semaine du 3 au 9 juillet*

Le lundi 3 et le vendredi 7 juillet, un ouvrier avec son tracteur 1 et le cultivateur

sont affectés à une opération de déchaumage. Ensuite, du mardi au jeudi, les deux ouvriers dotés du tracteur 2 et d'une moissonneuse batteuse se consacre à la récolte de l'orge de printemps. Enfin, le mardi et le vendredi, l'irrigation des parcelles de maïs se déroule de manière autonome en mobilisant le seul équipement d'irrigation disponible sur le bloc 1.

5.8 Bilan du chapitre

Dans ce chapitre nous avons apporté des réponses aux problèmes de décisions tactique et opérationnelle.

Décision tactique

Nous avons présenté comment le problème de décision tactique peut être abordé en utilisant des techniques de planification hiérarchique temporelle sous contrainte de ressources. L'heuristique de décomposition que nous avons proposé est une fonction globale qui permet d'expliciter les interdépendances, en matière de ressources, entre différents modes de conduites des cultures. Cette approche de planification nous permet de raisonner sur le temps de réalisation des tâches et sur les ressources nécessaires à leur exécution. Nous avons présenté notre algorithme de planification tactique (cf. Algorithme 11) qui permet de construire des plans tactiques flexibles basés sur des réseaux de contraintes temporelles simples (STN - *Simple Temporal Networks* (Dechter et al., 1991)).

Contrairement aux méthodes de planification tactique proposés par Attonaty et al. (1993) et Martin-Clouaire et Rellier (2009), nous introduisons des mécanismes permettant de raisonner numériquement sur le temps et les ressources. De plus, via l'heuristique de décomposition des tâches composées en tâches primitives, nous avons montré comment les effets globaux des itinéraires techniques peuvent être exploités lors de l'affectation des ITKs.

Cette proposition a pour avantage de mettre en adéquation les préférences de conduite des cultures d'un l'agriculteur et la charge de travail liée à la disponibilité effective des ressources de l'exploitation. En conséquence, nous pouvons à ce stade affirmer qu'en raisonnant sur les effets globaux des itinéraires techniques, l'agriculteur peut mettre en place un changement de la structure de l'exploitation (mise en place d'une politique d'achat de matériels agricole par exemple).

Notons toutefois que l'heuristique de décomposition proposée raisonne annuellement sur les consommations de ressources de chacune des décompositions. Cette approche peut s'avérer limitante. En effet, certaines périodes de l'année sont creuses tandis que d'autres sont particulièrement chargées. Le raisonnement annuel sur les disponibilités de ressources ne permet pas d'apprécier les périodes de fortes concurrences. Il serait donc envisageable d'améliorer l'heuristique afin de prendre en compte la nature des tâches et les priorités qui en découlent.

Décision opérationnelle

Nous avons montré comment l'algorithme de coloration de graphe peut être adapté (Algorithme 13) pour une résolution séquentielle du problème de décision opérationnel. Cela nous a permis de construire un ordonnancement à courts termes des opérations agricoles à réaliser. Contrairement à la majorité des travaux existants en agronomie, notre approche est capable d'anticiper l'organisation du travail sur l'ensemble de l'exploitation.

Notons toutefois que nous n'avons par évalué l'optimalité des ordonnancement. Il aurait été utile d'approfondir cette dimension de notre proposition afin de mieux positionner les performances de notre algorithme.

Dans le chapitre suivant, nous montrons comment les plans issus de la phase de planification opérationnelle peuvent être exécutés de manière distribuée. Nous présentons également la mise en œuvre globale du cadre.

Exécution distribuée de plans temporels en DSDE

6

Sommaire

5.1 Résolution des problèmes de décisions tactique et opérationnelle . . . 133
5.1.1 Rappel du problème de décision tactique . . . 133
5.1.2 Rappel du problème de décision opérationnelle . . . 134
5.1.3 Choix des approches de résolution . . . 135
5.2 État de l'art sur la planification hiérarchique temporelle 135
5.2.1 Les réseaux de tâches hiérarchiques . . . 135
5.2.2 Propagation des contraintes temporelles en planification . . . 138
5.2.3 Prise en compte de contraintes de ressources . . . 145
5.2.4 Bilan . . . 151
5.3 Représentation du domaine de planification tactique . . . 151
5.3.1 Prise en compte des effets . . . 151
5.3.2 Représentation des itinéraires techniques . . . 151
5.4 Affectation des ITKs et propagation des contraintes temporelles . . . 157
5.4.1 Définition des buts de la planification tactique . . . 157
5.4.2 Heuristique de décomposition du réseau de tâches . . . 158
5.4.3 Algorithme de planification tactique . . . 163
5.5 Application de l'approche de planification tactique . . . 165
5.5.1 Contexte des expérimentations . . . 166
5.5.2 Analyse des résultats . . . 168
5.5.3 Bilan sur la planification tactique . . . 175
5.6 Planification de la décision opérationnelle . . . 175
5.6.1 Modélisation du problème de décision opérationnelle . . . 176
5.6.2 Algorithme de résolution . . . 178
5.7 Application de l'approche de planification opérationnelle 179
5.7.1 Contexte des expérimentations . . . 181
5.7.2 Analyse des résultats . . . 181
5.8 Bilan du chapitre . . . 186

Dans ce chapitre nous présentons une nouvelle approche orientée modèles pour l'exécution de plans temporels. Cette approche s'inspire de l'exécutif proposé

dans RMPL. A la différence de ce dernier, nous nous basons sur le formalisme DSDE qui est un formalisme de modélisation et de simulation à événements discrets basé sur le formalisme DEVS issu des travaux de B. Zeigler. Dans un premier temps, nous introduisons les formalismes issus du cadre DEVS classique. Nous présentons ensuite la formalisation en DSDE et le fonctionnement de l'exécutif systémique que nous proposons. Cela nous permet d'aborder le fonctionnement globale de notre implémentation de l'architecture SAFIHR que nous avons précédemment présentée.

6.1 État de l'art sur les formalismes PDEVS et DSDE

Avant d'aborder la formalisation de notre système d'exécution de plans temporels nous présentons les formalismes PDEVS et DEVS utilisé dans cette thèse.. Pour cela nous présentons dans les sections 6.1.1 et 6.1.2.a les formalismes DEVS parallèle, DS-DEVS et DSDE.

6.1.1 Formalisme DEVS parallèle

6.1.1.a DEVS parallèle

Le formalisme DEVS parallèle (*Parallel Discrete EVent system specification* PDEVS) est un formalisme issu de DEVS classique proposée par Chow et Zeigler (1994). Ce formalisme propose des mécanismes pour la prise en compte de conflits entre les fonctions de transitions interne et externe. Pour ce faire, PDEVS laisse le choix au modélisateur de spécifier une fonction de conflit pour traiter les transitions interne et externe simultanées. De plus, afin de palier aux problèmes liés aux *événements simultanés* dans la version classique (cf. section 2.2.1), l'extension DEVS parallèle utilise des *bags* d'événements destinées à la fonction de transition externe. Ces bags permettent de collecter des événements émis à la même date. Ainsi, avec PDEVS, plusieurs événements externes peuvent être réalisés à la même date. Ces événements sont collectés et stockés comme des ensembles d'événements survenus à la même date. Les sorties réalisées par les modèles imminents [1] sont stockées dans des bags d'entrées notés I^b. Chacun des événements du bag est identifié par sa date d'occurrence. Aucune relation d'ordre n'est préconisée pour les événements appartenant à un même bag. On peut ainsi autoriser et dénombrer les évènements simultanés sur chaque port d'entrée. La prise en compte des événements du bag n'est autorisée qu'après les transitions internes de tous les modèles imminents. De ce fait, les transitions externes sont réalisées par des bags représentant ainsi la réponse agrégée des événements simultanés.

6.1.1.b Modèle DEVS atomique parallèle

Définition 6.1 (Modèle DEVS atomique parallèle) *Le modèle DEVS atomique parallèle* M *est décrit par un tuple* $\langle I, O, S, \tau, \delta_{int}, \delta_{ext}, \delta_{con}, \lambda \rangle$ *où*

I *est l'ensemble des ports* i *et des valeurs d'entrées,*
O *est l'ensemble des ports* o *et des valeurs de sorties,*
S *est l'ensemble des états partiels du système,*
$\tau : S \rightarrow \mathbb{R}_0^+$ *est la fonction d'avancement du temps,*
$\delta_{int} : S \times S$ *est la fonction de transition interne,*
$\delta_{ext} : Q \times I^b \rightarrow S$ *est la fonction de transition externe où,*
I^b *est l'ensemble des* bags *des entrées* I*,*
Q *est l'ensemble des états totaux,*
$Q = \{(s, e) | s \in S, 0 \leq e \leq \tau(s)\}$*,*
e *est le temps écoulé depuis la dernière transition,*
$\delta_{con} : S \times I^b \rightarrow S$ *est la fonction de conflit,*
$\lambda : S \rightarrow O^b$ *est la fonction de sortie*

De la même manière que pour la version classique, en absence d'événements sur les ports d'entrées, le système conserve un état passif jusqu'à la prochaine date

1. Les modèles imminents sont les modèles en conflits à une date donnée c'est-à-dire l'ensemble des modèles pour lesquels une transition interne est prévue à une même date (cf. section 2.2.2).

prévue pour la transition interne. Une sortie est alors réalisée suivie de la transition interne. Si un événement externe apparait sur l'un des ports d'entrée avant la date prévue pour la transition interne, l'état du système passe à $\delta_{ext}(s, e, i^b)$. À la différence de DEVS classique, la transition externe se base sur les bags d'événements provenant d'un ou de plusieurs modèles. Si un événement apparait sur I^b à $e = \tau(s)$, le système passe dans l'état $\delta_{con}(s, e, i^b)$. Le comportement de la fonction de conflit est défini par le modélisateur. Par défaut $\delta_{con} = \delta_{ext}(\delta_{int}(s, e), 0, i^b))$, donnant ainsi la priorité à la transition interne lors d'un conflit.

6.1.1.c Modèle DEVS couplé parallèle

La structure des modèles couplés dans l'extension DEVS parallèle est quasiment identique à celle de la version classique. Les modèles atomiques sont tous parallèles. On observe toutefois la suppression de la fonction de sélection.

Définition 6.2 (Modèle couplé DEVS parallèle) *Plus formellement, un modèle DEVS couplé M est décrit par un tuple $\langle I, O, N, \{M_n\}, \{\zeta_n\}, \{Z_{n,n'}\}\rangle$ où*

I est l'ensemble des ports et des valeurs d'entrées,

O est l'ensemble des ports et des valeurs de sorties,

N est l'ensemble des identifiants des modèles constituant le modèle couplé M, y compris l'identifiant « self » de M lui même,

M_n est un modèle DEVS (atomique/couplé) constituant au réseau hiérarchique et indexé par $n \in N$,

ζ_n est l'ensemble des modèles qui influencent le modèle n,

$Z_{n,n'}$ est une famille de fonctions de transfert telles que :

$$Z_{n,n'} : O_n \to I_{n'} \quad si \quad n, n' \in N \text{ et } n \in \zeta_{n'},$$
$$Z_{\mathsf{self},k} : I \to I_k \quad si \quad k \in N \text{ et } \mathsf{self} \in \zeta_k,$$
$$Z_{k,\mathsf{self}} : O_k \to O \quad si \quad k \in N \text{ et } k \in \zeta_{\mathsf{self}},$$

Une conséquence directe de la suppression de la fonction de sélection est la prise en compte des modèles imminents. En effet, tous les modèles imminents sont autorisés à effectuer une sortie.

D'après la propriété de fermeture sous couplage de DEVS parallèle, (cf. A.2.2) un modèle couplé $M = \langle I, O, N, \{M_n\}, \{\zeta_n\}, \{Z_{n,n'}\}\rangle$ peut être assimilé à sa résultante[2] $M_r = \langle I, O, S, \tau, \delta_{int}, \delta_{ext}, \delta_{con}, \lambda\rangle$.

6.1.2 Formalismes DEVS à structures dynamiques DS-DEVS et DSDE

6.1.2.a Formalisme DS-DEVS

Généralement lorsqu'on s'intéresse à l'étude des systèmes complexes, ces derniers sont décomposés en sous systèmes plus simples. Nous avons présenté dans les sections 2.2.1 et 6.1.1, les formalismes DEVS et parallèle DEVS. Dans ces formalismes, la structure du système est définie par une composition hiérarchique fixe de modèles couplés. Hors, l'un des reproches à DEVS classique et à DEVS parallèle est justement leurs incapacités à modifier dynamiquement leurs structures en cours de simulation. En effet, le comportement d'un modèle couplé est déterminé par celui des modèles atomiques le constituant. La structure du système est statique et est indépendante de son comportement. Cette caractéristique est très limitante notamment pour les applications du monde réel. Par exemple, une structure fixe

2. La résultante d'un modèle couplé est un modèle DEVS atomique auquel il peut être associé.

ne permet pas de modéliser efficacement la dynamique du parcellaire d'une exploitation agricole. En effet, considérons le parcellaire d'une exploitation agricole décrit par un modèle couplé constitué de modèles atomiques DEVS représentant les parcelles. D'une année à l'autre, une parcelle peut être découpée en parcelles plus petites ou fusionnée avec d'autres parcelles. Une structuration fixe du modèle couplé empêche l'ajout, la suppression de modèles et de connexions durant la simulation.

Plusieurs approches (Barros, 1995; Uhrmacher, 2001) ont été proposées afin de dépasser cette limite. Barros (1995) propose le formalisme *Dynamic Structure DEVS* (DS-DEVS Barros (1995, 1996, 1997, 2003)) basé sur DEVS classique et qui fournit des mécanismes pour le changement dynamique de la *structure* d'un modèle DEVS. Cette spécification n'effectue aucune modification de spécification des modèles atomiques. Dans le formalisme DS-DEVS, le modèle couplé contient un modèle particulier appelé *exécutive*. L'état de ce modèle contient un modèle couplé. Autrement dit, chaque état de l'exécutive est associé à une structure du modèle couplé dont la dynamique dépend des transitions internes et externes de l'exécutive.

Le changement dynamique de structures se base sur des réseaux de modèles DEVS atomiques. Les différences par rapport aux modèles couplés de DEVS sont que les listes de connexions et de modèles du réseau peuvent changer au cours du temps. Le réseau à structure dynamique est défini par un tuple $DSDEN = \langle \chi, M_\chi \rangle$ où χ est le nom de l'exécutive et M_χ le modèle associé à l'exécutive χ. Il détermine la manière dont la structure change au cours du temps.

Définition 6.3 (Modèle M_χ DS-DEVS) *Le modèle M_χ est défini par un tuple* $\left\langle I_\chi, O_\chi, S_\chi, s^0_\chi, \tau_\chi, \gamma, \Sigma^*, \delta_{int,\chi}, \delta_{ext,\chi}, \lambda_\chi \right\rangle$ *où*

I_χ *est l'ensemble des ports et des valeurs d'entrées,*
O_χ *est l'ensemble des ports et des valeurs de sorties,*
S_χ *est l'ensemble des états partiels du système,*
s^0_χ *est l'état partiel initial du système,*
$\tau_\chi : S_\chi \to \mathbb{R}^+_0$ *est la fonction d'avancement du temps,*
$\gamma : Q_\chi \to \Sigma^*$ *est la fonction de structure,*
Σ^* *est l'ensemble des structures,*
$\delta_{int,\chi} : S_\chi \times S_\chi$ *est la fonction de transition interne,*
$\delta_{ext,\chi} : Q_\chi \times I_\chi \to S_\chi$ *est la fonction de transition externe où :*
  Q_χ *est l'ensemble des états totaux,*
  $Q_\chi = \{(s_{\alpha,\chi}, e) | s_{\alpha,\chi} \in S_\chi, 0 \leq e \leq \tau(s_{\alpha,\chi})\}$
  e *est le temps écoulé depuis la dernière transition,*
$\lambda_\chi : S_\chi \to O_\chi$ *est la fonction de sortie.*

À l'exception de Σ^* et γ, cette structure est proche de celle des modèles atomiques classiques. L'état partiel $s^\alpha_\chi \in S_\chi$ contient des informations relatives au comportement et à la structure du réseau de modèles. Considérant un état partiel, s^α_χ, la structure du réseau de modèles $\Sigma^\alpha \in \Sigma^*$ est définie par $\Sigma^\alpha = \gamma(s^\alpha_\chi) = (I^\alpha, O^\alpha, N^\alpha, \{M^\alpha_n\}, \{\zeta^\alpha_n\}, \{Z^\alpha_{n,n'}\})$ où les paramètres sont ceux décrits dans la définition 2.8 (modèle DEVS couplé). Le réseau est défini par une composition hiérarchique des modèles DEVS atomiques et couplés.

Tout changement provoqué par l'appel de $\gamma(s^\alpha_\chi)$ se traduit par l'ajout ou la suppression de modèles et le changement des connexions entre modèles. Par exemple, la valeur I^α définit l'ensemble de ports d'entrées actifs. Cet ensemble est un sous-ensemble de I_χ. S'il est modifié, cela signifie que les événements admissibles par le modèle changent.

6.1.2.b Fermeture sous couplage

Barros (1995) démontre que le formalisme DS-DEVS est conforme aux spécifications DEVS et possède donc la capacité d'être couplé avec tout modèle DEVS. Du point de vue purement théorique, DS-DEVS profite de la fermeture sous couplage de DEVS. Cette propriété bien que correcte n'est pas réalisable en pratique. En effet, associer DS-DEVS à un modèle atomique DEVS, suppose l'explicitation de toutes les structures possibles. Celles-ci seront donc activées en fonction du comportement de l'exécutive. L'approche est simplement irréalisable car connaitre à priori les structures possibles du modèle est une tâche de modélisation difficile voir impossible à réaliser.

6.1.2.c Formalisme DSDE

Le formalisme DSDE *parallel Dynamic Structure Discrete Event system specification* Barros (1997, 1998) est une version parallèle de DS-DEVS. À l'instar de PDEVS, à chaque modèle est associée une fonction de conflit capable de prendre en compte des transitions interne et externe simultanées. Ce formalisme intègre également la notion de *bags* d'événements permettant ainsi de mieux appréhender la collecte et la gestion des événements qui sont émis à une même date. DSDE peut également piloter des modèles atomiques PDEVS.

Nous nous basons sur l'extension DSDE. La première raison de ce choix est qu'il offre plus de souplesses dans la modélisation de problèmes. La seconde raison est que le noyau de simulation de l'environnement VLE [3](Quesnel et al., 2009) sur lequel repose le projet RECORD [4] (Chabrier et al., 2007; Bergez et al., 2012), à la base de cette thèse, repose sur le formalisme DSDE.

Dans la suite de ce chapitre, nous montrons le principe de fonctionnement d'un exécutif de plans basé sur le formalisme DSDE.

6.2 Interactions entre le coordinateur et l'exécutif de plan

Le coordinateur centralisé s'occupe de la construction et de l'exécution du plan courant. La description des méthodes de planification est détaillée dans les chapitres 4 et 5. Dans cette section, nous proposons une approche systémique pour l'exécution du plan courant de l'agent. Cette approche, qui s'appuie sur le formalisme DSDE (cf. section 6.1.2.a), permet de représenter les tâches de l'agent comme des systèmes dynamiques ayant leur propre autonomie. Ces systèmes dynamiques interagissent entre eux afin de satisfaire les relations temporelles entre les tâches du plan courant. Nous pouvons ainsi assimiler l'exécutif du plan à un système dynamique distribué dans lequel chaque tâche s'exécute de manières indépendantes.

Pour ce faire, nous décrivons dans un premier temps la formalisation et l'exécution des différents éléments du plan. Cette description nous permet d'aborder le fonctionnement du coordinateur et ses interactions avec le plan courant de l'agent.

3. *Virtual Laboratory Environment*
4. http://www.inra.fr/record

6.2.1 Différence terminologique entre exécutive DEVS et exécutif de plan

Dans les sections suivantes, il est primordial de faire la distinction entre l'*exécutive DEVS* et l'*exécutif de plan*. Ces deux terminologies se rapportent à des éléments différents de l'architecture SAFIHR.

Premièrement, le terme *exécutive DEVS* se rapporte à la notion d'exécutive issue du formalisme DSDE (Barros, 1997). Ce terme définit le modèle particulier appelé *exécutive* contenu dans un modèle couplé DSDE. Comme décrit dans la section 6.1.2.a, l'état d'un exécutive DEVS est associé à la structure d'un modèle couplé DEVS. La dynamique de cette structure dépend des transitions internes et externes du modèle exécutive. Dans l'architecture SAFIHR, le coordinateur est un exécutive DEVS χ décrit par un modèle M_χ.

Par ailleurs, le terme *exécutif de plan* se rapporte au contrôleur d'exécution du plan courant. Ce terme est de ce fait proche du vocabulaire utilisé par exemple dans IxTeT-EXEC (Lemai, 2004). Ainsi, l'exécutif de plan intègre les dimensions temporelle et procédurale de l'exécution d'un plan. La première contrôle les dates de début et de fin des tâches contenues dans le plan. La seconde est quant à elle liée au (i) déclenchement de l'exécution de la tâche dans le système opérant et (ii) à l'établissement des préconditions d'activation et de fermeture portant sur les variables observées du système de croyance. Dans l'architecture SAFIHR, le plan courant π est contrôlé par l'exécutif de plan.

6.2.2 Coordinateur et l'exécutif de plan

Dans SAFIHR, le coordinateur du système intentionnel (*exécutive DEVS*) et le plan courant de l'agent sont définis par un modèle couplé DSDE (voir la Figure 6.1). Les entrées I_{IS} de ce modèle couplé sont connectées d'une part aux sorties du système de croyance $M_\mathcal{B}$ (cf. section 3.3.5) et d'autre part aux modèles DEVS atomiques de délibération (*WCSP solver*, *HTN planner* et *Ressource manager*) détaillés dans les chapitres 4 et 5. Les sorties O_A et O_χ sont respectivement celles de l'agent vers le système opérant et celles du coordinateur vers les modèles de délibération.

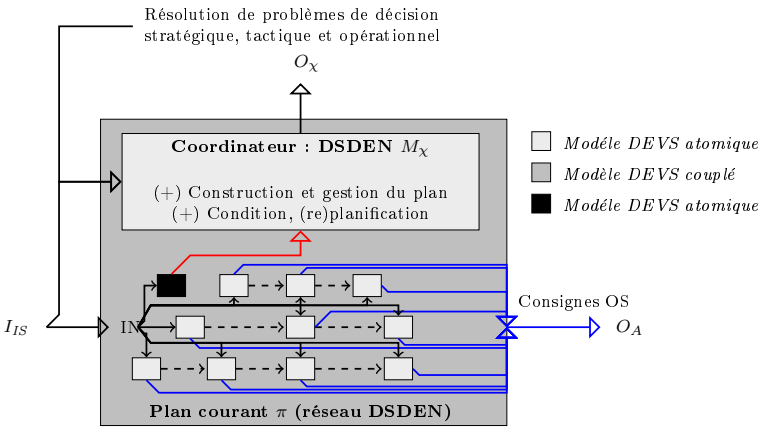


FIGURE 6.1 – *Coordinateur et exécutif distribué de plans temporels dans* SAFIHR

Le réseau DSDEN associé à l'exécutif χ décrit le plan courant en cours d'exécution. Chacun des modèles atomiques du réseau représente une tâche du plan courant de l'agent. Les relations temporelles entre les tâches sont établies via des modèles particuliers que nous décrivons dans la section 6.4.

Dans le réseau DSDEN, nous distinguons deux classes de modèles de tâches qui sont :

– les *modèles de tâches opérationnelles* (gris clair) : dont le rôle est d'envoyer des consignes paramétrées d'exécution de tâches au système opérant. Ces messages sont envoyés via les sorties O_A (cf. section 3.1.5.a).
– les *modèles de tâches de planification* (noir) : dont le rôle est d'envoyer au coordinateur M_χ, des alertes de déclenchement d'une résolution de problèmes de décision. Les tâches de planification peuvent être perçues comme les « choise point » de l'exécutif procédural PROPEL [5] (*PROcedure Planning and Execution Language* Levinson (1995)).

Ces modèles de tâches sont directement connectés au système de croyance via les ports d'entrées I_{IS} du système intentionnel. La Figure 6.1 montre les connexions entre les modèles de tâches, le système de croyances (cf. l'architecture complète représentée par la Figure 3.6) et le système opérant OS. Dans les trois sections suivantes, nous développons le fonctionnement de l'exécutif de plan distribué (cf. sections 6.2.3, 6.3 et 6.4) et celui du coordinateur (cf. section 6.5).

6.2.3 Description des tâches opérationnelles $\mathcal{A}$ et de planification $\mathcal{P}$

6.2.3.a Description des tâches opérationnelles $\mathcal{A}$ de l'exécutif

Les tâches opérationnelles sont des tâches duratives, exécutables sur l'environnement dont l'effet direct consiste à déclencher un processus opérationnel du système opérant. Ces tâches permettent de contrôler l'état du système opérant (ex : commencer le semis, arrêter l'irrigation) dans l'optique de perturber la dynamique du système bio-physique.

La description de l'état des tâches opérationnelles dans l'exécutif ajoute aux contraintes temporelles absolues [6] de la tâche, des informations relatives au contexte d'exécution. Rappelons que les contraintes temporelles d'une tâche a portent sur sa durée d_a et sur les dates de début $s_a \in [l_{s_a}, u_{s_a}]$ et de fin $f_a \in [l_{f_a}, u_{f_a}]$ (cf. section 5.2.2.b).

Définition 6.4 (État d'une tâche opérationnelle dans l'exécutif) *Dans l'exécutif, une tâche opérationnelle a est définie par un tuple $\langle id, C, Status, d_a, t_{start}, t_{end}, E\rangle$ où :*

▷ *id : l'identifiant de la tâche,*

▷ *$Status$: le statut de la tâche. L'ensemble des statuts possibles pour une tâche est* Init*,* Wait*,* Started*,* Failed*,* Finished *(cf. section 6.2.3.b),*

▷ *C : les conditions sur les transitions entre les statuts. Ces conditions sont les prédicats basés sur les événements reçus du système de croyance,*

▷ *d_a : durée d'exécution de la tâche,*

▷ *t_{start} et t_{end} : sont respectivement les intervalles d'ouverture ($t_{start} = [l_{s_a}, u_{s_a}[$) et de fermeture ($t_{end} = [l_{f_a}, u_{f_a}[$) des tâches (cf. section 6.2.3.c),*

▷ *E : les effets* directs *de l'exécution des tâches et les effets* attendus *du processus opérationnel associé à celle-ci (cf. section 6.2.3.d).*

6.2.3.b Statut des tâches opérationnelles

Comme l'indique la Figure 6.2, le statut de la tâche est défini par un automate à états finis dont les états sont *Init, Wait, Started, Failed, Finished.*

5. Une brève description est présentée dans la section 2.1.5.a.
6. Attention, il ne s'agit pas des contraintes de précédences entre tâches mais essentiellement des contraintes portant sur les dates de début et de fin.

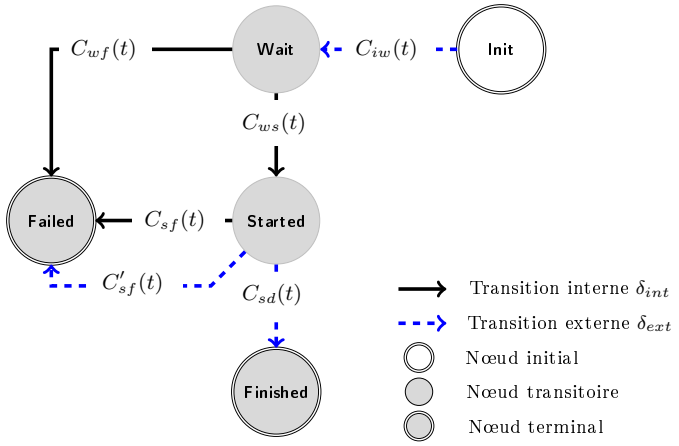


FIGURE 6.2 – *Automate de transition entre statut.*

L'automate décrit sur la Figure 6.2 montre les états et les transitions entre les statuts d'une tâche opérationnelle. Le statut *Init* est associé à la phase d'initialisation de la tâche. Le statut *Wait* décrit une phase dans laquelle les conditions d'exécution ne sont pas valides. Dès que ces conditions d'exécution sont valides, la tâche passe dans l'état *Started*. Ce statut indique que la tâche est en cours d'exécution par le système opérant. Dès l'instant que le processus opérationnel arrive à son terme, le statut passe à *Finished*. Par ailleurs, l'exécution d'une tâche n'est pas déterministe. Il peut y avoir des incertitudes de différentes natures qui empêchent la réalisation effective d'une tâche. Dans ces conditions, la tâche est annulée passant ainsi dans le statut *Failed*.

6.2.3.c Contraintes temporelles absolues d'une tâche opérationnelle

Dans l'exécutif, les contraintes temporelles absolues d'une tâche opérationnelle a, déterminent si la tâche est impérative ou non. Il s'agit de la date de :

1. début au plus tôt noté $l_{s_a} \in \mathbb{R}_0^+$,
2. début au plus tard noté $u_{s_a} \in \mathbb{R}_0^+$,
3. fin au plus tôt noté $l_{f_a} \in \mathbb{R}_0^+$,
4. fin au plus tard $u_{f_a} \in \mathbb{R}_0^+$.

l_{s_a} et u_{s_a} sont les bornes inférieure et supérieure définissant l'intervalle d'ouverture t_{start} de la tâche opérationnelle. De la même manière l_{f_a} et u_{f_a} sont les bornes inférieure et supérieure définissant l'intervalle de fermeture t_{end} de la tâche. À ces contraintes temporelles absolues se rajoute une contrainte temporelle durative d_a.

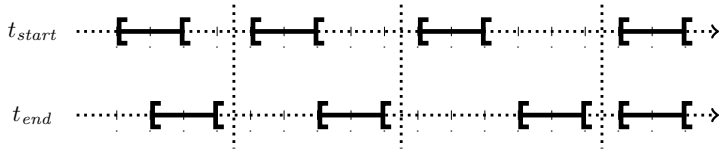


FIGURE 6.3 – *Les quatre configurations des contraintes temporelles absolues d'une tâche opérationnelle.*

Le Figure 6.3 montre les différentes configurations de la structure temporelle des tâches opérationnelles. Nous considérons par défaut que les bornes supérieures (u_{s_a} et u_{f_a}) sont exclues des intervalles. La justification de cette exclusion repose sur le fait qu'il est théoriquement impossible en DSDE d'assurer la réception de tous les bags événements datant d'une date donnée. En conséquence, la satisfaction

des contraintes relatives aux dates au plus tard ne pourrait être instantanément garantie.

6.2.3.d Les effets d'une tâche opérationnelle

Il existe deux classes d'effets *directs* pour une tâche opérationnelle. La première consiste à déclencher l'exécution de la tâche du système opérant (ex : commencer le semis). La seconde consiste à modifier les capacités et les disponibilités des ressources. Pour les mêmes raisons que celles évoquées dans la section 5.3.1 nous ne prenons pas en compte explicitement les effets *attendus* d'une tâche opérationnelle.

6.2.3.e Description des tâches de planification $\mathcal{P}$ de l'exécutif

Par définition, les tâches de planification sont décrites de la même manière que les tâches opérationnelles c'est-à-dire avec le tuple suivant : $\langle id, C, Status, d_a, t_{start}, t_{end}, E\rangle$. Cependant, contrairement aux tâches opérationnelles, les tâches de planification sont exécutables dans le monde virtuel. Elles peuvent être perçues comme les « *choise point* » de PROPEL. Ainsi, les tâches de planification représentent des points de décision directement intégrés dans le plan.

L'effet direct d'une tâche de planification est de déclencher la résolution d'un problème afin de modifier l'état interne de l'agent. Ces tâches modélisent en conséquence les buts de l'agent. Elles provoquent la construction, l'expansion ou la réduction du plan courant. Nous faisons l'hypothèse que l'exécution d'une tâche de planification est instantanée.

En résumé, dans l'exécutif, une tâche (opérationnelle ou de planification) est caractérisée par ses conditions d'exécutions, son statut d'exécution et les contraintes temporelles absolues et duratives auxquelles elle est soumise.

6.3 Formalisation PDEVS des tâches opérationnelles $\mathcal{A}$ et de planification $\mathcal{P}$

Dans cette section, nous présentons une version détaillée en PDEVS du comportement d'une tâche. Cette formalisation a été précédemment proposée dans Akplogan et al. (2010). Pour ce faire, nous nous basons sur les descriptions des tâches proposées dans les sections 6.2.3.a et 6.2.3.e. Nous suggérons de modéliser les tâches comme des modèles (M_T) atomiques PDEVS. Chaque tâche est donc décrite par un système dynamique.

Le comportement général des modèles de tâches est d'envoyer des consignes d'exécution de tâches au système. Ces consignes sont envoyées en fonctions de l'état courant de la tâche et des événements perçus sur ses ports d'entrées.

6.3.1 Interface des modèles génériques de tâches

Notre modélisation des interfaces des modèles de tâches est réalisée dans l'optique de prendre en compte les relations temporelles entre les tâches (cf. section 6.4). Le modèle générique de tâches M_T est formalisé comme suit.

$$
\begin{aligned}
M_T &= \langle I_T, O_T, S_T, \tau, \delta_{int}, \delta_{ext}, \delta_{con}, \lambda\rangle \\
I_T &= \{\mathsf{ACK}, \mathsf{Pcond}, I_S = \{I_{s_j}\}, I_F = \{I_{f_k}\} \text{ avec } \; j \in \mathbb{N} \text{ et } \; k \in \mathbb{N}\} \\
O_T &= \{\mathsf{OS}, \mathsf{start}, \mathsf{finish}, \mathsf{fail}\}
\end{aligned}
$$

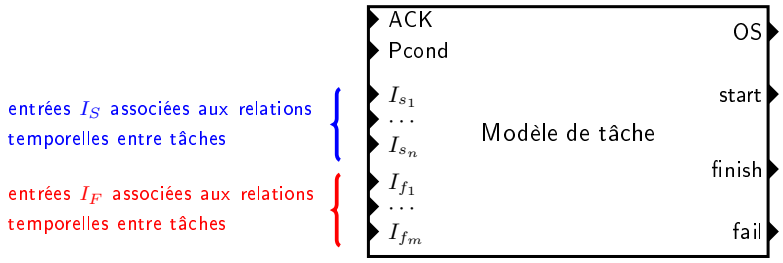


FIGURE 6.4 – *Interface générique des tâches opérationnelles et de planification.*

Les entrées ACK et Pcond sont connectées aux croyances issues du système de croyance $\mathcal{B}$. La première (ACK) est relative aux notifications liées à l'évolution des processus opérationnels du système opérant (ex : la tâche est réalisée). La seconde (Pcond) est associée aux croyances relatives à la dynamique des processus biophysiques (ex : les préconditions du semis sont valides). Les ensembles d'entrées I_{S_j} et I_{F_k} servent à la modélisation des relations temporelles entre tâches. Le premier (I_{S_j}) est destiné aux relations temporelles qui portent sur le démarrage de la tâche. Le second quant à lui est destiné aux relations temporelles qui portent sur la fermeture de la tâche. Dans la suite du document, nous notons J_s et J_f respectivement le nombre de ports I_{S_j} et I_{F_k}

La sortie OS est associée aux consignes d'exécution de tâches. Elle permet de modéliser l'effet direct de la tâche. Les sorties start, finish et fail sont respectivement des sorties de notifications associées au démarrage, à la fin et à l'échec de l'exécution de la tâche.

6.3.2 L'état d'un modèle de tâches

L'état S_T des tâches est défini par :

$$\begin{aligned} S_T &= \langle id, Status, t, [l_s, u_s[, [l_f, u_f[, C, \alpha_{OS}, \alpha_{start}, \alpha_{finish}, \alpha_{fail}, \\ &\quad \overrightarrow{\Omega_S}, \overrightarrow{\Omega_F}, \overrightarrow{\Theta_S}, \overrightarrow{\Theta_F}, \beta_S, \beta_F, \sigma \rangle \\ Status &= \{Init, Wait, Started, Finished, Failed\} \end{aligned}$$

Notons avant tout que les contraintes temporelles duratives ne sont pas représentées dans l'état S_T du modèle de tâche. En effet, la durée effective de l'exécution d'une tâche ne peut être connue qu'après sa mise en œuvre dans le système opérant. Ignorer ces contraintes pour l'exécution des tâches permet d'autoriser des retards inhérents aux processus opérationnels du système opérant.

Les variables $id, Status, [l_s, u_s[, [l_f, u_f[$ et C (cf. Figure 6.3) sont celles décrites dans la section 6.2.3. La variable t représente la date courante de la simulation. Les variables α_{OS}, α_{start}, α_{finish} et α_{fail} sont des booléens permettant d'activer les sorties respectivement vers les ports OS, start, finish et fail.

$\overrightarrow{\Omega_S}$ et $\overrightarrow{\Omega_F}$ sont des vecteurs de booléens qui permettent de valider les relations temporelles qui portent respectivement sur le début et la fin des tâches. $\overrightarrow{\Omega_S} = (\omega_{s1}, \cdots, \omega_{sn})_{J_s}$ où J_s est le nombre d'entrées I_{s_j}. Chaque booléen ω_{sj} est affecté à *vrai* si un événement externe est reçu sur le port d'entrée I_{s_j}. Ainsi, chaque port I_{s_j} modélise un opérateur *OU*. Lorsque plusieurs tâches sont connectées sur un seul et même port I_{s_j}, le premier événement reçu de l'un d'entre eux, valide la relation temporelle liée au démarrage de la tâche. L'ensemble des ports d'entrée I_{s_j} modélise un opérateur *ET*. Toutes les relations temporelles liées au démarrage de la tâche sont satisfaites dès qu'il y a au moins un événement externe sur chaque port I_{s_j}. $\overrightarrow{\Omega_F} = (\omega_{f1}, \cdots, \omega_{fm})_{J_f}$ est équivalent à $\overrightarrow{\Omega_S}$. La seule différence est que

contrairement à ce dernier, $\overrightarrow{\Omega_F}$ est relatif aux événements sur les ports I_{f_k} qui par définition portent sur les relations temporelles liées à la fermeture de la tâche.

Les variables $\overrightarrow{\Theta_S} = (\theta_{s1}, \cdots, \theta_{sn})_{J_s}$ et $\overrightarrow{\Theta_F} = (\theta_{f1}, \cdots, \theta_{sm})_{J_f}$ sont des vecteurs de réels positifs qui définissent les gardes temporelles respectivement sur le démarrage et la fermeture des tâches. Autrement dit, si un événement externe est reçu sur le port d'entrée I_{s_j} (respectivement I_{f_k}), chaque réel θ_{sj} (respectivement θ_{fj}) est assigné à une valeur contenue dans le message. Cette valeur indique une garde temporelle sur le délai de démarrage (respectivement le délai de fermeture) de la tâche.

Les variables β_S et β_F sont des booléens dont la valeur de vérité passe à *vrai* si toutes les contraintes liées aux relations temporelles de démarrage ou de fermeture sont satisfaites. Ainsi, $\beta_S = \bigwedge_{j=1}^{J_s} \omega_{sj}$ et $\beta_F = \bigwedge_{k=1}^{J_f} \omega_{fk}$.

Enfin, $\sigma \in \mathbb{R}_0^+$ permet de gérer la durée dans un état lorsque le modèle de tâche est perturbé par des événements externes. Ainsi, dans la suite de la formalisation, la fonction d'avancement du temps $\tau(S_T) = \sigma$.

6.3.3 Initialisation du modèle et transition de $Init$ à $Wait$

Les tâches sont initialisées par la fonction *init* dont le comportement est le suivant :

$$\begin{aligned} init : S_T \quad = \quad & (Init, [l_s, u_s[, [l_f, u_f[, C, \\ & \alpha_{OS} = \alpha_{start} = \alpha_{finish} = \alpha_{fail} = \beta_S = faux, \\ & \overrightarrow{\Omega_S} = \overrightarrow{\Omega_F} = \overrightarrow{faux},\ \overrightarrow{\Theta_S} = \overrightarrow{\Theta_F} = \overrightarrow{\infty^+}, \sigma = 0) \end{aligned} \tag{6.1}$$

Comme l'indique l'équation 6.2, si $|J_s| = 0$ et $|J_f| = 0$ c'est-à-dire, s'il n'existe aucun modèle de tâche dont l'exécution doit précéder celle de M_T, alors ce dernier passe de son état passif ($Init$) à un état actif ($Wait$). La fonction de transition interne, définie, ci-dessous, formalise la transition du statut $Init$ à $Wait$ lorsque la tâche est dans un état passif.

$$\begin{aligned} \delta_{int}(Init, J_s = 0 \wedge J_f = 0) : S_T \quad = \quad & (Wait, \\ & \sigma = \begin{cases} l_s - t & si\ t \leq l_s \\ u_s - t & si\ l_s < t < u_s \\ 0 & sinon \end{cases} \\ &) \end{aligned} \tag{6.2}$$

Dans le cas contraire (équation 6.3), le modèle de tâche reste dans un état passif tant que l'un au moins de ses prédécesseurs n'a pas émis un message. Ainsi, le modèle s'active lorsque le premier événement est perçu sur l'un des ports I_{s_j} ou I_{f_k}, le statut de la tâche, passe de *Init* à *Wait*.

$$\begin{aligned} \delta_{ext}((Init), e, (I_{s_j}, I_{f_k}))) : S_T \quad = \quad & (Wait, \alpha_{status} = \alpha_{start} = \alpha_{finish} = \alpha_{fail} = faux, \\ & \sigma = \begin{cases} l_s - t & si\ t \leq l_s \\ u_s - t & si\ l_s < t < u_s \\ 0 & sinon \end{cases} \\ &) \end{aligned} \tag{6.3}$$

6.3.4 Transition du statut $Wait$ vers $Started$ ou $Failed$

Les préconditions liées à la dynamique des processus biophysiques sont observées via le port Pcond. Les équations suivantes définissent les changements d'états en fonction des événements externes reçus du système de croyance $\mathcal{B}$ via le port d'entrée Pcond. Ils sont exploités pour la mise à jour des préconditions de démarrage $C_{ws}(t)$ de la tâche. Ainsi,

$$\begin{aligned}\delta_{ext}((Init \vee Wait), e, \mathsf{Pcond}) : S_T &= (C_{ws} = update(C_{ws}, e, \mathsf{Pcond}), \\ &\quad \sigma = \sigma - e)\end{aligned} \tag{6.4}$$

Lorsqu'un événement est reçu via un port d'entrée I_{s_j} alors que le statut du modèle de tâche est $Wait$, le booléen ω_{sj} passe à $vrai$. La garde temporelle θ_{sj} est mise à jour avec la valeur de garde v contenue dans le message. L'ancienne valeur de θ_{sj} est modifiée à condition que celle-ci soit supérieure à θ_{sj}. β_S est mis à jour comme l'indique l'équation 6.5.

$$\begin{aligned}\delta_{ext}((Wait), e, (I_{s_j})) &= (\omega_{sj} = vrai, \beta_S = \bigwedge_{j=1}^{J_s} \omega_{sj}, \theta_{sj} = min(\theta_{sj}, v), \\ &\quad \sigma = min(min(\overrightarrow{\Theta_S}), min(\overrightarrow{\Theta_F}), (\sigma - e))\end{aligned} \tag{6.5}$$

De la même manière, si un événement est reçu sur un port d'entrée I_{f_k} alors que le statut du modèle de tâche est à $Wait$, la mise à jour de l'état du modèle est réalisée suivant l'équation 6.6.

$$\begin{aligned}\delta_{ext}((Wait), e, (I_{f_k})) &= (\omega_{fk} = vrai, \beta_F = \bigwedge_{k=1}^{J_f} \omega_{fk}, , \theta_{fk} = min(\theta_{fk}, v), \\ &\quad \sigma = min(min(\overrightarrow{\Theta_S}), min(\overrightarrow{\Theta_F}), (\sigma - e))\end{aligned} \tag{6.6}$$

La fonction de transition interne (équation 6.7), définie ci-dessous, formalise la transition entre les statuts $Wait$ et $Started$ quand la tâche est active c'est-à-dire $\alpha_{OS} = vrai$.

$$\begin{aligned}&\delta_{int}(Wait, l_s \leq t < u_s, \\ &\qquad (t < u_f), \\ &\wedge(t < min(min(\overrightarrow{\Theta_S}), min(\overrightarrow{\Theta_F})), \\ &\qquad (C_{ws} \wedge \beta_S)) : S_T &= (Started, \alpha_{OS} = \alpha_{start} = vrai, \\ &&\quad \sigma = 0)\end{aligned} \tag{6.7}$$

Des sorties sont ensuite réalisées sur les ports OS et start à la date courante (équation 6.8 et 6.9). Le format du message $MsgToOS(S_T)$ envoyé sur OS est celui décrit dans la section 3.1.5.a et Figure 3.2.

Il permet de déclencher la réalisation de la tâche dans le système opérant. Le message envoyé sur le port start contient l'identifiant id du modèle de la tâche et un booléen ayant la valeur $vrai$.

$$\lambda((Started), \alpha_{OS} = true) : \mathsf{OS} = MsgToOS(S_T) \tag{6.8}$$

$$\lambda((Started), \alpha_{start} = true) : \mathsf{start} = MsgFail(S_T) \tag{6.9}$$

Une fois la sortie effectuée, la transition interne ci-dessous est réalisée.

$$\delta_{int}(Started) : S_T = (\alpha_{start} = faux, \sigma = min(min(\overrightarrow{\Theta_S}), min(\overrightarrow{\Theta_F}), (u_f - t))) \quad (6.10)$$

La tâche échoue sur une transition interne si les contraintes temporelles liées au démarrage ou à la fermeture ne sont plus respectées à la date courante. Dans ce cas, le modèle de tâche prend le statut $Failed$ (équations 6.11, 6.12 6.13, 6.14). Une sortie est réalisée instantanément sur le port de sortie fail (équation 6.15). Le message envoyé est constitué de l'identifiant id du modèle de tâche et un booléen ayant la valeur $vrai$.

$$\delta_{int}(Wait, t >= u_s) : S_T = (Failed, \alpha_{OS} = \alpha_{finish} = faux, \alpha_{fail} = vrai, \sigma = 0) \quad (6.11)$$

$$\delta_{int}((Wait), t >= u_f) : S_T = (Failed, \alpha_{OS} = \alpha_{finish} = faux, \alpha_{fail} = vrai, \sigma = 0) \quad (6.12)$$

$$\delta_{int}((Wait), t >= min(\overrightarrow{\Theta_S})) : S_T = (Failed, \alpha_{OS} = \alpha_{finish} = faux, \alpha_{fail} = vrai, \sigma = 0) \quad (6.13)$$

$$\delta_{int}((Wait), t >= min(\overrightarrow{\Theta_F})) : S_T = (Failed, \alpha_{OS} = \alpha_{finish} = faux, \alpha_{fail} = vrai, \sigma = 0) \quad (6.14)$$

$$\lambda((Wait), \alpha_{fail} = true) : \mathsf{fail} = MsgFail(S_T) \quad (6.15)$$

Une fois la sortie réalisée, la tâche retourne définitivement dans un état passif (équation 6.16).

$$\delta_{int}(Failed) : S_T = (\alpha_{fail} = faux, \sigma = \infty) \quad (6.16)$$

De la même manière, une tâche peut échouer sur une transition externe si un événement est reçu via le port ACK alors que le statut est à $Wait$. Le modèle passe du statut $Wait$ à $Failed$ (équation 6.17). Une sortie est réalisée sur le port fail puis le modèle retourne dans un état passif exactement comme dans le cas de l'échec sur une transition interne.

$$\delta_{ext}((Wait), e, \mathsf{ACK}) = (Failed, \alpha_{fail} = true,) \alpha_{OS} = \alpha_{finish} = faux, \sigma = 0) \quad (6.17)$$

6.3.4.a Transition du statut *Started* vers *Finished* ou *Failed*

La transition de $Started$ à $Finished$ survient quand un événement de notification de fin est reçu via le port ACK. La transition externe associée est définie par l'équation 6.18.

$$\delta_{ext}((Started), e, (\mathsf{ACK} = vrai)) = (Finished, \alpha_{finish} = true, \sigma = 0) \quad (6.18)$$

Une sortie est réalisée instantanément sur le port de sortie finish (équation 6.19). Comme dans le cas précédent, le message envoyé est constitué de l'identifiant id du modèle de tâche et un booléen ayant la valeur $vrai$.

$$\lambda((Started), \alpha_{finish} = true) : \mathsf{finish} = MsgFinish(S_T) \quad (6.19)$$

Une fois la sortie réalisée, le modèle retourne définitivement dans un état passif (équation 6.20).

$$\delta_{int}(Finished, \alpha_{finish} = true) : S_T \quad = \quad (\alpha_{OS} = \alpha_{finish} = faux, \sigma = \infty) \tag{6.20}$$

La transition de *Started* vers $Failed$ est réalisée sous deux conditions. Premièrement, lorsque l'événement externe de notification reçu via le port ACK indique un échec de la tâche (équation 6.21).

$$\begin{aligned} \delta_{ext}((Started), e, (\mathsf{ACK} = faux)) \quad &= \quad (Failed, \alpha_{fail} = true, \\ &\qquad \alpha_{OS} = \alpha_{finish} = faux, \sigma = 0) \end{aligned} \tag{6.21}$$

La deuxième situation d'échec intervient dans une transition interne. Elle est établie si les contraintes temporelles liées à la fermeture de la tâche ne soient plus respectées à la date courante (équation 6.22 et 6.23).

$$\begin{aligned} \delta_{int}((Started), t >= u_f) : S_T \quad &= \quad (Failed, \\ &\qquad \alpha_{OS} = \alpha_{finish} = faux, \\ &\qquad \alpha_{fail} = vrai, \sigma = 0) \end{aligned} \tag{6.22}$$

$$\begin{aligned} \delta_{int}((Started), t >= min(\overrightarrow{\Theta_F})) : S_T \quad &= \quad (Failed, \\ &\qquad \alpha_{OS} = \alpha_{finish} = faux, \\ &\qquad \alpha_{fail} = vrai, \sigma = 0) \end{aligned} \tag{6.23}$$

Dans les deux cas, une sortie est réalisée sur le port fail et le modèle retourne dans un état passif. Le mécanisme appliqué est équivalent à celui de la transition de $Wait$ vers $Failed$.

Dans cette section, nous avons formalisé, en PDEVS, les modèles de tâche. La dynamique de ces modèles permet de prendre en compte d'une part, des contraintes temporelles absolues liées au démarrage et à la fermeture des tâches et d'autre part, l'interface et l'interprétation des événements reçus permet de prendre en compte les contraintes liées aux relations temporelles entre différentes tâches.

6.4 Formalisation PDEVS des éléments de l'interpréteur du plan courant π

Dans cette section 6.4, nous introduisons les mécanismes qui nous permettent de modéliser et de simuler les relations temporelles de précédences et de synchronisation des tâches. Pour ce faire, nous décrivons dans un premier temps les types de relations temporelles entre les tâches. Cela nous permet de différentier les interprétations possibles des relations temporelles durant la simulation. Nous nous basons ensuite sur les différentes interprétations d'une relation temporelle pour proposer des modèles génériques capables de simuler de manière distribuée un plan temporel.

6.4.1 Types de relation entre les tâches du plan courant π

Le plan courant à exécuter est celui issu de la phase de planification opérationnelle. Comme nous l'avons présenté de la section 5.3.2.e le plan est représenté par un graphe orienté. Les nœuds du graphe sont des événements de début et de fin des tâches. Les arcs définissent les relations temporelles entre les tâches. Considérons

deux tâches a et b. Soient $s_a \in [l_{s_a}, u_{s_a}[$ et $f_a \in [l_{f_a}, u_{f_a}[$ deux instants qui définissent respectivement les instants de démarrage et de fin de la tâche a. $[l_{s_a}, u_{s_a}[$ et $[l_{f_a}, u_{f_a}[$ définissent les intervalles d'ouverture et de fermeture de la tâche a (cf. Définition 6.4). De la même manière, $s_b \in [l_{s_b}, u_{s_b}[$ et $f_b \in [l_{f_b}, u_{f_b}[$ définissent le démarrage et la fin de la tâche b.

À l'instar des treize relations primitives de Allen (1984) [7], les relations temporelles entre les tâches a et b peuvent être exprimées (cf. le Tableau 5.1 dans la section 5.2.2.b) par des contraintes temporelles *relatives*, portant sur les intervalles $[s_a, f_a[$, $[s_b, f_b[$

Dans le cas de l'architecture SAFIHR, nous partons du principe que les tâches a et b sont des systèmes dynamiques autonomes basés sur des événements discrets. Pour établir les relations temporelles entre a et b, nous ne pouvons observer que les événements liés au début et à la fin des tâches s_a, f_a, s_b et f_b. C'est justement en cela que les travaux de Vilain et al. (1986) (cf. section 5.2.2) sur la traduction des relations primitives de Allen en algèbre des instants apparaissent plus pertinentes à utiliser.

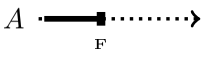
Afin de prendre en compte les décalages temporels *timelag*, nous définissons les variables XY_{ab}^{min} et XY_{ab}^{max} comme étant respectivement le décalage temporel minimal et maximal entre deux instants X et Y où X, Y sont des événements de début ou de fin des deux tâches a et b.

Dans le tableau 6.1, nous reprenons les relations primitives de Allen (ligne 1) en montrant leurs traductions en algèbre des instants (ligne 3) de Marc Vilain. Les lignes 5 et 6 représentent respectivement les décalages temporels et les contraintes qu'elles rajoutent. Enfin, la ligne 7 est une réécriture des relations temporelles qui exploite essentiellement les événements de début (S) et de fin (F). Sur la base de ces deux événements, nous pouvons identifier quatre types de relations temporelles atomiques qui sont : *FinishToStart* (**FS**), *StartToFinish* (**SF**), *StartToStart* (**SS**) et *FinishToFinish* **FF**. Ces relations atomiques décrivent les différentes combinaisons possibles entre les événements de début et de fin de deux tâches.

Afin des raisonner par simulation sur des relations temporelles **FS**, **SF**, **SS** et **FF** nous proposons dans la section 6.4.2 les différentes interprétations possibles des relations basées sur les deux événements S et F.

7. *a before b, a meet b, a overlap b, a start b, a finish b, a during b, a equal b, b after a, b is-met-by a, b is-overlapped-by a, b is-started-by a, b is-finished-by a, b includes a*. Les sept premières relations primitives (*before*, *meet*, *overlap*, *start*, *finish*, *during* et *equal*) permettent de retrouver par symétrie les six dernières.

TABLE 6.1 – *Traduction des relations temporelles en algèbre des instants avec prise en compte des* timelag

	Relations temporelles simples				*Relations temporelles composées*		
RELATIONS TEMPORELLES	a **before** b	a **meet** b	a **start** b	a **finish** b	a **equal** b	a **overlap** b	a **during** b
SCHÉMA							
TRADUCTION DE LA CONTRAINTE	$(s_a < f_a) \wedge (f_a < s_b) \wedge (s_b < f_b)$	$(s_a < f_a) \wedge (f_a = s_b) \wedge (s_b < f_b)$	$(s_a < f_a) \wedge (s_a = s_b) \wedge (s_b < f_b)$	$(s_a < f_a) \wedge (f_a = f_b) \wedge (s_b < f_b)$	$(s_a = s_b) \wedge (s_b < f_a) \wedge (f_a = f_b)$	$(s_a < s_b) \wedge (s_b < f_a) \wedge (f_a < f_b)$	$(s_a < f_a) \wedge (s_a > s_b) \wedge (f_a < f_b)$
TIMELAG	$[FS_{ab}^{min}, FS_{ab}^{max}[$	$[FS_{ab}^{min}, FS_{ab}^{max}[$	$[SS_{ab}^{min}, SS_{ab}^{max}[$	$[FF_{ab}^{min}, FF_{ab}^{max}[$	$[SS_{ab}^{min}, SS_{ab}^{max}[\wedge [FF_{ab}^{min}, FF_{ab}^{max}[$	$[SS_{ab}^{min}, SS_{ab}^{max}[\wedge [FF_{ab}^{min}, FF_{ab}^{max}[$	$[SS_{ba}^{min}, SS_{ba}^{max}[\wedge [FF_{ba}^{min}, FF_{ba}^{max}[$
	$f_a + FS_{ab}^{min} \leq s_b \wedge f_a + FS_{ab}^{max} > s_b$	$f_a + FS_{ab}^{min} = s_b \wedge f_a + FS_{ab}^{max} = s_b$	$s_a + SS_{ab}^{min} = s_b \wedge s_a + SS_{ab}^{max} = s_b$	$f_a + FF_{ab}^{min} = f_b \wedge f_a + FF_{ab}^{max} = f_b$	$s_a + SS_{ab}^{min} = s_b \wedge s_a + SS_{ab}^{max} = s_b \wedge f_a + FF_{ab}^{min} = f_b \wedge f_a + FF_{ab}^{max} = f_b$	$s_a + SS_{ab}^{min} \leq s_b \wedge s_a + SS_{ab}^{max} > s_b \wedge f_a + FF_{ab}^{min} \leq f_b \wedge f_a + FF_{ab}^{max} > f_b$	$s_b + SS_{ba}^{min} \leq s_a \wedge s_b + SS_{ba}^{max} > s_a \wedge f_a + FF_{ab}^{min} \leq f_b \wedge f_a + FF_{ab}^{max} > f_b$
RÉÉCRITURE DES RELATIONS TEMPORELLES	$(a, \mathbf{FS}, b) \wedge FS_{ab}^{min} > 0 \wedge FS_{ab}^{max} > 0$	$(a, \mathbf{FS}, b) \wedge FS_{ab}^{min} = 0 \wedge FS_{ab}^{max} = 0$	$(a, \mathbf{SS}, b) \wedge SS_{ab}^{min} = 0 \wedge SS_{ab}^{max} = 0$	$(a, \mathbf{FF}, b) \wedge FF_{ab}^{min} = 0 \wedge FF_{ab}^{max} = 0$	$(a, \mathbf{SS}, b) \wedge SS_{ab}^{min} = 0 \wedge SS_{ab}^{max} = 0 \wedge (a, \mathbf{FF}, b) \wedge FF_{ab}^{min} = 0 \wedge FF_{ab}^{max} = 0$	$(a, \mathbf{SS}, b) \wedge SS_{ab}^{min} > 0 \wedge SS_{ab}^{max} > 0 \wedge (a, \mathbf{FF}, b) \wedge FF_{ab}^{min} > 0 \wedge FF_{ab}^{max} > 0$	$(b, \mathbf{SS}, a) \wedge SS_{ba}^{min} > 0 \wedge SS_{ba}^{max} > 0 \wedge (a, \mathbf{FF}, b) \wedge FF_{ab}^{min} > 0 \wedge FF_{ab}^{max} > 0$

6.4.2 Différentes interprétations des relations basées sur les événements S et F

Les relations temporelles basées sur les événements de début et de fin peuvent être prises en compte soit comme une *précédence* soit comme une *synchronisation* sur ces deux événements **S** et **F**. La raison principale de cette double interprétation est qu'il s'agit ici de tester en cours de simulation la validité des contraintes temporelles et ceci de manière distribuée.

6.4.2.a Contraintes de précédence

La relation temporelle entre les événements **S** et **F** peut être interprétée comme une *précédence* à condition qu'elle engendre un lien de causalité entre l'apparition des événements de début ou de fin des tâches a et b. Par exemple, le début du semis de blé a lieu après le labour.

On notera cette dimension causale de la relation temporelle par le symbole $\prec$. Ainsi,

- $(a, \prec \mathbf{FS}, b)$ *sera interprété par :* b commence après la fin de a. On définit pour ceci, FS_{ab}^{min} et FS_{ab}^{max} respectivement le décalage temporel minimal et maximal entre la fin de a et le début de b.
- $(a, \prec \mathbf{SF}, b)$ *sera interprété par :* b fini après le début de a. On définit pour ceci, SF_{ab}^{min} et SB_{ab}^{max} respectivement le décalage temporel minimal et maximal entre le début de a et la fin de b.
- $(a, \prec \mathbf{SS}, b)$ *sera interprété par :* b commence après le début de a. On définit pour ceci, SS_{ab}^{min} et SS_{ab}^{max} respectivement le décalage temporel minimal et maximal entre le début de a et celui de b.
- $(a, \prec \mathbf{FF}, b)$ *sera interprété par :* b fini après la fin de a. On définit pour ceci, FF_{ab}^{min} et FF_{ab}^{max} respectivement le décalage temporel minimal et maximal entre la fin de a et celle de b.

Pour chacune des relations temporelles ci-dessus, la contrainte sera considérée comme non satisfaite si la condition décrite dans la colonne CONTRAINTES DE PRÉCÉDENCE (colonne 2) du tableau 6.2 n'est pas respectée.

6.4.2.b Contraintes de synchronisation

Contrairement au cas précédent, la relation temporelle entre les événements **S** et **F** ne constituent pas des contraintes de précédence. Ces relations doivent être interprétées comme des synchronisations entre à **S** et **F**. Autrement dit, il s'agit d'une interprétation parallèle des événements de début et de fin dans laquelle l'ordre n'a pas d'importance. On notera cette dimension parallèle de la relation temporelle par le symbole $\prec \,.\, \succ$. Ainsi,

- $(a, \prec \mathbf{FS} \succ, b)$ *sera interprété par :* la fin de a et le début de b doivent intervenir avec un décalage temporel compris entre $[FS_{ab}^{min}, FS_{ab}^{max}[$,
- $(a, \prec \mathbf{SF} \succ, b)$ *sera interprété par :* le début de a et la fin de b doivent intervenir avec un décalage temporel compris entre $[SF_{ab}^{min}, SF_{ab}^{max}[$,
- $(a, \prec \mathbf{SS} \succ, b)$ *sera interprété par :* le début de a et celui de b doivent intervenir avec un décalage temporel compris entre $[SS_{ab}^{min}, SS_{ab}^{max}[$,
- $(a, \prec \mathbf{FF} \succ, b)$ *sera interprété par :* la fin de a et celle de b doivent intervenir avec un décalage temporel compris entre $[FF_{ab}^{min}, FF_{ab}^{max}[$.

De la même manière que dans le cas précédent, la contrainte sera considérée comme non satisfaite si la condition décrite dans la CONTRAINTES DE SYNCHRONISATION (colonne 3) du tableau 6.2 n'est pas respectée.

TABLE 6.2 – *Contraintes de précédence et de synchronisation*

RELATIONS TEMPORELLES	CONTRAINTES DE PRÉCÉDENCE ($\prec$)	CONTRAINTES DE SYNCHRONISATION ($\prec$. $\succ$)
$(a, \mathbf{FS}, b)$ $[FS_{ab}^{min}, FS_{ab}^{max}[$	$(FS_{ab}^{min} + f_a \leq s_b < FS_{ab}^{max} + f_a)$	$(FS_{ab}^{min} + f_a \leq s_b < FS_{ab}^{max} + f_a) \vee$ $(FS_{ab}^{min} + s_b \leq f_a < FS_{ab}^{max} + s_b)$
$(a, \mathbf{SF}, b)$ $[SF_{ab}^{min}, SF_{ab}^{max}[$	$(SF_{ab}^{min} + s_a \leq f_b < SF_{ab}^{max} + s_a)$	$(SF_{ab}^{min} + s_a \leq f_b < SF_{ab}^{max} + s_a) \vee$ $(SF_{ab}^{min} + f_b \leq s_a < SF_{ab}^{max} + f_b)$
$(a, \mathbf{SS}, b)$ $[SS_{ab}^{min}, SS_{ab}^{max}[$	$(SS_{ab}^{min} + s_a \leq s_b < SS_{ab}^{max} + s_a)$	$(SS_{ab}^{min} + s_a \leq s_b < SS_{ab}^{max} + s_a) \vee$ $(SS_{ab}^{min} + s_b \leq s_a < SS_{ab}^{max} + s_b)$
$(a, \mathbf{FF}, b)$ $[FF_{ab}^{min}, FF_{ab}^{max}[$	$(FF_{ab}^{min} + f_a \leq f_b < FF_{ab}^{max} + f_a)$	$(FF_{ab}^{min} + f_a \leq f_b < FF_{ab}^{max} + f_a)$ $\vee$ $(FF_{ab}^{min} + f_b \leq f_a < FF_{ab}^{max} + f_b)$

6.4.3 Types de modèles et interprétation en ligne des relations temporelles

Pour exécuter le plan courant π du réseau DSDEN en maintenant la consistance des relations temporelles entre tâches, nous proposons de modéliser les relations **FS**, **SF**, **SS** et **FF** par des modèles atomiques PDEVS. La simulation de ces relations révèle des caractéristiques particulières selon la nature de l'interprétation ($\prec$ ou $\prec$. $\succ$).

6.4.3.a Interprétation basée sur les précédences

Soit $XY_{ab}^{min} + X_b \leq Y_b < XY_{ab}^{max} + X_b$ (avec X, Y des événements de début ou de fin) la forme générale des contraintes de précédence décrites dans la colonne CONTRAINTES DE PRÉCÉDENCE du tableau 6.2. Si la relation $(a, \prec \mathbf{XY}, b)$ est simulée par un modèle autonome nommé $\prec \mathbf{XY}$, la consistance de la contrainte ne peut être que partiellement simulée par le modèle $\prec \mathbf{XY}$. En effet, le début ou la fin de b est précédé du début ou de la fin de a. À la date t, si l'événement X_b apparaît, seule la première composante de la contrainte $(XY_{ab}^{min} + X_b \leq Y_b)$ peut être validée. X_b précédant Y_b, cette affirmation se justifie. La deuxième composante $(Y_b < XY_{ab}^{max} + X_b)$ de la contrainte serait dans ces conditions validée à posteriori par le modèle de la tâche b. Dans notre formalisation des modèles de tâches, les équations 6.5, 6.6, 6.7, 6.13, 6.14 et 6.23 permettent de valider cette deuxième composante de la contrainte. Ainsi, le modèle générique des relations temporelles de types $\prec \mathbf{XY}$ que nous proposons, valide essentiellement la contrainte $(XY_{ab}^{min} + X_b \leq t)$. La colonne INTERPRÉTATIONS BASÉES SUR LES PRÉCÉDENCES du tableau 6.3 décrit, pour chaque type de relations, la contrainte à satisfaire.

6.4.3.b Interprétation par synchronisation

Contrairement au cas précédent, les contraintes de synchronisation (colonne CONTRAINTES DE SYNCHRONISATION du tableau 6.2) sont décrites par une conjonction de deux contraintes de précédence. L'ordre d'exécution de a et b est dans ce cas réversible. La consistance globale des relations temporelles de types $\prec \mathbf{XY} \succ$ peut-être simulée par un modèle autonome nommé $\prec \mathbf{XY} \succ$. Le modèle générique des

TABLE 6.3 – *Interprétations séquentielles et parallèles des relations temporelles entre les événements* **S** *et* **F**

RELATIONS TEMPORELLES	INTERPRÉTATIONS BASÉES SUR LES PRÉCÉDENCES ($\prec$)	INTERPRÉTATIONS PAR SYNCHRONISATION ($\prec . \succ$)
$(a, \mathbf{FS}, b)$ $[FS_{ab}^{min}, FS_{ab}^{max}[$	$FS_{ab}^{min} + f_a \leq t$	$(min(f_a, s_b) + FS_{ab}^{min} \leq t)$ $(t < min(f_a, s_b) + FS_{ab}^{max})$ $\wedge$
$(a, \mathbf{SF}, b)$ $[SF_{ab}^{min}, SF_{ab}^{max}[$	$SF_{ab}^{min} + s_a \leq t$	$(min(s_a, f_b) + SF_{ab}^{min} \leq t)$ $(t < min(s_a, f_b) + SF_{ab}^{max})$ $\wedge$
$(a, \mathbf{SS}, b)$ $[SS_{ab}^{min}, SS_{ab}^{max}[$	$SS_{ab}^{min} + s_a \leq t$	$(min(s_a, s_b) + SS_{ab}^{min} \leq t)$ $(t < min(s_a, s_b) + SS_{ab}^{max})$ $\wedge$
$(a, \mathbf{FF}, b)$ $[FF_{ab}^{min}, FF_{ab}^{max}[$	$FF_{ab}^{min} + f_a \leq t$	$(min(f_a, f_b) + FF_{ab}^{min} \leq t)$ $(t < min(f_a, f_b) + FF_{ab}^{max})$ $\wedge$

relations temporelles de types $\prec \mathbf{XY} \succ$ que nous proposons assure que le décalage temporel entre les événements reçus est compris dans l'intervalle $[XY_{ab}^{min}, XY_{ab}^{max}[$. La colonne INTERPRÉTATION PAR SYNCHRONISATION du tableau 6.3 présente pour chacune des relations temporelles la contrainte à satisfaire pour conserver une cohérence locale sur le plan courant π en cours d'exécution.

6.4.4 Formalisation DEVS des modèles de contraintes de précédence

Les relations temporelles entre tâches sont modélisées par des modèles PDEVS. L'interface des modèles M_R de relations temporelles est conçue de manière à observer les événements de début et de fin issus des modèles de tâches M_T (cf. section 6.3). Ainsi,

$$\begin{aligned} M_R &= \langle I_R, O_R, S_T, \tau, \delta_{int}, \delta_{ext}, \delta_{con}, \lambda \rangle \\ I_T &= \{I_X = \{I_{x_j}\}, \text{avec} \quad j \in \mathbb{N}\} \\ O_T &= \{\mathsf{valid}\} \end{aligned}$$

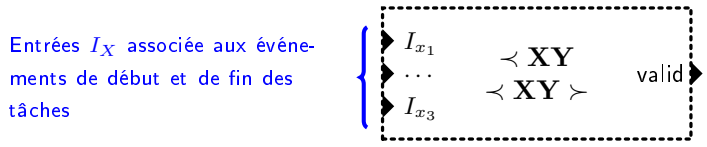


FIGURE 6.5 – *Interface générique des modèles de relation temporelle*

L'ensemble d'entrées I_{x_j} est destiné aux événements de début et de fin des tâches. Le port de sortie valid sert à la notification de la validité des contraintes décrites dans la tableau 6.3.

L'état S_R des tâches est défini par :

$$\begin{aligned} S_T &= \langle Status, XY_{ab}^{min}, XY_{ab}^{max}, \alpha_{valid}, \overrightarrow{\Omega}, \beta, \sigma \rangle \\ Status &= \{Init, Idle, End\} \end{aligned}$$

Les variables XY_{ab}^{min} et XY_{ab}^{max} sont celles décrites dans la section précédente. Elles se réfèrent aux décalages temporels minimum et maximum. Le *Status* est décrit par un automate à trois états dans lequel les états initial et final sont respectivement *Init* et *End* (cf. Figure 6.6).

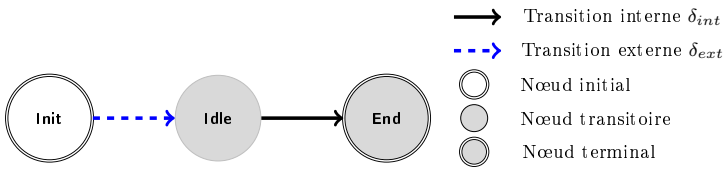


FIGURE 6.6 – *Statut des modèles de relations temporelles.*

Comme dans le cas des modèles de tâche, $\overrightarrow{\Omega}$ est un vecteur de booléens qui permet de valider les relations temporelles en entrée du modèle. $\overrightarrow{\Omega} = (\omega_{x1}, \cdots, \omega_{xn})_J$ où J est le nombre d'entrées I_{x_j}. De la même manière que pour les tâches, ω_{xj} est affecté à *vrai* si un événement externe est reçu sur le port d'entrée I_{x_j}. La variable β est un booléen dont la valeur passe à *vrai* si au moins un événement est reçu sur chacun des ports d'entrée I_{x_j}. $\sigma \in \mathbb{R}_0^+$ permet de gérer le temps passé dans un état par le modèle. En conséquence, la fonction d'avancement du temps s'écrit : $\tau(S_R) = \sigma$.

Le modèle de relation est initialisé avec les valeurs de décalage temporel (équation 6.24).

$$init : S_R \quad = \quad (Init, XY_{ab}^{min}, XY_{ab}^{max}, \alpha_{valid} = faux, \overrightarrow{\Omega} = \overrightarrow{faux}, \sigma = 0) \tag{6.24}$$

Le modèle reste dans l'état $Init$ tant que l'une des tâches dont il observe les événements n'émet pas de message. Il passe dans l'état $Idle$ dès la réception du premier passage sur l'un des ports I_{x_j} (équation 6.25).

$$\delta_{ext}((Init), e, I_{x_j})) : S_R \quad = \quad (Idle, \sigma = \infty) \tag{6.25}$$

Lorsqu'un événement est reçu sur le port d'entrée I_{x_j} alors que le statut du modèle est $Idle$, le booléen ω_{xj} passe à *vrai*. β et σ sont ensuite mis à jour comme l'indique l'équation 6.26. L'expression $\sigma = XY_{ab}^{min}$ est celle qui permet d'interpréter et de valider une relation temporelle basée sur la précédence (type $\prec$ **XY**). La traduction littérale de la mise à jour de σ est la suivante : *si à la date t le modèle a reçu l'ensemble des événements issus des précédents, alors la première composante* $XY_{ab}^{min} + X_b \leq Y_b$ *de la contrainte de précédence (cf tableau 6.3) sera valide dans* XY_{ab}^{min} *unité de temps. Dans le cas contraire,* σ *est mis à jour en fonction de l'avancement du temps.*

$$\begin{aligned} \delta_{ext}((Idle), e, (I_{x_j})) \quad = \quad & (\omega_{xj} = vrai, \beta = \bigwedge_{j=1}^{J} \omega_{xj}, \\ & \sigma = \begin{cases} XY_{ab}^{min} & si\ (\beta = vrai) \wedge (\sigma = \infty) \\ \sigma - e & sinon \end{cases} \\ &) \end{aligned} \tag{6.26}$$

Si dans l'état $Idle$ une transition interne est réalisée avec $\beta = vrai$ on en déduit que la contrainte de précédence est valide. Le statut du modèle passe à End et son état est mis à jour conformément à l'équation 6.27.

$$\delta_{int}((Idle), \beta = vrai) : S_T \quad = \quad (End, \alpha_{valid} = vrai, \sigma = 0) \tag{6.27}$$

$$\lambda((End), \alpha_{valid} = true) : \mathsf{valid} \quad = \quad Msg(vrai, (XY_{ab}^{max} - XY_{ab}^{min})) \tag{6.28}$$

$$\delta_{int}(End) : S_R \quad = \quad (\alpha_{valid} = faux, \sigma = \infty) \tag{6.29}$$

La fonction de transition interne affecte la valeur 0 à σ. Une sortie est alors réalisée instantanément sur le port valid (équation 6.28). Le message envoyé sur valid contient l'unité de temps relatif pendant laquelle la deuxième composante de la contrainte de précédence restera valide. Après la sortie, une nouvelle transition interne est réalisée pour désactiver le modèle.

6.4.5 Formalisation DEVS des modèles de contraintes de synchronisation

Les relations temporelles définissant les contraintes de synchronisation sont modélisées suivant les mêmes principes que celles qui définissent les contraintes de précédence. Toutefois, trois différences existent. La première est relative à la mise à jour de la variable σ dans la fonction de transition externe c'est-à-dire, l'équation 6.30. En effet dans le cas de l'interprétation par synchronisation (relation de types (type $\prec$ **XY** $\succ$), $\sigma = XY_{ab}^{max} - XY_{ab}^{min}$), la traduction littérale de la mise à jour de σ est celle-ci : *dès que le premier événement apparaît à la date t sur l'un des ports d'entrées, le modèle se donne au maximum* $XY_{ab}^{max} - XY_{ab}^{min}$ *unités de temps pour observer l'ensemble des autres événements auxquels il est soumit. À l'issu de ce délai, le contrainte de synchronisation sera considérée comme non satisfaite.* Le comportement de la transition externe est celui définit par l'équation 6.30.

$$\begin{aligned} \delta_{ext}((Idle), e, (I_{x_j})) \quad = \quad & (\omega_{xj} = vrai, \beta = \bigwedge_{j=1}^{J} \omega_{xj}, \\ & \sigma = \begin{cases} 0 & si\ \beta = vrai \\ XY_{ab}^{max} - XY_{ab}^{min} & si\ \sigma = \infty \\ \sigma - e & sinon \end{cases} \\ &) \end{aligned} \tag{6.30}$$

Ensuite, même si $\beta = faux$, le statut du modèle passe à End et l'état du modèle est mis à jour comme l'indique l'équation 6.31.

$$\delta_{int}((Idle)) : S_T \quad = \quad (End, \alpha_{valid} = vrai, \sigma = 0) \tag{6.31}$$

Enfin, deux types de sorties sont instantanément réalisables sur le port valid. Une sortie est réalisée, si au moins un événement est observé sur l'ensemble des ports d'entrées ($\beta = vrai$) avant la fin du délai. La notification de validité de la synchronisation est envoyée sur valid (équation 6.32). Dans le cas contraire, la notification indique une invalidité de la synchronisation (équation 6.33).

$$\lambda((End), (\alpha_{valid} = true) \wedge (\beta = vrai)) : \mathsf{valid} \quad = \quad Msg(vrai, \infty) \tag{6.32}$$

$$\lambda((End), (\alpha_{valid} = true) \wedge (\beta = faux)) : \mathsf{valid} \quad = \quad Msg(faux, \infty) \tag{6.33}$$

6.4.6 Quelques exemples de relations temporelles entre tâches

Nous représentons ci-après les motifs de base des différentes relations temporelles. Les modèles en gris sont ceux dont l'exécution valide les contraintes de précédence (cf. Figures 6.7, 6.8, 6.9 et 6.10) et de synchronisation (cf. Figures 6.11, 6.13, 6.12 et 6.14). Le plan courant représenté par le réseau DSDEN est construit en combinant ces différents motifs.

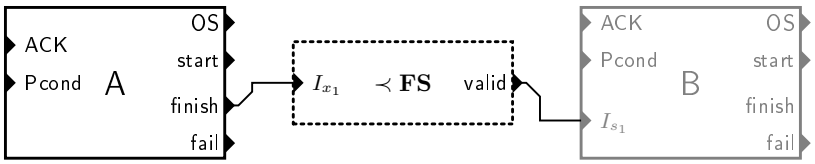


FIGURE 6.7 – $(a, \prec \mathbf{FS}, b)$. *L'événement de fin f_a précède le début s_b.*

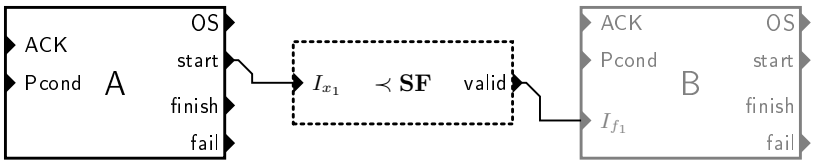


FIGURE 6.8 – $(a, \prec \mathbf{SF}, b)$. *L'événement de début s_a précède la fin f_b.*

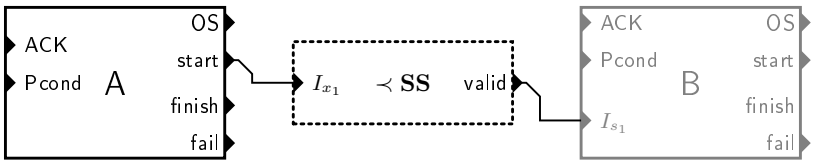


FIGURE 6.9 – $(a, \prec \mathbf{SS}, b)$. *L'événement de début s_a précède le début s_b.*

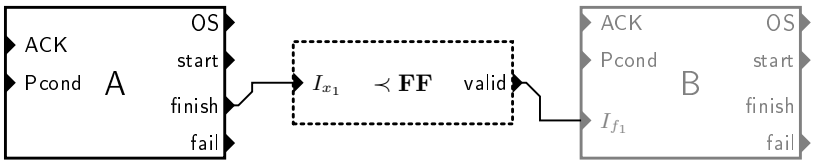


FIGURE 6.10 – $(a, \prec \mathbf{FF}, b)$. *L'événement de fin f_a précède la fin f_b.*

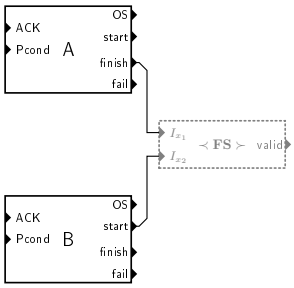


FIGURE 6.11 – $(a, \prec \mathbf{FS} \succ, b)$. *Synchronisation sur les événements f_a et s_b.*

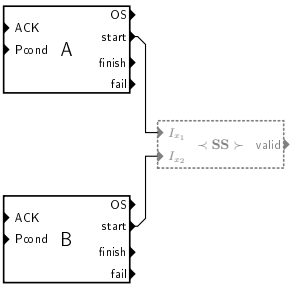


FIGURE 6.12 – $(a, \prec \mathbf{SS} \succ, b)$. *Synchronisation sur les événements s_a et s_b.*

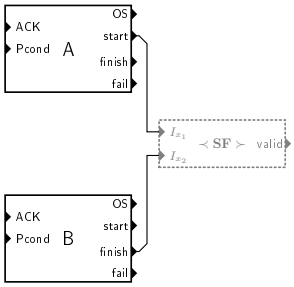


FIGURE 6.13 – $(a, \prec \mathbf{SF} \succ, b)$. *Synchronisation sur les événements s_a et f_b*

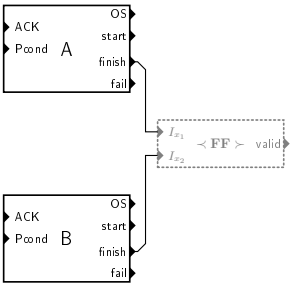


FIGURE 6.14 – $(a, \prec \mathbf{SS} \succ, b)$. *Synchronisation sur les événements f_a et f_b*

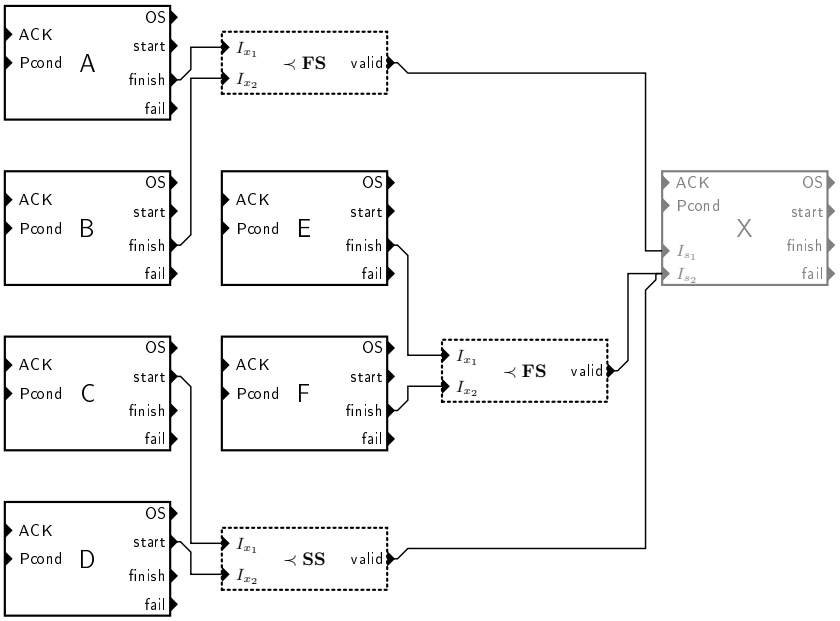


FIGURE 6.15 – $((a \wedge b) \prec \mathbf{FS}, X) \bigwedge((C \wedge D) \prec \mathbf{SS}, X) \vee (E \wedge F) \prec \mathbf{FS}, X))$

6.5 Fonctionnement du coordinateur : $\langle \chi, M_\chi \rangle$

Nous décrivons dans cette section, les mécanismes qui permettent de construire durant la simulation le réseau de modèles de tâches PDEVS. Ces mécanismes sont intégrés dans le coordinateur (cf. Figure 6.1). Ce dernier est modélisé par un exécutive DSDE nommé χ. En effet, les modèles exécutives sont les seuls capables de modifier en ligne (ajout ou suppression de connexions ou de modèles) la structure d'un réseau de modèles.

Formellement, le modèle M_χ du coordinateur est défini par le tuple :

$$M_\chi = \langle I_\chi, O_\chi, S_\chi, s^0_\chi, \tau_\chi, \beta, \Sigma^*, \delta_{int,\chi}, \delta_{ext,\chi}, \delta_{con,\chi}, \lambda_\chi \rangle$$

Dans la suite du manuscrit, nous décrivons chacun des éléments du modèle M_χ.

6.5.1 Interface du modèle du coordinateur M_χ

Conformément à la définition 6.3, M_χ est un modèle atomique DEVS parallèle. I_χ est l'union de I_{IS} et du port d'entrée associé aux modèles de tâches de planification (arc rouge de la Figure 6.1). La Figure 6.16 décrit de manière détaillée l'interface de M_χ. Les entrées *BS predicat*, *OS predicat fail*, *OS predicat finish* et *OS predicat start* sont respectivement destinées aux croyances relatives :

1. à l'état du système biophysique BS,
2. aux opérations en situation d'échec,
3. aux opérations qui viennent de commencer,
4. aux opérations qui viennent de se terminer.

L'entrée *plan* est destinée aux alertes envoyées par les tâches de planification pour la résolution de problèmes. *solved* est quant à lui réservé à la notification de la fin d'une résolution de problème et à l'envoi du plan correspondant. Les sorties *strategic, tactic* et *operational* servent à l'initialisation et au déclenchement de la résolution des problèmes de décisions stratégique, tactique et opérationnelle.

L'état partiel $s^\alpha_\chi \in S_\chi$ du coordinateur contient d'une part l'ensemble des informations relatives à la structure du réseau DSDEN. D'autre part, $s^\alpha_\chi \in S_\chi$ intègre l'ensemble des connaissances spécifiques de l'agriculteur.

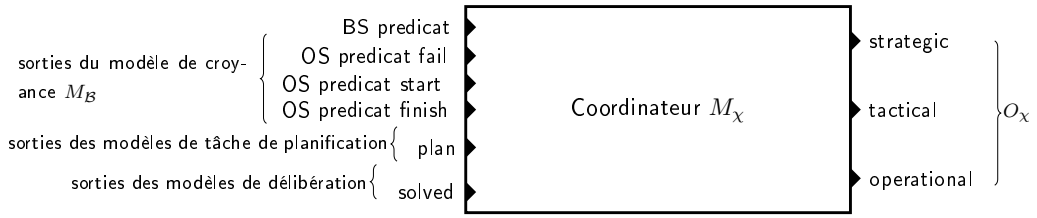


FIGURE 6.16 – *Interface du modèle M_χ du coordinateur.*

6.5.2 Description des états partiels s_χ^α

En début de simulation, le coordinateur est initialisé avec les données contenues dans la base de connaissances (cf. Figure 3.6). Un exemple de base de connaissances est celui de l'exploitation virtuelle décrite dans la section 1.6. Il s'agit notamment des connaissances portant sur la structure de l'environnement (ex : structure de l'exploitation agricole, l'historique des parcelles, les capacités de ressource, les cultures productibles etc.). À ces informations s'ajoutent les connaissances spécifiques à la construction et à l'exécution de plan courant. Il s'agit dans ce cas des tâches opérationnelles $\mathcal{A}$ (ex : Labour, Semis, etc.), des tâches de planification $\mathcal{P}$ (Stratégique, Tactique, Opérationnelle), des conditions d'activation des tâches contenue dans les itinéraires techniques. Enfin, le coordinateur dispose de l'ensemble des contraintes et des préférences qui définissent les objectifs de production de l'agent (cf. section 1.6.4).

Ces informations sont incluses dans l'état partiel initial s_χ^0 du coordinateur M_χ. La structure initiale Σ_0 du coordinateur est initialisée avec une tâche de planification. Dans notre cas, il s'agira d'un modèle de tâche de planification stratégique dont le but est de déclencher une résolution de problème de décision stratégique.

Formellement, les états partiels s_χ^α sont définis comme ci-dessous.

$$s_\chi^\alpha = (I^\alpha, O^\alpha, N^\alpha, \{M_n^\alpha\}, \{\zeta_n^\alpha\}, \{Z_{n,n'}^\alpha\}, \mathcal{B}_\chi, \Pi_\chi, \mathsf{pstate})$$

À l'exception des variables $\mathcal{B}_\chi$, Π_χ, pstate, la définition de l'état partiel $s_\chi^\alpha \in S_\chi$ correspond exactement à celle de la définition 6.3. En d'autres termes, le tuple $(I^\alpha, O^\alpha, N^\alpha, \{M_n^\alpha\}, \{\zeta_n^\alpha\}, \{Z_{n,n'}^\alpha\})$ décrit le comportement et la structure du réseau DSDEN c'est-à-dire le plan courant π de l'agent présenté dans la section 6.4.

Par ailleurs, la variable $\mathcal{B}_\chi$ modélise la base de connaissances décrite ci-dessus et les croyances envoyées via les ports issus du système de croyances $M_\mathcal{B}$.

Π_χ regroupe les trois niveaux de plans issus des décisions stratégique, tactique et opérationnelle. On notera Π_χ^{str}, Π_χ^{tac} et Π_χ^{ope} chacun de ces différents plans. Π_χ^{str} est un ensemble de graphes orientés et unidirectionnels dans lequel les nœuds sont des cultures. Chacun des graphes est associé à une parcelle donnée décrivant ainsi la succession de cultures adoptée. Π_χ^{tac} est un ensemble de graphes orientés unidirectionnels de tâches. Chaque graphe décrit l'itinéraire technique adopté pour un couple de culture/parcelle donnée. Les nœuds du graphe décrive les intervalles de début et de fin issus de la planification tactique. Les arcs représentent les timelag entre deux tâches reliées par des contraintes de précédences. Enfin, Π_χ^{ope}, à l'instar de Π_χ^{tac}, est un graphe orienté unidirectionnel de tâches. Contrairement à Π_χ^{tac} il n'existe qu'un seul graphe dans Π_χ^{ope} pour l'ensemble des parcelles de l'exploitation. Les plans Π_χ^{str} et Π_χ^{tac} sont des plans abstraits. Par opposition à ces derniers, Π_χ^{ope} est entièrement instancié c'est-à-dire quelque soit la tâche $a \in \Pi_\chi^{ope}$, les variables temporelles ($s_a \in [l_{s_a}, u_{s_a}[$ et $f_a \in [l_{f_a}, u_{f_a}[$) et de ressources sont fixées.

La variable pstate = {INIT, STR_NEED, STR_REQ, STR_DONE, TAC_NEED, TAC_REQ, TAC_DONE, OPE_NEED, OPE_REQ, OPE_DONE, IDLE} détermine les besoins de planification du coordinateur. Elle est affectée à :

- ▷ INIT *:* pour initialiser le mécanisme d'entrelacement des différentes phases de décision,
- ▷ STR_NEED *:* si l'état partiel s_χ du coordinateur impose une planification stratégique,
- ▷ STR_REQ *:* si une planification stratégique a été initiée par le coordinateur,
- ▷ STR_DONE *:* si la planification stratégique a été réalisée,
- ▷ TAC_NEED *:* si l'état partiel s_χ du coordinateur impose une planification tactique,
- ▷ TAC_REQ *:* si une planification tactique a été initiée par le coordinateur,
- ▷ TAC_DONE *:* si la planification tactique a été réalisée,
- ▷ OPE_NEED *:* si l'état partiel s_χ du coordinateur impose une planification opérationnelle,
- ▷ OPE_REQ *:* si une planification opérationnelle a été initiée par le coordinateur,
- ▷ OPE_DONE *:* si la planification tactique a été opérationnelle,
- ▷ IDLE *:* si le coordinateur attends des événements issus du réseau DSDEN.

6.5.3 Formalisation du coordinateur M_χ

6.5.3.a La fonction d'avancement du temps τ_χ

Comme l'indique l'équation 6.34, si l'état partiel s_χ du coordinateur impose une initialisation, celle-ci est réalisée instantanément. Dans l'état pstate = IDLE, le coordinateur reste passif en attendant un événement issu du réseau DSDEN (équation 6.35). Dès que l'état partiel du coordinateur impose une planification, celle-ci est initiée instantanément (équation 6.36). Par ailleurs, si une planification est initiée, le coordinateur retourne dans un état passif (équation 6.37). Enfin, dès que la planification est réalisée, une modification de la structure du réseau DSDEN doit être instantanément effectuée (équation 6.38).

$$\tau_\chi(s^\alpha_\chi, \mathsf{pstate} = \mathsf{INIT}) = 0 \quad (6.34)$$
$$\tau_\chi(s^\alpha_\chi, \mathsf{pstate} = \mathsf{IDLE}) = \infty \quad (6.35)$$
$$\tau_\chi(s^\alpha_\chi, \mathsf{pstate} \in \{\mathsf{STR\_NEED}, \mathsf{TAC\_NEED}, \mathsf{OPE\_NEED}\}) = 0 \quad (6.36)$$
$$\tau_\chi(s^\alpha_\chi, \mathsf{pstate} \in \{\mathsf{STR\_REQ}, \mathsf{TAC\_REQ}, \mathsf{OPE\_REQ}\}) = \infty \quad (6.37)$$
$$\tau_\chi(s^\alpha_\chi, \mathsf{pstate} \in \{\mathsf{STR\_DONE}, \mathsf{TAC\_DONE}, \mathsf{OPE\_DONE}\}) = 0 \quad (6.38)$$

6.5.3.b La fonction de sortie λ_χ

Les sorties du coordinateur permettent de paramétrer puis de déclencher un modèle DEVS atomique de délibération. Il s'agit dans notre cas de l'un des modèles suivant :

- *StrategicPlanning* qui encapsule un système de génération automatique des instances de planification stratégique et leur résolution avec Toulbar2 (cf. Chapitre 4),
- *TacticalPlanning* qui encapsule l'algorithme de planification tactique (cf. Chapitre 5, Section 5.4)
- *OperationnalPlanning* qui encapsule l'algorithme de planification opérationnelle (cf. Chapitre 5, Section 5.6)

Comme l'indique les équations 6.39, 6.40 et 6.41, en fonction de la valeur de pstate, et selon le type de planification, une partie de l'état de M_χ est envoyée sur l'une des sorties strategic, tactical ou operationnal.

$$\lambda(\mathsf{STR\_NEED}) : \mathsf{strategic} = Msg(s_\chi) \quad (6.39)$$

$$\lambda(\mathsf{TAC\_NEED}) : \mathsf{tactical} = Msg(s_\chi) \quad (6.40)$$

$$\lambda(\mathsf{STR\_NEED}) : \mathsf{operationnal} = Msg(s_\chi) \quad (6.41)$$

6.5.3.c Dynamique des structures Σ^* et fonction de structure γ

La structure Σ^α qui définit le réseau DSDEN est construite principalement à partir du plan opérationnel Π_χ^{ope} et des modèles de tâches de planification (*StrategicPlanning*, *TacticalPlanning* et *OperationnalPlanning*). La modification du réseau est réalisée par la fonction de structure $\gamma(s_\chi^\alpha, \pi)$. Dès qu'un plan opérationnel est trouvé, la fonction de structure est appelée avec $\pi = \Pi_\chi^{ope}$ afin de construire la nouvelle structure Σ^α. Le coordinateur exploite, pour ce faire, les modèles de tâches et de relations temporelles présentées dans les sections 6.3 et 6.4. Ainsi, l'appel de la fonction de structure $\gamma(s_\chi^\alpha, \Pi_\chi^{ope})$ permet de :

- construire, pour chaque tâche a du plan Π_χ^{ope}, le modèle PDEVS de tâche associé en définissant :
  - les ports d'entrées/sorties et les connexions conformément à la description présentées dans la section 6.3,
  - les intervalles $[l_{s_a}, u_{s_a}[$ et $[l_{f_a}, u_{f_a}[$
- construire, pour chaque relation temporelle du plan Π_χ^{ope}, le modèle PDEVS de relations temporelles entre tâches, conformément à la description présentées dans la section 6.4.

De la même manière, si $\pi \in \{StrategicPlanning, TacticalPlanning, OperationnalPlanning\}$ une tâche de planification est rajoutée à Σ^α suivant le principe ci-dessus décrit. Ainsi, la structure Σ^α du réseau DSDEN est obtenue par une composition de modèles PDEVS de tâches reliées par des modèles de relations temporelles.

6.5.3.d La fonction de transition interne δ_{ext}

La fonction de transition interne est appelée à la date $e = \tau(s_\chi^\alpha)$ et en absence d'événements sur l'un des ports d'entrées I_χ. Ainsi, lorsque le coordinateur est resté dans l'état partiel s_χ^α pendant $\tau(s_\chi^\alpha)$ unité de temps, il réalise une sortie puis effectue sa fonction de transition interne.

Transition de l'état INIT vers IDLE À l'initialisation du coordinateur, la fonction de transition interne ajoute une tâche de planification stratégique au réseau DSDEN (équation 6.42).

$$\delta_{int}(\mathsf{pstate} = \mathsf{INIT}) : s_\chi = (\mathsf{IDLE}, \Sigma^\alpha \leftarrow \gamma(s_\chi^0, StrategicPlanning)) \quad (6.42)$$

Transition d'un état XXX_NEED vers XXX_REQ Si l'état partiel s_χ du coordinateur impose une planification stratégique, tactique ou opérationnelle, celle-ci a due être initiée durant la précédente sortie du coordinateur. Comme l'indique

les équations 6.43, 6.44 et 6.45, le modèle M_χ passe alors dans l'état XXX_REQ associé.

$$\delta_{int}(\mathsf{pstate} = \mathsf{STR\_NEED}) : s_\chi = (\mathsf{STR\_REQ}) \quad (6.43)$$
$$\delta_{int}(\mathsf{pstate} = \mathsf{TAC\_NEED}) : s_\chi = (\mathsf{TAC\_REQ}) \quad (6.44)$$
$$\delta_{int}(\mathsf{pstate} = \mathsf{OPE\_NEED}) : s_\chi = (\mathsf{OPE\_REQ}) \quad (6.45)$$

Transition d'un état XXX_DONE vers IDLE Dans le cas où la planification stratégique a été réalisée (pstate = STR_DONE) le modèle passe dans un état IDLE. Une première tâche de planification tactique est ajoutée au réseau DSDEN (équation 6.46). Cette tâche de planification servira au déclenchement de la prochaine décision tactique. Par ailleurs, une seconde tâche de planification stratégique est ajoutée au réseau DSDEN. Elle servira au déclenchement de la *prochaine*[8] décision stratégique. Ce principe est également utilisé dès que la planification tactique vient d'être réalisée (pstate = TAC_DONE, équation 6.47). Dans ce cas, une première tâche de planification opérationnelle puis une seconde tâche de planification tactique sont ajoutées au réseau DSDEN.

$$\begin{aligned}\delta_{int}(\mathsf{pstate} = \mathsf{STR\_DONE}) : s_\chi = (&\mathsf{IDLE}, \\ &\Sigma^\alpha \leftarrow \gamma(s_\chi, TacticalPlanning) \\ &\Sigma^\alpha \leftarrow \gamma(s_\chi, StrategicPlanning))\end{aligned} \quad (6.46)$$

$$\begin{aligned}\delta_{int}(\mathsf{pstate} = \mathsf{TAC\_DONE}) : s_\chi = (&\mathsf{IDLE}, \\ &AddTo(\Pi^{ope}_\chi, OperationnalPlanning) \\ &AddTo(\Pi^{ope}_\chi, TacticalPlanning) \\ &\Sigma^\alpha \leftarrow \gamma(s_\chi, \Pi^{ope}_\chi))\end{aligned} \quad (6.47)$$

Dès qu'une planification tactique a été réalisée, (pstate = OPE_DONE) le modèle passe dans un état IDLE. Le réseau DSDEN est ensuite mis à jour avec le plan opérationnel courant (équation 6.48).

$$\delta_{int}(\mathsf{pstate} = \mathsf{OPE\_DONE}) : s_\chi = (\mathsf{IDLE}, \Sigma^\alpha \leftarrow \gamma(s_\chi, \Pi^{ope}_\chi)) \quad (6.48)$$

6.5.3.e La fonction de transition externe δ_{ext}

Durant la simulation, si un événement externe apparaît sur l'un des ports d'entrées $i \in I_\chi$ avant la fin de la phase transitoire ($t \leq e \leq t + \tau(s^\alpha_\chi)$), l'état total du système passe à $\delta_{ext}(s^\alpha_\chi, e, i)$. Chaque événement reçu est traité en fonction de son port d'apparition.

Entrée BS predicat Sur la base des événements reçus via le port d'entrée BS predicat, les variables d'état du coordinateur relatives à l'état du système biophysique BS sont mises à jour (équation 6.49).

$$\delta_{ext}(s_\chi, e, \mathsf{BS\ predicat}) : s_\chi \leftarrow (\mathcal{B}_\chi = f(\mathcal{B}_\chi, e, \mathsf{BS\ predicat}.value) \quad (6.49)$$

8. La prochaine planification stratégique est programmée à la fin de l'horizon de planification stratégique courant. Dans notre cas, il s'agit de la fin de la quatrième année

Entrée plan Le message envoyé par une tâche de planification a est reçu via le port plan. Il informe le coordinateur qu'une décision stratégique, tactique ou opérationnelle est imminente. Comme l'indique l'équation 6.50, le coordinateur M_χ passe de l'état IDLE à l'état XXX_NEED associé. L'état partiel s_χ est mis à jour ($Update(s_\chi)$) afin de préparer les paramètres à envoyer au modèle DEVS atomique de résolution du problème de décision imminent. La fonction $Cleanup(\Sigma$,a) est ensuite appelée afin de supprimer la tâche de planification a du réseau DSDEN.

$$\delta_{ext}((\text{pstate} = \text{IDLE}), e, \text{plan}) \quad : \quad s_\chi \leftarrow \begin{cases} STR\_NEED & si\ a{=}StrategicPlanning \\ TAC\_NEED & si\ a{=}TacticalPlanning \\ OPE\_NEED & si\ a{=}OperationnalPlanning \end{cases} \quad (6.50)$$
$$Update(s_\chi)$$
$$Cleanup(\Sigma^\alpha, a)$$

La fonction $Update(s_\chi)$ détermine l'horizon de planification, le plan courant et/ou les objectifs a envoyer. Premièrement, l'horizon est défini en fonction du type de décision. Pour la décision stratégique (plusieurs années) et tactique (1 année), l'horizon de planification est fixe et donné en entrée du problème . L'horizon de la décision opérationnelle est déterminé en fonction de la structure du réseau temporel décrivant le plan tactique Π_χ^{tac}. Par défaut, cet horizon est d'une semaine au maximum.

Le plan envoyé pour la décision stratégique correspond à l'historique des assolements. Le plan envoyé pour la décision tactique correspond au plan stratégique de l'année courant auquel s'ajoute le plan tactique en cours d'exécution. Enfin, le plan envoyé pour la décision opérationnelle correspond à une partie du plan tactique courant réalisable sur l'horizon de la planification. À ces informations se rajoutes d'autres, notamment celles liées aux objectifs de production, aux capacités de ressources etc.

Entrée solved Lorsqu'un événement est reçu sur le port solved, le coordinateur M_χ passe de l'état XXX_REQ à l'état XXX_DONE associé. Le plan associé est mis à jour suivant l'une des équations 6.51, 6.52 et 6.53.

$$\delta_{ext}((\text{pstate} = \text{STR\_REQ}), e, \text{solved}) \quad : \quad s_\chi \leftarrow (STR\_DONE, \Pi_\chi^{str} = \text{solved}.plan) \quad (6.51)$$
$$\delta_{ext}((\text{pstate} = \text{TAC\_REQ}), e, \text{solved}) \quad : \quad s_\chi \leftarrow (TAC\_DONE, \Pi_\chi^{tac} = \text{solved}.plan) \quad (6.52)$$
$$\delta_{ext}((\text{pstate} = \text{OPE\_REQ}), e, \text{solved}) \quad : \quad s_\chi \leftarrow (OPE\_DONE, \Pi_\chi^{ope} = \text{solved}.plan) \quad (6.53)$$

Entrée OS predicat start Dés qu'une tâche opérationnelle a démarre, un message est envoyé sur le port d'entrée OS predicat start. Comme l'indique l'équation 6.54, le coordinateur M_χ reste dans l'état IDLE. La tâche a est ajoutée à la liste des tâches en cours d'exécution.

$$\delta_{ext}((\text{pstate} = \text{IDLE}), e, \text{OS predicat start}) \quad : \quad s_\chi \leftarrow (AddTo(\pi_{start}, a)) \quad (6.54)$$

Entrée OS predicat finish Dès qu'une tâche opérationnelle a est terminée et qu'un message de fin est envoyé sur le port OS predicat finish, celle-ci est supprimée des plans tactique et opérationnel. Le coordinateur M_χ reste dans l'état IDLE. La fonction $Cleanup(\Sigma$,a) est ensuite appelée afin de supprimer la tâche de planification a du réseau DSDEN.

$$\begin{aligned}\delta_{ext}((\text{pstate} = \text{IDLE}), e, \text{OS predicat finish}) \quad : \quad & s_\chi \leftarrow (DeleteFrom(\Pi^{ope}_\chi, a) \\ & DeleteFrom(\Pi^{tac}_\chi, a)) \\ & Cleanup(\Sigma^\alpha, a)\end{aligned} \tag{6.55}$$

Entrée OS predicat fail En cas d'échec d'une tâche opérationnelle, un message reçu via le port OS predicat fail. Dans notre cas, deux stratégies RETRIES/RECOVERY sont envisageables pour la gestion des situations d'échec. La stratégie RETRIES permet de reprogrammer la tâche tant que possible. Elle est donc appliqué par défaut. Cette stratégie est la seule applicable pour l'ensemble des opérations agricoles autres que les semis.

Pour la stratégie RETRIES, le coordinateur M_χ passe de l'état IDLE à l'état OPE_NEED. Ainsi, une replanification opérationnelle est instantanément programmée. La tâche en situation d'échec et les tâches en attente d'exécution sont supprimées du réseau DSDEN.

$$\begin{aligned}\delta_{ext}((\text{pstate} = \text{IDLE}), e, \text{OS predicat fail}) \quad : \quad & s_\chi \leftarrow ((\text{OPE\_NEED}) \\ & Cleanup(\Sigma^\alpha, a) \\ & Cleanup(\Sigma^\alpha, a' \notin \pi_{start}))\end{aligned} \tag{6.56}$$

Cette stratégie RETRIES est également applicable par défaut aux situations d'échecs des opérations critiques telles que le semis. Cependant, si le contrainte temporelle d'exécution du semis ne sont plus satisfaite, [9] la stratégie RECOVERY est appliquée. Elle consiste à réaliser une replanification totale. Il s'agit notamment de trouver un plan stratégique permettant de substituer la culture prévue par une autre. Ainsi, pour la stratégie RECOVERY, le coordinateur M_χ passe de l'état IDLE à l'état STR_NEED. Une replanification stratégique est instantanément programmée. La tâche en situation d'échec et les tâches en attente d'exécution sont supprimées du réseau DSDEN.

$$\begin{aligned}\delta_{ext}((\text{pstate} = \text{IDLE}), e, \text{OS predicat fail}) \quad : \quad & s_\chi \leftarrow ((\text{STR\_NEED}) \\ & Cleanup(\Sigma^\alpha, a) \\ & Cleanup(\Sigma^\alpha, a' \notin \pi_{start}))\end{aligned} \tag{6.57}$$

6.6 Application

Dans cette section nous présentons les expérimentations menées pour l'entrelacement des trois phases de planifications et le contrôle d'exécution des tâches. Ces expérimentations nous permettent de montrer le fonctionnement globale du cadre Safihr. Elles permettent également de valider l'hypothèse selon laquelle l'exécutif du plan peut être assimilé à un système dynamique distribué dans lequel chacune des tâches de l'agent peut être représentée comme un système dynamique

9. Ça peut être le cas si la date courante est supérieure à la date de fin au plus tard du semis

autonome. Afin de satisfaire les contraintes liées aux relations temporelles entre les événements de début et de fin des tâches du plan courant, nous avons identifié et formalisé deux interprétations : précédence et synchronisation (cf. section 6.4.2). Pour les expérimentions présentées ci-dessous, seules les relations temporelles interprétées comme des contraintes de précédences ont été implémentées.

Cette section application est divisée en deux sous sections. La première décrit le contexte expérimental. La seconde quant à elle, illustre le fonctionnement global du cadre.

6.6.1 Contexte des expérimentations

Les expérimentations ont été menées en utilisant les instances B1[1-4]-LU15(*), B[1-4]-LU30(*) et B[1-4]-LU60(*). Pour ces différentes instances, les contextes des expérimentions pour les phases de planification stratégique, tactique et opérationnelle sont respectivement celles décrites dans les sections 4.6, 5.5 et 5.7. Les simulations ont été réalisées sur une machine doté d'un processeur Intel(R) Xeon(R) de 2.27 GHz.

6.6.2 Quelques résultats de simulation

6.6.2.a Dynamique d'état de l'exécutive pour l'instance B1[1-4]-LU15(*) en 1961

Nous présentons ci-dessous une partie de la trace d'exécution liée au fonctionnement du coordinateur. Les données présentée porte sur la période du 1961-01-01 au 1961-02-26. Le mot clé « *buildDEVSTask* » correspond à la mis à jour de la structure du réseau DSDEN.

```
1  ;;Début de la simulation :  1961−01−01
2    ;; Initialisation  du coordinateur
3  pstate = INIT at Dim. 1961−01−01
4  τ(INIT) =0  at 1961−01−01
5    ;;Construction/Ajout du modèle de tâche pour la planification  stratégique
6  (δint) buildDEVSTask : StrategicPlanning_0 [planning−task] at 1961−01−01
7    pstate = IDLE at Dim. 1961−01−01
8  τ(IDLE) =oo at 1961−01−01
9    ;;Message envoyé par StrategicPlanning_0 pour déclencher la planification  stratégique
10 (δext) StrategicPlanning_0 : start at 1961−01−01 | STATE = started
11   ;;Le coordinateur passe dans un état STR_NEED
12   pstate = STR_NEED  at Dim. 1961−01−01
13   Deletion of  StrategicPlanning_0 at 1961−01−01
14 τ(STR_NEED) =0 at 1961−01−01
15   ;;Message envoyé au planificateur stratégique  le  1961−01−01 et attente d'une réponse
16 (δint) pstate = STR_REQ  at Dim. 1961−01−01
17 τ(STR_REQ) =oo at 1961−01−01
18   ;;Réponse reçue du planificateur  stratégique
19     ;;Le coordinateur passe dans un état STR_DONE
20 (δext) pstate = STR_DONE  at Dim. 1961−01−01
21 τ(STR_DONE) =0  at 1961−01−01
22   ;;Construction/Ajout des modèles de tâches pour la planification  tactique  et  stratégique
23 (δint) buildDEVSTask : TacticalPlanning_1 [planning−task] at 1961−01−01
24 (δint) buildDEVSTask : StrategicPlanning_2 [planning−task] at 1961−01−01
25 τ(IDLE) =oo  at 1961−01−01
26   ;;Message envoyé par TacticalPlanning_1 pour déclencher la planification  tactique
27 (δext) TacticalPlanning_1 : start at 1961−01−01 | STATE = started
28   ;;Le coordinateur passe dans un état TAC_NEED
29   pstate = TAC_NEED  at Dim. 1961−01−01
30   Deletion of  TacticalPlanning_1 at 1961−01−01
31 τ(TAC_NEED) =0 : at 1961−01−01
32   ;;Message envoyé au planificateur tactique  le  1961−01−01 et attente d'une réponse
```

```
33 (δint) pstate = TAC_REQ at Dim. 1961−01−01
34 τ(TAC_REQ) =oo at 1961−01−01
35   ;;Réponse reçue du planificateur tactique
36     ;;Le coordinateur passe dans un état TAC_DONE
37 (δext) pstate = TAC_DONE at Dim. 1961−01−01
38 τ(TAC_DONE) =0 at 1961−01−01
39   ;;Construction/Ajout des modèles de tâches pour la planification opérationnelle et tactique
40 (δint) buildDEVSTask : OperationnalPlanning_2 [planning−task] at 1961−01−01
41 (δint) buildDEVSTask : TacticalPlanning_3 [planning−task] at 1961−01−01
42 τ(IDLE) =oo at 1961−01−01
43   ;;Message envoyé par OperationnalPlanning_2 pour déclencher la planification opérationnelle
44 (δext) OperationnalPlanning_2 : start at 1961−01−01 | STATE = started
45   ;;Le coordinateur passe dans un état OPE_NEED
46   pstate = OPE_NEED at Dim. 1961−01−01
47   Deletion of OperationnalPlanning_2 at 1961−01−01
48 τ(OPE_NEED) =0 : at 1961−01−01
49 (δint) pstate = IDLE at Dim. 1961−01−01
50   ;;Aucune tâche à réaliser cette semaine
51   Nothing to do FROM 1961−01−02 to 1961−01−08
52 (δint) buildDEVSTask : OperationnalPlanning_4 [planning−task] at 1961−01−01
53 τ(IDLE) =oo : at 1961−01−01
54 (δext) OperationnalPlanning_4 : start at 1961−01−08 | STATE = started
55   ;;Le coordinateur passe dans un état OPE_NEED
56   pstate = OPE_NEED at Dim. 1961−01−08
57   Deletion of OperationnalPlanning_4 at 1961−01−08
58 (δint) pstate = IDLE at Dim. 1961−01−08
59   ;;Aucune tâche à réaliser cette semaine
60   Nothing to do FROM 1961−01−09 to 1961−01−15
61 (δint) buildDEVSTask : OperationnalPlanning_5 [planning−task] at 1961−01−08
62 τ(IDLE) =oo : at 1961−01−08
63   .............
64
65 (δext) OperationnalPlanning_9 : start at 1961−02−12 | STATE = started
66   ;;Le coordinateur passe dans un état OPE_NEED
67   pstate = OPE_NEED at Dim. 1961−02−12
68   Deletion of OperationnalPlanning_9 at 1961−02−12
69 τ(OPE_NEED) =0 at 1961−02−12
70 (δint) pstate = OPE_NEED at Dim. 1961−02−12
71   ;;Message envoyé au planificateur opérationnel le 1961−02−12 et attente d'une réponse
72 (δint) pstate = OPE_REQ at Dim. 1961−02−12
73 τ(OPE_REQ) =oo at 1961−02−12
74   ;;Réponse reçue du planificateur opérationnel
75     ;;Le coordinateur passe dans un état OPE_DONE
76 (δext) pstate = OPE_DONE at Dim. 1961−02−12
77 τ(OPE_DONE) =0 at 1961−02−12
78   ;; Construction du réseau de tâche courant
79 (δint)
80     ;;Construction/Ajout des tâches opérationnelles aux réseaux de tâches de la parcelle 4
81     buildDEVSTask: Plowing−OP_VD_0_2 on 4 [operationnal−task] at 1961−02−12
82     buildDEVSTask: Sowing−OP_VD_1_3 on 4 [operationnal−task] at 1961−02−12
83     buildDEVSTask: Fertilization−OP1_VD_2_4 on 4 [operationnal−task] at 1961−02−12
84     ;;Construction/Ajout des relations temporelles de types FS aux réseaux de tâches de la
           parcelle 4
85     buildDEVSTask: FS0 on 4 Plowing−OP_VD_0_2 → Sowing−OP_VD_1_3 at
           1961−02−12
86     buildDEVSTask: FS1 on 4 Sowing−OP_VD_1_3 → Fertilization−OP1_VD_2_4 at
           1961−02−12
87
88     ;;Construction/Ajout des tâches opérationnelles aux réseaux de tâches de la parcelle 5
89     buildDEVSTask: Plowing−OP_VD_0_5 on 5 [operationnal−task] at 1961−02−12
90     buildDEVSTask: Sowing−OP_VD_1_6 on 5 [operationnal−task] at 1961−02−12
91     ;;Construction/Ajout des relations temporelles de types FS aux réseaux de tâches de la
           parcelle 5
```

```
92      buildDEVSTask: FS2 on 5 Plowing−OP_VD_0_5 → Sowing−OP_VD_1_6 at
            1961−02−12
93      ;;Construction/Ajout des tâches opérationnelles aux réseaux de tâches de la parcelle 12
94      buildDEVSTask: Plowing−OP_VD_0_7 on 12 [operationnal−task] at 1961−02−12
95      ;;Construction/Ajout du modèle de tâche pour la prochaine planification opérationnelle
96      buildDEVSTask : OperationnalPlanning_10 [planning−task] at 1961−02−12
97  τ(IDLE) =oo : at 1961−02−12
98
99      ;; Notifications portant sur le début des tâches
100   (δext) Plowing−OP_VD_0_2 is started at 1961−02−14
101   (δext) Plowing−OP_VD_0_5 is started at 1961−02−15
102     ;; Notifications portant sur la fin des tâches
103   (δext) Plowing−OP_VD_0_2 is done at 1961−02−15
104     ;; Notifications portant sur les contraintes de précédences valides
105   (δext) FS0 is done at 1961−02−15
106     ;; Notifications portant sur le début des tâches
107   (δext) Sowing−OP_VD_1_3 is started at 1961−02−15
108     ;; Notifications portant sur la fin des tâches
109   (δext) Plowing−OP_VD_0_5 is done at 1961−02−16
110     ;; Notifications portant les contraintes de précédences valides
111   (δext) FS2 is done at 1961−02−16
112     ;; Notifications portant sur la fin des tâches
113   (δext) Sowing−OP_VD_1_3 is done at 1961−02−16
114   (δext) FS1 is done at 1961−02−16
115     ;; Notifications portant sur le début des tâches
116   (δext) Fertilization−OP1_VD_2_4 is started at 1961−02−16
117   (δext) Sowing−OP_VD_1_6 is started at 1961−02−17
118   (δext) Plowing−OP_VD_0_7 is started at 1961−02−17
119     ;; Notifications portant sur la fin des tâches
120   (δext) Fertilization−OP1_VD_2_4 is done at 1961−02−17
121   (δext) Sowing−OP_VD_1_6 is done at 1961−02−18
122   (δext) Plowing−OP_VD_0_7 is done at 1961−02−18
123
124   ;;Message envoyé par OperationnalPlanning_10 pour déclencher une nouvelle planification
          opérationnelle
125 (δext) OperationnalPlanning_10 : start at 1961−02−19 | STATE = started
126   ;;Le coordinateur passe dans un état OPE_NEED
127   pstate = OPE_NEED at Dim. 1961−02−19
128   Deletion of OperationnalPlanning_10 at 1961−02−19
129   ;;Suppression de modèles de tâche déjà réalisées
130   Deletion of FS0 at 1961−02−19
131   Deletion of FS1 at 1961−02−19
132   Deletion of FS2 at 1961−02−19
133   Deletion of Fertilization−OP1_VD_2_4 at 1961−02−19
134   Deletion of Plowing−OP_VD_0_2 at 1961−02−19
135   Deletion of Plowing−OP_VD_0_5 at 1961−02−19
136   Deletion of Plowing−OP_VD_0_7 at 1961−02−19
137   Deletion of Sowing−OP_VD_1_3 at 1961−02−19
138   Deletion of Sowing−OP_VD_1_6 at 1961−02−19
139 τ(OPE_NEED) =0 : at 1961−02−19
140   ;;Message envoyé au planificateur opérationnel le 1961−02−19 et attente d'une réponse
141 (δint) pstate = OPE_REQ at Dim. 1961−02−19
142
143 τ(OPE_REQ) =oo at 1961−02−19
144   ;;Réponse reçue du planificateur opérationnel
145     ;;Le coordinateur passe dans un état OPE_DONE
146 (δext) pstate = OPE_DONE at Dim. 1961−02−19
147 τ(OPE_DONE) =0 at 1961−02−19
148   ;; Construction du réseau de tâche courant
149 (δint)
150     buildDEVSTask: Fertilization−OP1_VD_2_9 on 5 [operationnal−task] at 1961−02−19
151     buildDEVSTask: Sowing−OP_VD_1_10 on 12 [operationnal−task] at 1961−02−19
152     buildDEVSTask: Fertilization−OP1_VD_2_11 on 12 [operationnal−task] at
            1961−02−19
```

```
153     buildDEVSTask: FS3 on 12 SEQ constraint between Sowing−OP_VD_1_10 and
            Fertilization−OP1_VD_2_11 at 1961−02−19
154     buildDEVSTask: Plowing−OP_VD_0_12 on 13 [operationnal−task] at 1961−02−19
155     buildDEVSTask: Sowing−OP_VD_1_13 on 13 [operationnal−task] at 1961−02−19
156     buildDEVSTask: Fertilization−OP1_VD_2_14 on 13 [operationnal−task] at
            1961−02−19
157     buildDEVSTask: FS4 on 13 SEQ constraint between Plowing−OP_VD_0_12 and
            Sowing−OP_VD_1_13 at 1961−02−19
158     buildDEVSTask: FS5 on 13 SEQ constraint between Sowing−OP_VD_1_13 and
            Fertilization−OP1_VD_2_14 at 1961−02−19
159     buildDEVSTask : OperationnalPlanning_11  [PL task] at 1961−02−19
160
161      (δext) Plowing−OP_VD_0_12 is started at 1961−02−20
162      (δext) Sowing−OP_VD_1_10 is started at 1961−02−20
163      (δext) Fertilization−OP1_VD_2_9 is started at 1961−02−20
164      (δext) Fertilization−OP1_VD_2_9 is done at 1961−02−20
165      (δext) Plowing−OP_VD_0_12 is done at 1961−02−21
166      (δext) Sowing−OP_VD_1_10 is done at 1961−02−21
167      (δext) FS4 is done at 1961−02−21
168      (δext) FS3 is done at 1961−02−21
169      (δext) Fertilization−OP1_VD_2_11 is started at 1961−02−21
170      (δext) Sowing−OP_VD_1_13 is started at 1961−02−22
171      (δext) Fertilization−OP1_VD_2_11 is done at 1961−02−22
172      (δext) Sowing−OP_VD_1_13 is done at 1961−02−23
173      (δext) FS5 is done at 1961−02−23
174      (δext) Fertilization−OP1_VD_2_14 is started at 1961−02−23
175 ;;Message envoyé par OperationnalPlanning_11 pour déclencher une nouvelle planification
        opérationnelle
176 (δext) OperationnalPlanning_11 : start at 1961−02−26 | STATE = started
177   ;;Le coordinateur passe dans un état OPE_NEED
178   pstate = OPE_NEED  at Dim. 1961−02−26
179   Deletion of  OperationnalPlanning_11 at 1961−02−26
180   ;;Suppression de modèles de tâche déjà  réalisées
181   Deletion of  FS3  at 1961−02−26
182   Deletion of  FS4  at 1961−02−26
183   Deletion of  FS5  at 1961−02−26
184   Deletion of  Fertilization−OP1_VD_2_11 at 1961−02−26
185   Deletion of  Fertilization−OP1_VD_2_14 at 1961−02−26
186   Deletion of  Fertilization−OP1_VD_2_9 at 1961−02−26
187   Deletion of  Plowing−OP_VD_0_12 at 1961−02−26
188   Deletion of  Sowing−OP_VD_1_10 at 1961−02−26
189   Deletion of  Sowing−OP_VD_1_13 at 1961−02−26
190 τ(OPE_NEED) =0 : at 1961−02−26
191   ;;Message envoyé au planificateur opérationnel  le  1961−02−26 et attente d'une réponse
192 (δint) pstate = OPE_REQ  at Dim. 1961−02−26
193
194 τ(OPE_REQ) =oo at 1961−02−26
195   ;;Réponse reçue du planificateur  opérationnel
196     ;;Le coordinateur passe dans un état OPE_DONE
197 (δext) pstate = OPE_DONE  at Dim. 1961−02−26
198 τ(OPE_DONE) =0  at 1961−02−26
199   ;;  Construction du réseau de tâche courant
200    ..............
```

Listing 6.1 – *-LU15(*)]Fonctionnement du coordinateur - B1[1-4]-LU15(*)*

6.6.2.b Entrelacement des phases de décision stratégique, tactique et opérationnelle

Instance B1[1-4]-LU15(*) : Les données présentées ci-dessous (cf. Listing 6.2) montrent pour différentes dates de planification le type et l'horizon des décisions prises. On observe notamment qu'une planification stratégique (*horizon futur de 4*

ans) suivie d'une planification tactique (horizon de 1 an) sont réalisées le premier janvier de l'année 1961. La mise en œuvre opérationnelle du plan tactique résultant est ensuite réalisée tout au long de l'année. Les deux premières dates de planifications (stratégique, tactique) sont prédéfinies dans la description du problème initial (cf. Annexe A.4.4). Les planifications opérationnelles sont réalisées tous les dimanches et porte sur un horizon d'une semaine. Si au cours d'une semaine, une tâche en cours d'exécution échoue, une replanification opérationnelle est réalisée sur le reste de la semaine.

```
DatePlanif    HorizonDébut HorizonFin  Type de planification
-------------------------------------------------------------------------
1961-01-01    1961-01-01   1964-12-31  Stratégique
1961-01-01    1961-01-01   1962-01-01  Tactique
1961-02-12    1961-02-13   1961-02-19  Opérationnelle
1961-02-19    1961-02-20   1961-02-26  Opérationnelle
1961-02-26    1961-02-27   1961-03-05  Opérationnelle
1961-04-09    1961-04-10   1961-04-16  Opérationnelle
1961-04-15    1961-04-15   1961-04-16  Replanification opérationnelle
1961-04-16    1961-04-17   1961-04-23  Opérationnelle
1961-04-23    1961-04-24   1961-04-30  Opérationnelle
1961-04-30    1961-05-01   1961-05-07  Opérationnelle
1961-05-14    1961-05-15   1961-05-21  Opérationnelle
1961-05-21    1961-05-22   1961-05-28  Opérationnelle
1961-06-04    1961-06-05   1961-06-11  Opérationnelle
1961-06-11    1961-06-12   1961-06-18  Opérationnelle
1961-06-18    1961-06-19   1961-06-25  Opérationnelle
1961-06-25    1961-06-26   1961-07-02  Opérationnelle
1961-07-02    1961-07-03   1961-07-09  Opérationnelle
1961-07-09    1961-07-10   1961-07-16  Opérationnelle
1961-07-16    1961-07-17   1961-07-23  Opérationnelle
1961-07-23    1961-07-24   1961-07-30  Opérationnelle
1961-07-30    1961-07-31   1961-08-06  Opérationnelle
1961-08-13    1961-08-14   1961-08-20  Opérationnelle
1961-08-20    1961-08-21   1961-08-27  Opérationnelle
1961-09-03    1961-09-04   1961-09-10  Opérationnelle
1961-09-10    1961-09-11   1961-09-17  Opérationnelle
1961-09-24    1961-09-25   1961-10-01  Opérationnelle
1961-10-01    1961-10-02   1961-10-08  Opérationnelle
1961-10-04    1961-10-04   1961-10-08  Replanification opérationnelle
1961-10-08    1961-10-09   1961-10-15  Opérationnelle
1961-10-13    1961-10-13   1961-10-15  Replanification opérationnelle
1961-10-15    1961-10-16   1961-10-22  Opérationnelle
1961-10-19    1961-10-19   1961-10-22  Replanification opérationnelle
1961-10-22    1961-10-23   1961-10-29  Opérationnelle
....
```

Listing 6.2 – *-LU15(*)]Dates et horizons de planification de l'année 1961 - B1[1-4]-LU15(*)*

Considérant l'instance B1[1-4]-LU15(*), quatre replanifications opérationnelles, basées sur la stratégie RETRIES, ont été réalisées durant l'année 1961. Elles ont eux lieu :

- ▷ *le 1961-04-15 :* suite à l'échec d'un semis de maïs la veille sur la parcelle élémentaire 3
- ▷ *le 1961-10-04 :* suite à l'échec d'un désherbage de colza d'hiver la veille sur la parcelle élémentaire 6
- ▷ *le 1961-10-13 :* suite à l'échec d'un semis de blé d'hiver la veille sur la parcelle élémentaire 10

▷ *le 1961-10-19 :* suite à l'échec de la récolte du maïs la veille sur la parcelle élémentaire 2

Ces quatre situations d'échecs font suite à des conditions climatiques défavorables. La Figure 6.17 récapitule les différentes dates, horizon et type de planification pour l'année 1961.

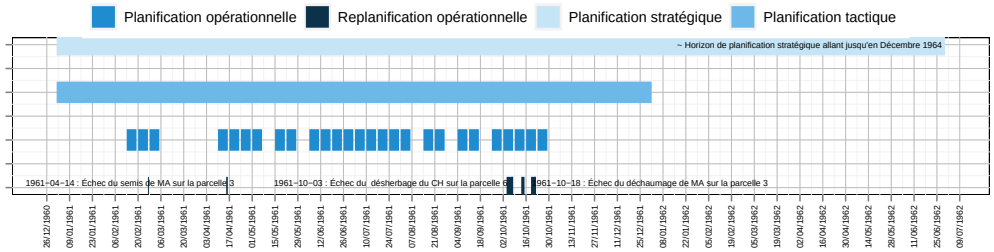


FIGURE 6.17 – *Dates et horizons de planification pour l'instance B1[1-4]-LU15(*)*

Instance B1[1-4]-LU30(*) : La Figure 6.18 récapitule les différentes dates, horizon et type de planification pour l'année 1961.

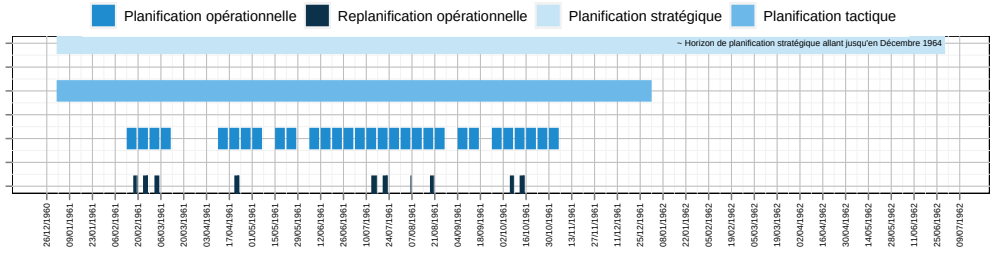


FIGURE 6.18 – *Dates et horizons de planification pour l'instance B1[1-4]-LU30(*)*

Instance B1[1-4]-LU60(*) : La Figure 6.19 récapitule les différentes dates, horizon et type de planification pour l'année 1961.

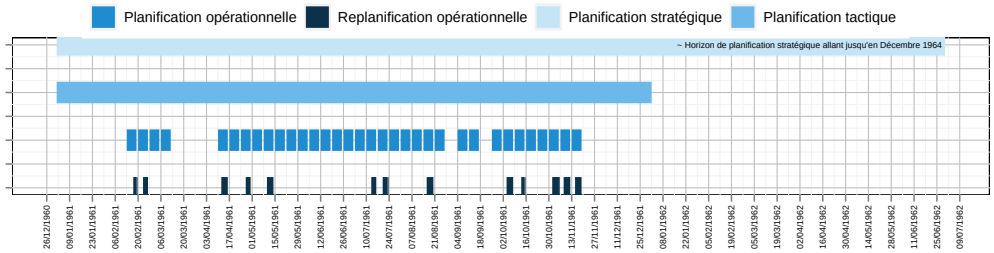


FIGURE 6.19 – *Dates et horizons de planification pour l'instance B1[1-4]-LU60(*)*

Notons que pour l'instance B1[1-4]-LU15(*), le nombre de replanifications est relativement faible comparé au deux autres. Cela s'explique notamment par la nature de notre système opérant. En effet, pour cette instance, les parcelles sont de grandes tailles. L'exécution des tâches s'étend généralement sur plusieurs jours. Le système opérant que nous utilisons étant instantané, il est donc impossible de prendre en compte les changements d'états survenus après le début de l'exécution des tâches.

6.6.2.c Dynamique d'état des tâches

Considérant une semaine du 12 au 18-02-1961, le tableau 6.4 représente l'évolution du statut des modèles de tâches contenus dans le réseau DSDEN. L'ensemble des tâches sont dans un état *Wait* dès le Dimanche 11-02-1961 (date de planification du plan opérationnel de la semaine). Pour cette semaine, le semis de la parcelle 5 et le labour de la parcelle 12 s'étendent au samedi. Ceci s'explique par le fait que les tâches qui commencent en semaine ne sont pas interrompues durant le week-end.

TABLE 6.4 – *Dynamique d'état des modèles de tâches opérationnelles sur la semaine du 1961-02-11 au 1961-02-18*

	Orge de printemps					
	LU-4			LU-5		LU-12
	Plowing	*Sowing*	*Fertilization*	*Plowing*	*Sowing*	*Plowing*
1961-02-11	Wait	Wait	Wait	Wait	Wait	Wait
1961-02-12	Wait	Wait	Wait	Wait	Wait	Wait
1961-02-13	**Started**	Wait	Wait	Wait	Wait	Wait
1961-02-14	Finished	**Started**	Wait	**Started**	Wait	Wait
1961-02-15	Finished	Finished	**Started**	Finished	Wait	Wait
1961-02-16	Finished	Finished	Finished	Finished	**Started**	**Started**
1961-02-17	Finished	Finished	Finished	Finished	Finished	Finished
1961-02-18	-	-	-	-	-	-

Contrairement aux plans opérationnels qui ont une granularité d'une heure, les modèles de tâches reposent sur une granularité d'une journée. En conséquence, durant l'exécution des tâches, nous perdons toute la précision de la planification opérationnelle. Cette dégradation de la précision est liée aux caractéristiques de nos systèmes opérants et biophysiques.

6.7 Bilan sur le chapitre

Dans ce chapitre, nous avons proposé une approche systémique, basée sur le formalisme DEVS à structure dynamique (DSDE), pour l'exécution du plan courant de l'agent. Nous avons également montré un mécanisme événementiel pour l'entrelacement des trois phases de planifications et le contrôle d'exécution des tâches.

L'exécutif de plan temporel représente les tâches comme des systèmes dynamiques autonomes. Ces tâches s'exécutent de manière décentralisée. Nous avons proposé puis formalisé un modèle générique de tâche capable de prendre en compte les contraintes temporelles de démarrage et de fin. Les modèles de tâches interagissent via des modèles de relations temporelles que nous avons également présentés. Ces modèles de relations temporelles sont essentiellement basés sur les événements de début et de fin des tâches du plan. Nous avons identifié et formalisé deux types d'interprétations des relations temporelles : précédence et synchronisation. Les relations temporelles sont interprétées comme des contraintes de *précédence* si elles intègrent un lien de causalité entre l'apparition des événements de début ou de fin de deux tâches. Les relations temporelles sont interprétées comme des contraintes de *synchronisation* si elles intègrent une dimension parallèle dans l'apparition des événements de début et de fin.

L'exécutif proposé contient un coordinateur centralisé. Ce coordinateur est capable de modifier en ligne la structure du réseau de modèle de tâche en cours d'exécution. Nous avons présenté le fonctionnement du coordinateur centralisé. Cette proposition permet de construire et de contrôler des réseaux de modèle de tâches.

Conclusion générale

Démarche suivie

Dans le cadre de cette thèse, nous nous sommes intéressés à la modélisation et à la simulation des processus de décision d'un agent agriculteur dont le but est de conduire des systèmes de culture d'une exploitation agricole. Cette problématique nous impose une prise en compte du comportement de l'agriculteur face à ses différentes classes de problème de décision. Nous soutenons que la reproduction du processus de décision d'un agriculteur consiste avant tout à résoudre en ligne les différents problèmes auxquels il se confronte.

À ce jour, les systèmes à base de règles sont généralement utilisés dans les modèles proposés en agronomie. Les approches d'agents à base de plans et capables de prendre en compte les ressources de l'exploitation restent faiblement exploitées. Nous avons expliqué que ces approches sont tout à fait pertinentes voire indispensables pour la problématique visée. Notre proposition s'inscrit pleinement dans cette dernière approche.

Nous avons établi un lien direct entre les problématiques de décision liées à la conduite des systèmes de culture à l'échelle de l'exploitation agricole et celle, généralement étudiée en intelligence artificielle, relative à de la conception des systèmes complexes autonomes opérant dans un environnement dynamique et incertain.

Nous avons décomposé ces problèmes de décisions en trois catégories :

- la planification dans le temps et dans l'espace des cultures sur l'ensemble l'exploitation agricole (*décision stratégique*),
- la planification des modes de conduites des cultures (*décision tactique*),
- l'allocation dynamique des ressources matérielles et humaines aux opérations agricoles journalières (*décision opérationnelle*).

Ces classes de problèmes sont de nature très différentes. De plus, elles portent sur des horizons de décision variables allant de quelques jours à plusieurs années. Nous avons mis en œuvre pour ce faire, des mécanismes d'interaction entre la planification et l'exécution des plans stratégiques, tactiques et opérationnelles. Ainsi, nous réduisons, le processus de décision de l'agriculteur à l'entrelacement des trois classes de problèmes de planification : *stratégique, tactique* et *opérationnelle*.

Nous avons choisi de construire un système complexe autonome capable d'entrelacer continuellement des phases de planification et d'exécution. Pour y arriver, nous avons privilégié les approches issues de la robotique autonome permettant d'intégrer les spécificités de la décision dans les systèmes de culture à l'échelle de l'exploitation.

Cette démarche nous a amené à centrer notre étude sur des architectures robotiques pertinentes pour la conception de systèmes complexes autonomes. L'architecture CLARAty (*Coupled Layer Architecture for Robotic Autonomy* (Volpe et al., 2000, 2001; Estlin et al., 2001)), utilisée dans le système CLEaR (*Closed-Loop Execution and Recovery* (Fisher et al., 2000; Estlin et al., 2001)) nous a fortement inspiré car celle-ci intègre la planification et l'exécution au niveau décisionnel,

permettant ainsi de réduire la frontière entre ces deux éléments. De plus, nous exploitons l'idée de la hiérarchisation de l'horizon de planification proposée dans CLEaR. Cela nous permet de mettre en œuvre des niveaux de planification à long terme (pour la décision stratégique), à moyen terme (pour la décision tactique) et à court terme (pour la décision opérationnelle). Nous avons proposé d'utiliser pour chaque classe de décision, des approches de planification spécifiques.

Afin de faire coopérer ces planificateurs spécifiques aux sein de notre même système, nous exploitons les mécanismes proposés dans l'architecture IDEA (*Intelligent Distributed Execution Architecture* Muscettola et al. (2002); Dias (2003)) sur l'interaction entre différents planificateurs. Chaque planificateur est alors vu comme un système de contrôle indépendant.

La mise en œuvre de cet ensemble de mécanisme nous a permis de proposer un cadre complet destiné à la reproduction :

- de la dynamique d'interaction entre des entités agronomiques autonomes, hétérogènes comprenant un agent agriculteur,
- de la construction et de l'exécution des plans d'action d'agent agriculteur.

Aperçu global des apports de la thèse

Une architecture modulaire nommée Safihr

Nous avons proposé une nouvelle architecture pour la simulation des systèmes complexes autonomes. Nommée Safihr (**S***imulation-based* **A***rchitecture* **F***or* **I***nterleaving* **H***eterogeneous decisions in* **R***eal world problems*), cette architecture modulaire est destinée à la modélisation d'un agent, vu comme un système hiérarchique dynamique et distribué explicitement en interaction avec son environnement. Ainsi, chacun des trois types de problèmes de décision de l'agriculteur est résolu par un module spécifique. Dans l'architecture Safihr, les modules de décision et leurs mécanismes d'interaction reposent entièrement sur une version parallèle et à structures dynamiques du cadre de modélisation et de simulation à événements discrets (DSDE Barros (1998)).

Cette nouvelle architecture intègre principalement :

- un module de gestion des états de croyance de l'agent,
- un module de gestion d'un ensemble d'algorithmes spécifiques aux différents problèmes de décision,
- un module d'exécution de plan, représenté comme un réseau de modèles de tâches.

Une première implémentation de l'architecture Safihr a été proposée dans le cadre de cette thèse. Les formalisations proposées dans ce manuscrit peuvent servir de bases pour d'autres implémentations de Safihr. De plus, l'architecture est suffisamment souple pour prendre en compte d'autres approches de résolution des problèmes de planifications aussi bien stratégiques, tactiques qu'opérationnels.

Planification stratégique

Pour aborder le problème de planification stratégique, nous avons proposé une approche basée sur les réseaux de contraintes pondérées. Nous avons montré la manière dont les contraintes dures et les préférences d'un agriculteur peuvent être abordées sous la forme d'une optimisation de fonction objective globale. Les résultats obtenus montrent qu'avec le solver *Toulbar2*, des solutions peuvent être trouvées en temps raisonnable pour des problèmes d'allocation de culture de petites et moyennes tailles.

Cette approche nous a permis de résoudre le problème de décision stratégique dans sa globalité et de manière optimale tout en rendant plus explicite la prise en compte des dimensions spatiales et temporelles de l'allocation de culture. L'approche proposée nous semble plus prometteuse que celles proposées par :

- Annetts et Audsley (2002) car dans ces travaux, l'allocation de cultures repose essentiellement sur l'optimisation de la marge brute,
- Bachinger et Zander (2007) car dans ces travaux, les connaissances prisent en compte se résument aux filtres agronomiques du types : règles de succession, quantités de produits phytosanitaires dans le sol etc.
- El-Nazer et McCarl (1986); Dogliotti et al. (2003) car dans ces travaux, l'allocation de cultures se limite à la recherche des successions de cultures admissibles,
- Castellazzi et al. (2008) car dans ces travaux, ni l'optimalité de l'allocation, ni la complétude des solutions optimales n'ont été recherchées.

Planification tactique

Nous avons montré comment la décision du choix des modes de conduite des cultures peut être abordée comme un ensemble de problèmes de planification hiérarchique temporelle sous contrainte de ressources. La propagation des contraintes temporelles est réalisée de manière incrémentale au fur et à mesure de l'expansion des réseaux de tâches hiérarchiques. L'heuristique de décomposition que nous proposons est une fonction globale qui permet d'expliciter les interdépendances (liées aux ressources) entre différents modes de conduites des cultures. Cette approche de planification nous a permis de représenter et de raisonner sur le temps et les ressources. Les plans tactiques proposés sont des plans flexibles basés sur les réseaux de contraintes temporelles simples (STN - *Simple Temporal Networks* (Dechter et al., 1991)).

Notre approche de planification tactique semble être plus prometteuse que celles proposées par :

- Attonaty et al. (1993); Martin-Clouaire et Rellier (2009) car nous introduisons un raisonnement numérique sur le temps et les ressources,
- Martin-Clouaire et Rellier (2009) car contrairement à ce dernier, nous avons proposé une heuristique formelle pour la réduction des tâches composées en tâches primitives, permettant ainsi de montrer comment les effets globaux des itinéraires techniques peuvent être pris en compte.

Planification opérationnelle

Pour aborder le problème de planification opérationnelle, nous avons proposé un modèle de programmation linéaire en nombres entier et un algorithme de résolution séquentielle permettant de construire l'ordonnancement à court terme des opérations agricoles. Cette approche nous a permit d'intégrer, contrairement aux travaux existants Attonaty et al. (1993); Martin-Clouaire et Rellier (2009), la notion d'anticipation dans la décision de l'organisation du travail sur l'ensemble de l'exploitation.

Exécutif distribué de plan temporel en DSDE

Nous avons proposé une approche systémique pour l'exécution du plan courant de l'agent. Notre exécutif peut être vue comme un système dynamique distribué. L'approche proposée se base sur le formalisme DSDE. Chacune des tâches s'exécute de manière indépendante. Nous avons introduit les mécanismes permettant de faire

interagir des modèles de tâche autonomes dans l'optique de maintenir la consistance des contraintes temporelles du plan courant de l'agent.

Cette proposition permet de concevoir des réseaux de tâches modulaires capables d'assurer de manière décentralisée les relations temporelles entre tâches. L'approche proposée est novatrice car elle permet d'affirmer qu'il est possible de réaliser un système capable de simuler un agriculteur dans ses différentes décisions.

Adéquation avec le domaine d'application

Les mises en œuvre réalisées sur notre cas d'étude, basé sur une exploitation virtuelle, illustrent l'adéquation de notre proposition avec le domaine applicatif. Aucun des travaux existants ne propose un cadre aussi complet pour la simulation du processus de décision de l'agriculteur. Via notre cas d'étude, nous avons montré comment les différents concepts spatiaux/organisationnels (exploitation, blocs fonctionnels, îlots fonctionnels, parcelles) et temporels (choix des rotations pluriannuelle, assolement annuelle, actions journalières) peuvent être pris en compte en agronomie. En dépit de l'absence d'une analyse comparative, les perspectives ouvertes par les approches de résolutions que nous proposons nous semble prometteuses.

Perspectives

Les travaux présentés dans ce manuscrit constituent une première approche validant nos mécanismes de reproduction du comportement d'un agriculteur. Nous nous sommes concentrés sur une exploitation virtuelle, afin de nous assurer que toutes les spécificités des problèmes de conduite de culture à l'échelle de l'exploitation on été bien prises en compte.

Pour la suite, nous comptons explorer trois grands axes relatifs :
- au renforcement de la pertinence agronomique,
- à l'exploitation de l'architecture Safihr,
- l'approfondissement des approches de planification.

Renforcement de la pertinence agronomique

Expérimentation sur un cas réel L'exploitation virtuelle étudiée dans ce manuscrit est issue d'une série d'enquêtes (Dury et al., 2011). Elle caractérise de ce fait différents types d'exploitation réelles. Nous envisageons de poursuivre des expérimentations sur des exploitations réelles. En effet, en considérant des conditions pédoclimatiques, les cultures ou les types de sol des exploitions réelles, nous sommes persuadés que le problème de décision, notamment la dimension stratégique, peut être plus simple. La finalité de cette piste de travail est de montrer que bien souvent, les problèmes réels en agronomie peuvent être plus simples en terme de planification stratégique.

Comparaison avec d'autres approches traditionnelles en Agronomie De nombreux travaux en agronomie ont exploré différentes pistes généralement spécifiques à chacun des problème de décision. Nous envisageons pour la suite une étude comparative dont la finalité sera de confronter le gain obtenu par des approches planifiées reposant sur les travaux en IA (ex : planification de l'organisation spatio-temporelle, planification des tâches) et les approches réactives (ex : règles de décision) communément utilisées. Ainsi, nous pouvons confronter l'exhaustivité de nos approches à la pratique des agriculteurs. Au travers des cette étude,

nous espérons pouvoir justifier l'intérêt d'utiliser nos résultats comme une base de référence permettant d'évaluer la qualité des règles d'allocation de cultures, d'affectation des itinéraires techniques et d'organisation du travail.

Extension à d'autres problème d'agronomie Nous pensons que le cadre et les méthodes proposés peuvent être exploités pour d'autres types de problèmes en agronomie. Par exemple, les systèmes de gestion de troupeaux, les exploitations mixtes, les systèmes de maraîchage(Mireille Navarrete et Marianne Le Bail, 2007) etc. peuvent être expérimentés. Cette piste de travail nous permettra d'éprouver les limites du cadre.

Architecture Safihr

Multiplication des approches de résolution L'architecture Safihr est conçue dans l'optique de faciliter, durant la simulation, l'entrelacement de différents systèmes de décision. Nous envisageons de montrer par la suite comment des systèmes de décision concurrents peuvent être peuvent être exploités dans afin de résoudre un même problème de décision. Nous pouvons envisager à terme, des heuristiques de sélection des méthodes de résolutions les plus adaptées à un types de problèmes donnés. Cette piste de travail présente au moins deux avantage. D'une part, elle permettra, conformément aux ambitions du projet RECORD (Bergez et al., 2012), d'enrichir notre bibliothèque de méthodes de résolution. D'autre part, elle facilitera la mise en œuvre des études comparatives entre différentes méthodes de résolution.

Outils finalisé Nous souhaitons nous baser sur cette architecture afin de proposer un outil entièrement finalisé, destiné à des utilisateurs finaux qui seraient des agronomes voir des agriculteurs.

Les premiers pourraient ainsi disposer d'un ensemble de méthodes permettant d'évaluer les pratiques des agriculteurs, d'étudier leurs systèmes de cultures et surtout de concevoir des nouveaux modes de conduites des systèmes. Cela permettra de conseiller les agriculteurs pour la prise en compte des objectifs globaux tels que les facteurs environnementaux etc.

Les seconds pourraient également exploiter des applications basées sur l'architecture Safihr afin de décrire la structure et les états des parcelles puis d'utiliser notre outil via des assistants personnels d'aide à la décision.

Approfondissement des approches de planification

Étude comparative des approches de planification stratégique Nous avons mentionné les limites de notre approche de planification stratégique sur des problèmes de grandes tailles. Nous sommes convaincus que d'autres formulations du problème pourraient offrir de meilleurs performances. Afin de valider cette intuition, nous avons à ce jour entreprit d'explorer d'autres formulations du problème. Il s'agit notamment de proposer une formulation en programmation linéaire en nombres entiers (formulation IP) du modèle de planification stratégique. Cela nous permettra d'étendre le modèle WCSP proposé. Nous avons déjà entreprit des quelques travaux de ce sens. Des premiers résultats assez prometteurs ont été récemment décrits dans Akplogan et al. (2012b).

Intégration des périodes critiques dans l'heuristique de décomposition utilisé pour la planification tactique Le système de planification tac-

tique que nous avons proposé intègre un heuristique de décomposition permettant de raisonner de manière globale sur l'utilisation des ressources. Nous pouvons donc raisonner annuellement sur les consommations de ressources de chaque décomposition. Par la suite, nous comptons intégrer les périodes critiques c'est-à-dire les périodes d'utilisation intensives de ressources. En effet, certaines périodes de l'année sont creuses en raison des opérations agricoles potentiellement exécutables. D'autres sont particulièrement chargées. Dans ces conditions, un raisonnement annuel sur les disponibilités de ressources ne permet pas d'apprécier les périodes de fortes concurrences. Nous envisageons d'explorer de nouvelles pistes de travail capables de repousser cette limite de notre proposition.

Évaluation des jours disponibles Pour l'instant, lors des phases de planification tactique et opérationnelle, nous ne prenons pas en compte les jours disponibles pour la réalisation de opérations agricole. Or, l'incertitude intrinsèque de la conduite des systèmes de culture à l'échelle de l'exploitation agricole nous permet de dire avec certitude que notre approche est excessivement optimiste. Nous savons que tous les jours d'une année ne sont pas disponibles. Ceci est lié à la réalité de la variabilité climatique et au type de sol. Ces caractéristiques impactent en particulier la portance des parcelles et implicitement l'accès à ces dernières.

À ce stade, nous pensons qu'il est possible de déduire les jours disponibles d'un apprentissage de plusieurs scénarios climatiques. La finalité de cette piste de travail serait de reconsidérer non seulement les volumes horaires de disponibilité des ressources matérielles mais aussi les jours disponibles pour l'ordonnancement des tâches quotidiennes.

Annexes

A

Sommaire

Démarche suivie 227
Aperçu global des apports de la thèse 228
Perspectives 230

A.1 Description des opérations agricoles

Nous présentons ci-dessous une description simplifiée et non exhaustive des principales opérations agricoles pour la conduite des cultures annuelles.

Travail du sol : consiste à préparer la parcelle dès la récolte de la culture précédente. Le travail du sol affecte la majorité des processus physiques, chimiques et biologiques qui se déroulent dans le sol. En fonction de la nature du travail du sol réalisé, l'action visera la structure du sol, l'enfouissement des matières organiques, la destruction des mauvaises herbes ou l'amélioration de la circulation de l'eau. Il existe plusieurs types de travail du sol :

- le *déchaumage* : opération agricole superficielle de préparation du sol. Elle consiste à arracher et enterrer les plantes ou les chaumes de la culture précédente. Les outils généralement utilisés pour cette opération sont des outils à dents ou disques appelés déchaumeurs.
- le *décompactage* : opération agricole de travail du sol en profondeur permettant de fragmenter le sol sans retournement. Ce sont des outils à dents qui sont généralement utilisés et appelés décompacteurs.
- le *labour* : opération agricole de travail du sol en profondeur permettant de découper puis de retourner d'une bande de terre au moyen d'une charrue.
- le *travail superficiel* : opération consistant à travailler le sol en surface en affinant la terre en vue de préparer le lit de semence. Les outils utilisés sont animés (type herse rotative) ou non (type vibroculteur).

Amendement et fertilisation de fond avant semis : consiste à entretenir la fertilité chimique et organique du sol et peut être sous forme minérale ou organique :

- la *fertilisation minérale* : opération agricole de fertilisation consistant à apporter des engrais phospho-potassiques (P et K) sous forme minérale dans le sol. Les outils utilisés sont des épandeurs à engrais centrifuges.
- le *chaulage* : opération visant à corriger l'acidité d'un sol par apport de carbonates de calcium.
- la *fertilisation organique* : opération agricole de fertilisation consistant à épandre le fumier, compost ou lisier sur les parcelles. Les outils généralement utilisés sont les remorques épandeurs à fumier ou des tonnes à lisiers.

Semis : opération agricole consistant à mettre une graine en terre pour qu'elle germe et se développe. Différents types de semoirs existent suivant les graines semées (petites ou grosses graines) et les techniques de semis (semis direct ou non).

Buttage : opération agricole permettant de mettre la terre en butte le long du rang afin d'entourer la base des plantes. Cette opération est réalisée dans l'optique de favoriser le développement de la plante en absence de lumière. C'est le cas en particulier pour les pommes de terre. Des outils spécifiques appelés buteuses sont utilisés.

Désherbage : opération agricole permettant d'éviter la concurrence des adventices c'est-à-dire des mauvaises herbes, vis à vis de la culture principale de la parcelle. Cette opération agricole est réalisée en prévention afin d'éviter l'apparition de mauvaises herbes ou au coup par coup en adaptant l'intervention en fonction des adventices dominants. On distingue deux techniques de désherbage qui sont :

- le *désherbage mécanique* : opération agricole de désherbage qui consiste à detruire mécaniquement les adventices. Les outils généralement utilisés sont les herses etrilles (sur céréales) et les bineuses (sur plantes sarclées comme le maïs)
- le *désherbage chimique* : opération agricole de désherbage qui se base sur l'apport de substances chimiques appelées herbicides. Les outils généralement utilisés sont les pulvérisateurs.

Fertilisation de croissance : consiste à apporter aux cultures pendant leur croissance les éléments nécessaires au développement des plantes. Cette opération permet d'accroître le rendement et la qualité des cultures. La fertilisation de croissance est généralement réalisée sous forme minérale et consiste essentiellement à apporter de l'azote (N). Les outils utilisés sont les pulvérisateurs (engrais liquides) et les distributeurs d'engrais (engrais solides).

Protection ravageurs consiste à traiter les cultures pendant leur croissance afin de lutter contre les maladies des plantes et contre de nombreux insectes nuisibles à la culture. On distingue trois types de protection qui sont :

- la *protection par molluscicides* : opération agricole effectuée durant la croissance dont le but est d'empêcher les mollusques tels que les limaces de dévorer les feuilles. Le traitement est réalisé en apportant des molluscicides de manière localisée sur le pourtour des parcelles ou sur l'ensemble de la surface de la parcelle. Les outils généralement utilisés sont des épandeurs centrifuges.
- la *protection par insecticide* : opération agricole effectuée durant la croissance dont le but est d'empêcher le développement d'une forte population d'insectes ravageurs des plantes (type pucerons). Le traitement est réalisé en apportant des insecticides. Les outils généralement utilisés sont les pulvérisateurs.
- la *protection par fongicide* : opération agricole effectuée durant la croissance dont le but est d'empêcher le développement de champignons (ex : rouille, septoriose). Le traitement est réalisé en pulvérisant des fongicides sur les plantes au moyen de pulvérisateurs.

Régulation de croissance : opération agricole de durcissage des tiges permettant de palier aux risques de chutes causées par une élongation ou croissance excessive des plantes. Les outils généralement utilisés sont les pulvérisateurs.

Irrigation : opération agricole consistant à apporter artificiellement l'eau aux cultures afin de compenser l'insuffisance des précipitations naturelles et éviter des déficits en eau à des phases critiques du cycle de croissance. Les outils généralement utilisés sont les rampes et les pivots d'irrigation.

Défanage : opération agricole dont le but est de faire faner la feuille afin de stopper la croissance de la plante avant la récolte. Le défanage est généralement réalisé par pulvérisation sur les parties aériennes de la plante d'un produit de synthèse.

Récolte : opération agricole dont le but est de récupérer les graines, les racines et tubercules ou les fourrages une fois leur maturité atteinte. Les outils généralement utilisés sont les moissonneuses-batteuses pour les céréales, les arracheuses pour les tubercules et divers outils spécifiques pour la récolte des fourrages.

A.2 Discrete EVent system specification

A.2.1 Fermeture sous couplage de DEVS classique

Soit un modèle DEVS couplé M constitué de N modèles DEVS atomiques. Zeigler et al. (2000) montrent que M peut être associé à un modèle DEVS atomique $M_r = \langle I, O, S, \tau, \delta_{int}, \delta_{ext}, \lambda \rangle$ appelé la *résultante*. L'ensemble des états partiels S de la résultante est défini par le produit cartésien des ensembles d'états totaux Q_n des modèles $n \in N$ pris indépendamment. Ainsi, $S = \underset{n \in N}{\times} Q_n$ et $s = \{(s_n, e_n)\}$. La fonction d'avancement du temps $\tau(s) = \underset{n \in N}{\min}(\sigma_n)$ où $\sigma_n = \tau_n(s_n) - e_n$ définit le temps restant avant la prochaine transition interne de n. Dans ces conditions, le sous ensemble imm de modèles imminents est défini par l'ensemble des modèles pour lesquels $\sigma_n = \tau(s)$. Soit $n^* = \mathsf{select}(imm)$ le modèle sélectionné pour effectuer une sortie et une transition interne. La fonction de transition interne $\delta_{int}(s)$ de la résultante M_r est une application de $S \to S$ tel que $\delta_{int}(s) = s' = \{(s'_n, e'_n)\}$ avec

$$\begin{aligned}
(s'_n, e'_n) = &\quad (\delta_{int,n}(s_n), 0) && \text{si } n = n^* \\
= &\quad (\delta_{ext,n}(s_n, (e_n + \tau(s)), i_n), 0) && \text{si } n^* \in \zeta_n \text{ et } i_n \neq \emptyset \\
= &\quad (s_n, (e_n + \tau(s))) && \text{sinon}
\end{aligned}$$

Ainsi, à t, la fonction de transition interne de la résultante change l'état du modèle n^* en l'autorisant à réaliser une transition interne. Les modèles n influencés par n^* effectuent une transition externe et mettent à jour leurs états sur la base des sorties réalisées par n^*. Les autres modèles mettent à jours le temps e_n écoulé depuis la dernière transition.

La fonction de transition externe de $\delta_{ext}(s)$ de la résultante M_r est définie de $Q \times I \to S$ tel que $\delta_{ext}(s, e, i) = s' = \{(s'_n, e'_n)\}$ avec

$$\begin{aligned}
(s'_n, e'_n) = &\quad (\delta_{ext,n}(s_n, (e_n + e), i_n), 0) && \text{si } \mathsf{self} \in \zeta_n \\
= &\quad (s_n, (e_n + e)) && \text{sinon}
\end{aligned}$$

Les modèles n influencés par la résultante elle même (self) effectuent une transition externe et mettent à jour leurs états sur la base des événements d'entrée de M_r. Les autres modèles mettent à jour le temps e_n écoulé en l'augmentant de e. Enfin la sortie $\lambda(s) = \emptyset$ si $n^* \notin \zeta_{\mathsf{self}}$. Dans le cas contraire, $\lambda(s)$ dépend de la fonction de transfert $Z_{n^*,\mathsf{self}} : O_{n^*} \to O$. On transfert ainsi les sorties du modèle imminent sélectionné n^* aux sorties de la résultante M_r suivant la définition de $Z_{n^*,\mathsf{self}}$.

A.2.2 Fermeture sous couplage de DEVS parallèle

Un modèle couplé $M = \langle I, O, N, \{M_n\}, \{\zeta_n\}, \{Z_{n,n'}\} \rangle$ peut être assimilé à sa résultante $M_r = \langle I, O, S, \tau, \delta_{int}, \delta_{ext}, \delta_{con}, \lambda \rangle$. Les variables I, O, S et τ sont définies de manière analogue à ceux de la fermeture sous couplage de DEVS classique (cf. annexe A.2.1). L'état s est égal à $\{(s_n, e_n)\}$. La fonction d'avancement du temps $\tau(s) = \underset{n \in N}{\min}(\sigma_n)$ où $\sigma_n = \tau_n(s_n) - e_n$ définit le temps restant avant la prochaine transition interne de n. Soit imm le sous ensemble de modèles imminents. Soit $inf(s)$ l'ensemble des modèles ayant reçu au moins un message en entrée.

À chaque transition, les sous-modèles de M peuvent être regroupés en quatre sous-ensembles. Le sous ensemble $int(s)$ représente l'ensemble des sous-modèles imminents de M n'ayant reçu aucun événement externe. Le sous-ensemble $ext(s)$ représente l'ensemble des sous-modèles non imminents ayant reçu un événement

externe. Le sous ensemble $conf(s)$ représente l'ensemble des sous-modèles imminents ayant reçu un événement externe tandis que $un(s)$ contient les sous-modèles restants.

$$\begin{cases} imm(s) & = \{n | \sigma_n = \tau(s)\} & \textit{imminents} \\ inf(s) & = \{n | n' \in \zeta_n, n' \in imm(s) \wedge i_n^b \neq \emptyset\} & \textit{ayant reçu un message} \\ & \text{avec } i_n^b = \{Z_{n',n}(\lambda_{n'}(s_{n'})) | n' \in imm(s) \cap \zeta_n\} & \\ conf(s) & = imm(s) \cap inf(s) & \textit{en conflit} \\ int(s) & = imm(s) - inf(s) & \textit{imminents et } i_n^b = \emptyset \\ ext(s) & = inf(s) - imm(s) & \textit{non imminents et } i_n^b \neq \emptyset \\ un(s) & = N - imm(s) - inf(s) & \textit{modèles restants} \end{cases}$$

La fonction de sortie est obtenue en collectant les sorties de tous les modèles imminents. Ainsi,

$$\lambda(s) = \{Z_{n,\mathsf{self}}(\lambda_n(s_n)) | n \in imm(s) \wedge n \in \zeta_{\mathsf{self}}\}$$

La fonction de transition interne de la résultante M_r est l'agrégation des quatre types de transition qui sont :

- la transition interne des sous-modèles $n \in int(s)$,
- la transition externes des sous-modèles $n \in ext(s)$,
- la transition de conflit des sous-modèles $n \in conf(s)$,
- la mise à jour de l'avancement du temps des modèles $n \in un(s)$.

Ainsi, sous l'hypothèse qu'aucun événement n'ait été reçu par M, la fonction de transition interne $\delta_{int}(s)$ de la résultante M_r est définie par $\delta_{int}(s) = s' = \{(s'_n, e'_n)\}$ avec

$$\begin{array}{lll} (s'_n, e'_n) & = (\delta_{int,n}(s_n), 0) & \text{si } n \in int(s) \\ & = (\delta_{ext,n}(s_n, e_n + \tau(s), i_n^b), 0) & \text{si } n \in ext(s) \\ & = (\delta_{con,n}(s_n, i_n^b), 0) & \text{si } n \in conf(s) \\ & = (s_n, e_n + \tau(s)) & \text{dans les autres cas} \end{array}$$

Le bag d'événements i^b reçu est fusionné aux bags d'événements i_n^b de chacun des sous-modèles n tel que $\mathsf{self} \in \zeta_n$. En conséquence la fonction de transition externe de $\delta_{ext}(s)$ de la résultante M_r est définie par $\delta_{ext}(s, e, i^b) = s' = \{(s'_n, e'_n)\}$ avec $0 < e < \tau(s)$

$$\begin{array}{lll} (s'_n, e'_n) & = (\delta_{ext,n}(s_n, e_n + e, i_n^b), 0) & \text{pour } \mathsf{self} \in \zeta_n \\ & = (s_n, e_n + e)(s)) & \text{sinon} \end{array}$$

Enfin, la fonction de conflit est appelée dans le cas où le modèle couplé reçoit un événement externe à la même date qu'une transition interne prévue par au moins un sous-modèle. Il s'avère que la différence entre les fonctions de transition δ_{conf} et δ_{int} de la résultante est essentiellement liée aux bags i^b reçus du réseau à la date $\tau(s)$. On peut donc étendre l'ensemble $inf(s)$ aux modèles influencés par les événement reçus du réseau. Ce nouvel ensemble sera nommé $inf'(s)$. Sur cette base on redéfinit les quatre groupes de modèles $conf'(s)$, $int'(s)$, $ext'(s)$ et $un'(s)$ ayant respectivement la même sémantique que $conf(s)$, $int(s)$, $ext(s)$ et $un(s)$. Ainsi :

$$\begin{cases} inf'(s) & = \{n | n' \in \zeta_n, n' \in imm(s) \wedge i^b_n \neq \emptyset\} \\ & \text{avec } i^b_n = \{Z_{n',n}(\lambda_{n'}(s_{n'})) | n' \in imm(s) \cap \zeta_n\} \uplus \{Z_{\mathsf{self},n}(i) | i \in i^b \wedge \mathsf{self} \in \zeta_n\} \\ conf'(s) & = imm(s) \cap inf'(s) \\ int'(s) & = imm(s) - inf'(s) \\ ext'(s) & = inf'(s) - imm(s) \\ un(s) & = N - imm(s) - inf'(s) \end{cases}$$

Notons que l'opération d'union disjointe des bags ($\uplus$), définissant i^b_n conserve l'origine des événements permettant ainsi de garder l'ensemble des événements en palliant à la perte d'événements induite par l'opérateur d'union classique. D'une part, le bag $\{Z_{n',n}(\lambda_{n'}(s_{n'}))|n' \in imm(s) \cap \zeta_n\}$ collecte toutes les sorties des modèles imminents. D'autre part, le bag $\{Z_{\mathsf{self},n}(i)|i \in i^b \wedge \mathsf{self} \in \zeta_n\}$ contient les événements envoyés aux sous-modèles depuis le réseau. On peut ainsi définir la fonction de conflit $\delta_{conf}(s, i^b) = s' = \{(s'_n, e'_n)\}$ avec :

$$\begin{aligned} (s'_n, e'_n) & = (\delta_{int,n}(s_n), 0) && \text{si } n \in int'(s) \\ & = (\delta_{ext,n}(s_n, e_n + \tau(s), i^b_n), 0) && \text{si } n \in ext'(s) \\ & = (\delta_{con,n} s_n, i^b_n), 0) && \text{si } n \in conf'(s) \\ & = (s_n, e_n + ta(s)) && \text{sinon} \end{aligned}$$

A.3 Analyse des solutions d'assolement : proportions de cultures

A.3.1 Instance « B1234-LU15-ALL-star » : blocs fonctionnels 1,2,3 et 4 en interdépendance

Table A.1 – *Instance B1234-LU15-ALL-star*

	Année 6				Année 7				Année 8				Année 9			
	BH	OP	MA	CH	BH	OP	MA	CH	BH	OP	MA	CH	BH	OP	MA	CH
1	60	48	24	48	96	36	48	0	60	60	24	36	84	36	48	0
2	60	48	24	48	96	36	48	0	60	60	24	36	84	36	48	0

Fin des solutions de l'instance

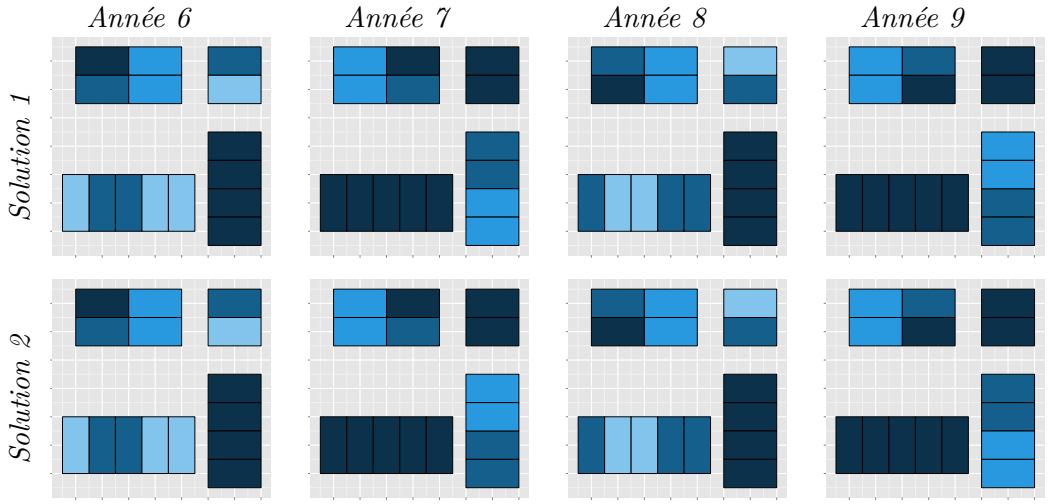


Figure A.1 – *Ensemble des solutions de l'instance B1234-LU15-ALL(*) : Blè d'hivers, Orge de printemps, Maïs, Colza d'hivers*

A.3.2 Instance « B1234-LU30-ALL-star » : blocs fonctionnels 1,2,3 et 4 en interdépendance

Table A.2 – *Instance B1234-LU30-ALL-star*

	Année 6				Année 7				Année 8				Année 9			
	BH	OP	MA	CH	BH	OP	MA	CH	BH	OP	MA	CH	BH	OP	MA	CH
1	66	48	24	42	96	30	48	6	66	54	24	36	78	48	48	0
2	66	48	24	42	96	30	48	6	66	54	24	36	78	48	48	0
3	66	48	24	42	90	36	48	6	66	54	24	36	84	42	48	0

... suite sur la page suivante ...

Table A.2 – *Instance B1234-LU30-ALL-star*

	Année 6				Année 7				Année 8				Année 9			
	BH	OP	MA	CH	BH	OP	MA	CH	BH	OP	MA	CH	BH	OP	MA	CH
4	66	48	24	42	96	30	48	6	66	54	24	36	78	48	48	0
5	66	48	24	42	96	30	48	6	66	54	24	36	78	48	48	0
6	66	48	24	42	90	36	48	6	66	54	24	36	84	42	48	0
7	66	48	24	42	96	30	48	6	66	54	24	36	78	48	48	0
8	66	48	24	42	96	30	48	6	66	54	24	36	78	48	48	0
9	66	48	24	42	90	36	48	6	66	54	24	36	84	42	48	0
10	66	48	24	42	96	30	48	6	66	54	24	36	78	48	48	0
11	66	48	24	42	96	30	48	6	66	54	24	36	78	48	48	0
12	66	48	24	42	90	36	48	6	66	54	24	36	84	42	48	0

Fin des solutions de l'instance

FIGURE A.2 – *Sept des douze des solutions de l'instance B1234-LU30-ALL(*) :* ■ *Blè d'hivers,* ■ *Orge de printemps,* ■ *Maïs,* ■ *Colza d'hivers*

A.3.3 Instance « B1234-LU60-ALL-star » : blocs fonctionnels 1,2,3 et 4 en interdépendance

Table A.3 – *Instance B1234-LU60-ALL-star*

	Année 6				Année 7				Année 8				Année 9			
	BH	OP	MA	CH	BH	OP	MA	CH	BH	OP	MA	CH	BH	OP	MA	CH
1	69	48	24	39	99	24	48	9	69	51	24	36	72	57	48	0
2	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
3	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
4	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
5	69	48	24	39	99	24	48	9	69	51	24	36	72	57	48	0
6	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
7	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
8	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
9	69	48	24	39	99	24	48	9	69	51	24	36	72	57	48	0
10	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
11	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
12	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
13	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
14	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
15	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
16	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
17	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
18	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
19	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
20	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
21	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
22	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
23	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
24	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
25	69	48	24	39	99	24	48	9	69	51	24	36	72	57	48	0
26	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
27	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0

. . . suite sur la page suivante . . .

Table A.3 – *Instance B1234-LU60-ALL-star*

	Année 6				Année 7				Année 8				Année 9			
	BH	OP	MA	CH	BH	OP	MA	CH	BH	OP	MA	CH	BH	OP	MA	CH
28	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
29	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
30	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
31	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
32	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
33	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
34	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
35	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
36	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
37	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
38	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
39	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
40	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
41	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
42	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
43	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
44	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
45	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
46	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
47	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
48	69	48	24	39	87	36	48	9	69	51	24	36	84	45	48	0
49	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
50	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
51	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
52	69	48	24	39	87	36	48	9	69	51	24	36	84	45	48	0
53	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
54	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
55	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
56	69	48	24	39	87	36	48	9	69	51	24	36	84	45	48	0

. . . suite sur la page suivante . . .

Table A.3 – *Instance B1234-LU60-ALL-star*

	Année 6				Année 7				Année 8				Année 9			
	BH	OP	MA	CH	BH	OP	MA	CH	BH	OP	MA	CH	BH	OP	MA	CH
57	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
58	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
59	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
60	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
61	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
62	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
63	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
64	69	48	24	39	87	36	48	9	69	51	24	36	84	45	48	0
65	81	36	24	39	99	24	48	9	81	39	24	36	96	33	48	0
66	81	36	24	39	99	24	48	9	81	39	24	36	96	33	48	0
67	81	36	24	39	99	24	48	9	81	39	24	36	96	33	48	0
68	81	36	24	39	99	24	48	9	81	39	24	36	96	33	48	0
69	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
70	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
71	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
72	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
73	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
74	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
75	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
76	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
77	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
78	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
79	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
80	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
81	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
82	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
83	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
84	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
85	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0

. . . suite sur la page suivante . . .

Table A.3 – *Instance B1234-LU60-ALL-star*

	Année 6				Année 7				Année 8				Année 9			
	BH	OP	MA	CH	BH	OP	MA	CH	BH	OP	MA	CH	BH	OP	MA	CH
86	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
87	69	48	24	39	87	36	48	9	69	51	24	36	84	45	48	0
88	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
89	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
90	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
91	69	48	24	39	87	36	48	9	69	51	24	36	84	45	48	0
92	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
93	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
94	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
95	69	48	24	39	87	36	48	9	69	51	24	36	84	45	48	0
96	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
97	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
98	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
99	69	48	24	39	87	36	48	9	69	51	24	36	84	45	48	0
100	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
101	69	48	24	39	99	24	48	9	69	51	24	36	72	57	48	0
102	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
103	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
104	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
105	69	48	24	39	99	24	48	9	69	51	24	36	72	57	48	0
106	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
107	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
108	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
109	69	48	24	39	99	24	48	9	69	51	24	36	72	57	48	0
110	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
111	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
112	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
113	69	48	24	39	99	24	48	9	69	51	24	36	72	57	48	0
114	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0

. . . suite sur la page suivante . . .

Table A.3 – *Instance B1234-LU60-ALL-star*

	Année 6				Année 7				Année 8				Année 9			
	BH	OP	MA	CH	BH	OP	MA	CH	BH	OP	MA	CH	BH	OP	MA	CH
115	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
116	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
117	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
118	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
119	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
120	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
121	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
122	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
123	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
124	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
125	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
126	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
127	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
128	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
129	69	48	24	39	96	27	48	9	69	51	24	36	75	54	48	0
130	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
131	69	48	24	39	93	30	48	9	69	51	24	36	78	51	48	0
132	69	48	24	39	90	33	48	9	69	51	24	36	81	48	48	0
133	81	36	24	39	99	24	48	9	81	39	24	36	96	33	48	0
134	81	36	24	39	99	24	48	9	81	39	24	36	96	33	48	0
135	81	36	24	39	99	24	48	9	81	39	24	36	96	33	48	0
136	81	36	24	39	99	24	48	9	81	39	24	36	96	33	48	0

Fin des solutions de l'instance

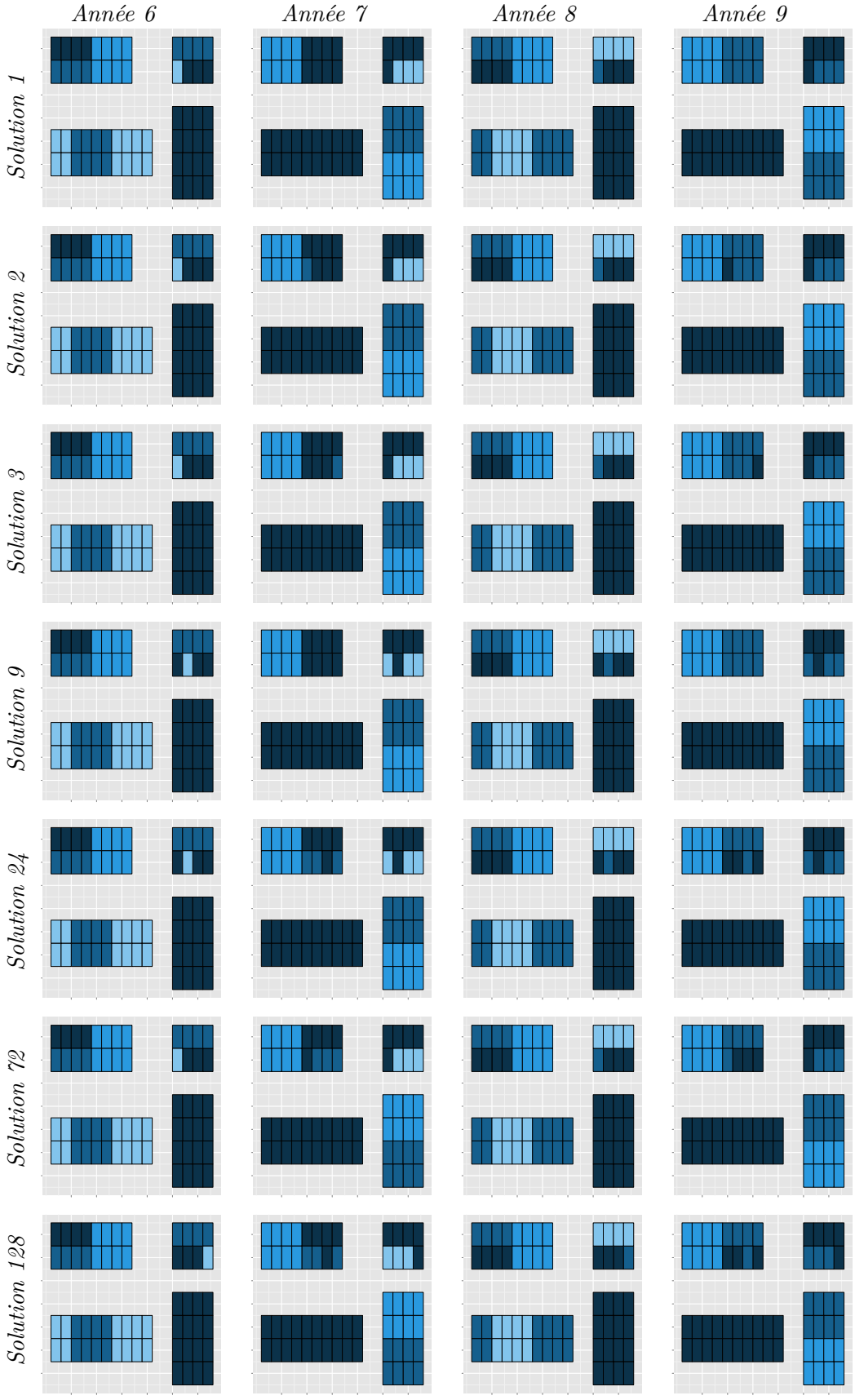


FIGURE A.3 – *7 des 136 des solutions de l'instance B1234-LU60-ALL(*) :* *Blè d'hivers,* *Orge de printemps,* *Maïs,* *Colza d'hivers*

A.4 Langage de description du domaine

En nous basant sur la représentation pddl proposé par Fox et Long (2003), on distinguera deux façons de représenter les tâches primitives. La première permet de représenter les tâches primitives non duratives. La seconde permet de représenter les tâches primitives duratives.

A.4.1 Représentation de tâches primitives non duratives

Les tâches primitives non duratives s'expriment par <action-def> décrit ci-dessous.

```
(<action-def>   ::= (:action  <action-symbol>
                              :parameters (<typed list (variable)>)
                              <action-def body>
)
```

La tâche est définit par le mot clé : *action*, un nom <name> un ensemble de paramètres : *parameters* puis le corps <action-def body> de la tâches.

```
<action-symbol>       ::= <name>
<action-def body>     ::= [:precondition <GD>]
                          [:effect <effect>]
```

Dans le corps, le mot clé : *precondition* permet de définir les conditions d'activation de la tâches. : *precondition* est associé à un <GD> (goal descriptor) défini ci dessous.

```
<GD>                  ::= () | (atomic formula(term)) | <literal(term)>
                          | (and <GD>*) | (or <GD>) | (imply <GD> <GD>)
                          | (exists (<typed list (variable)>*) <GD>)
                          | (forall (<typed list (variable)>*) <GD>)
                          | <f-comp>  ;;fluents

<atomic formula(t)> ::= (<predicate> t*)
<literal(t)>        ::= <atomic formula(term)> |
                        (not <atomic formula(term)>)

<term>                ::= <name> | <variable>
<variable>            ::= ?<name>
<predicate>           ::= <name>
<f-comp>              ::= (<binary-comp> <f-exp> <f-exp>)
<binary-comp>         ::= > | < | = | >= | <=
<f-exp>               ::= (<binary-op> <f-exp> <f-exp>) | <number> |
                          (- <f-exp> ) | <f-head>
<binary-op>           ::= + | - | * | /
<f-head>              ::= (<function-symbol> <term>*) | <function-symbol>
<function-symbol>     ::= <name>
```

En somme, on peut avoir des préconditions vides. Elles peuvent contenir des prédicats, des formules, des quantificateurs existentiels, universels, des expressions comportant des opérateurs (comparaison, arithmétique) binaires (>,<,=,>=,<=, +, -, *, /), des opérateurs logiques (and, or, imply).

Les effets d'une tâche primitive non durative sont exprimé par le mot clé : $effect$ auquel est associé <effect>, décrit ci dessous.

```
<effect>              ::= () | (and <c-effect>*) | <c-effect>

<c-effect>            ::= (forall (<variable>*) <effect>) | <c-effect>
                          | (when <GD> <cond-effect> |
                          <p-effect>
```

```
<cond-effect>          ::= (and <p-effect>*) | <p-effect>

<p-effect>             ::= (<assign-op> <f-head> <f-exp>)
                           | (not <atomic formula(term)>)
                           | <atomic formula(term)>

<assign-op>            ::= assign | scale-up | scale-down
                           | increase | decrease
```

Une action peut ne pas avoir des effets. De la même manière, les effets d'une action peuvent être exprimés par un quantificateur universel. Les effets conditionnels sont exprimés par le mot clé *when* auquel on associe une condition de la forme <GD>. Enfin, un effet peut être exprimé comme une conjonction d'effet. Dans la description des effets, le mot clé *assign* est un opérateur d'assignation simple. Les opérateurs $scale-up$, $scale-down$, *increase* et *decrease* sont respectivement les équivalents de $*=$, $/=$, $+=$ et $-=$ en C.

A.4.2 Représentation de tâches primitives duratives

Les tâches primitives duratives s'exprime par <durative-action-def> décrit ci-dessous.

```
(<durative-action-def>  ::= (:durative-action  <da-symbol>
                              :parameters (<typed list (variable)>)
                              <da-def body>
)
```

```
<da-symbol>                ::= <name>
<da-def body>              ::= :duration <duration-constraint>
                               :condition <da-GD>
                               :effect <da-effect>
```

Dans une tâche durative, les conditions d'exécution définissent à la fois les conditions d'activations, les conditions d'ouverture de fermeture, la durée. La description native en pddl de tâches primitives est reduite à des conjonctions d'expression temporelles défini comme ci dessous.

```
<da-GD>                ::= () | <timed-GD> | (and <timed-GD>+)
<timed-GD>             ::= (at <time-specifier> <GD>)
                             |(over <interval> <GD>)
<time-specifier>       ::= start | end
<interval>             ::= all
```

La durée d'une tâche peut être égale à une expression ou contrainte par une inégalité. Les contraintes permettent d'imposer des limites temporelles sur la durée de la tâche. Ces contraintes peuvent être explicitement rattachées à début ou à la fin de la tâche, indiquant ainsi le contexte d'évaluation des contraintes. Ainsi, la durée d'une tâche durative peut être exprimée comme ci dessous.

```
<duration-constraint>             ::= ()|(and <simple-duration-constraint
    >+)
                                        | <simple-duration-constraint>

<simple-duration-constraint> ::= (<d-op> ?duration <d-value>)
                                    | (at <time-specifier>
                                          <simple-duration-constraint
                                              >)
<d-op>                           ::= <= | >= | =
<d-value>                        ::= <number> | <f-exp>
```

Les effets d'une tâche duratives expriment en plus de ceux de tâches primitives non duratives l'instant (start, end) auquel l'effet est associé.

```

<da-effect>       ::= () | (and <da-effect>*) | <timed-effect>
                         | (forall (<variable>* ) <da-effect>)
                         | (when <da-GD> <timed-effect>)
                         | (<assign-op> <f-head> <f-exp-da>)

<timed-effect>   ::=  (at <time-specifier> <a-effect>)
                       | (at <time-specifier> <f-assign-da>)
                       | (<assign-op-t> <f-head> <f-exp-t>)

<f-assign-da>    ::= (<assign-op> <f-head> <f-exp-da>)

<f-exp-da>       ::= (<binary-op> <f-exp-da> <f-exp-da>)
                       | (- <f-exp-da>)| ?duration | <f-exp>
```

A.4.3 Les tâches composées

```
 (<compound-task-def> ::=      (:compound-task <ct-symbol>
                                 :parameters (<typed list (variable)>)
                                 <ct-def body>
 )
```

```
<ct-symbol>                  ::= <name>
<ct-def body>                ::= :methods (<desc-method>*)
<desc-method>                ::= (:method <m-symol>
                                       :parameters (<typed list (variable)>
                                          )
                                       <m-def body>)
```

```
<m-symbol>                   ::= <name>
<m-def body>                 ::= :condition <m-GD>
                                  :tasks (<task-def>*)

<task-def>                   ::= <task>|
                                   | (when [<def-constraint>] <task>)
                                   | (forall (<variable>*) <task>)
<def-constraint>             ::=  (and <constraint>*) | (not <constraint>)
<constraint>                 ::=  (<binary-comp> <c-exp> <c-exp>)
<c-exp>                      ::=  (<binary-op> <c-exp> <c-exp>) | <number>
                                   | (<binary-comp> (<const-head> <task>)
                                       <c-exp>)
                                   | (<time-specifier> <task>)
<time-specifier>             ::=  ?start | ?end
<const-head>                 ::=  start | end
<task>                       ::=  <action-symbol> | <da-symbol> | <ct-
    symbol>
```

A.4.4 Exemple d'un domaine décrivant l'ITK de Maïs

```
 ;; Opération de semis du maïs
(:durative-action Sowing-MA
 :parameters(?p - Plot ?cdate - Date ?tractor2 - Tractor2 ?worker - Worker
     ?seeddrill - Seeddrill)
 ;; Durée de l'action
 :duration (= ?duration (* (area ?p) (speed-sowing ?tractor2)) )
 :condition (and
```

```
    ;; Prédicat indiquant si le semis est réalisable
    (RuleSowing)
    ;; Contrainte temporelle d'ouverture de l'action
   (at start (and (>= (date) 99 ) (<= (date)  +oo )))
   ;; Contrainte temporelle de fermeture de l'action
    (at end (and (>= (date) -oo ) (<= (date)  129 )))
   )
  :effect (and
       ;; Consommation des ressources: un ouvrier, un tracteur et un
           semoir
       (at start (decrease ?tractor2 1))
       (at start (decrease ?worker 1))
       (at start (decrease ?seeddrill 1))
       ;; Production des ressources: un ouvrier, un tracteur et un semoir
       (at end (increase ?tractor2 1))
       (at end (increase ?worker) 1)
       (at end (increase ?seeddrill 1))
   )
)
```

```
  ;; Opération de labour du maïs
(:durative-action Plowing-MA
 :parameters(?p - Plot ?cdate - Date ?tractor1 - Tractor1 ?worker - Worker
     ?plow - Plow)
 :duration (= ?duration (* (area ?p) (speed-plowing ?tractor1)) )
 :condition (and
    (RulePlowing)
    (at start (and (>= (date) 99 ) (<= (date)  +oo )))
    (at end (and (>= (date)  -oo ) (<= (date)  129 )))
   )
  :effect (and (at start (decrease ?tractor1 1))
       (at start (decrease ?worker 1))
       (at start (decrease ?plow 1))
       (at end (increase ?tractor1 1))
       (at end (increase ?worker 1))
       (at end (increase ?plow 1))
     )
)
```

```
  ;; Opération de fertilisation du maïs
(:durative-action Fertilization-MA1
 :parameters(?p - Plot ?cdate - Date ?tractor2 - Tractor2 ?worker - Worker
     ?spray - Spray)
 :duration (= ?duration (* (area ?p) (speed-fertilization ?tractor2)) )
 :condition (and
    (RuleFertilization)
    (at start (and (>= (date) -oo ) (<= (date)  +oo )))
    (at end (and (>= (date) -oo ) (<= (date)  +oo )))
   )
  :effect (and (at start (decrease ?tractor2 1))
       (at start (decrease ?worker 1))
       (at start (decrease ?spray 1))
       (at end (increase ?tractor2 1))
       (at end (increase ?worker 1))
       (at end (increase ?spray 1))
     )
)
```

```
  ;; Opération de fertilisation du maïs
```

```
(:durative-action Fertilization-MA2
 :parameters(?p - Plot ?cdate - Date ?tractor2 - Tractor2 ?worker - Worker
      ?spray - Spray)
 :duration (= ?duration (* (area ?p) (speed-fertilization ?tractor2)) )
 :condition (and
    (RuleFertilization)
    (at start (and (>= (date) 154 ) (<= (date)  +oo )))
    (at end (and (>= (date) -oo ) (<= (date)  174)))
   )
  :effect (and (at start (decrease ?tractor2 1))
       (at start (decrease ?worker 1))
       (at start (decrease ?spray 1))
       (at end (increase ?tractor2 1))
       (at end (increase ?worker 1))
       (at end (increase ?spray 1))
     )
)
```

```
  ;; Opération de fertilisation du maïs
(:durative-action Fertilization-MA22
 :parameters(?p - Plot ?cdate - Date ?tractor2 - Tractor2 ?worker - Worker
      ?spray - Spray)
 :duration (= ?duration (* (area ?p) (speed-fertilization ?tractor2)) )
 :condition (and
    (RuleFertilization)
    (at start (and (>= (date) 124 ) (<= (date)  +oo)))
    (at end (and (>= (date) -oo) (<= (date)  154)))
   )
  :effect (and (at start (decrease ?tractor2 1))
       (at start (decrease ?worker 1))
       (at start (decrease ?spray 1))
       (at end (increase ?tractor2 1))
       (at end (increase ?worker 1))
       (at end (increase ?spray 1))
     )
)
```

```
  ;; Opération de désherbage du maïs
(:durative-action Weeding-MA
 :parameters(?p - Plot ?cdate - Date ?tractor2 - Tractor2 ?worker - Worker
      ?spray - Spray)
 :duration (= ?duration (* (area ?p) (speed-weeding ?tractor2)) )
 :condition (and
    (RuleWeeding)
    (at start (and (>= (date) -oo ) (<= (date)  +oo )))
    (at end (and (>= (date) -oo ) (<= (date)  +oo )))
   )
  :effect (and (at start (decrease ?tractor2 1))
       (at start (decrease ?worker 1))
       (at start (decrease ?spray 1))
       (at end (increase ?tractor2 1))
       (at end (increase ?worker 1))
       (at end (increase ?spray 1))
     )
)
```

```
  ;; Opération d'irrigation du maïs
(:durative-action Irrigation-MA
 :parameters(?p - Plot ?cdate - Date ?waterEquipment - WaterEquipment ?
     water - Water)
```

```
 :duration (= ?duration (* (area ?p) (speed-irrigation ?waterEquipment)) )
 :condition (and
    (RuleIrrigation)
    (at start (and (>= (date) 134  ) (<= (date)  +oo )))
    (at end (and (>= (date) -oo  ) (<= (date)  313 )))
   )
  :effect (and (at start (decrease ?waterEquipment 1))
       (at start (decrease ?water 16))
       (at end (increase ?waterEquipment 1))
    )
)
```

```
  ;; Opération de récolte du maïs
(:durative-action Harvesting-MA
 :parameters(?p - Plot ?cdate - Date ?tractor2 - Tractor2 ?worker - Worker
     ?harvester - Harvester)
 :duration (= ?duration (* (area ?p) (speed-harvesting ?tractor2)) )
 :condition (and
    (RuleHarvesting)
    (at start (and (>= (date) 284  ) (<= (date)  +oo)))
    (at end (and (>= (date) -oo) (<= (date)  314 )))
   )
  :effect (and (at start (decrease ?tractor2 1))
       (at start (decrease ?worker 2))
       (at start (decrease ?harvester 1))
       (at end (increase ?tractor2 1))
       (at end (increase ?worker 2))
       (at end (increase ?harvester 1))
    )
)
```

```
  ;; Opération de déchaumage du maïs
(:durative-action Harrowing-MA
 :parameters(?p - Plot ?cdate - Date ?tractor2 - Tractor2 ?worker - Worker
     ?cultivator - Cultivator)
 :duration (= ?duration (* (area ?p) (speed-harrowing ?tractor2)) )
 :condition (and
    (RuleHarrowing)
    (at start (and (>= (date) 284  ) (<= (date)  +oo )))
    (at end (and (>= (date) -oo ) (<= (date)  319 )))
   )
  :effect (and (at start (decrease ?tractor1 1))
       (at start (decrease ?worker 1))
       (at start (decrease ?cultivator 1))
       (at end (increase ?tractor1 1))
       (at end (increase ?worker 1))
       (at end (increase ?cultivator 1))
    )
)
```

```
;; Tâche composée représentant le maïs.
(:task MA
 :parameters(?parcelle - Parcelle)
(:method minWorkingDays ;; Décomposition pour l'ITK $\downarrow travail$
:condition ()
:tasks (
   ((> ?start (end Sowing-MA) (3 +oo)) Weeding-MA_1)
   ((> ?start (end Weeding-MA_1) (0 +oo)) Fertilization-MA1)
   ((> ?start (end Fertilization-MA1) (0 +oo)) Weeding-MA_2)
```

```
10      ((> ?start (end Sowing-MA) (15 +oo)) Weeding-MA_2)
11      ((> ?start (end Sowing-MA) (15 +oo)) Fertilization-MA1)
12      ((> ?start (end Weeding-MA_2) (0 +oo)) Fertilization-MA2)
13      ((> ?start (end Weeding-MA_2) (0 +oo)) Irrigation-MA)
14      ((> ?start (end Irrigation-MA) (9 +oo)) Irrigation-MA)
15      ((> ?start (end Irrigation-MA) (3 +oo)) Harvesting-MA)
16      ((> ?start (end Fertilization-MA2) (0 +oo)) Harvesting-MA)
17  ))
18
19  (:method minChemicalInput ;; Décomposition pour l'ITK $\downarrow intrant$
20  :condition ()
21  :tasks (
22      ((> ?start (end Plowing-MA) (0 +oo)) Sowing-MA)
23      ((> ?start (end Sowing-MA) (3 +oo)) Weeding-MA_1)
24      ((> ?start (end Weeding-MA_1) (0 +oo)) Weeding-MA_2)
25      ((> ?start (end Sowing-MA) (15 +oo)) Weeding-MA_2)
26      ((> ?start (end Weeding-MA_2) (0 +oo)) Fertilization-MA22)
27      ((> ?start (end Weeding-MA_2) (0 +oo)) Irrigation-MA)
28      ((> ?start (end Irrigation-MA) (9 +oo)) Irrigation-MA)
29      ((> ?start (end Irrigation-MA) (3 +oo)) Harvesting-MA)
30      ((> ?start (end Fertilization-MA22) (0 +oo)) Harvesting-MA)
31      ((> ?start (end Harvesting-MA) (0 +oo)) Harrowing-MA)
32  ))
33
34  (:method maxYield ;; Décomposition pour l'ITK $\uparrow rendement$
35  :condition ()
36  :tasks (
37      ((> ?start (end Plowing-MA) (0 +oo)) Sowing-MA)
38      ((> ?start (end Sowing-MA) (3 +oo)) Weeding-MA_1)
39      ((> ?start (end Weeding-MA_1) (0 +oo)) Fertilization-MA1)
40      ((> ?start (end Fertilization-MA1) (0 +oo)) Weeding-MA_2)
41      ((> ?start (end Sowing-MA) (15 +oo)) Weeding-MA_2)
42      ((> ?start (end Sowing-MA) (15 +oo)) Fertilization-MA1)
43      ((> ?start (end Weeding-MA_2) (0 +oo)) Fertilization-MA2)
44      ((> ?start (end Weeding-MA_2) (0 +oo)) Irrigation-MA)
45      ((> ?start (end Irrigation-MA) (9 +oo)) Irrigation-MA)
46      ((> ?start (end Irrigation-MA) (3 +oo)) Harvesting-MA)
47      ((> ?start (end Fertilization-MA2) (0 +oo)) Harvesting-MA)
48      ((> ?start (end Harvesting-MA) (0 +oo)) Harrowing-MA)
49  )))
```

A.5 Plans opérationnels

A.5.1 Instance « B[1-4]-LU30(*) »

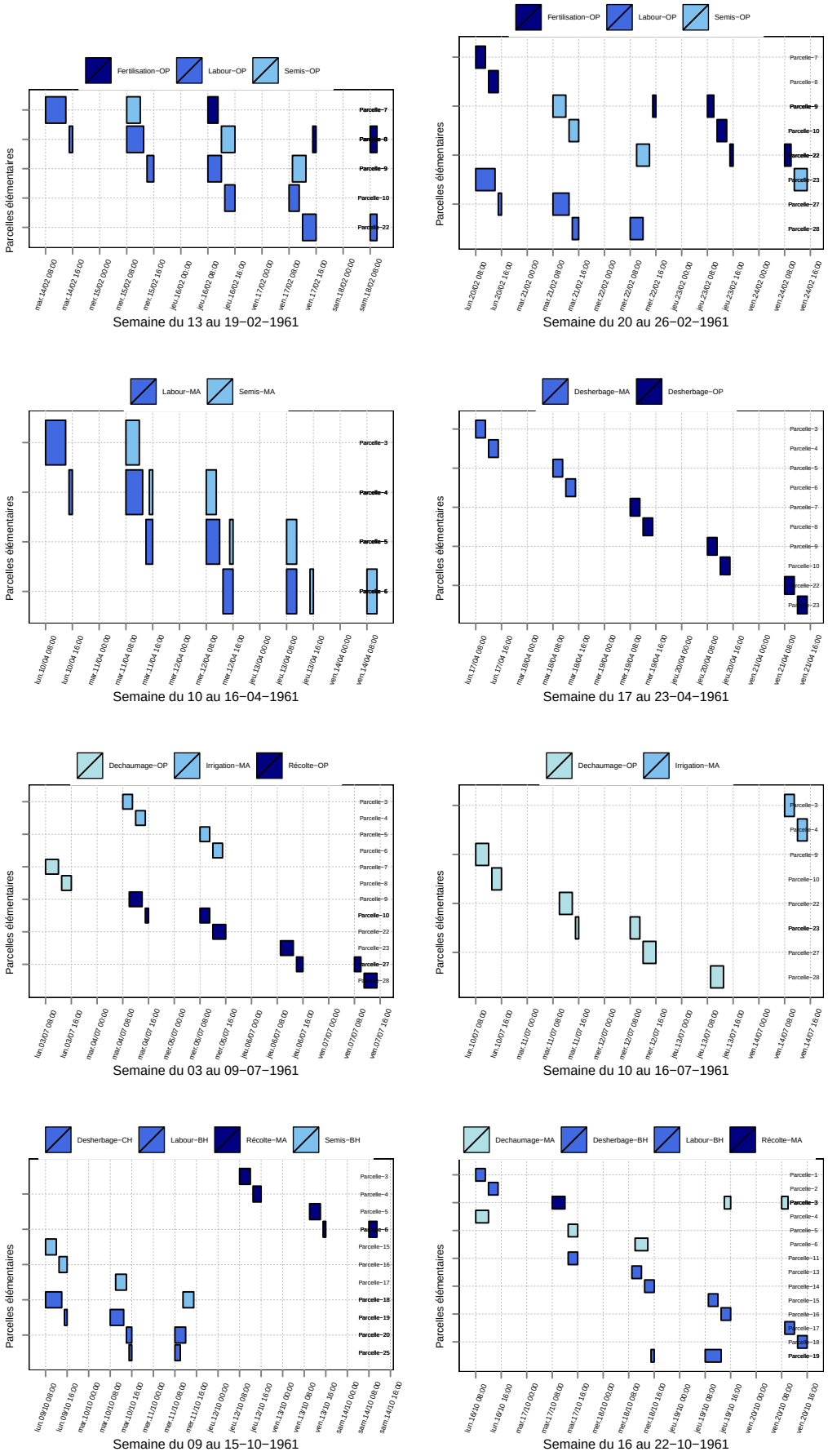


FIGURE A.4 – *Instance B[1-4]-LU30(*) : plan opérationnel obtenus pour 6 semaines de l'année 1961*

A.5.2 Instance « B[1-4]-LU60(*) »

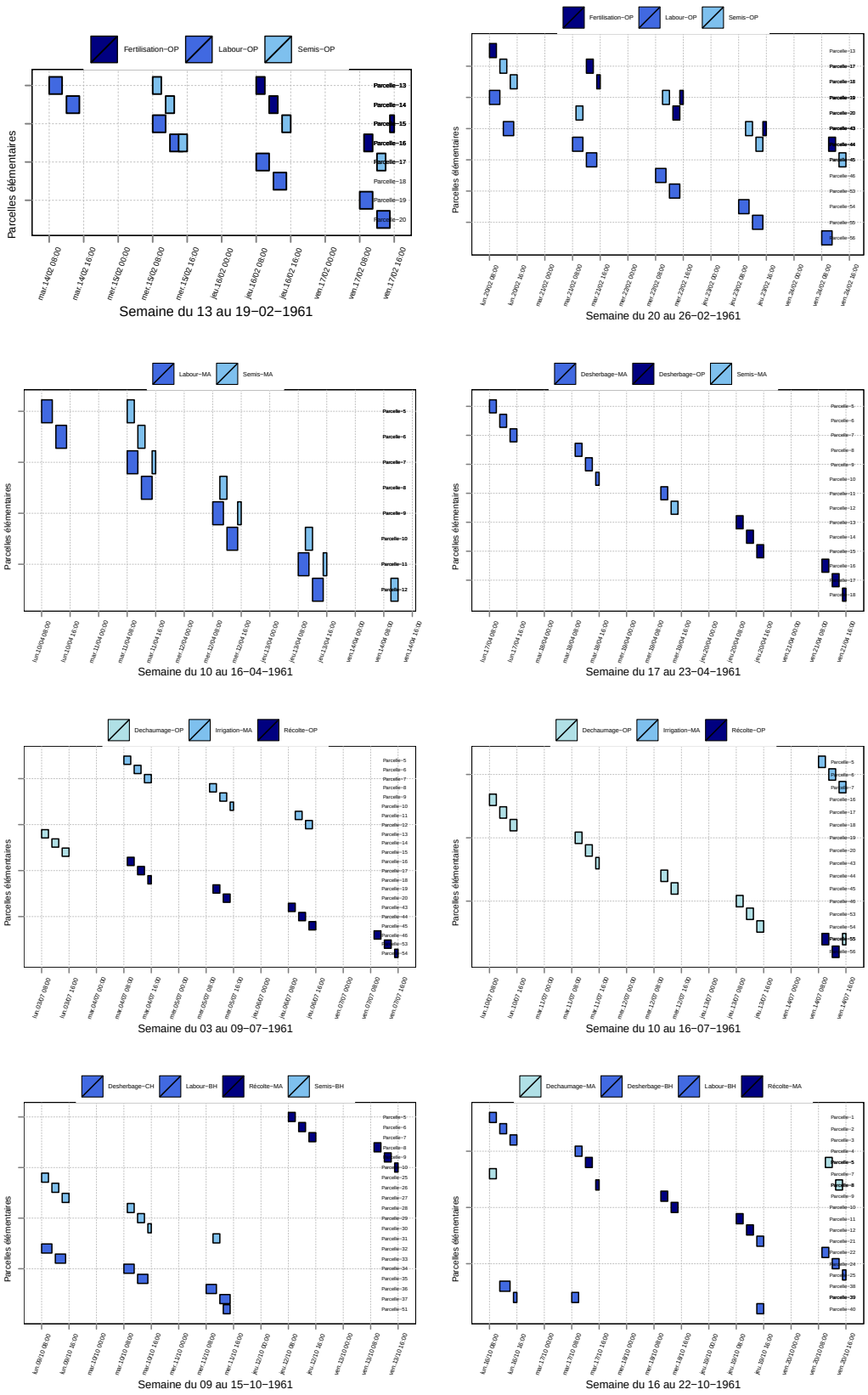


FIGURE A.5 – *Instance B[1-4]-LU60(*) : plan opérationnel obtenus pour 6 semaines de l'année 1961*

Index

état, 65, 191
évènement discrets, 64
    DEVS, 64
îlot
    structurel, 9

adventice, 234
affectation
    complète, 46
    partielle, 46
agent, 2
    délibératif, 3
    intentionnel, 3
    réactif, 3
API
    VLE, 68
arc-consistant, 49
architecture
    réactives, 53
    SPA, 52
    subsumption, 53

bag, 191, 194
bloc fonctionnel, 12

caractéristique biophysique, 9
consistance d'arc, 49
contrainte
    relaxation, 96
contrainte temporelle
    Simple Temporal Networks STN, 138, 139
    Temporal Constraint Satisfaction Problem, 138, 139
culture, 9
    monoculture, 11
    monoculture partielle, 11, 129
    système de culture, 14

délai de retour, 10, 11

effet précédent, 10

exploitation agricole, 9

filtrage, 49
fonction
    de conflit, 191, 194
    de correspondance, 42, 82, 84, 85
    de structure, 193
    de transition externe, 193
    de transition externe, 65, 191
    de transition interne, 65, 191, 193

graphe de distance, 140

hiérarchie d'abstraction, 49
HTN, 135
    Hierarchical Task Networks, 50
    méthode, 51, 135
    opérateur, 51, 136
    tâche composée, 51, 135
    tâche primitive, 51, 135

itinéraire technique
    ITK, 13
itinéraire technique, 13, 19, 20

modèle
    atomique, 64
    couplé, 64, 66
        imminent, 67, 191, 192, 236–238
        résultante, 67, 192, 236, 237
    influencé, 66
    influenceur, 66
    structure, 66

objet, 82
    composé, 82
    simple, 82
opérations agricoles, 13, 19, 20, 234

parcellaire, 9
parcelle, 9
port, 65
portée, 46

processus opérationnel, 75

représentation
    classique, 41
    variable d'état, 42
ressource, 43
    consommable, 10, 44, 145
    continue, 44
    contraintes de ressources, 145
    discrète, 44
    fonction d'extension de ressource, 147
    fonction de consommation, 43
    fonction de production, 43
    non-partageable, 44
    partageable, 44
    réutilisable, 10, 44, 145

séquence de culture, 11
    rotation, 11
STRIPS, 41, 52

tâche
    de planification, 196
    de planification, 78, 79, 213
    opérationnelle, 78, 79, 196, 213

zone cultivable, 9

Bibliographie

Mahuna Akplogan, Jerome Dury, Simon De Givry, Gauthier Quesnel, Alexandre Joannon, et Frédérick Garcia. Résolution du problème d'allocation de culture par satisfaction de contraintes pondérées. Dans *Journées Francophones de Programmation par Contraintes 2012 - JFPC*, pages 5–14, 2012a. (Cité page 88.)

Mahuna Akplogan, Jerome Dury, Simon De Givry, Gauthier Quesnel, Alexandre Joannon, Arnaud Reynaud, Jacques Eric Bergez, et Frédérick Garcia. A weighted csp approach for solving spatio-temporal planning problem in farming systems. Dans *Preferences and Soft Constraints*, pages 1–15, 2011. (Cité page 88.)

Mahuna Akplogan, Simon De Givry, Jean-Philippe Metivier, Gauthier Quesnel, Alexandre Joannon, et Frédérick Garcia. Solving the Crop Allocation Problem using Hard and Soft Constraints. *RAIRO Operations Research*, 2012b. Submited to the Special issue from a selection of papers presented in the JFPC2012. (Cité page 231.)

Mahuna Akplogan, Gauthier Quesnel, Frédérick Garcia, Alexandre Joannon, et Roger Martin-Clouaire. Towards a deliberative agent system based on devs formalism for application in agriculture. Dans *SummerSim*, pages 250–257, 2010. (Cité page 198.)

R. Alami, R. Chatila, S. Fleury, M. Ghallab, et F. Ingrand. An architecture for autonomy. *International Journal of Robotics Research*, 17 :315–337, 1998. (Cité pages 53 et 54.)

James F. Allen. Towards a general theory of action and time. *Artificial Intelligence*, 23(2) :123–154, July 1984. ISSN 0004-3702. (Cité pages 42, 138 et 204.)

David Allouche, Christian Bessière, Patrice Boizumault, Simon de Givry, Patricia Gutierrez, Samir Loudni, Jean-Philippe Métivier, et Thomas Schiex. Filtering decomposable global cost functions. Dans *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, Toronto, Canada, 2012. AAAI Press. (Cité page 129.)

Jose A. Ambros-Ingerson et Sam Steel. Integrating planning, execution and monitoring. Dans *AAAI*, pages 83–88, 1988. (Cité page 78.)

J. E Annetts et E. Audsley. Multiple objective linear programming for environmental farm planning. *Journal of the Operational Research Society*, 53(9) :933–943, 2002. ISSN 0160-5682. (Cité pages 27 et 229.)

J.M. Attonaty, M.H. Chatelin, J.C. Poussin, et L.G. Soler. Advice and decision support systems in agriculture : new issues. Dans *Farm level information systems*, pages 89–101, Woudschoten, Zeist, The Netherlands, 1993. (Cité pages 1, 28, 186 et 229.)

C. Aubry, A. Biarnes, F. Maxime, et F. Papy. Modélisation de l'organisation technique de la production dans l'entreprise agricole : la constitution de systèmes de culture dans le bassin parisien. *Etud. Rech. Syst. Agraires Dév.*, 31 :25–43, 1998a. (Cité pages 9, 10 et 11.)

C. Aubry, A. Biarnes, F. Maxime, et F. Papy. Modélisation de l'organisation technique de la production dans l'exploitation agricole : la constitution de systèmes de culture. *Etudes et Recherches sur les Systèmes Agraires et le Développement*, 31 :25–43, 1998b. (Cité pages 11 et 12.)

C. Aubry, F. Papy, et A. Capillon. Modelling decision-making processes for annual crop management. *Agricultural Systems*, 56(1) :45–65, 1998c. (Cité pages 13 et 17.)

Christine Aubry. *Gestion de la sole d'une culture dans l'exploitation agricole. Cas du blé d'hiver en grande culture dans la région picarde.* PhD thesis, NA P-G, Paris, 1995. (Cité page 15.)

Chritine Aubry. Une modélisation de la gestion de de production dans l'exploitation agricole. *Revue Française de Gestion*, 129 :32–46, 2000. (Cité page 12.)

Fahiem Bacchus et Michael Ady. Planning with resources and concurrency a forward chaining approach. Dans *Proceedings of the 17th international joint conference on Artificial intelligence - Volume 1*, pages 417–424, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1-55860-812-5, 978-1-558-60812-2. (Cité page 45.)

Johann Bachinger et Peter Zander. Rotor, a tool for generating and evaluating crop rotations for organic farming systems. *European Journal of Agronomy*, 26 (2) :130 –143, 2007. ISSN 1161-0301. (Cité pages 27 et 229.)

F. J. Barros. Dynamic Structure Discret Event System Specification : Formalism, Abstract Simulators and Applications. 13(1) :35–46, 1996. (Cité page 193.)

F. J. Barros. Modeling Formalisms for Dynamic Structure Systems. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 7 :501 – 515, october 1997. (Cité pages 80, 193, 194 et 195.)

F. J. Barros. Dynamic Structure Multiparadigm Modeling and Simulation. *ACM Transactions on Modeling and Computer Simulation*, 13(3) :259–275, 2003. (Cité page 193.)

Fernando J. Barros. Dynamic structure discrete event system specification : a new formalism for dynamic structure modeling and simulation. Dans *WSC '95 : Proceedings of the 27th conference on Winter simulation*, pages 781–785, Washington, DC, USA, 1995. IEEE Computer Society. ISBN 0-7803-3018-8. (Cité pages 193 et 194.)

Fernando J. Barros. Abstract simulators for the dsde formalism. Dans *Proceedings of the 30th conference on Winter simulation*, WSC '98, pages 407–412, Los Alamitos, CA, USA, 1998. IEEE Computer Society Press. ISBN 0-7803-5134-7. (Cité pages 194 et 228.)

Tania Bedrax-weiss, Conor McGann, et Sailesh Ramakrishnan. Formalizing resources for planning. Dans *In Proceedings of the ICAPS-03 Workshop on PDDL*, 2003. (Cité page 44.)

Nicolas Beldiceanu et Evelyne Contejean. Introducing Global Constraints in CHIP. *Mathl. Comput. Modelling*, 20(12) :97–123, 1994. (Cité page 129.)

Nicolas Beldiceanu, Irit Katriel, et Sven Thiel. Filtering algorithms for the same constraint. Dans *CPAIOR*, pages 65–79, 2004. (Cité pages 97 et 99.)

J. E. Bergez, P. Chabrier, C. Gary, M. H. Jeuffroy, D. Makowski, G. Quesnel, E. Ramat, H. Raynal, N. Rousse, D. Wallach, P. Debaeke, P. Durand, M. Duru, J. Dury, P. Faverdin, C. Gascuel-Odoux, et F. Garcia. An open platform to build, evaluate and simulate integrated models of farming and agro-ecosystems. *Environmental Modelling & Software*, 2012. (Cité pages 2, 194 et 231.)

J.-E. Bergez, P. Debaeke, J.-M. Deumier, B. Lacroix, D. Leenhardt, P. Leroy, et D. Wallach. MODERATO : an object-oriented decision tool for designing maize irrigation schedules. *Ecological Modelling*, 137(1) :43 – 60, 2001. ISSN 0304-3800. (Cité pages 1, 12, 28 et 81.)

C. Bolte, J. van Evert, et A. Lamaker. The modcom modular simulation system. *European Journal of Agronomy*, 18 :333–343, 2003. (Cité page 1.)

R. Peter Bonasso, David Kortenkamp, David P. Miller, et Marc Slack. Experiences with an architecture for intelligent, reactive agents. *Journal of Experimental and Theoretical Artificial Intelligence*, 9 :237–256, 1997. (Cité page 53.)

Blai Bonet et Héctor Geffner. Planning as heuristic search : New results. Dans *IN PROCEEDINGS OF ECP-99*, pages 360–372. Springer, 1999. (Cité page 45.)

R. Brooks. A robust layered control system for a mobile robot. *IEEE Journal on Robotics and Automation*, 2(1) :14–23, Mars 1986. ISSN 0882-4967. (Cité page 53.)

Rodney A. Brooks. Intelligence without representation. *Artif. Intell.*, 47(1-3) : 139–159, 1991. ISSN 0004-3702. (Cité page 53.)

D. G. Bullock. Crop rotation. *Critical Reviews in Plant Sciences*, 11(4) :309–309, 1992. (Cité page 11.)

Edmund Kieran Burke et Sanja Petrovic. Recent research directions in automated timetabling. *European Journal of Operational Research*, 140 :266–280, 2002. (Cité page 175.)

M.S. Castellazzi, G.A. Wood, P.J. Burgess, J. Morris, K.F. Conrad, et J.N. Perry. A systematic representation of crop rotations. *Agricultural Systems*, 97(1-2) :26 – 33, 2008. ISSN 0308-521X. (Cité pages xiv, 12, 27, 121 et 229.)

Luis A. Castillo, Juan Fernández-Olivares, Óscar García-Pérez, et Francisco Palao. Temporal enhancements of an htn planner. Dans *CAEPIA*, pages 429–438, 2005. (Cité pages 135, 144 et 154.)

Luis A. Castillo, Juan Fernández-Olivares, Óscar García-Pérez, et Francisco Palao. Efficiently handling temporal knowledge in an htn planner. Dans *ICAPS*, pages 63–72, 2006. (Cité page 144.)

P. Chabrier, F. Garcia, R. Martin-Clouaire, G. Quesnel, et H. Raynal. Toward a simulation modeling platform for studying cropping systems management : the record project. Dans *International Congress on Modelling and Simulation, International Society for Computer Simulation*, pages 10 –13, Christchurch. New Zealand., 2007. (Cité pages 2, 68 et 194.)

M. H. Chatelin, C. Aubry, J. C. Poussin, J. M. Meynard, J. Massé, N. Verjux, Ph. Gate, et X. Le Bris. DéciBlé, a software package for wheat crop management simulation. *Agricultural Systems*, 83(1) :77 – 99, 2005. ISSN 0308-521X. (Cité pages 1 et 28.)

Steve Chien. Integrated ai in space : The autonomous sciencecraft on earth observing one. Dans *AAAI*, pages 1513–1516, 2006. (Cité page 59.)

Steve Chien, Russell Knight, Andre Stechert, Rob Sherwood, et Gregg Rabideau. Using iterative repair to improve responsiveness of planning and scheduling. Dans *Proceedings of the Fifth International Conference on Artificial Intelligence Planning and Scheduling*, pages 300–307, 2000. (Cité pages xiv, 56 et 59.)

A.C.H. Chow et B.P. Zeigler. Parallel DEVS : a parallel, hierarchical, modular, modeling formalism. Dans *Proceedings of the 26th conference on Winter simulation*, pages 716–722, Orlando, Florida, United States, 1994. (Cité pages 67 et 191.)

T.H. Cormen, C.E. Leiserson, et R.L. Rivest. *Introduction to Algorithms*. Mit Electrical Engineering and Computer Science Series. Mit Press, 1990. ISBN 9780262530910. (Cité page 143.)

M.J. Cros, M. Duru, F. Garcia, et R. Martin-Clouaire. Simulating rotational grazing management. *Journal of Environment International*, page 27, 2001. (Cité page 28.)

Ken Currie et Austin Tate. O-plan : the open planning architecture. *Artif. Intell.*, 52(1) :49–86, 1991. ISSN 0004-3702. (Cité pages 50 et 146.)

Simon de Givry. *Optimisation combinatoire dans les réseaux de fonctions de coût*. Habilitation à dirigé des recherches, Université Toulouse 3 Paul Sabatier, Mars 2011. (Cité page 49.)

Simon de Givry, Matthias Zytnicki, Federico Heras, et Javier Larrosa. Existential arc consistency : getting closer to full arc consistency in weighted csps. Dans *Proceedings of the 19th international joint conference on Artificial intelligence*, IJCAI'05, pages 84–89, San Francisco, CA, USA, 2005. Morgan Kaufmann Publishers Inc. URL http://dl.acm.org/citation.cfm?id=1642293.1642307. (Cité page 88.)

Lavindra de Silva et Lin Padgham. A comparison of bdi based real-time reasoning and htn based planning. Dans *IN 17TH AUSTRALIAN JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE*, pages 1167–1173. Springer, 2004. (Cité page 137.)

Lavindra de Silva, Sebastian Sardina, et Lin Padgham. First principles planning in bdi systems. Dans *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS '09, pages 1105–1112, Richland, SC, 2009. International Foundation for Autonomous Agents and Multiagent Systems. ISBN 978-0-9817381-7-8. (Cité page 137.)

D. de Werra. An introduction to timetabling. *European Journal of Operational Research*, 19(2) :151–162, 1985. (Cité page 177.)

Rina Dechter. *Constraint processing*. Elsevier Morgan Kaufmann, 2003. ISBN 978-1-55860-890-0. (Cité pages 143 et 145.)

Rina Dechter, Itay Meiri, et Judea Pearl. Temporal constraint networks. *Artif. Intell.*, 49 :61–95, May 1991. ISSN 0004-3702. (Cité pages 43, 138, 139, 140, 143, 186 et 229.)

Olivier Despouys et François Felix Ingrand. Propice-plan : Toward a unified framework for planning and execution. Dans *Proceedings of the 5th European Conference on Planning (ECP '99)*, pages 278–293, London, UK, 1999. Springer-Verlag. ISBN 3-540-67866-2. (Cité pages 57 et 58.)

Nina K. Detlefsen. Crop rotation modelling. Dans *Proceedings of the EWDA-04 European workshop for decision problems in agriculture and natural resources. Silsoe Research Institute, England*, pages 5 –14, 2004. (Cité page 27.)

M. Bernardine Dias. A real-time rover executive based on model-based reactive planning. Dans *In The 7th International Symposium on Artificial Intelligence, Robotics and Automation in Space*, 2003. (Cité pages xiv, 61, 62, 70 et 228.)

S. Dogliotti, W. A. H. Rossing, et M. K. van Ittersum. Rotat, a tool for systematically generating crop rotations. *European Journal of Agronomy*, 19(2) :239 – 250, 2003. ISSN 1161-0301. (Cité pages 10, 11, 12, 27 et 229.)

Patrick Doherty et Jonas Kvarnström. Talplanner : A temporal logic based planner. *AI MAGAZINE*, 22 :2001, 2001. (Cité page 45.)

Colleen Doucet, Susan E Weaver, Allan S Hamill, et Jianhua Zhang. Separating the effects of crop rotation from weed management on weed density and diversity. *Weed Science*, 47(6) :729–735, 1999. (Cité page 10.)

Brian Drabble, Austin Tate, et South Bridge. The use of optimistic and pessimistic resource profiles to inform search in an activity based planner. Dans *In Proceedings of AIPS-94*, pages 243–248. AAAI Press, 1994. (Cité page 146.)

Jérôme Dury, Noémie Schaller, Frédérick Garcia, Arnaud Reynaud, et Jacques Eric Bergez. Models to support cropping plan and crop rotation decisions. a review. *Agronomy for Sustainable Development*, July 2011. ISSN 1774-0746. (Cité pages 17, 28, 100 et 230.)

Stefan Edelkamp, Jörg Hoffmann, et Michael Littman. The language for the 2004 international planning competition. http://www.tzi.de/~edelkamp/ipc-4/, 2004. (Cité page 145.)

L. Edwards, G. Richter, B. Bernsdorf, R.-G. Schmidt, et J. Burney. Measurement of rill erosion by snowmelt on potato fields under rotation in prince edward island (canada). *Agricultural Institute of Canada*, 78(3) :449—458, 1998. (Cité page 10.)

Talaat El-Nazer et Bruce A. McCarl. The choice of crop rotation : A modelling approach and case study. *American Journal of Agricultural Economics*, 68(1) : 127–136, 1986. ISSN 00029092. (Cité pages 27 et 229.)

Kutluhan Erol, James Hendler, et Dana S. Nau. HTN planning : Complexity and expressivity. Dans *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94)*, volume 2, pages 1123–1128, Seattle, Washington, USA, 1994. AAAI Press/MIT Press. URL citeseer.ist.psu.edu/24780.html. (Cité pages 45 et 50.)

Tara Estlin, Rich Volpe, Issa Nesnas, Darren Mutz, Forest Fisher, Barbara Engelhardt, et Steve Chien. Decision-making in a robotic architecture for autonomy. Dans *In Proceedings of the International Symposium on Artificial Intelligence, Robotics, and Automation in Space (iSAIRAS*, 2001. (Cité pages 55, 59, 70 et 227.)

Richard E Fikes, Peter E Hart, et Nils J Nilsson. Learning and executing generalized robot plans. *Artificial Intelligence*, 3(4) :251–288, 1972. (Cité page 52.)

Richard E. Fikes et Nils J. Nilsson. Strips : A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2(3-4) :189–208, 1971. (Cité pages 41, 44 et 52.)

Alberto Finzi, Félix Ingrand, et Nicola Muscettola. Model-based executive control through reactive planning for autonomous rovers. Dans *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, 2004. (Cité pages xiv et 63.)

R. James Firby. Task networks for controlling continuous processes. Dans *In Proceedings of the Second International Conference on AI Planning Systems*, pages 49–54, 1994. (Cité page 57.)

F Fisher, R Knight, B Engelhardt, S Chien, et N Alejandre. A planning approach to monitor and control for deep space communications. *2000 IEEE Aerospace Conference Proceedings Cat No00TH8484*, pages 311–320, 2000. (Cité pages 55, 59, 70 et 227.)

L. R. Ford et D. R. Fulkerson. Maximal Flow through a Network. *Canadian Journal of Mathematics*, 8 :399–404, 1956. (Cité page 92.)

Maria Fox et Derek Long. PDDL2.1 : an extension to PDDL for expressing temporal planning domains. *Journal of Artificial Intelligence Research*, 20 :2003, 2003. (Cité pages 44, 152 et 248.)

Jeremy Frank et Ari Jónsson. Constraint-based attribute and interval planning. *Journal of Constraints, Special Issue on Constraints and Planning*, 8 :339–364, 2003. (Cité page 138.)

Free Software Fundation. GNU Operating System : GNU's Not Unix, 1984. http://www.gnu.org. (Cité page 68.)

Erann Gat. Integrating planning and reacting in a heterogeneous asynchronous architecture for controlling real-world mobile robots. Dans *AAAI*, pages 809–815, 1992. (Cité page 53.)

Erann Gat. Esl : a language for supporting robust plan execution in embedded autonomous agents. *1997 IEEE Aerospace Conference*, pages 319–324, 1997. (Cité pages 54 et 57.)

Erann Gat. Three-layer architectures. Dans *Artificial intelligence and mobile robots*, pages 195–210. MIT Press, Cambridge, MA, USA, 1998. ISBN 0-262-61137-6. (Cité page 53.)

Alfonso Gerevini, Alessandro Saetti, et Ivan Serina. Planning through stochastic local search and temporal action graphs in lpg. *J. Artif. Intell. Res. (JAIR)*, 20 : 239–290, 2003. (Cité page 45.)

Malik Ghallab. Planification et décision. Dans J. P. Laumond, éditeur, *La robotique mobile*, Chapitre 5, pages 259–296. Hermes, 2001. (Cité page 41.)

Malik Ghallab et Hervé Laruelle. Representation and control in ixtet, a temporal planner. Dans *AIPS*, pages 61–67, 1994. (Cité pages 54 et 138.)

Malik Ghallab, Dana Nau, et Paolo Traverso. *Automated Planning : Theory & Practice*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004. ISBN 1558608567. (Cité pages 41, 42, 45, 136 et 137.)

Matt Ginsberg. *Essentials of artificial intelligence*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1994. ISBN 1-55860-221-6. (Cité page 41.)

Arturo González-Ferrer, Annette ten Teije, Juan Fernández-Olivares, et Krystyna Milian. Careflow planning : From time-annotated clinical guidelines to temporal hierarchical task networks. Dans *AIME*, pages 265–275, 2011. (Cité page 154.)

Karen Zita Haigh et Manuela M. Veloso. Planning, execution and learning in a robotic agent. Dans *In Proceedings of the Fourth International Conference on Artificial Intelligence Planning Systems*, pages 120–127. AAAI Press, 1998. (Cité page 78.)

W. K. Klein Haneveld et A. W. Stegeman. Crop succession requirements in agricultural production planning. *European Journal of Operational Research*, 166 (2) :406 – 429, 2005. ISSN 0377-2217. (Cité page 27.)

E. O Heady. The economics of rotations with farm and production policy applications. *Journal of Farm Economics*, pages 645–664, 1948. (Cité page 27.)

Malte Helmert. The fast downward planning system. *Journal of Artificial Intelligence Research*, 26 :191–246, 2006. (Cité page 45.)

Willem Jan van Hoeve, Gilles Pesant, et Louis-Martin Rousseau. On global warming : Flow-based soft global constraints. *J. Heuristics*, 12(4-5) :347–373, 2006. (Cité pages 96, 97 et 113.)

Jörg Hoffmann et Bernhard Nebel. The ff planning system : Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, 14 :2001, 2001. (Cité page 45.)

François Felix Ingrand et Frederic Py. An execution control system for autonomous robots. Dans *ICRA*, pages 1333–1338, 2002. (Cité page 54.)

François Félix Ingrand, Raja Chatila, Rachid Alami, et Frédéric Robert. Prs : A high level supervision and control language for autonomous mobile robots. Dans *In IEEE International Conference on Robotics and Automation, Mineapolis*, 1996. (Cité pages 54 et 58.)

S. Irnich et G. Desaulniers. Shortest path problems with resource constraints. Dans G. Desaulniers, Jacques Desrosiers, et M.M. Solomon, éditeurs, *Column Generation*, GERAD 25th Anniversary Series, Chapitre 2, pages 33–65. Springer, 2005. (Cité pages 112 et 147.)

Takeshi Itoh, Hiroaki Ishii, et Teruaki Nanseki. A model of crop planning under uncertainty in agricultural management. *International Journal of Production Economics*, 81-82 :555–558, 2003. ISSN 0925-5273. (Cité page 27.)

M. K. van Ittersum et R. Rabbinge. Concepts in production ecology for analysis and quantification of agricultural input-output combinations. *Field Crops Research*, 52(3) :197–208, 1997. (Cité page 12.)

A. Joannon, F. Papy, P. Martin, et V. Souchère. Planning work constraints within farms to reduce runoff at catchment level. *Agriculture, Ecosystems and Environment*, 111 :13–20, 2005a. ISSN 0167-8809. (Cité page 24.)

Alexandre Joannon. *Coordination spatiale des systèmes de culture pour la maîtrise de processus hydrologiques. Cas du ruissellement érosif dans les bassins versants agricoles du Pays de Caux, Haute Normandie.* PhD thesis, Paristech-INAPG - ED 435 Agriculture, Alimentation, Biologie, Environnements et Santé, Février 2004. UMR INRA/INA P-G SAD APT, F-78550 Thiverval Grignon, France. (Cité pages xiv, 24 et 25.)

Alexandre Joannon, V Souchère, et M Tichit. Analyse de la gestion spatialisée de l'exploitation agricole à partir de l'utilisation du parcellaire. Dans C Laurent et P Thinon, éditeurs, *Agricultures et territoires, série Aménagement et gestion du territoire*, pages 155–174. Ed. Hermès, 2005b. (Cité page 9.)

Sue Ellen Johnson et Eric Toensmeier. How expert organic farmers manage crop rotations. Dans Charles L. Mohler et Sue Ellen Johnson, éditeurs, *Crop rotation on organic farms : a planning manual*, Chapitre 2, pages 3–20. Natural Resource Agriculture and Engineering Service (NRAES) Cooperative Extension, 2009. ISBN 978-193-339-521-0. (Cité page 16.)

George Katsirelos, Nina Narodytska, et Toby Walsh. The weighted grammar constraint. *Annals OR*, 184(1) :179–207, 2011. (Cité page 129.)

W. K. Kein Haneveld et A. W. Stegeman. Crop succession requirements in agricultural production planning. *European Journal of Operational Research*, 166 : 406–429, 2005. (Cité page 27.)

Phil Kim, Brian C. Williams, et Mark Abramson. Executing reactive, model-based programs through graph-based temporal planning. Dans *Proceedings of the 17th international joint conference on Artificial intelligence - Volume 1*, pages 487–493, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1-55860-812-5, 978-1-558-60812-2. (Cité pages xiv, 60, 61 et 70.)

Philippe Laborie. Algorithms for propagating resource constraints in ai planning and scheduling : Existing approaches and new results. *Artificial Intelligence*, 143 (2) :151–188, 2003. ISSN 0004-3702. (Cité pages 44 et 146.)

Philippe Laborie et Malik Ghallab. Ixtet : an integrated approach for plan generation and scheduling. Dans *INRIA/IEEE Symposium on Emerging Technologies and Factory Automation - Volume 1*, pages 485–495, 1995. (Cité page 146.)

S. M. LaValle. *Planning Algorithms.* Cambridge University Press, Cambridge, U.K., 2006. (Cité page 41.)

P.-Y. Le Gal, C.-H. Moulin, M. Navarrete, et J. Wery. Modelling framework to design innovative agricultural production systems. Dans *Proceedings of Farming Systems Design 2007 (FSD-07)*, pages 64–65, 2007. (Cité page 74.)

Jimmy Ho-Man Lee et Ka Lun Leung. Consistency techniques for flow-based projection-safe global cost functions in weighted constraint satisfaction. *J. Artif. Intell. Res. (JAIR)*, 43 :257–292, 2012. (Cité page 88.)

Solange Lemai. *IXTET-EXEC : planning, plan repair and execution control with time and resource management*. PhD thesis, Institut National Polytechnique de Toulouse - INPT, 06 2004. 04432. (Cité pages 54, 78, 146 et 195.)

P. Leroy, J. M Deumier, et C. Jacquin. IRMA : un simulateur de l'organisation des chantiers d'irrigation. *Perspectives agricoles n. 228*, pages 76–83, 1997. (Cité pages 1 et 28.)

P. Leroy et C. Jacquin. Un logiciel pour le choix de l'assolement sur le périmètre irrigable d'une exploitation. Dans *17. Conférence régionale européenne sur l'irrigation et le drainage ICID/CIID*, pages 61–72, Varna, Bulgarie, 1994. (Cité page 27.)

B. Leteinturier, J.L. Herman, F. de Longueville, L. Quintin, et R. Oger. Adaptation of a crop sequence indicator based on a land parcel management system. *Agriculture, Ecosystems & Environment*, 112(4) :324–334, March 2006. (Cité pages 10, 11, 12, 19, 110 et 117.)

Richard Levinson. A general programming language for unified planning and control. *Artif. Intell.*, 76 :319–375, July 1995. ISSN 0004-3702. (Cité pages 57, 58 et 196.)

Roger Martin-Clouaire et Jean-Pierre Rellier. Représentation et interprétation de plans de production flexibles. Dans *Journées Francophones de Planification, Décision et Apprentissage pour la conduite de systèmes*, Toulouse, France, Mai 2006. (Cité pages 21 et 22.)

Roger Martin-Clouaire et Jean-Pierre Rellier. Modelling and simulating work practices in agriculture. *International Journal of Metadata, Semantics and Ontologies*, 2009. (Cité pages 1, 10, 28, 74, 151, 186 et 229.)

F. Maxime, J. M. Mollet, et F. Papy. Aide au raisonnement de l'assolement en grande culture. *Cah Agric*, 4 :351–62, 1995. (Cité page 17.)

B. A McCarl, W. V Candler, D. H Doster, et P. R Robbins. Experiences with farmer oriented linear programming for crop planning. *Canadian Journal of Agricultural Economics/Revue canadienne d'agroeconomie*, 25(1) :17–30, 1977. (Cité page 27.)

Drew Mcdermott, Malik Ghallab, Adele Howe, Craig Knoblock, Ashwin Ram, Manuela Veloso, Daniel Weld, et David Wilkins. PDDL - The Planning Domain Definition Language. Rapport technique, CVC TR-98-003/DCS TR-1165, Yale Center for Computational Vision and Control, 1998. (Cité pages 44 et 152.)

P Meseguer, F Rossi, et T Schiex. Soft constraints processing. Dans F Rossi, P van Beek, et T Walsh, éditeurs, *Handbook of Constraint Programming*. Elsevier, 2006. (Cité pages 47 et 88.)

Jean-Philippe Métivier, Patrice Boizumault, et Samir Loudni. Solving nurse rostering problems using soft global constraints. Dans *CP*, pages 73–87, 2009. (Cité pages 90 et 94.)

Mireille Navarrete et Marianne Le Bail. Saladplan : a model of the decision-making process in lettuce and endive cropping. *Agronomy for Sustainable Development*, 27(3) :209–221, 2007. (Cité page 231.)

U Montanari. Networks of constraints : fundamental properties and application to picture processing. *Information Science*, 7 :95–132, 1974. (Cité pages 45 et 46.)

R. C. Muchow, T. R. Sinclair, et J. M. Bennett. Temperature and Solar Radiation Effects on Potential Maize Yield across Locations. *Agron J*, 82(2) :338–343, 1990. URL http://agron.scijournals.org/cgi/content/abstract/agrojnl;82/2/338. (Cité page 81.)

R. I Muetzelfeldt et J. Massheder. The simile visual modelling environment. *European Journal of Agronomy*, 18 :345–358, 2003. (Cité page 1.)

Nicola Muscettola. Hsts : Integrating planning and scheduling. Rapport technique, Robotics Institute, Pittsburgh, PA, March 1993. (Cité page 54.)

Nicola Muscettola, Gregory A. Dorais, Chuck Fry, Richard Levinson, et Christian Plaunt. Idea : Planning at the core of autonomous reactive agents. Dans *in Proceedings of the 3rd International NASA Workshop on Planning and Scheduling for Space*, 2002. (Cité pages 61, 70 et 228.)

Nicola Muscettola, P Pandurang Nayak, Barney Pell, et Brian C Williams. Remote agent : to boldly go where no ai system has gone before. *Artificial Intelligence*, 103(1-2) :5–47, 1998. (Cité pages 53 et 56.)

D. Nau, Y. Cao, A. Lotem, et Muqoz M. Avila. Shop : Simple hierarchical ordered planner. Dans *15th International Joint Conference on Artificial Intelligence*, pages 968–973, 1999. (Cité pages 50 et 154.)

Dana Nau, Okhtay Ilghami, Ugur Kuter, J. William Murdock, Dan Wu, et Fusun Yaman. Shop2 : An htn planning system. *Journal of Artificial Intelligence Research*, 20 :379–404, 2003. (Cité pages 45, 50 et 154.)

Thomas Nesme, Françoise Lescourret, Stéphane Bellon, et Robert Habib. Is the plot concept an obstacle in agricultural sciences ? a review focussing on fruit production. *Agriculture, Ecosystems and Environment*, 138(3-4) :133 – 138, 2010. ISSN 0167-8809. (Cité page 9.)

Nils J. Nilsson. *Principles of Artificial Intelligence*. Springer, 1982. ISBN 978-3-540-11340-9. (Cité page 52.)

Christos H. Papadimitriou et Kenneth Steiglitz. *Combinatorial optimization : algorithms and complexity*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1982. ISBN 0-13-152462-3. (Cité page 143.)

Francois Papy. Interdépendance des systèmes de culture dans l'exploitation agricole. Dans E Malézieux, G Trébuil, et M Jaeger, éditeurs, *Modélisation des agrosystèmes et aide à la décision*, pages 51–74. CIRAD-INRA, collection Repères, 2001. (Cité pages 15, 16 et 23.)

Jung-Min Park, Insup Song, Young-Jo Cho, et Sang-Rok Oh. A hybrid control architecture using a reactive sequencing strategy for mobile robot navigation. Dans *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'99)*, pages 1279–284, 1999. (Cité page 53.)

Edwin P. D. Pednault. ADL and the State-Transition Model of Action. *Journal of Logic and Computation*, 4 :467–512, 1994. (Cité page 44.)

Gilles Pesant. A regular language membership constraint for finite sequences of variables. Dans *CP*, pages 482–495, 2004. (Cité pages 89, 90, 91 et 105.)

Michael Pinedo. Interval scheduling, reservations, and timetabling. Dans Thomas V. Mikosch, Sidney I. Resnick, et Stephen M. Robinson, éditeurs, *Planning and Scheduling in Manufacturing and Services*, Springer Series in Operations Research and Financial Engineering, pages 205–228. Springer New York, 2005. ISBN 978-0-387-27399-0. (Cité page 177.)

Gauthier Quesnel. *Approche formelle et opérationnelle de la multi-modélisation et de la simulation des systèmes complexes – Apports pour les systèmes multi-agents*. Informatique, École doctorale de l'Université du Littoral - Côte d'Opale, 2006. URL http://vle.univ-littoral.fr/fr/index.php/Publications. (Cité pages 2, 68 et 70.)

Gauthier Quesnel, Raphaël Duboz, et Éric Ramat. The Virtual Laboratory Environment – An operational framework for multi-modelling, simulation and analysis of complex dynamical systems. *Simulation Modelling Practice and Theory*, 17 :641–653, April 2009. (Cité pages 2, 68, 70 et 194.)

Gregg Rabideau, Russell Knight, Steve Chien, Alex Fukunaga, et Anita Govindjee. Iterative repair planning for spacecraft operations using the aspen system. Dans *In Proceedings of the International Symposium on Artificial Intelligence Robotics and Automation in Space (ISAIRAS*, 1999. (Cité pages 45 et 59.)

A. S. Rao et M. P. Georgeff. BDI-agents : from theory to practice. Dans *Proceedings of the First Intl. Conference on Multiagent Systems*, San Francisco, 1995. (Cité page 77.)

Patrick Riley et George Riley. SPADES — a distributed agent simulation environment with software-in-the-loop execution. Dans S. Chick, P. J. Sánchez, D. Ferrin, et D. J. Morrice, éditeurs, *Winter Simulation Conference Proceedings*, volume 1, pages 817–825, 2003. (Cité page 67.)

M. D. A. Rounsevell. A review of soil workability models and their limitations in temperate regions. *Soil Use and Management*, 9(1) :15–20, 1993. ISSN 1475-2743. (Cité page 24.)

Stuart J. Russell et Peter Norvig. *Artificial Intelligence - A Modern Approach (3. internat. ed.)*. Pearson Education, 2010. ISBN 978-0-13-207148-2. (Cité pages 3 et 41.)

Jean-Charles Régin. Generalized arc consistency for global cardinality constraint. Dans *Proceedings of the thirteenth national conference on Artificial intelligence - Volume 1*, AAAI'96, pages 209–215. AAAI Press, 1996. (Cité pages 93, 94, 95 et 112.)

Earl D. Sacerdoti. Planning in a hierarchy of abstraction spaces. *Artif. Intell.*, 5 (2) :115–135, 1974. (Cité page 49.)

Earl D. Sacerdoti. The nonlinear nature of plans. Dans *IJCAI*, pages 206–214, 1975. (Cité page 49.)

Ruhul Sarker et Tapabrata Ray. An improved evolutionary algorithm for solving multi-objective crop planning models. *Computers and Electronics in Agriculture*, 68(2) :191–199, 2009. (Cité page 27.)

Bernd Schattenberg et Adelinde M. Uhrmacher. Planning agents in james. *Proceedings of the IEEE*, 89(2) :158–173, 2001. (Cité page 68.)

Thomas Schiex. Arc consistency for soft constraints. Dans *Proceedings of Principles and Practice of Constraint Programming - CP*, pages 411–424, 2000. (Cité page 49.)

Thomas Schiex, Hélène Fargier, et Gerard Verfaillie. Valued Constraint Satisfaction Problems : Hard and Easy Problems. Dans Chris Mellish, éditeur, *IJCAI'95 : Proceedings International Joint Conference on Artificial Intelligence*, Montreal, 1995. (Cité pages 46 et 47.)

Alexander Schrijver. *Combinatorial Optimization - Polyhedra and Efficiency*. Springer, 2003. (Cité page 92.)

Michel Sebillotte. Le concept de modèle général et la prise de décision dans la conduite d'une culture. *C.R. Académie d'Agriculture Française*, 129 :71–80, 1988. (Cité page 15.)

Michel Sebillotte. Système de culture, un concept opératoire pour les agronomes. *Un point sur les systèmes de culture. Paris : Inra éditions*, pages 165–190, 1990. (Cité pages 10, 11, 12, 13 et 14.)

Michel Sebillotte et Louis-G Soler. Le concept de modèle général et la compréhension du comportement de l'agriculteur. *C.R. Académie d'Agriculture française*, pages 59–70, 1974. (Cité page 13.)

Jeremy G. Siek, Lee-Quan Lee, et Andrew Lumsdaine. *The Boost Graph Library : User Guide and Reference Manual*. Addison-Wesley, 2002. URL http://www.awprofessional.com/title/0201729148. (Cité pages 163 et 168.)

Reid Simmons et David Apfelbaum. A task description language for robot control. Dans *in Proceedings of the Conference on Intelligent Robots and Systems (IROS)*, 1998. (Cité pages 56, 57 et 59.)

Reid Simmons, Sanjiv Singh, Frederik Heger, Laura M. Hiatt, Seth C Koterba, Nicholas Melchior, et Brennan Peter Sellner. Human-robot teams for large-scale assembly. Dans *Proceedings of the NASA Science Technology Conference 2007 (NSTC-07)*, 2007. (Cité page 53.)

Stephen Smith et Marcel Becker. An ontology for constructing scheduling systems. Dans *Working Notes of 1997 AAAI Symposium on Ontological Engineering*. AAAI Press, March 1997. (Cité pages 10 et 44.)

Snow et Lovatt. A general planner for agro-ecosystem models. *Comput. Electron. Agric.*, 60(2) :201–211, 2008. ISSN 0168-1699. (Cité page 28.)

B. Stroustrup. *The C++ Programming Language*. Addison Wesley, 1986. (Cité page 68.)

A. Tate. Generating project networks. Dans *Proceedings of the Fifth International Joint Conference on Artificial Intelligence (IJCAI-77)*, pages 888–893, 1977. (Cité page 49.)

Austin Tate, Brian Drabble, et Richard Kirby. O-plan2 : an open architecture for command, planning and control. Dans *Intelligent Scheduling*, pages 213–239. Morgan Kaufmann, 1994. (Cité page 146.)

A. Uhrmacher et B. Kullick. Plug and Test Software Agents in Virtual Environments. Dans *WSC '00 : Proceedings of the 32nd conference on Winter simulation*, pages 1722–1729, San Diego, CA, USA, 2000. Society for Computer Simulation International. ISBN 0-7803-6582-8. (Cité pages 67 et 70.)

A. M. Uhrmacher. Dynamic structures in modeling and simulation : a reflective approach. *ACM Trans. Model. Comput. Simul.*, 11 :206–232, April 2001. ISSN 1049-3301. (Cité page 193.)

S. A. Vere. Planning in time : Windows and durations for activities and goals. Dans J. Allen, J. Hendler, et A. Tate, éditeurs, *Readings in Planning*, pages 297–318. Kaufmann, San Mateo, CA, 1990. (Cité page 50.)

Steven A. Vere. Planning in time : Windows and durations for activities and goals. *IEEE Trans. Pattern Anal. Mach. Intell.*, 5 :246–267, Mars 1983. ISSN 0162-8828. (Cité page 42.)

Marc Vilain, Henry Kautz, et Peter Beek. Constraint propagation algorithms for temporal reasoning. Dans *Readings in Qualitative Reasoning about Physical Systems*, pages 377–382. Morgan Kaufmann, 1986. (Cité pages 43, 138 et 204.)

R. Volpe, I. Nesnas, T. Estlin, D. Mutz, R. Petras, et H. Das. The claraty architecture for robotic autonomy. *In Proceedings of the 2001 IEEE Aerospace Conference*, March 2001. (Cité pages 55 et 227.)

R Volpe, Issa Nesnas, T Estlin, D Mutz, R Petras, et H Das. Claraty : Coupled layer architecture for robotic autonomy. Rapport technique, D-19975, Jet Propulsion Laboratory, 2000. (Cité pages 55 et 227.)

Eg Wijnands. Crop rotation in organic farming : theory and practice. Dans *Designing and testing crop rotations for organic farming Proceedings from an international workshop*, pages 21–35. Danish Research Centre for Organic Farming, 1999. (Cité page 17.)

David E. Wilkins. Domain-independent planning : representation and plan generation. *Artif. Intell.*, 22(3) :269–301, 1984. ISSN 0004-3702. (Cité pages 50 et 146.)

David E. Wilkins. *Practical planning : extending the classical AI planning paradigm*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988. ISBN 0-934613-94-X. (Cité pages 58 et 146.)

Brian C. Williams, Seung Chung, et Vineet Gupta. Mode estimation of model-based programs : monitoring systems with complex behavior. Dans *Proceedings of the 17th international joint conference on Artificial intelligence - Volume 1*, IJCAI'01, pages 579–585, San Francisco, CA, USA, 2001a. Morgan Kaufmann Publishers Inc. ISBN 1-55860-812-5, 978-1-558-60812-2. (Cité page 60.)

Brian C. Williams, Phil Kim, Michael Hofbaur, Jonathan How, Jon Kennell, Jason Loy, Robert Ragno, John Stedl, et Aisha Walcott. Model-based reactive programming of cooperative vehicles for mars exploration. Dans *PROC. OF THE INTERNATIONAL SYMPOSIUM ON ARTIFICIAL INTELLIGENCE, ROBOTICS AND AUTOMATION IN SPACE*, 2001b. (Cité pages 60 et 140.)

Brian C. Williams, P. Pandurang Nayak, et Urang Nayak. A model-based approach to reactive self-configuring systems. Dans *In Proceedings of AAAI-96*, pages 971–978, 1996. (Cité page 54.)

Håkan L. S. Younes et Reid G. Simmons. On the role of ground actions in refinement planning. Dans *AIPS*, pages 54–62, 2002. (Cité page 138.)

Bernard P. Zeigler. *Theory of Modeling and Simulation*. John Wiley, 1976. (Cité pages 64 et 70.)

Bernard P. Zeigler, Tag Gon Kim, et Herbert Praehofer. *Theory of modeling and simulation : Integrating Discrete Event and Continuous Complex Dynamic Systems*. Academic Press, 2000. (Cité pages 2, 64, 67, 70 et 236.)

Titre Approche modulaire pour la planification continue. *Application à la conduite des systèmes de culture*

Résumé Au cours des dernières années, dans l'optique de faire face aux changements environnementaux et économiques, les chercheurs en agronomie ont développé de nombreux modèles conceptuels relatifs à la conduite des systèmes de production agricoles. La simulation et le contrôle de ces systèmes permettent de reproduire les dynamiques d'interaction entre des entités autonomes et hétérogènes comprenant un agent (agriculteur). Ces travaux sont essentiels aujourd'hui où l'utilisation de la simulation devient de plus en plus prépondérante dans la conception des systèmes complexes tels que sont les systèmes de production agricole. Dans le cadre de cette thèse, nous nous sommes intéressés à la conception des systèmes complexes autonomes opérant dans un environnement dynamique et incertain. Nous expliquons que la conduite des systèmes de culture d'une exploitation agricole est un problème multi-échelles spatiales et temporelles. La structure de l'exploitation étant fixée, elle intègre trois types de décision qui sont : l'allocation des cultures (à long terme), choix du mode de conduite des cultures, les itinéraires technique (à moyen terme) et l'ordonnancement des opérations agricoles journalières (à court terme). Ces problèmes de décision étant de nature différente, nous avons développé pour chacun d'eux des méthodes de planification spécifiques. Nous proposons d'aborder l'allocation des cultures comme un problème de satisfaction de contraintes pondérées (WCSP) où l'utilité de l'allocation est évaluée par une fonction de coût globale. Les modes de conduite des cultures étant prédéfinis, nous utilisons une approche de planification temporelle et hiérarchique (HTN et STN) dans laquelle l'heuristique de décomposition est une fonction globale permettant ainsi, de prendre en compte les inter-dépendances entre les effets de chaque mode de conduite. Enfin, l'approche que nous proposons pour l'ordonnancement des opérations agricoles est basée sur un modèle de programmation linéaire. Pour appréhender ces différents problèmes de décision, nous proposons une architecture systémique nommée « Safihr ». Celle-ci est capable de prendre en compte l'entrelacement en ligne de plusieurs planificateurs spécifiques. Cette architecture repose sur le cadre des systèmes à événements discrets (DEVS). L'agent agriculteur est vu comme un système hiérarchique , dynamique et distribué en interaction avec son environnement physique. Chacun des planificateurs est vu comme un système de contrôle indépendant. Safihr intègre les mécanismes permettant de faire coopérer différents planificateurs spécifiques aux sein d'un même système. Cette architecture a été testée sur une exploitation virtuelle de 180 ha, cultivant 4 cultures annuelles, chacune pouvant être cultivée selon 3 itinéraires techniques différents.

Mots-clés Systèmes à événement discrets, CSP pondérés, Planification hiérarchique temporelle, Allocation de culture, Conduite des systèmes de culture, Agent orienté simulation

Title Modular approach for continuous planning. *Application to cropping systems management*

Abstract During the last years, in order to deal with environmental and economical changes, agronomists have developed several conceptual models of farming systems management. Simulation of farming systems enable to reproduce the dynamic of interactions between autonomous and heterogeneous entities including a farmer agent. This type of research is essential today since simulation is leading the design of complex systems such as a cropping systems. In this work, we address the challenge of the development of autonomous complex systems operating in a dynamic and uncertain environment. We only consider a farm with a cropping system and a determined structure. We explain that decision-making in farming systems is a complex issue in which decisions are joined up, through various spatial (from farm to plots) and temporal (from several years to several days) scales. These decisions are usually grouped in three classes: strategic, tactical and operational decisions. Strategic decisions are long-term (several years) planning problems in which, knowing biophysical and structural constraints, crops are assigned to plots over a fixed horizon. Tactical decisions are mid-term (several months) planning problems in which, knowing crops to be grown, a crop management systems is assigned to each pair of plot and crop. Finally, operational decisions are short-term (several weeks or days) scheduling problems that could be summarized as scheduling daily farm operations to timely control crop production processes. Considering, the inherent feature of these decision-making problems, we developed for each of them a specific planning technique. Strategic decisions are address as a Weighted Constraint Satisfaction Problem (WCSP) in which the relevance of crops allocation is assessed by a global objective function. Tactical decisions are address as hierarchical and temporal planning problems, based on Hierarchical Tasks Networks (HTN) and Simple Temporal Network (STN). We introduced a new decomposition heuristic into HTN framework which enable to take into account the interdependence between crops production techniques. Finally, we proposed to tackle operational decisions by using linear programming techniques. To interleave these decision-making problems, we introduce a new modular architecture call Safihr "Smulation based Architecture For Interleaving Heterogeneous decisions in Real world problems". The proposed architecture is a model-based approach relying on DEVS (Discrete EVent System specification) formalism. Thus, the farmer agent is seen as a distributed hierarchical dynamic systems which explicitly interacts with the environment. Safihr, integrate mechanisms to easily make several planners to cooperate. The architecture have been tested on a virtual farm.

Keywords Discrete events systems, Weighted CSP, Temporal hierarchical planning, Crops allocation, Cropping systems management, Agent-based simulation