Office National d'Études
et de Recherches Aérospatiales

Centre d'Études et de Recherche de l'École Nationale Supérieure
de l'Aéronautique et de l'Espace à Toulouse

Un Modèle de Systèmes Hybrides pour le Contrôle de Processus Temps-Réel Interrogeables

Martin Adelantado   Frédéric Boniol   Augusto Burgueño   Simon de Givry
Florence Pagani

Publication présentée à :

Journées Formalisation des Activités Concurrentes
Toulouse, IRIT, 7-8 février 1996.

Département d'Études et de Recherches en Informatique

2, avenue Édouard Belin - BP 4025 - 31055 Toulouse Cedex
Tél. 62.25.25.91 - Fax. 62.25.25.93

Un Modèle de Systèmes Hybrides pour le Contrôle de Processus Temps-Réel Interrogeables *

Martin Adelantado Frédéric Boniol Augusto Burgueño Simon de Givry
Florence Pagani

CERT-ONERA, Département d'Informatique,
2 av. E. Belin BP4025, 31055 Toulouse cedex, France.
e-mail: saturne@cert.fr

Résumé

M. Dean et Th. Boddy ont récemment proposé dans [5] le concept de processus *interrogeables*, *i.e.* capable de fournir un résultat approximatif à tout instant en cours d'exécution, comme moyen de répondre à la double exigence des systèmes temps-réel, à savoir respecter des échéances strictes et enchaîner des calculs potentiellement longs et coûteux. D'un point de vue opérationnel, une tâche interrogeable est pilotée au sein d'un système temps-réel par un *noyau réactif* en fonction des échéances et de la qualité de ses résultats intermédiaires. Une telle structuration en termes de noyaux réactifs et tâches interrogeables permet d'allier les propriétés de réactivité, prédictibilité et adaptativité aux échéances, mais en revanche complexifie la nature du contrôle réactif. L'objectif de cet article est de caractériser et définir formellement les mécanismes réactifs mis en œuvre pour le contrôle de tâches interrogeables. Nous définissons les *systèmes hybrides à horloges dynamiques*, comme une sous-classe des systèmes hybrides proposés par R. Alur *et al.* dans [2, 9], et montrons qu'ils répondent exactement à la problématique du contrôle réactif/interrogeable.

Mots-clefs : systèmes hybrides, systèmes réactifs, temps-réel, algorithmes interrogeables.

1 Introduction

Les prochaines générations de systèmes temps-réel devront allier deux caractéristiques généralement contradictoires : d'une part réagir en "temps-réel" et continûment à l'évolution de l'environnement externe, et d'autre part réagir "intelligemment" en échafaudant des calculs parfois longs et coûteux (voire dont la terminaison n'est pas garantie) pour répondre à des situations complexes ou imprévues.

De façon générale, tout système temps-réel peut être caractérisé par trois propriétés :

- La *réactivité* du système face à l'environnement qu'il contrôle, *i.e.* la capacité à réagir instantanément aux événements apparaissant dans l'environnement externe,
- La *ponctualité* du système face aux échéances qui lui sont imposées. Ces échéances pouvant être dynamiques, la propriété de ponctualité définit la capacité du système à s'adapter dynamiquement à de nouvelles échéances, propriété encore appelée *adaptativité aux échéances*,
- La *prédictibilité* comportementale du système. Cette propriété caractérise la capacité à prévoir le comportement du système en fonction de l'évolution de son environnement, et par suite la possibilité de prouver formellement la correction logique et temporelle de l'application.

*Une version préliminaire de cet article a été publiée au congrès AFCET "Modélisation des systèmes réactifs" Brest 28-29 mars 1996.

A cela s'ajoutent les aptitudes que l'on peut attendre d'un système embarqué évolué, comme par exemple la gestion de bases de connaissances, la fusion de données, la capacité à s'adapter à des situations nouvelles ou imprévues, ...

Historiquement, la propriété de *ponctualité* (respect des échéances) a primé sur toutes les autres et a encouragé la communauté scientifique, d'une part à orienter ses recherches vers la course aux performances de façon à réduire les temps d'exécution, et d'autre part à exclure des systèmes temps-réel tout traitement dont les temps de calcul ne sont pas maîtrisés. Cependant, il est maintenant admis que, même si l'optimisation des techniques de traitement est souhaitable, cette course aux performances ne pourra jamais par elle seule assurer la *prédictibilité* du système ni *l'adaptativité* aux échéances [12].

En réponse à ce constat, M. Dean et Th. Boddy ont récemment proposé le concept d'algorithme interrogeable, *i.e.* capable de retourner une réponse approximative à chaque instant (quel que soit le temps d'exécution effectivement consommé) et dont la qualité (ou précision) croît avec le temps [5]. Cette idée a été raffinée par Adelantado *et al* en un modèle général de système temps-réel, le modèle SATURNE, alliant réactivité, prédictibilité et adaptativité aux échéances [1]. L'idée générale de ce modèle repose sur le principe de séparation réactif/interrogeable, conjecturant que tout système temps-réel peut être décrit par la composition d'un sous-système réactif prédictible contrôlant un ensemble de tâches de traitement interrogeables. A chaque interaction avec l'environnement externe, le sous-sytème réactif réagit en produisant des réponses réflexes et/ou en activant des tâches interrogeables. Lorsque l'échéance (instant auquel il faut produire une réponse) est atteinte, et/ou lorsque la qualité des résultats en cours d'élaboration (par les tâches) est suffisante, le sous-sytème réactif interroge ces tâches qui produisent alors un résultat éventuellement approximatif. Contrairement aux systèmes temps-réel classiques pour lesquels on essaie de garantir un temps de réponse, l'idée de cette approche est de garantir à tout instant une réponse dont la qualité croît avec le temps. En ce sens, le paradigme réactif/interrogeable constitue une réponse intéressante aux applications confrontées à des échéances dynamiques ou mettant en œuvre des traitements algorithmiques complexes dont le temps de calcul n'est pas borné. Il oblige toutefois, d'une part à repenser le problème de la spécification et de l'expression des tâches algorithmiques dans un cadre interrogeable, et d'autre part à définir le comportement logique et temporel du sous-système réactif contrôlant des tâches interrogeables. Nous nous intéressons dans cet article au second point.

Une tâche interrogeable peut être vue par le sous-système réactif comme un processus continu, caractérisé par une fonction croissante qualifiant ses résultats intermédiaires en fonction du temps d'exécution effectivement consommé. D'autre part, le sous-système réactif interagit avec l'environnement externe par émission et réception de signaux discrets, et envoie des requêtes aux traitements algorithmiques. Le sous-système réactif peut donc être assimilé à un système hybride [9, 2] combinant à la fois des composantes discrètes (les signaux et les requêtes) et des composantes continues (les temps d'exécution des tâches interrogeables et leur fonction de qualité). L'objectif de cet article est de formaliser la notion de contrôle réactif/interrogeable. Nous définissons une sous-classe des systèmes hybrides, appelée *système hybride à horloges dynamiques* (SHHD), et nous montrons que la spécification d'un noyau réactif contrôlant des tâches interrogeables peut se ramener à la définiton d'un SHHD.

L'article est organisé comme suit. La section 2 présente le paradigme réactif/interrogeable, et revient en particulier sur le concept de tâche interrogeable et sur la nature du contrôle réactif. La section 3 présente le concept de système hybride et définit formellement la classe des SHHDs. Un exemple de système temps-réel mettant en œuvre des tâches interrogeables et un SHHD est présenté en section 4.

2 Le temps-réel par une approche mixte réactif/interrogeable

Le paradigme *réactif/interrogeable* est un raffinement du principe de séparation *réactif/transformationnel* proposé par David Harel et Amir Pnueli dans [7]. Selon ce paradigme, un système temps réel peut être décrit par la composition de deux sous-systèmes communicants : un noyau réactif contrôlant un sous-système transformationnel, lui même composé de tâches interrogeables [1]. Le sous-système réactif assure une interaction permanente avec l'environnement, contrôle le temps et calcule les échéances, tandis que les tâches interrogeables mettent en œuvre les traitements contraints par le temps.

Nous détaillons dans cette section le concept de tâche interrogeable et caractérisons les mécanismes réactifs nécessaires pour le contrôle de telles tâches.

Le lecteur interessé par une description plus détaillée du modèle de description mixte *réactif/interrogeable* pourra se reporter à [1].

2.1 Concept *anytime* et algorithmes interrogeables

Traitement *anytime* de l'information

Le concept de traitement *anytime* de l'information, initialement proposé par M. Dean et T. Boddy [5], est issu des travaux tentant de coupler les domaines de l'Intelligence Artificielle et du Temps-Réel. Ce concept repose sur les observations suivantes :

1. une décision de qualité dégradée mais satisfaisante (selon une métrique à définir) et élaborée à temps est souvent préférable à une décision optimale élaborée trop tard ;
2. un modèle de *réflexion-décison-action* doit capturer la notion de consommation du temps disponible pour l'exécution des traitements.

De manière générale, le concept *anytime* définit un modèle de traitement de l'information reposant sur des mécanismes réactifs et algorithmiques permettant de diriger une exécution en distribuant le temps disponible sur un ensemble de tâches, de manière à maximiser une fonction d'utilité (ou de satisfaction) donnée. Cette approche est une généralisation des notions traditionnelles de processus de traitement et de procédure algorithmique. A l'inverse des procédures classiques qui fournissent un résultat unique au bout d'un temps donné, une procédure *anytime* est capable de fournir une infinité de résultats (éventuellement non-distincts), un à chaque instant. En ce sens, une procédure *anytime* est une fonction de l'espace des entrées et de la demi-droite temporelle R^+ vers l'espace des sorties.

Algorithmes interrogeables

La notion d'algorithme interrogeable constitue sans nul doute le fondement essentiel sur lequel repose le modèle de traitement *anytime*. Un algorithme interrogeable est dans sa définition originelle, un algorithme capable de produire à tout instant un résultat de qualité croissante avec la durée de délibération dont il dispose [5]. Il est de ce fait interrogeable à tout instant par un agent réactif.

La qualité des résultats élaborés par un algorithme interrogeable peut être représentée par un *profil de qualité* en fonction du temps d'exécution effectivement consommé. Cette fonction de qualité doit être évaluée par une métrique indépendante de l'utilisation des résultats. Plusieurs métriques générales peuvent être identifiées :

- $Q(r)$ = degré de certitude de la correction du résultat r (un résultat peut être sûr, probable, possible, improbable ...) ;
- $Q(r)$ = précision du résultat approché par rapport au résultat exact, comme c'est le cas par exemple, dans les développements en séries limitées ;
- $Q(r)$ = degré de détail du résultat, en particulier lorsque le résultat est composé d'un ensemble d'attributs dont certains sont partiellement ou pas du tout instanciés.

De façon générale, on peut distinguer deux types de profil de qualité :

1. une distribution probabiliste du type $P_A(X,\delta)(q)$ définissant la probabilité que le résultat élaboré par l'algorithme interrogeable A après une durée d'exécution δ pour les données d'entrées X soit de qualité q,

2. une distribution exacte du type $Q_A(X,\delta)$ définissant pour un algorithme interrogeable A la qualité du résultat attendu après une durée d'exécution δ pour les données d'entrées X.

Nous nous limiterons dans cet article à ce dernier type de profil de qualité.

Définition 1 *Un algorithme interrogeable (et déterministe) A est défini comme une fonction (partielle)*

$$\begin{array}{rccl} f_A & : & D_A \times \mathsf{R}^+ & \rightsquigarrow & C_A \\ & & (X,\delta) & \mapsto & Y \end{array}$$

où D_A, R^+ et C_A sont respectivement l'espace des entrées, le domaine temporel dense, et l'espace des sorties de l'algorithme A. Le profil de qualité de A est défini par une fonction normée sur $[0,1]$

$$\begin{array}{rccl} Q_A & : & D_A \times \mathsf{R}^+ & \rightsquigarrow & [0,1] \\ & & (X,\delta) & \mapsto & q \end{array}$$

et croissante sur le temps, i.e. telle que :

$$\forall X \in D_A,\ \forall \delta,\delta' \in \mathsf{R}^+,\quad \delta \geq \delta' \Rightarrow Q_A(X,\delta) \geq Q_A(X,\delta')$$

□

Il existe de nombreuses techniques algorithmiques classiques appropriées au codage d'algorithmes interrogeables. Citons par exemple les techniques numériques itératives, les techniques neuronales, les heuristiques de recherche dans les arbres de décision ...

Exemple

Considérons la méthode de tri par insertion, présentée en figure 1. Le principe de cet algorithme est le suivant :

1. la liste à trier est rangée dans `liste_entree` ;

2. le résultat est construit par itérations successives dans `liste_resultat` ;

3. à chaque itération sur i, on insère le $i^{ème}$ élèment de la liste "à sa place"parmi les $i-1$ premiers.

Par construction, après chaque itération sur i, `liste_resultat` est composée par un préfixe de longueur i complètement trié, et d'un reste de longueur $n-i$ non trié. Notons $\texttt{liste\_resultat}_i$ la liste produite à la fin de l'itération i :

$$\texttt{liste\_resultat}_i = \underbrace{(a_1,\ldots,a_i)}_{\text{préfixe trié}} \cdot \overbrace{(a_{i+1},\ldots,a_n)}^{\text{reste à trier}}$$

Soit $f_{T_{inser}}$ la fonction interrogeable correspondant à l'algorithme ci-dessus, et retournant à chaque interrogation l'état courant de `liste_resultat`. On définit la qualité d'une liste partiellement triée par :

$$q(\texttt{liste}) \stackrel{\text{def}}{=} \frac{l}{length(\texttt{liste})}$$

```
read(liste_entree);
n := lenght(liste_entree);

liste_travail := liste_entree;
for(i=1; i<=n; i++)
  {
    liste_resultat := liste_travail
    for(j=2; j>=2; j--)
      {
        if liste_travail[j] < liste_travail[j-1]
        then
          {
            temp := liste_travail[j];
            liste_travail[j] := liste_travail[j-1]
            liste_travail[j-1] := temp;
          }
        else
            break;   /* on sort directement du "for(j=i; j>=2; j--)" */
        endif
      }
  }
```

Figure 1 : algorithme de tri par insertion

où l est la longueur du plus long préfixe trié de `liste`. Par construction, après chaque itération $i = k$, la longueur du plus grand préfixe trié de `liste_resultat` est au moins k. En conséquence, la qualité de `liste_resultat`, après l'itération $i = k$ et tant que l'itération $i = k + 1$ n'est pas achevée, est constante et au moins égale à k/n, où n est la longueur de la liste initiale à trier.

D'autre part, soit δ_t le temps (maximun) nécessaire à l'exécution du test :

```
if liste_travail[j] < liste_travail[j-1]
```

et soit δ_p le temps (maximun) nécessaire à l'exécution des instructions, *i.e.* à l'exécution de la permutation :

```
{
  temp := liste_travail[j];
  liste_travail[j] := liste_travail[j-1];
  liste_travail[j-1] := temp;
}
```

Notons $\delta \stackrel{\text{def}}{=} \delta_t + \delta_p$. Alors par construction, la $k^{i\grave{e}me}$ itération sur i ($i = k+1$) dure (dans le pire cas) $k\delta$ (en faisant abstraction du temps nécessaire pour réaliser l'affectation de `liste_travail` à `liste_resultat`).

On définit à l'instant t le profil de qualité du résultat de l'algorithme interrogeable T_{insert} par la qualité minimale $Q_{T_{inser}}(X, t)$ calculée comme suit :

$$\left\{ \begin{array}{ll} Q_{T_{inser}}(X,t) = \frac{k}{length(X)} & \text{si } t \in [t_k, t_{k+1}[\text{ et } t < t_n \\ Q_{T_{inser}}(X,t) = 1 & \text{si } t \geq t_{n+1} \end{array} \right. \quad \text{avec} \quad \left\{ \begin{array}{l} t_1 = 0 \\ t_k = t_{k-1} + k\delta \quad \text{pour} \quad k > 0 \end{array} \right.$$

où X est la liste à trier, et t_k désigne la date de fin d'exécution de l'itération k.

Considérons par exemple une instanciation de l'algorithme de tri par insertion sur la liste suivante :

$$\texttt{liste\_entree} \stackrel{\text{def}}{=} (2, 1, 4, 5, 3)$$

Les résultats partiels fournis au cours du temps sont :

Pendant la première itération ($i = 2$) : tant que la première itération n'est pas terminée, `liste_résultat` est identique à `liste_entrée`. La qualité de `liste_résultat` est donc $q = 1/5$. Le temps consommé par cette itération est exactement δ :

$$\forall t \in [0, \delta[\quad \texttt{liste\_resultat}(t) \; = \; (2) \cdot (1, 4, 5, 3) \quad \Rightarrow \quad q = 1/5$$

Pendant la seconde itération ($i = 3$) : à l'issue de la première itération `liste_résultat` est égal à $(1, 2, 4, 5, 3)$. Sa qualité est donc $q = 4/5$. Par définition de l'algorithme, la seconde itération dure δ_t (la durée du test) :

$$\forall t \in [\delta, \delta + \delta_t[\quad \texttt{liste\_resultat}(t) \; = \; (1, 2) \cdot (4, 5, 3) \quad \Rightarrow \quad q = 4/5$$

Pendant la troisième itération ($i = 4$) : la seconde itération n'a pas modifié `liste_résultat`. Sa qualité est donc inchangée. D'autre part, le temps comsommé par la troisième itération est également δ_t (la durée du test) :

$$\forall t \in [\delta + \delta_t, \delta + 2\delta_t[\quad \texttt{liste\_resultat}(t) \; = \; (1, 2, 4) \cdot (5, 3) \quad \Rightarrow \quad q = 4/5$$

Pendant la quatrième itération ($i = 5$) : la troisième itération n'a pas modifié `liste_résultat`. Sa qualité est donc inchangée. En revanche, le temps comsommé par la quatrième itération est $2\delta_t + \delta_t$, soit 2 cycles (test et permutation) puis un test d'arret :

$$\forall t \in [\delta + 2\delta_t, 3\delta + 3\delta_t[\quad \texttt{liste\_resultat}(t) \; = \; (1, 2, 4, 5) \cdot (3) \quad \Rightarrow \quad q = 4/5$$

Fin de l'algorithme : le résultat final est enfin atteint à l'issue de la quatrième itération :

$$\forall t \in [3\delta + 3\delta_t, \infty[\quad \texttt{liste\_resultat}(t) \; = \; (1, 2, 3, 4, 5) \quad \Rightarrow \quad q = 1$$

Le profil réel de la qualité du résultat intermédiaire produit au cours de cette exécution, ainsi que le profil de qualité minimale, sont représentés en figure 2 (nous avons supposé pour simplifier la figure que $\delta_t = \delta_p$).

Par soucis de simplification, nous nous limiterons dans la suite de l'article à des algorithmes interrogeables dont le profil de qualité est une fonction du type $Q_A(t)$ ne dépendant pas des données d'entrées.

2.2 Le contrôle réactif/interrogeable

Le concept de tâche interrogeable permet d'envisager de nouvelles stratégies de contrôle temps réel. Par définition, une tâche interrogeable étant consultable à tout instant, le sous-système réactif peut piloter les processus de traitement en fonction de :

leurs échéances : une tâche ayant dépassé le temps qui lui était imparti (éventuellement variable et dépandant de l'environnement), est interrompue pour remettre son résultat (échéance "dure") ;

la qualité de leurs résultats : une tâche ayant déjà produit un résultat dont la qualité dépasse un certain seuil (éventuellement variable), est interrompue pour remettre son résultat ;

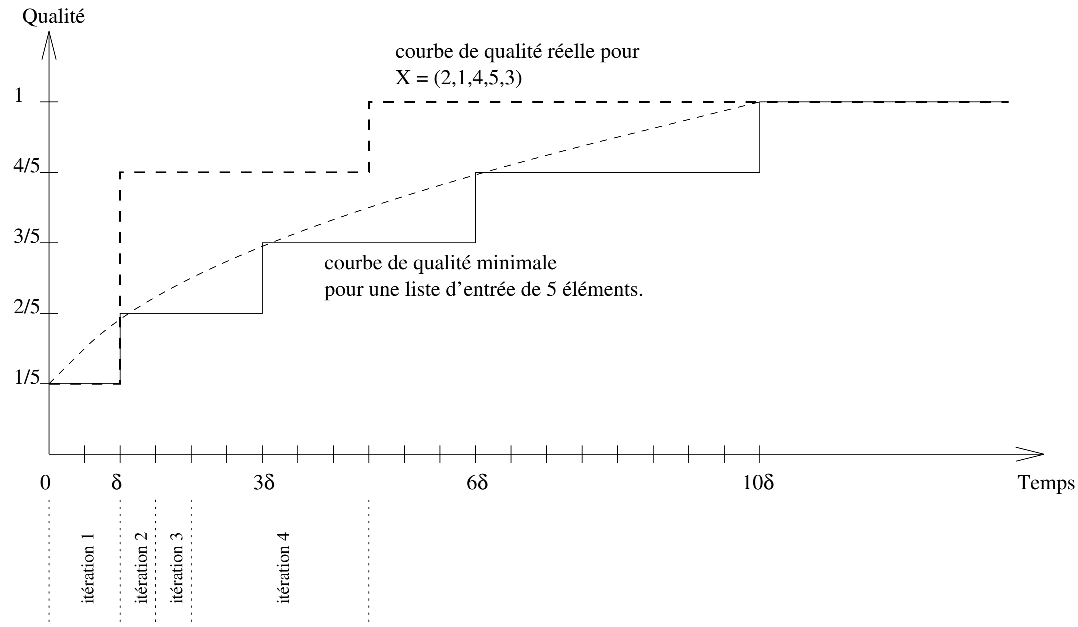


Figure 2 : profil de qualite de l'algorithme de tri par insertion

d'un compromis échéance/qualité : dans certains cas (échéance "molle"), un résultat produit après son échéance peut quand même être "plus utile" qu'un moins bon résultat à l'échéance, l'utilité augmentant avec la qualité mais décroissant avec le temps (après l'échéance).

En conséquence, on peut identifier un certain nombre de mécanismes réactifs primitifs pour le contrôle de tâches interrogeables :

1. activation, suspension et reprise de tâches sur occurrences d'événements,
2. consultation et arrêt de tâches sur occurrences d'événements,

où les événements mentionnés peuvent être :

3. les signaux émis par l'environnement externe,
4. la fin d'une tâche,
5. le dépassement d'une échéance temporelle,
6. le dépassement d'un seuil de qualité donné par le résultat d'une tâche en cours d'exécution.

Les points 1, 3, 4 et 5 correspondent aux mécanismes de contrôle discret que l'on retrouve habituellement dans les noyaux réactifs des systèmes temps-réel classiques. De tels mécanismes peuvent être décrits dans des formalismes abstraits comme les *automates temporisés* de R. Alur *et al.* [3], les *automates chronométrés* de O. Roux *et al.* [11], ou plus concrètement dans des langages de programmation du type Esterel [4] ou Electre [10]. En revanche, les points 2 et 6 ci-dessus découlent directement du caractère continu des tâches interrogeables, et ne peuvent être exprimés dans de tels formalismes.

R. Alur *et al.* ont récemment proposé dans [2, 9] le concept de *système hybride*, comme une extension des automates temporisés. Les systèmes hybrides sont généralement utilisés pour exprimer le contrôle d'un environnement qui évolue continuement avec le temps, comme par exemple un thermostat, un bruleur de gaz .... Si on assimile les tâches interrogeables à des processus continus évoluant dans l'environnement externe, alors le sous-système réactif contrôlant ces tâches peut être décrit comme un système hybride.

Nous identifions dans la section suivante une sous-classe des systèmes hybrides, appelés *systèmes hybrides à horloges dynamiques*, permettant la spécification du contrôle réactif d'un ensemble de tâches interrogeables.

3 Systèmes hybrides à horloges dynamiques

Les systèmes hybrides sont des systèmes concurrents et réactifs qui évoluent en exécutant des transitions discrètes, ou en réalisant des transformations continues. Le comportement temporel des composantes continues est défini dans chaque état du système par un ensemble de relations ou d'équations (par exemple des équations différentielles), alors que les composantes discrètes sont contrôlées par des programmes (par exemple un ensemble de règles).

Le modèle des systèmes hybrides, proposé par A. Olivero dans [9] et sur lequel repose la suite de l'article, est une généralisation des systèmes de transitions temporisés en ajoutant à l'ensemble des horloges un ensemble de variables d'états à valeur dans R. Chacune de ces variables évolue selon des lois et un rythme propres. En particulier, à l'inverse des horloges d'un système de transitions temporisé, la valeur des variables d'un système hybride peut indifféremmemt croître ou diminuer avec l'écoulement du temps. Rappelons enfin que le domaine temporel des systèmes hybrides est dense (l'ensemble R^+).

Le formalisme des systèmes hybrides offre un cadre très général pour décrire le contrôle réactif et temps-réel d'un environnement analogique continu. Nous dégageons dans cet article un sous-ensemble de ce formalisme, appelés *systèmes hybrides à horloges dynamiques* (SHHDs), pour le contrôle réactif et temps-réel de tâches interrogeables. Informellement, nous restreignons l'ensemble des variables d'états d'un SHHD à :

- des horloges évoluant uniformément et de façon synchrone avec le temps-réel ;
- des horloges asynchrones dont le rythme de croissance est défini par des équations différentielles ;
- des variables d'états à valeur dans $[0,1]$ définies par des fonctions croissantes sur les horloges asynchrones.

Les horloges asynchrones (resp. les variables d'état à valeur dans $[0,1]$) dénotent à chaque instant la durée d'exécution (resp. la qualité) des tâches interrogeables. Intuitivement, l'idée sous-jacente et initialement présentée par O. Roux *et al.* dans [11] est de contrôler et modéliser le comportement d'une tâche interrogeable en agissant sur la dérivée temporelle de l'horloge dénotant sa durée d'exécution. La qualité de cette tâche sera ensuite calculée à chaque instant en fonction de la valeur ce cette horloge.

3.1 Notations et définitions préliminaires

Notons respectivement R^+, Z et N l'ensemble des réels positifs ou nuls, l'ensemble des entiers relatifs, et l'ensemble des entiers positifs ou nuls.

Définition 2 *Soit $X = \{x_1, \ldots, x_n\}$ un ensemble fini de variables à valeur dans R^+, on appelle valuation de X une fonction $\tilde{x}$ qui associe à chaque variable x_i une valeur dans R^+. $\tilde{x}(x_i)$ est la valeur de x_i selon la valuation $\tilde{x}$.*

Soit $A \subset X$, soit $\tilde{x}$ une valuation de X, on note "$\tilde{x}[A := 0]$" la valuation définie par :

$$\forall i = 1 \dots n \qquad \tilde{x}[A := 0](x_i) \stackrel{\text{def}}{=} \begin{cases} 0 & si\ x_i \in A \\ \tilde{x}(x_i) & sinon \end{cases}$$

Soit $t \in \mathsf{R}^+$, soit "op" une opération arithmétique sur R^+, on note "$(\tilde{x}\ op\ t)$" la valuation définie par :

$$\forall i = 1, \dots, n \quad (\tilde{x}\ op\ t)(x_i) \stackrel{\text{def}}{=} \tilde{x}(x_i)\ op\ t$$

On note enfin $\tilde{X}$ l'ensemble de toutes les valuations de X.

□

Définition 3 *Soit $X = \{x_1, \dots, x_n\}$ un ensemble fini de variables à valeur dans R^+, on note P_X l'ensemble des prédicats sur les variables de X défini par les règles inductives suivantes :*

- *"**true**" appartient à P_X ;*
- *soit $I \subseteq \{1, \dots, n\}$, soient $\alpha_i \in \mathsf{Z}$ pour $i \in I$, soit $c \in \mathsf{Z}$, et soit $\sim \in \{=, \neq, <, \leq, >, \geq\}$, alors*

  $$\text{“}\sum_{i \in I} \alpha_i x_i \sim c\text{”} \quad \textit{est un prédicat de } P_X \ ;$$

- *soient p_1 et p_2 deux prédicats de P_X, alors "$p_1 \wedge p_2$" est un prédicat de P_X ;*
- *soit p un prédicat de P_X, alors "$\neg p$" est un prédicat de P_X ;*

Soit $\tilde{x} \in \tilde{X}$, soit $p \in P_X$, l'expression "$p[\tilde{x}]$" représente la valeur du prédicat p dont les composantes sont évaluées par $\tilde{x}$, et en utilisant l'interprétation classique de la logique. La valuation $\tilde{x}$ satisfait le prédicat p, noté $\tilde{x} \in p$ si et seulement si $p[\tilde{x}]$ est vrai.

□

Définition 4 *Soit $X = \{x_1, \dots, x_n\}$ un ensemble fini de variables à valeur dans R^+, on note ∂X l'ensemble des dérivées des variables de X par rapport au temps :*

$$\partial X \stackrel{\text{def}}{=} \{\ \dot{x}_1, \dots, \dot{x}_n\ \}$$

L'expression "$\dot{x}$" dénote la dérivée de la variable x par rapport au temps, i.e. $\dot{x} \stackrel{\text{def}}{=} \frac{dx}{dt}$ □

3.2 Syntaxe

Nous commençons par introduire la syntaxe formelle des SHHDs.

Définition 5 *Un système hybride à horloges dynamiques (SHHD) est un tuple $H =< S, s_0, X, \tilde{x}_0, T, I, \Phi >$ où :*

- *S est un ensemble fini de sommets, et $s_0 \in S$ est le sommet initial de H.*
- *$X = (X_s, X_{as}, X_v)$ est l'ensemble des horloges et des variables de H où :*
  - *X_s est un ensemble fini d'horloges synchrones à valeur dans R^+, et évoluant au même rythme que le temps-réel, i.e. telles que :*

    $$\forall x \in X_s, \ \dot{x} = 1 \ \textit{dans chaque sommet} \ ;$$

- X_{as} *est un ensemble fini d'horloges asynchrones à valeur dans* R^+, *et évoluant selon un rythme variable et éventuellement non déterministe tel que :*

$$\forall a \in X_{as},\ 0 \leq \dot{a} \leq 1 \text{ dans chaque sommet ;}$$

- X_v *est un ensemble fini de variables d'états à valeur dans* $[0,1]$.

Notons $\tilde{X} \stackrel{\text{def}}{=} \tilde{X}_s \times \tilde{X}_{as} \times \tilde{X}_v$. *On notera alors* $\tilde{x} = (\tilde{x}_s, \tilde{x}_{as}, \tilde{x}_v)$ *les valuations de* $\tilde{X}$ *où* $\tilde{x}_s$, $\tilde{x}_{as}$, $\tilde{x}_v$ *sont respectivement des valuations des horloges synchrones, des horloges asynchrones, et des variables d'états à valeur dans* $[0,1]$.

- $\tilde{x}_0 \in \tilde{X}$ *est la valuation initiale des horloges et des variables de* H.
- $T \subseteq (S \times E \times (P_{X_s \cup X_{as} \cup X_v}) \times 2^{(X_s \cup X_{as})} \times S)$ *est un ensemble fini de transitions où :*
  - E *est un ensemble fini d'événements ;*
  - $P_{X_s \cup X_{as} \cup X_v}$ *est l'ensemble des prédicats sur les horloges de* X_s, X_{as} *et sur les variables de* X_v.
- $I : S \rightarrow P_{X_s \cup X_{as} \cup X_v}$ *une fonction qui associe à chaque sommet un prédicat sur les horloges de* X_s, X_{as} *et sur les variables de* X_v, *appelé invariant du sommet.*
- $\Phi = (\Phi_{as}, \Phi_v)$ *les lois d'évolutions en chaque sommet où :*
  - $\Phi_{as} : S \rightarrow P_{\partial X_{as}}$ *est une fonction qui associe à chaque sommet un prédicat sur les dérivées des horloges de* X_{as}, *caractérisant leur évolution avec le passage du temps dans le sommet ;*
  - $\Phi_v : S \rightarrow (\tilde{X}_{as} \rightarrow \tilde{X}_v)$ *est une fonction qui associe à chaque sommet une fonction calculant la valeur des variables de* X_v *à partir de la valeur des horloges de* X_{as}.

□

Plus informellement, un SHHD est un système de transitions dont les sommets sont caractérisés par des invariants qui dénotent les conditions à respecter pour rester dans ce sommet. Ces invariants sont des prédicats pouvant porter sur l'ensemble des variables de SHHD, à savoir les valeurs des horloges synchrones de X_s (le temps-réel), les valeurs des horloges asynchrones de X_{as} (le temps d'exécution de chaque tâche) et les valeurs des variables d'états de X_v (la qualité des tâches).

De même, les transitions d'un SHHD sont étiquetées par un événement, par un prédicat conditionnant le franchissement de la transition, et par un ensemble d'horloges (synchrones ou asynchrones) remises à zéro lors du tir de la transition. Les conditions de franchissement d'une transition peuvent porter sur la durée d'exécution des tâches (horloges asynchrones), la qualité de leur résultat (variables de X_v), ainsi que le respect des échéances (horloges synchrones). De plus, seules les horloges synchrones ou asynchrones peuvent être réinitialisées (à zéro) lors du franchissement d'une transition. La qualité du résultat d'une tâche interrogeable ne peut pas être modifiée par une action explicite et discrète. La remise à zéro d'une horloge asynchrone correspond intuitivement à l'arrêt de la tâche correspondante.

Enfin l'évolution temporelle des horloges asynchrones est spécifiée dans chaque sommet par un ensemble de lois différentielles (portant sur les dérivées de ces horloges). Ces lois sont exprimées en chaque sommet s par le prédicat $\Phi_{as}(s)$. La sémantique implicite associée à ces prédicats est que tant que le système se trouve dans un sommet s, les horloges asynchrones évoluent selon les lois dénotées par $\Phi_{as}(s)$. Par exemple, le prédicat "$\dot{a} = 0$" signifie que la tâche correspondante est suspendue. De même, le prédicat

"$\dot{a} + \dot{b} = 1$" signifie que les tâches A et B sont simultanément actives et se partagent de façon non-déterministe la ressource d'exécution.

On note dans la suite la loi d'évolution des valuations dans un sommet s par la relation $\Phi(s) \subset (\tilde{X} \times \mathsf{R}^+ \times \tilde{X})$ définie par :

$$((\tilde{x}_s,\ \tilde{x}_{as},\ \tilde{x}_v), t, (\tilde{x}'_s,\ \tilde{x}'_{as},\ \tilde{x}'_v)) \in \Phi(s) \ \ \textbf{ssi} \ \begin{cases} \tilde{x}'_s = \tilde{x}_s + t \\ \Phi_{as}(s)[(\tilde{x}'_{as} - \tilde{x}_{as})/t] \text{ est vrai} \qquad \text{pour tout } t > 0 \\ \tilde{x}'_v = \Phi_v(s)(\tilde{x}'_{as}) \end{cases}$$

et

$$((\tilde{x}_s,\ \tilde{x}_{as},\ \tilde{x}_v), 0, (\tilde{x}'_s,\ \tilde{x}'_{as},\ \tilde{x}'_v)) \in \Phi(s) \ \ \textbf{ssi} \ \ (\tilde{x}'_s = \tilde{x}_s \text{ et } \tilde{x}'_{as} = \tilde{x}_{as} \text{ et } \tilde{x}'_v = \tilde{x}'_{as})$$

Plus informellement, $(\tilde{x}, t, \tilde{x}') \in \Phi(s)$ si et seulement si $\tilde{x}'$ est une valuation obtenue à partir de $\tilde{x}$ en laissant passer une durée t conformément aux lois d'évolution $\Phi_{as}(s)$ et $\Phi_v(s)$ en vigueur dans le sommet s.

3.3 Sémantique

Avant de donner la sémantique des SHHDs de façon formelle, nous présentons informellement leur comportement. Soit $H =< S, X, T, I, \Phi >$ un SHHD :

- un état du système est déterminé par un sommet $s \in S$ et par une valuation $\tilde{x} = (\tilde{x}_s, \tilde{x}_{as}, \tilde{x}_v)$ telle que l'invariant de s soit satisfait par $\tilde{x}$. Notons un tel état par la paire $(s, \tilde{x})$;
- la transition $< s, e, p, A, s' >\in T$ peut être franchie à partir de l'état $(s, \tilde{x})$ si le prédicat p est satisfait par la valutation $\tilde{x}$. L'état résultant est $(s', \tilde{x}')$ où $\tilde{x}'$ est la nouvelle valuation obtenue à partir de $\tilde{x}$ en affectant zéro aux variables de A ;
- le franchissement d'une transition ne prend pas de temps ;
- dans chaque sommet, les valeurs des variables changent avec le temps en respectant les lois d'évolutions Φ ;
- le système peut séjourner dans un sommet s tant que l'invariant $I(s)$ du sommet est satisfait.

Notons Q l'ensemble des états du système :

$$Q \stackrel{\text{def}}{=} \{(s, \tilde{x}) \ \mid \ s \in S,\ \tilde{x} \in \tilde{X},\ \text{et } I(s)[\tilde{x}] \text{ est vrai}\}$$

On définit la sémantique des SHHDs par une relation $\longrightarrow \subseteq (Q \times (E \cup \mathsf{R}^+) \times Q)$ traduisant le comportement réactif et temporel d'un SHHD. On note

$$(s, \tilde{x}) \ \stackrel{e}{\longrightarrow} \ (s', \tilde{x}') \quad \text{avec } e \in E$$

pour exprimer que le système évolue de l'état $(s, \tilde{x})$ à l'état $(s', \tilde{x}')$ en tirant instantanément la transition étiquetée par e. De même, on note

$$(s, \tilde{x}) \ \stackrel{t}{\longrightarrow} \ (s, \tilde{x}') \quad \text{avec } t \in \mathsf{R}^+$$

pour exprimer que le système évolue de l'état $(s, \tilde{x})$ à l'état $(s, \tilde{x}')$ en laissant passer une durée t.

La relation "$\longrightarrow$" est la plus petite relation incluse dans $(Q \times (E \cup \mathsf{R}^+) \times Q)$ vérifiant les deux règles suivantes :

Tir d'une transition : une transition est tirable si et seulement si son prédicat est satisfait par la valuation courante. L'état résultant est composé par le sommet cible de la transition et par la valuation obtenue à partir de la valuation courante en affectant zéro aux variables et horloges contenues dans A :

$$\frac{< s, e, p, A, s' > \in T \quad p[\tilde{x}] \quad I(s)[\tilde{x}] \quad I(s')[\tilde{x}[A := 0]]}{(s, \tilde{x}) \xrightarrow{e} (s', \tilde{x}[A := 0])} \tag{1}$$

Ecoulement du temps : un système peut évoluer en laissant passer le temps tout en restant dans le sommet courant tant que l'invariant du sommet est satisfait par la valeur des variables et des horloges. Les horloges synchrones de $\tilde{X}_s$ évoluent alors uniformément au rythme du temps réel. L'évolution des horloges asynchrones de X_{as} est contrainte par le prédicat $\Phi_{as}(s)$ portant sur les dérivées de ces variables. Enfin, la nouvelle valuation des variables réelles de X_v est une fonction de la nouvelle valuation des horloges asynchrones :

$$\frac{t \in \mathsf{R}^+ \quad (\tilde{x}, t, \tilde{x}') \in \Phi(s) \quad I(s)[\tilde{x}] \quad I(s)[\tilde{x}']}{(s, \tilde{x}) \xrightarrow{t} (s, \tilde{x}')} \tag{2}$$

3.4 Discussion

La sémantique formelle étant exposée, nous discutons maintenant des propriétés de déterminisme et d'additivité des SHHDs, ainsi que du problème de l'accessibilité d'un état.

3.4.1 Non-déterminisme

Contrairement aux automates temporisés de R. Alur *et al.* [3], un SHHD n'est pas déterministe par rapport au temps. Considérons un sommet $s \in S$ et une valuation $\tilde{x} = (\tilde{x}_s, \tilde{x}_{as}, \tilde{x}_v)$, la valuation $\tilde{x}' = (\tilde{x}'_s, \tilde{x}'_{as}, \tilde{x}'_v)$ obtenue à partir de $\tilde{x}$ en laissant passer une durée non nulle t dans le sommet s est telle que :

$$\begin{cases} (i) & \tilde{x}'_s = \tilde{x}_s + t \\ (ii) & \Phi_{as}(s)[(\tilde{x}'_{as} - \tilde{x}_{as})/t] \text{ est vrai} \\ (iii) & \tilde{x}'_v = \Phi_v(s)(\tilde{x}'_{as}) \end{cases}$$

Le calcul de $\tilde{x}'_s$ et de $\tilde{x}'_v$ est déterministe. En revanche il peut exister dans le cas général plusieurs valuations $\tilde{x}'_{as}$ satisfaisant la loi d'évolution (ii) ci-dessus.

Cet indéterminisme peut se justifier par le fait que dans un environnement asynchrone et concurrent, les vitesses d'exécution des processus de traitement peuvent être variables et différentes de 1 (par rapport au temps-réel), voire même indéterminée. Considérons par exemple deux tâches A et B. Notons respectivement $a(t)$ et $b(t)$ la part du temps d'exécution consommé à l'instant t par chaque tâche. On peut alors identifier 4 comportements possibles (dans un environnement mono-processeur) :

(i) A et B sont inactives : alors $\dot{a} = 0$ et $\dot{b} = 0$;

(ii) seule A est active : alors $\dot{a} = 1$ et $\dot{b} = 0$;

(iii) seule B est active : alors $\dot{a} = 0$ et $\dot{b} = 1$;

(iv) A et B sont actives simultanément et en parallèle : alors $\dot{a} + \dot{b} = 1$.

Les trois premiers cas sont déterministes. En revanche le dernier cas est une généralisation continue du modèle sémantique de la concurrence, dite “sémantique par entrelacement”, des langages asynchrones classiques. On retrouve alors le non-déterminisme inhérent à un tel modèle sémantique.

D'autre part, du point de vue du contrôle réactif, il peut être intéressant de modéliser un certain indéterminisme sur la vitesse d'exécution d'une tâche donnée, à un instant donné, ne serait-ce parce que la vitesse réelle de la tâche dépend concrètement de paramètres d'implémentation parfois difficilement maîtrisables tels que l'ordonnanceur, la charge des réseaux de communication ...

3.4.2 Additivité

Les SHHDs, tels qu'ils ont été définis, ne sont pas additifs par rapport au temps. Considérons par exemple un sommet s et l'invariant suivant :

$$I(s) \stackrel{\text{def}}{=} (x_1 > 5 \ \vee \ x_1 < 3)$$

avec $x_1 \in X_s$. Considérons la valuation $\tilde{x}$ telle que $\tilde{x}(x_1) = 2$. Alors $I(s)[\tilde{x}]$ est vrai. De même $I(s)[\tilde{x}+4]$ est vrai. En revanche $I(s)[\tilde{x}+2]$ est faux. En conséquence, d'après la règle 2 il existe $\tilde{x}''$ tel que :

$$(s, \tilde{x}) \ \xrightarrow{4} \ (s, \tilde{x}'')$$

Mais il n'existe pas $\tilde{x}'$ tel que :

$$(s, \tilde{x}) \ \xrightarrow{2} \ (s, \tilde{x}')$$

Ce qui est contraire à la sémantique intuitive de l'écoulement du temps.

Plus généralement, le problème vient du fait que les invariants ne sont pas nécessairement *connexes* par rapport à l'écoulement du temps. On restreint donc la classe des SHHDs aux systèmes dont les sommets sont caractérisés par un invariant connexe, *i.e.* vérifiant la définition suivante :

Définition 6 *Soit un sommet s, soit $\Phi(s)$ la loi d'évolution des valuations dans le sommet s. Soit $I(s)$ l'invariant du sommet s. Soit une valuation $\tilde{x}$ telle que $I(s)[\tilde{x}]$ soit vrai. On dit que $I(s)$ est connexe par rapport à l'écoulement du temps si et seulement si pour tout $t, t' \in \mathsf{R}^+$ et pour toute valuation $\tilde{x}'' \in \tilde{X}$:*

$$(\tilde{x}, t+t', \tilde{x}'') \in \Phi(s) \ \wedge \ I(s)[\tilde{x}''] \ \ \textbf{ssi} \ \ \left(\begin{array}{c} \exists \tilde{x}' \in \tilde{X} \ (\tilde{x}, t, \tilde{x}') \in \Phi(s) \wedge \ I(s)[\tilde{x}'] \\ \wedge \ (\tilde{x}', t', \tilde{x}'') \in \Phi(s) \end{array} \right)$$

En d'autres termes, un invariant est connexe par rapport au temps si et seulement si pour toute valuation $\tilde{x}''$ obtenue à partir de $\tilde{x}$ en laissant passer une durée arbitraire telles que $\tilde{x}$ et $\tilde{x}''$ satisfont l'invariant, alors les valuations intermédiaires parcourrues entre $\tilde{x}$ et $\tilde{x}''$ satisfont également l'invariant.

Si on restreint la classe des SHHDs aux systèmes dont les sommets sont caractérisés par un invariant connexe par rapport au temps, on peut alors montrer que les SHHDs ainsi construits sont additifs par rapport au temps. Si le système peut évoluer d'un état $(s, \tilde{x})$ vers un état $(s, \tilde{x}')$ en laissant passer une durée t, et de $(s, \tilde{x}')$ vers $(s, \tilde{x}'')$ en laissant passer une durée t', alors il peut évoluer de $(s, \tilde{x})$ vers $(s, \tilde{x}'')$ en laissant passer une durée $t + t'$, et réciproquement :

$$\begin{array}{l} \forall s \in S, \ \forall \tilde{x}, \tilde{x}'' \in \tilde{X}, \ \forall t, t' \in \mathsf{R}^+ \\ (s, \tilde{x}) \ \xrightarrow{t+t'} \ (s, \tilde{x}'') \ \ \textbf{ssi} \ \ \exists \tilde{x}' \in \tilde{X}, \ (s, \tilde{x}) \ \xrightarrow{t} \ (s, \tilde{x}') \ \xrightarrow{t'} \ (s, \tilde{x}'') \end{array}$$

3.4.3 Indécidabilité du problème de l'accessibilité

Le problème de l'accessibilité, *i.e.* déterminer si un état $(s, \tilde{x})$ est accessible à partir de l'état initial par une succession de transitions discrètes et de transformations continues, est la question centrale du problème de la vérification des systèmes hybrides. En particulier, la vérification d'une propriété d'invariance revient à vérifier s'il existe des états ne vérifiant pas la propriété d'invariance mais qui sont accessibles à partir de l'état initial. La non-existence de tels états démontre la propriété d'invariance correspondante.

Malheureusement le problème de l'accessibilité est indécidable pour les systèmes hybrides en général, et en particulier pour la classe restreinte des *systèmes hybrides linéaires à pente constante* (SHLPCs) développée par A. Olivero dans [9]. Or la classe des SHHDs contient la classe des SHLPCs. Le problème de l'accessibilité est donc également indécidable pour les SHHDs[1].

En conséquence, le problème de la vérification de propriétés d'invariances est indécidable pour les SHHDs, et plus généralement pour les systèmes temps-réel construit selon le paradigme *réactif/interrogeable*.

4 Exemple

Considérons un système de multi-conférence à trois agents *Pierre*, *Paul* et *Jean*. Ce dernier utilise un récepteur mobile et reçoit simultanément des messages de Pierre et de Paul respectivement sur les canaux C_A et C_B. Jean étant mobile, la qualité de réception via les canaux C_A et C_B peut varier au cours du temps (surgissement d'obstacle entre Pierre et Jean et/ou entre Paul et Jean, apparitions temporaires d'interférences avec d'autres sources dans l'environnement, ...). Il est donc nécessaire, pour maintenir une communication correcte, d'estimer et corriger en permanence les erreurs de transmission sur chaque canal. Cette correction doit être faite par le récepteur mobile.

Notons respectivement A et B les tâches estimant l'erreur de transmission sur les canaux C_A et C_B. Notons $a(t)$ et $b(t)$ respectivement les durées d'exécution de A et B à un instant t. Notons C la tâche corrigeant les erreurs estimées par A et B. Notons enfin $c(t)$ la durée d'exécution de C à un instant t. On notera dans la suite a, b et c au lieu de $a(t)$, $b(t)$ et $c(t)$.

On suppose que les tâches A et B sont interrogeables. Notons $q_a = Q_A(a)$ et $q_b = Q_B(b)$ les qualités des résultats produits par les tâches respectivement A et B après des durées d'exécution respectives a et b.

Le système de correction d'erreurs composé par les tâches A B et C et embarqué dans le récepteur mobile de Jean répond aux spécifications suivantes :

1. l'erreur de réception doit être estimée et corrigée sur chaque canal au plus tôt toutes les $25ms$ (milliseconde) et au plus tard tout les $40ms$; En d'autres termes, le fonctionnement du système de correction doit être cyclique et son temps de cycle doit être compris entre $25ms$ et $40ms$;

2. les tâches A, B et C ne disposent que d'un processeur pour s'exécuter.

D'autre part, les tâches A, B et C sont caractérisées par les propriétés suivantes :

3. A et B sont interrogeables mais nécessitent chacune une durée minimale d'exécution de $10ms$ pour produire un résultat approximatif applicable (de qualité non nulle) :

$$Q_A(a) > 0 \textbf{ ssi } a \geq 10ms \quad \text{et} \quad Q_B(b) > 0 \textbf{ ssi } b \geq 10ms$$

[1] On peut également obtenir directement ce résultat on montrant que le problème de l'accessibilité d'un état d'un SHHD peut se ramener au problème de l'arrêt d'une machine à n compteurs (n étant le nombre d'horloges asynchrones) (voir par exemple les démonstrations proposées dans [2] et [8]). Or le problème de l'arrêt d'une machine à n compteurs est indécidable dès que $n \geq 2$. On en déduit donc que le problème de l'accessibilité est indécidable pour les SHHDs.

4. un résultat approximatif de qualité 0.75 ou plus (resp. 0.85 ou plus) produit par A (resp. B) est suffisant pour obtenir un correction satisfaisante ;

5. C n'est pas interrogeable et s'exécute en exactement $10ms$.

Le problème à résoudre est alors d'ordonnancer les tâches A, B et C de manière à ce que le temps de cycle soit effectivement compris entre $25ms$ et $40ms$, et d'autre part que les résultats fournis par A et B soient de qualité non nulle.

Considérons le SHHD présenté en figure 3. Ce système décrit le contrôle réactif des tâches A B et C embarqué dans le récepteur mobile de façon à satisfaire les spécifications ci-dessus. Ce SHHD est composé de :

1. un ensemble de sommets $S = \{s_0, s_1, s_2, s_3, s_4\}$, s_0 étant le sommet initial,
2. une horloge synchrone $X_s = \{t\}$,
3. trois horloges asynchrones $X_{as} = \{a, b, c\}$,
4. deux variables d'état q_a et q_b, telles que dans chaque sommet $q_a = Q_A(a)$ et $q_b = Q_B(b)$ où Q_A et Q_B sont des fonctions croissantes de R^+ dans $[0, 1]$,
5. un ensemble de signaux $E = \{communication\_on, arret\_a, arret\_b, end\_c\}$.

Initialement, le système attend le signal *communication_on* pour commencer le cycle *estimation-correction*. Au début du cycle (sommet s_1) les tâches A et B sont activées simultanément sur la même ressource d'exécution. Le temps de cycle étant au plus de $40ms$ et au moins de $25ms$, et la durée de la tâche C étant exactement de $10ms$, le temps imparti à A et B et donc au plus $30ms$ et au moins $15ms$. Les tâches A et B sont donc interronpues (et interrogées) soit lorsque $t = 30$, soit lorsque $t \geq 15$ et lorsque la qualité de leur résultat est suffisant pour provoquer une correction satisfaisante. Enfin, lorsque A et B ont été interrompues, C est activée (sommet s_4). Le cycle recommence alors lorsque C est terminée. Le schéma figure 3 illustre ce comportement.

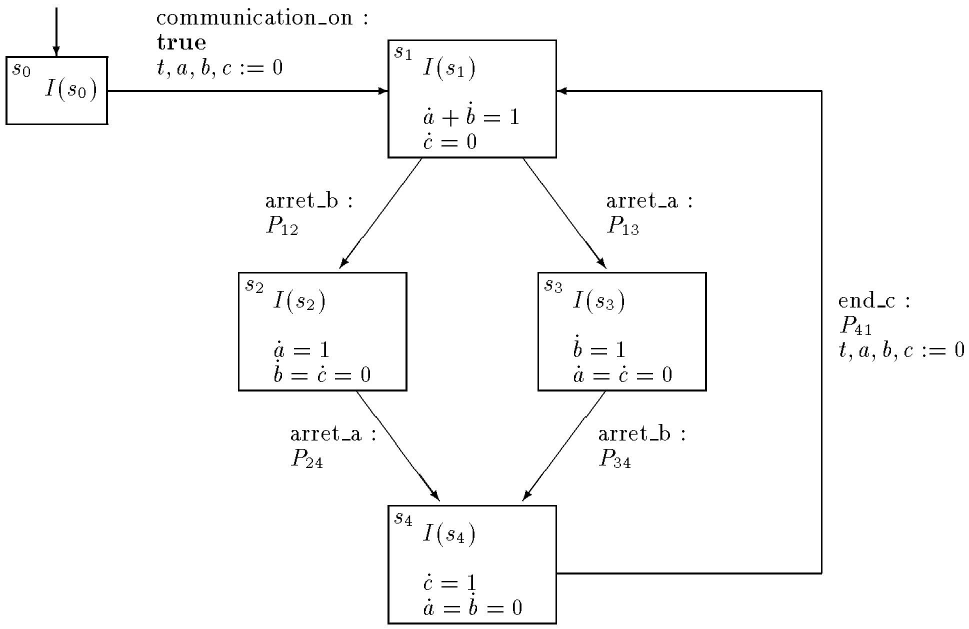
On peut alors montrer que, d'une part en sortie de s_4 l'horloge t (*i.e.* la durée du cycle) est comprise entre $25ms$ et $40ms$, et d'autre part en s_4 les horloges a et b (durées d'exécution de A et B) sont supérieures à $10ms$ et donc que q_a et q_b sont strictement positives.

5 Conclusion

L'objectif de cet article est de caractériser et définir formellement les mécanismes de contrôle de tâches interrogeables. Nous avons défini pour ce faire une sous-classe des systèmes hybrides appelé *systèmes hybrides à horloges dynamiques*. Une tâche interrogeable est caractérisée dans un tel formalisme par deux variables continues : une horloge asynchrone croissante dénotant sa durée d'exécution, et une variable d'état croissante à valeur dans $[0, 1]$ dénotant la qualité de ses résultats intermédiaires. Le formalisme des SHHDs répond complètement à la problématique du contrôle *réactif/interrogeable*.

Or, le problème de l'accessibilité, et donc la vérification de propriétés d'invariance, est indécidable pour les SHHDs. En conséquence, la vérification de telles propriétés sur un système temps-réel *réactif/interrogeable* sera en général indécidable. Face à ce constat, il existe en général deux approches possibles et complémentaires :

- soit définir des sous-classes de SHHDs pour lesquelles le problème de l'accessibilité devient décidable, c'est l'approche suivie par exemple par Kesten *et al.* [8] et Roux *et al.* [11],



avec :

$$P_{12} = (t = 30) \vee (q_b \geq 0.85 \wedge t \geq 15) \qquad I(s_0) = \mathbf{true}$$
$$P_{13} = (t = 30) \vee (q_a \geq 0.75 \wedge t \geq 15) \qquad I(s_1) = (t \leq 30) \wedge (t \leq 15 \vee q_a \leq 0.75 \vee q_b \leq 0.85)$$
$$P_{24} = (t = 30) \vee (q_a \geq 0.75 \wedge t \geq 15) \qquad I(s_2) = (t \leq 30) \wedge (t \leq 15 \vee q_a \leq 0.75)$$
$$P_{34} = (t = 30) \vee (q_b \geq 0.85 \wedge t \geq 15) \qquad I(s_3) = (t \leq 30) \wedge (t \leq 15 \vee q_b \leq 0.85)$$
$$P_{41} = (c = 10) \qquad I(s_4) = (c \leq 10)$$

Figure 3 : exemple de SHHD

- soit au contraire utiliser des algorithmes de vérification par approximation reposant par exemple sur des techniques d'interprétation abstraite, c'est l'approche suivie par Halbwachs *et al* [6].

Notre prochain objectif est de dépasser les limites structurelles posées par le caractère indécidable du problème de l'accessibilité, et donc de la vérification de propriétés d'invariance, en expérimentant les deux approches ci-dessus dans le cadre des SHHDs.

Bibliographie

[1] M. Adelantado and S. de Givry. Reactive/anytime agents: Towards intelligent agents with real-time performance. In *IJCAI'95 Workshop on Anytime Algorithms and Deliberation Scheduling*, Montreal, Canada, August, 21-25 1995.

[2] R. Alur, C. Courcoubetis, N. Halbwachs, T. Henzinger, P. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138(1), February 1995.

[3] R. Alur and D. Dill. The theory of timed automata. *LNCS 600 : Real-time : Theory in practice*, pages 45–73, 1991.

[4] G. Berry and G. Gonthier. The Esterel Synchronous Programming Language: Design, Semantics, Implementation. *Science Of Computer Programming*, 19(2):87–152, 1992.

[5] M. Boddy. *Solving time-dependent problems:a decision-theoretic approach to planning in dynamic environments*. PhD thesis, Brown University, Providence, Rhode Island 02912, May 1991.

[6] N. Halbwachs, P. Raymond, and Y.E. Proy. Verification of linear hybrid systems by means of convex approximations. In *2nd AMAST Workshop on Real-Time Systems*, Bordeaux, France, June 14-16 1995.

[7] D. Harel and A. Pnueli. On the development of reactive systems. In *Logic and Models of Concurrent Systems*. Proc NATO Advanced Study Institute on Logics and Models for Verification and Specification of Concurrent Systems (NATO ASI Series F vol. 13), 1985.

[8] Y. Kesten, A. Pnueli, J. Sifakis, and S. Yovine. Integration graphs: A class of decidable hybrid systems. In *Workshop on Theory of Hybrid Systems, LNCS 736*, pages 179–208, Lyngby, Denmark, June 1993.

[9] A. Olivero. *Modélisation et analyse des systèmes temporisés et hybrides*. PhD thesis, Institut National Polytechnique de Grenoble, september 1994.

[10] O. Roux, D. Creusot, F. Cassez, and J.P. Elloy. Le langage réactif asynchrone ELECTRE. *Technique et Science de l'Informatique*, 11(5):35–66, 1992.

[11] O. Roux and V. Rusu. Verifying time-bounded properties for ELECTRE reactive programms with stopwatch automata. In *Workshop on Hybrid Systems and Autonomous Control*, Cornell University, Ithaca (New-York, USA), October 1994.

[12] J.A. Stankovic. Misconceptions about real-time computing. *Computer*, 1988.