European Network of Excellence in AI Planning

The PLANET Roadmap on AI Planning and Scheduling

Ruth Aylett, Susanne Biundo,
Malik Ghallab, Simon de Givry,
Paul Kearney, Lee McCluskey

(Editors)

This document is part of the Final Report for the Network of Excellence PLANET, Project No. 28439
It is continuosly updated, further versions will be available in the download section of the PLANET web pages.

June 2001

Preface

Planning and Scheduling is the field of Artificial Intelligence that is concerned with all aspects of the system-supported or fully automated synthesis, execution, and monitoring of courses of actions, activities, and tasks. With that, it provides a technology for increasing the autonomy of systems by making them more flexible, robust, and adaptive. Consequently, it has a particularly large application potential in a variety of industrial and administrative areas including the growing *e*-business and *e*-work sectors.

This road map document aims to take stock of current exploitation of the technology and points out future research and development steps for both improving the technology in current applications and widening the spectrum of future ones. The road map is joint work by members of PLANET, the *European Network of Excellence in AI Planning* – funded by the European Union under the *Esprit* programme from October 1998 to December 2000.

The road map addresses a number of themes under the global view of aiming to extend the exploitation of the technology far beyond the current stage. To this end, two main aspects are considered: further technological developments required and a number of key application areas for various industrial and business sectors.

The road map comprises five chapters. Two are devoted to methodological themes, namely *Knowledge Engineering* and *On-line Scheduling*, another three are concerned with the application areas of *Robotics*, *Intelligent Manufacturing*, and *Workflow Management*, respectively.

Each chapter comprehensively reviews the State of the Art in the respective sector. It identifies directions for further exploitations of the technology and – guided by these – suggests improvements of planning and scheduling methodologies and techniques. Finally, each chapter provides development plans for focused research and development actions in the short-, medium-, and long-term run.

Each chapter of the road map corresponds to a so-called *Technical Co-ordination Unit* (TCU) of PLANET. These units were built to provide a forum for network members working in the particular areas. Each TCU organised series of coherent and focused workshops, among others. Parts of their results are reflected in this road map document, the chapters of which are edited by the respective TCU chairs, namely **Ruth Aylett**, **Simon De Givry**, **Malik Ghallab**, **Paul Kearney**, and **Lee McCluskey**.

The following authors contributed to this road map: Rachid Alami (LAAS-CNRS, Toulouse, France), Ricardo Aler (University Carlos III de Madrid, Spain), Ruth Aylett (Salford University, UK), Michael Beetz (University of Bonn, Germany), Susanne Biundo (University of Ulm, Germany), Daniel Borrajo (University Carlos III de Madrid, Spain), Luis Castillo (University of Granada, Spain), Amedeo Cesta (IP-CNR, Rome, Italy), Alexandra Coddington (Salford University, UK), Patrick Doherty (Linköping University, Sweden), Patrick Fabiani (ONERA, Toulouse, France), Simon De Givry (Thales LCR, Orsay, France), Miguel Angel Garcia (University Rovira i Virgili, Tarragona, Spain), Malik Ghallab (LAAS-CNRS, Toulouse, France), Patrik Haslum (Linköping University, Sweden), Joachim Hertzberg (GMD Bonn, Germany), Felix Ingrand (LAAS-CNRS, Toulouse, France), Peter Jarvis (SRI International, Menlo Park, USA), Paul Kearney (British Telecom, Ipswich, UK), Jonas Kvarnström (Linköping University, Sweden), David Lesaint (British Telecom, Paris, France), Roger Mampey (ONERA, Toulouse, France), Nicola Matino (Centro Ricerche Fiat, Turin, Italy), Nikolay Mehandjiev (University of Manchester, UK), Lee McCluskey (University of Huddersfield, UK), Yannick Meiller (ONERA, Toulouse, France), Alessandro Saffiotti (Örebro University, Sweden), Bernd Schattenberg (University of Ulm, Germany), Ulrich Scholz (Darmstadt University of Technology, Germany), Gérard Verfaille (ONERA, Toulouse, France).

June 2001 **Susanne Biundo**

Co-ordinator of PLANET

Contents

Part I.

Robot Action Planning

Malik Ghallab

(LAAS-CNRS, Toulouse, France)

with contributions for the state of the art from Daniel Borrajo (Universidad Carlos III de Madrid, Spain), Patrick Fabiani (ONERA, Toulouse, France), José Manuel Molina (Universidad Carlos III de Madrid, Spain), Araceli Sanchis (Universidad Carlos III de Madrid, Spain), and Bernd Schattenberg (University of Ulm, Germany), and contributions from the TCU nodes that are described in the last section.

1. Introduction

1.1. Planning in robotics

Planning in robotics is a computational activity, among several others, carried out only when needed for achieving a task or reaching a goal. It is neither central nor is it focused into a single system. Planning within a robot takes several forms and uses different types of representations. The purpose of these various forms of planning is in general to synthesize an abstract trajectory in some search space, predicting outcome, choosing and organizing actions of different types for reaching goals or for optimizing some utility functions.

In order to perform prediction and choices for achieving specified goals and utilities, robot planning relies on:

- Models of the environment and of the robot capabilities;
- Specification of required goals and/or utility criteria;
- Online input from sensors and communication channels.

The specifics of planning in robotics, as compared to other application domains of planning, are mainly the need to handle

- Heterogeneous partial models of the environment and of the robot, as well as noisy and partial state information acquired through sensors and communication channels
- Direct integration of planning to acting and sensing

Robotics without planning corresponds basically to handcoding the environment structure, the robot skills, and its strategies into a purely reactive control. This is a perfectly feasible approach as long as the handcoding is inexpensive and reliable enough for the application at hand. This will be the case for a well-structured and stable environment and for a robot carrying out repeatedly a single task or a fairly reduced set of tasks, with a limited man-robot interaction. Learning capabilities, supervised or autonomous, and programming aids, such as hardware devices (e.g., memorizing the motion of a pantomime), or software tools (e.g., graphical programming interfaces), may extend the scope of applicability of the approach. However, if a robot has to face a diversity of tasks and/or a variety of environments, then planning capabilities will significantly simplify and robustify its programming and augment its usefulness. Planning capabilities are not to be opposed to reactive activities of a robot, handcoded or learned, neither they have to be opposed to learning capabilities. They have to be closely integrated to them. For all forms of robot planning, such integration is a highly challenging problem.

Among the various forms of robot planning, there is in particular

- Path and motion planning
- Perception planning
- Navigation planning
- Manipulation planning
- Task planning
- Communication planning

Path and motion planning is concerned with the synthesis of a geometric path, from a starting position to a goal, and of a control trajectory along that path that specifies the state variables of a robot and their derivatives in the configuration space, taking into account the kinematic and dynamic constraints. This is a very well studied area [115], which appears today to be quite mature and offering a wide range of efficient and reliable methods.

Perception planning is concerned with information gathering plans. It arises in tasks such as modeling environments or objects, identifying objects, localizing the robot, or more generally identifying the current state the environment. Perception planning addresses questions such as which information is needed and when, where to look for it, which sensors are most adequate for this particular task, and how to use them. It requires models of available sensors, of their capabilities and constraints. It relies on decision theory for problems of which and when information is needed, on mathematical programming and constraint satisfaction for the viewpoint selection and sensor modality control.

Navigation planning combines the two previous problems. Its purpose is to synthesize a strategy or policy that combines localization primitives and sensor-based motion primitives, e.g., visual servoing, in order to reach a goal or to explore an area.

Manipulation planning is similar to navigation. The output strategy here combines sensory-motor primitives using forces and touch (haptics), vision, range and other sensors to handle objects and assemblies.

Task planning corresponds to the classical AI planning problems, with general-purpose state-transition operators. However, Robotics task planning has to deal with time and resource allocation, dynamic environments, uncertainty and partial state knowledge, and mostly incremental planning with consistent integration to acting and sensing.

Communication planning arises in multi-robots cooperation and in Man-machine interaction. It addresses issues such as when and how to query needed information, which feedback should be provided.

PLANET is not active in all these research areas. Robot Planning TCU has to be aware of them and of their common features, which are mainly the handling of

- Uncertainty
- On-line constraints
- Dynamic environments and feedback loops
- Cooperation and multi-agency

PLANET TCU on Robot Planning has to build on top of the various robot planning forms and to promote Integration of sensory motor capabilities with deliberative, goal-oriented capabilities. This requires a consistent Integration of heterogeneous representations for modeling and reasoning on space, time, kinematic and dynamics, on the physics of sensors, on uncertainty, on logical properties and various domain constraints, including computational constraints. It requires the integration of various forms of planning together, and to learning, in order to extend sensory-motor controllers.

1.2. Situation of the domain

Today, the maturity of robot planning is mainly at the level of some planning components. Path and motion planning corresponds to a well mature area, relying on computational geometry and using efficiently probabilistic algorithms. It is already deployed in robotics and in other application areas such as CAD or computer animation. Task planning benefits from a wealth of algorithms in the classical framework (Strips and ADL), with techniques such as heuristic search, disjunctive refinement, or SAT coding. It is not widely deployed in robotics for various reasons, among which the restrictive assumptions and expressiveness of the classical framework. Perception planning is a younger and much more open area, although some focused problems are well solved, e.g. the viewpoint selection problem with mathematical programming techniques.

Another well mature area is that of high level reactive controllers. These are rule-based or procedure-based systems that permit preprogrammed goal-directed and event-reactive behaviors. They are implemented in PRS, RAP, Propice, SRCs and other similar systems, which are well advanced and well integrated to the sensory-motor level. Most laboratory robots run on them.

An important aspect of the maturity of the field corresponds to possible technology transfer. Within robotics, there is certainly a wide potential of technology development as demonstrated by few industrial projects within the industry. However the market development for robots, beyond the highly structured and specifically engineered area of manufacturing robots, is today very slow. There are several limiting factors for industrial deployment of robot planning technology, e.g. the development of sensory-motor functions, the reliability and security problems, and the cost problem which undermine the development of many service robot applications. The technology transfer of reactive controllers is a first step that is already going on, within and outside robotics, e.g., in transportation systems. Special purpose navigation and perception planning capabilities can also be transferred in other domains. The usefulness of path and motion planning in CAD, in computer animation and graphics has already been demonstrated. Perception planning is of interest in surveillance systems for example. Tasks planning with time and resources can be applied in manufacturing for process planning, workflow management, or network management. Integrated planning and execution systems have been deployed in well-structured domains such as the management of autonomous spacecraft, e.g., the Deep Space One experiment. This is a very important application area, it will be addressed by a specific TCU within PLANET.

1.3. What needs to be done

A main challenge in robot planning is that of integration, that is not only *system integration* of components into a consistent architecture, but also and mainly the design of representations and algorithms that permit such integration. In particular, research should focus on issues such as:

- Planning with information gathering and sensing
- Heterogeneous planning techniques
- Heterogeneous representations
- Access to world state through sensing
- Execution models
- Dynamic planning and acting
- Planning and sensing
- Planning depth and on-line constraints
- Incremental planning with fault detection, diagnosis, recovery
- Reaction and deliberation architectures
- Planning and learning

Another important research topic is the evaluation of integrated planning systems and the analysis of their performance. One should be able to answer questions such as when and why a robot architecture with planning may perform better than without. Performance criteria such as the type and variability of sets of environments and sets of tasks dealt with, of the achieved robustness, of the cost of programming and verification of the robot software should be studied. Experimental setups, comparison, benchmarks should be developed and disseminated.

Typical robotics projects that address some of the above issues are for example:

- Cooperating service robots for tasks such as transportation, surveillance, cleaning, search and gathering of object, e.g., office assistants
- Surveillance and monitoring of the traffic network
- Exploration robots, environment monitoring

Several current projects at the Robot Planning TCU nodes fall within these categories. These are in particular:

- Rhino, a robotics museum tourguides at the Univ. of Bonn
- Hilare, cooperating service robots at LAAS - CNRS
- Supermarket cleaning robots at Siemens
- Jerry, a robot arm for manipulation tasks in space, at IRST and the Univ. of Genoa
- MAKRO a robot for sewage maintenance operations at GMD
- WITAS, a traffic surveillance aerial robot at the Univ. of Linköping

- MACTA, cooperating robots for an automated laboratory, at the Univ. of Salford
- RoboCup, cooperative acting and sensing in robotics soccer, at the Univ. of Freiburg

These projects are presented in detail in section 3 of this roadmap together with the various research topics currently investigated at the TCU nodes. Section 2 develops a state of the art focused on some essential issues, complementary to the good surveys and reference works existing in the literature on basic robot planning components, e.g. on Path and motion planning or on Task planning. The Roadmap surveys current work on the Integration of planning, acting and sensing (section 2.1), on handling uncertainty in robot planning (section 2.3), and on the Integration of planning and learning (section 2.2).

2. State of the Art

2.1. Integration of Planning, Acting & Sensing

Section proposed by Bernd Schattenberg, University of Ulm

2.1.1. Planning and Acting

The generation of plans for autonomous systems like robots and software agents cannot be separated from their execution. Mostly, actions are intended to be performed within dynamic and unpredictable environments. This makes any plan extremely susceptible for failures. One approach to overcome this problem is described in another part of this document: The domain model explicitly represents uncertainties in the environment specification and provides strategies for all execution problems. However, the techniques for handling uncertainty depend on an explicit and accurate domain analysis, e.g. probability distributions.

A more pragmatic approach is to combine a classical plan generation process with the plan's execution. The execution component continuously reports the planning system about the progress being made, including action failures and unexpected world changes. The transferred information enables the planning system to adapt (partly) committed plans or to generate new (sub-)plans. All of the presented approaches agree, that this is necessary for the flexibility and robustness of the overall system. Another benefit is the way domain models are created: Failures do not have to be foreseen explicitly.

Surprisingly, this technique was not adopted in many planning and execution systems. Only few up-to-date planners are capable of continuous interaction with their execution clients.

As a rule, the control of autonomous hard- or software agents is done today by reactive systems. We find architectures with symbolic representations like *reactive action packages* in RAPRAP [86] and that used in the *procedural reasoning system* PRSPRS [104]. Also very popular in Robotics is SAPHIRA [148] with its fuzzy controllers and derivatives of the *subsumption architecture* [56], which are characterized by *layers of competence*.

All these approaches have in common the notion of *behaviors* of the execution components. The term behaviors addresses fixed pre-compiled patterns of actions which are selected depending on the actual situation of the executor. As a first step towards the integration of a *deliberative* planning process into such a plan execution environment, there are systems where the executors send "off-line" queries to the built-in planners, e.g. [93], [102] or [158].

An alternative system design puts the execution control into the planner itself. Such systems typically generate plans to perform the entire task and monitor their realization afterwards. One of the first planners following this approach was PLANEX [85]. It uses the STRIPS planning engine and controls plan execution by monitoring state changes at execution time. In case of an execution failure, it continues to execute independent, unaffected parts of the plan. If no such independent sub-plan is left, a new (complete) plan is generated. A more recent representative of this technique is the *execution agent* of O-Plan [67].

Many researchers argued that repair plans have to be provided to enhance performance. In the case of execution failures, they allow to complete the original task, starting from the current failed state. Unfortunately, planners in most frameworks typically get unspecific feedback information like "action *x* has failed". This means that computational expensive "from-scratch" re-planning has to be done. To make re-planning more tractable, some systems provide their execution layer with reactive plan repair capabilities, e.g. [160].

In [125] planning problems are regarded as specifications for complex actions. The generation of plans corresponds to their execution in form of logic programs. IPEM [23] (*Integrated Planning, Execution and Monitoring*) was the first system to integrate non-linear planning and plan execution. Partial plans are completed stepwise using classical non-linear techniques. After every such step a sequence of executable actions is determined. Their execution leads to a new current state which is the basis for the next completion step in the planning sub-system.

The *Continuous Planning and Execution Framework* CPEF [134] is a system with emphasis on the interaction of planning and execution in an asynchronously working architecture. It relies on the planning system SIPE-2 [168] and PRS [104] for execution management. In this framework plans can be generated to arbitrary levels of refinement and then be manipulated at runtime by the executor component. *Manipulating* means translating into sequences of executable actions or repairing of flaws.

The SAGE-system [109] tightly integrates plan execution into the planning process. This is done by extending the non-linear planner UCPOP [143] by two new types of flaws: *un-executed* and *executing*. The latter is introduced to express the fact that actions are not instantaneous. An action cannot be executed until all of its preconditions have been satisfied and achieved by executing preceding actions. After an action is marked executable, SAGE delays its execution as long as possible in order to avoid committing prematurely to a partially constructed plan. The planner runs continuously and returns results as soon as they are obtained. SAGE provides an advanced re-planning capability. It introduces user-defined and domain-specific failure handlers which remove a failed portion of the plan and update the domain model accordingly. This avoids repeating failures during re-planning and re-executing.

The reactive planning system BURTON [169] is developed for autonomous spacecraft control. It uses compiled temporal logical descriptions of state transition models of the various spacecraft components and generates from them reliably correct instruction sequences. It is the first system to use consequently symbolic representation of the application domain down to the lowest level of plan execution.

The *New Millennium Remote Agent* NMRA [142], the predecessor of *Burton*, is the first AI planning system to be used in an autonomous spacecraft. The system controls Deep Space One, which has been launched in October 1998 to explore the asteroid 1992KD. NMRA does not have the elaborated modeling approach of BURTON yet, but although some modules of the probe are controlled exclusively by an on-board execution module, the high-level control is the planning system itself and has been successfully tested several times during its mission.

2.1.2. Planning and Sensing

The last section addressed many problems concerning performing tasks in uncertain and changing domains and it proposed some solutions by integrating planning into execution control. However, the basic assumption of most of these systems is that at any time complete and correct knowledge is available for free, which of course does not hold in any realistic environment. Therefore, this section will deal with the aspect of acquiring information that is necessary for a successful mission accomplishment.

It is of particular importance for robot planning systems to keep knowledge about the environment consistent and up to date. In this context abstract and artificial looking cost-models known from agent-literature get very concrete and obviously mission-critical. The costs for obtaining information are directly recognizable and understandable: It may take a robot many actions to perceive knowledge about its environment. Furthermore, some of the sensing actions may be irreversible. An example for information gathering is moving to a distant location and then using an optic sensor to check an object's presence. But the reader may also think of performing some chemical experiment to determine the concentration of a volatile toxin.

Two basic methodologies will be discussed in the following two sections: we will refer to them as *Planning for Sensing* and *Sensing for Planning*.

2.1.3. Planning for Sensing

This methodology uses planning to support the sensory actions of an agent or robot, and it can be found in most existing systems. The basic idea is to add knowledge gathering actions to a contingency planning approach. These sensing actions satisfy *knowledge goals* which are unknown domain variables in open preconditions. Such frameworks represent the outcome of sensing by conditional branching. Please note, that the term *conditional* branching is used to distinguish alternative action sequences and the planning search tree. Many terms in this area are not uniform: E.g. some approaches call the root node of a conditional branch *contingency*, some do so for the paths along the branches.

The semantics of action nodes and conditional branches are typically that of *possible worlds* in an epistemic modal logic. Unknown domain variables induce several successor worlds, one for each possible value. Sensory actions allow to separate the successor worlds, i.e. in terms of the logic to "know" that an expression has a given value.

An example: The planner inserts a sensing action for an *unknown* Boolean-valued information in the current partial plan. This creates two conditional branches, representing a partitioning of the successor worlds: One in which the value of the fact is *known* to be true, and one in which it is *known* to be false. Execution of this sensing action enables the agent to *know* at execution time in which partition of the possible worlds it is and to pursue the correct plan path.

The logic based planner presented in [96] can be viewed as the implementation of the methodology described above. Actions are modeled in an epistemic description logic framework [95]. It distinguishes between *moving actions* and *sensing actions*. The *moving actions* are formalized as in classical planning by domain constraints and precondition, effect and frame axioms. The *sensing actions* are using additional axioms involving epistemic operators. Therefore, the formalism can express that after performing a sensing action an agent knows whether some property holds or not, as well as that the sensing itself has side effects.

Related to this point of view is the one taken up in SAGE [109]. It is based on UCPOP [143] and serves as a query planner for the SIMS information mediator [24], which aims at providing flexible and efficient access to large numbers of information sources. The domain is modeled in a KL-ONE type language to determine the relevant sources for a given query, i.e. the contents and characteristic features of a database or knowledge source are represented in a description logic. SAGE incorporates a mechanism called *run-time variables* [23]. These place holders are located in effects of operators to collect values returned by their actions at execution time. Once the values are bound to such variables by successfully executed sensing actions they can be used by the plan execution component in the following plan steps. SAGE delays working on any open condition that involves such run-time variables, and waits for sensed information to update its domain model.

A very important feature of SAGE is its ability to introduce "unforced" additional sub-goals to queries. Thus the system is actively seeking for new information [108], in order to adapt its formal cost model for information sources. This mechanism enables SAGE to compare query plans and to prune the sets of databases relevant to subsequent sub-goals more effectively. PUCCINI [98] is

another partial order planner that performs sensing. PUCCINI serves like its well known predecessor XII as a planning engine for an Internet soft- bot which has correct but massively incomplete information about its environment. The algorithms are based on the idea that a precondition is true either because it is observed to be true or it has been made true by an action. The authors introduce *observational links* that do not induce additional ordering constraints: this enables preconditions to be supported by prior observation or later verification.

The action representation language SADL [97] is used, a derivative of ADL [141]. It supports universally quantified information goals and universally quantified, conditional, and observational effects. The effects are *world state changes* and *world state reports*, the latter assign values to *run-time variables* (see above). The use of a three- valued logic allows to model uncertainty explicitly, which is again described in terms of a possible world semantics.

The authors introduce three annotations for literals in preconditions and goals. One indicates classical goals with the intention to achieve them "by whatever means possible". If they contain run-time variables the agent has to learn the variable's truth value . The second annotation is used to describe so-called maintenance goals, i.e. fluents that must not be changed. The last annotation indicates to sense a fluent at the time the goal is introduced in the plan. Together with the first annotation this can formalize tidiness goals like "restore the initial value". Any un- annotated literal is marked to depend only on the state of the world and not on the agent's knowledge.

An important aspect of this work is the identification of a special, but large class of problem domains, wherein the presented formalism is regarded tractable, because actions can be encoded without knowledge preconditions: the *knowledge-free Markov domains*. In these domains the actions' effects only depend on the state of the world at the time of plan execution. Therefore, all knowledge sub-goals will be of the same form: The agent needs to know the value of a fluent exactly at the time the action is to be executed, and it does not matter if the agent affects the fluent by side effects of sensing.

CASSANDRA [147] is also based on the UCPOP [143] algorithm and is able to handle uncertainty about actions' outcomes. As in classical conditional planning, *secondary preconditions* represent context-dependent effects of the actions. Together with "unknown" preconditions this models uncertain effects. It introduces condition-action rules called *decision steps* and *information gathering steps*. The decision steps make the agent's choices explicit which conditional branches in the plan it should follow. The decisions are based on the knowledge provided by the execution of information gathering steps. Therefore, decision preconditions are treated as knowledge goals and are added to plans each time new sources of uncertainty are encountered. The rule base is built up incrementally during the planning process from those uncertainties that are used to establish preconditions in the given conditional branches. CASSANDRA's labeling system for uncertainties in a plan is similar to that of PUCCINI (see above). Positive labels are propagated to denote plan elements that contribute to goal achievement in the current conditional branch and negative ones that for those which prevent goal achievement. Unlabeled elements mark "context neutral" actions which may or may not be executed on this path. Threat resolution is done in a classical manner by promotion, demotion, separation via non-codesignation, and preservation, i.e. generation of a sub-goal to disable the threatening effect by placing it in a separate conditional branch (*conditioning* in [144]). In contrast to most other conditional planners, CASSANDRA is able to reunify the proper conditional branches in partial plans.

Sensory Graph-Plan SGP [166] is an extension of the well-known GRAPH-PLAN system [48]. It handles planning problems with uncertain initial states and with actions that combine causal and sensory effects. The underlying semantics is again that of possible worlds which the system tries to make distinguishable for an plan-executing component. To this end, specific epistemic formulas are used to express that an agent knows in which world it currently is.

Each action has the usual causal effects plus zero or more *observational effects* denoted by arbitrary logical expressions composed of GRAPH-PLAN propositions. Such effects return one bit of information when executed: True if the logical expression is true in the world immediately prior to execution, and False otherwise. Consequently, action instances are linked to preconditions in two separate conditional contexts. This situation is mirrored in a layered planning graph with each layer representing one possible world.

Another planner that has been extended to handle contingencies and information acquisition is the probabilistic planner BURIDAN [113]. The resulting system [79], allows actions with causal and informational effects. The input is a probabilistic distribution over the initial states, a goal expression, a set of probabilistic action descriptions and a probability threshold. This makes it possible to define noisy sensors. The system produces a contingent plan that makes the goal expression true with a probability of at least the threshold. Again, the possible worlds produced by observations and uncertain action effects are divided in equivalence classes by a context-based labeling mechanism. Each action in the plan is annotated with a context label which is generated by observations and inherited by previous steps. Labels determine under which circumstances an action should be performed.

The methodologies used in this system differ in some way from conventional techniques. In particular, all operations on the plan aim at increasing the probability for goal satisfaction. Threat resolution in this context is done not only by the classical promotion and demotion mechanisms: *Confrontation* adopts the triggers of "benign" consequences as goals which has the effect of decreasing the probability of the threatening consequence. *Conditioning* is a technique from classical conditional planning [144]. It ensures the threatening step never to be executed in the same execution trace as the producer or consumer of the threatened link. This technique is used within branches that connect information producing actions to subsequent actions, indicating which observation labels of the first permit execution of the second: The system can separate the context of the threatening step from the context of either the link's producer or consumer.

2.1.4. Sensing for Planning

The systems described so far consider the execution component's knowledge only at the time the plan is actually executed. The approaches of the following section try to "complete" the integration of the planning system in the overall architecture by establishing a bi-directional flow of information and thereby providing the planner with domain knowledge acquired at run-time? This paradigm – *sensing for planning* – incorporates information producing processes into planners and is motivated by problem descriptions of realistic domains.

In general, search space for conventional conditional planning explodes for relatively small ranges of the contingencies. There are many examples, we give a simple one here: The robot has to get the right key for a door identifiable by an in advance unknown 5-digit number. Even a few doors make this problem intractable. A pragmatic approach in this situation would abstract from the key's identification and solve the contingency by "looking at" the door lock at execution time. Then it could plan how to get the key – of course assumed, that this can all be done without getting stuck in dead-ends.

But there is made another assumption by all systems that is not trivial at all and very hard to relax: The sensory input is given as logical predicates. The question how to build symbolic information out of sensor readings is a wide field of research but out of the scope of this document.

In [30] the authors use constraint satisfaction techniques to determine the value of preconditions in a UCPOP-like planner [143]. The CSPs are formulated to ensure consistency of the partially ordered plan- sequences during the planning process.

The system incorporates *interactive constraints* for acquiring unknown domain values at run-time. Such binary constraints are treated as follows: If no variable is associated with a domain, then the constraint is suspended. If both are already associated, then the constraint is propagated as in a classical CSP. In case of only one variable being "unknown" knowledge acquisition is performed which leads either to some characterization of the unknown variable's domain or to a failure. Finally, both variables can be instantiated and the Boolean value of the constraint can be calculated. BUMP

[138] is an agenda-based planning system that follows a more classical planning paradigm. Sensing actions are represented in the same way as non-sensory processes. The preconditions of such sensor operators can be used for set-up actions. Following a STRIPS-style representation, Add-lists contain at least the sensed information and Delete-lists refer to additional side-effects of the sensors. BUMP uses special dummy constants for values that will be acquired by sensor readings. Any subsequently processed goals that refer to one of these constants will be deferred until the executor has obtained the reading. Execution begins as soon as all goals in the partial plan have been either solved or deferred. When selecting the actions for execution the controller prefers sensor processes over other parallel plan steps. Once a requested sensor reading is obtained, BUMP is restarted immediately with the new information. The new plan is returned, and the process proceeds until BUMP has found a complete plan.

In this context the authors discuss the problems in finding a suitable and efficient sensing strategy: The planner has to select the appropriate goals for deferral at the appropriate time. Therefore, they present an execution cost-model for plans with sensing actions and its empirical evaluation [137].

2.1.5. Conclusion

Although much progress has been made in the field of integrating planning, sensing and acting into one common framework, there are many aspects still not addressed in this area: There exist only few formal models of the information gathering process during planning. The integration of new knowledge, that perhaps even rules out already made commitments in the plan, is an open question, especially when looking at correctness or even completeness of a given planning method. Another question is the executability of plan steps; today's systems rely on benign environments, heuristic information or make strong assumptions about the domains. A third aspect is the combination of all three concepts: planning, sensing and acting. The systems so far only deal with two of them. We think that an interleaving planning approach combined with information acquisition during plan generation would have strong synergetic effects towards stable and efficient autonomous systems.

2.2. Learning in Robotics

Section proposed by Daniel Borrajo, José Manuel Molina, and Araceli Sanchis, University Carlos III of Madrid

2.2.1. Introduction

Many robotic systems applied in industry are autonomous mobile robots working in stationary environments, i.e. automatic floor-cleaning, automatic assembly, transporting parts in a factory, etc. Other applications of robotic systems involve interactions with dynamic environments, where the autonomous robot deals with unexpected events, such as tour-guiding robots. The successful operation in such environments depends on the ability of adaptation to the changes. Thus, for most agent-based tasks, having a perfect domain theory (model) of how the actions of the agent affect the environment is usually an ideal.

We differentiate here three ways of providing such models to agents (planners/controllers):

- High level planning, learning and control. One of the first approaches to planning, learning, and execution within autonomous robotic tasks was the Shakey robot [136]. It had a planner, STRIPS, and a learning method based on compacting the solutions (plans) to given problems into new operators that could be used in future planning steps, as well as means for replanning in case things went wrong [85]. However, it was soon discovered that approaches based on the classical paradigms (abstraction, planning, heuristic search, etc.) were not completely suitable for unpredictable and dynamic environments.
- Manual design of reactive planners. Other approaches consider reaction as the new paradigm to build intelligent systems. One classical instance of this kind of architecture is the subsumption architecture which was proposed by Brooks [56] and has been successfully implemented on several robots. The base of the subsumption architecture is a piece of code called "behavior" or "skill". Each behavior produces an action (reacts) in a given situation, and the global

control of the "planning" system is a composition of behaviors. Different systems, from finite state machines to fuzzy controllers [105], have been used for the implementation of these behaviors. In most cases, the way in which these behaviors were built was by careful and painstaking "ad-hoc" manual design of skills by a human [55, 66].

- Automatically acquiring skills for reactive planners. Another approach that solves the above mentioned disadvantages of manual design consists on automatically learning those behaviors. There have been already many different approaches for learning skills in robotic tasks, such as neural networks [131], genetic techniques [76, 111, 124, 130], evolutionary strategies for configuring neural networks [43, 121], inductive collaborative techniques [92], or reinforcement learning techniques [122, 152].

Another task that can be learned consists on acquiring maps for navigation purposes, as in the XAVIER robot [110], or learning to improve path planning as in the ROGUE system applied to the same robot [101]. As an example, the RHINO robot employed learning capabilities for localization and navigation[161, 59]. Another example is the work by one the previous authors [163] that uses neural networks and naive Bayesian learning for generating grid based maps as well as topological ones.

In this document we will cover some examples of the use of machine learning for robotic tasks, mainly for learning skills. Other books and surveys cover in more detail many of these aspects [65, 89, 151, 165], as well the sequence of workshops on learning robots.

2.2.2. Neural networks

The concept of designing new computational structures using all the available information on the very efficient world of biology, specially the case of neural networks [88, 54] is a fascinating approach. The most interesting feature of this approach is that biological neural networks have been designed to be adapted to the real world [99, 135]. This suggests the biological approach as the most appropriate to solve the problems existing in the mobile robotics field [94, 69]. The redundancy, the parallelism and the resulting robustness of the computational structure are the interesting points of the biological approach. These mechanisms are not only exploited inside the brain structure, but also at the level of groups of animals, giving very interesting results in collective behavior of insects, for instance.

One way to use a neural network for learning a robot skill consists in the use of a feed forward network with n input units and m output units [128]. The inputs are usually related to the robot sensoring mechanism, while the output units represent the actions to be performed, such as wheel velocities. The use of a neural network controller has several advantages [128]: neural networks are resistant to noise, exist in real environments, can generalize their ability to new situations, and could easily exploit several ways of learning during its lifetime

Another successful approach to learning a robotic skill by using neural networks has been the RALPH and ALVINN systems [145, 146]. In this case, the learning task was to learn how to drive an autonomous vehicle in a highway.

2.2.3. Evolutionary Computation in Controllers

In the last few years, new approaches that involve a form of simulated evolution have been proposed in order to build autonomous robots that can perform useful tasks in unstructured environments [57, 64, 123] . The big interest in this approach is due to the dissatisfaction with traditional robotic and Artificial Intelligence approaches, and their belief that interesting robots may be too difficult to design. Thus, it would appear reasonable to use an automatic procedure, such a genetic algorithm, that gradually builds up the control system of an autonomous agent by exploiting the variations in the interactions between the environment and the agent itself. It remains to be determined if it is feasible.

In particular, two questions should be answered: what to evolve? and how to evolve it? The choice of what to evolve is controversial. Some authors have proposed to evolve controllers in the form of explicit programs in some high-level language. Brooks [57] proposes to use an extension of Koza's genetic programming technique [112]. Other authors propose to evolve controller rules using a form of classifier system [74, 77, 75, 130], or using a fuzzy controller [124]. Finally, others propose to evolve weights in neural network controllers, fixing the architecture [26, 43, 44, 87, 128].

2.2.4. Reinforcement learning

Currently, one of the most common approach to learning robots behavior is based on reinforcement learning techniques [106, 156, 153] within the Markov Decision Processes (MDP) paradigm [122, 155, 164], or Partially Observable Markov Decision Processes (POMDP) [61]. The main objective of reinforcement learning is to automatically acquire knowledge to better decide what action an agent should perform at any moment to optimally achieve a goal. Among many different reinforcement learning techniques, the most cited one is Q-learning [164]. Usually, they integrate reinforcement learning, planning and execution based on approximated dynamic programming. In order to define any reinforcement learning problem, some issues have to be considered [129]:

- **Delayed reward.** Other learning systems require knowing for each learning episode/instance (perceived state and action to be taken) the class they belong to (yes, it achieves the goal, or not, it does not achieve the goal). Many times, the agent does not immediately know whether it has achieved its goal after execution an action, since it might be only achieved (or not) after the execution of several actions. This creates the *temporal credit assignment problem*, that consists on "a posteriori" marking which actions from a sequence of actions have a positive or negative impact on achieving or not the goals. These marks allow the proper reinforcement of the learning instance.

- **Exploration versus exploitation.** In many discovery/learning tasks, there is a relation between using pre- acquired knowledge when planning (exploitation), or trying new alternatives (exploration). In those cases, an exploration strategy must be defined, and it influences the learning rate and its quality.

- **Partially observable states** In many situations, sensors of agents provide only partial information of the domain. This problem implies that an agent could receive from the environment states with different levels of information, that could make the agent not to differentiate some states from others, or perceiving two different states as equals.

- **Nondeterministic rewards and actions.** In many environments, the execution of an action in a given state does not always arrive to the same state, because of noise associated to sensors, and actuators. This implies that the reward of executing an action in a given state, can be different depending on if the next observed state is a goal state or not.

- **Integration of several acquired skills.** In most domains, different behaviors are learned, and the problem is how to combine those acquired behaviors. Most of the times, the solution involves a decomposition of the learning task in subtasks, and later learn the skills associated to those subtasks [152].

- **Representation of action descriptions.** In the case of classical reinforcement learning techniques, there is no representation of the actions, since they are not used for explicit planning. It is only needed to keep the action name, and the procedure for executing each action in the environment.

- **Global vs. local reinforcement.** The reinforcement procedure for most techniques is local to an operator (the term operator is understood here as the connection between two states through the execution of an action).

- **Representation of states.** In most cases, instantiated states are used, or non-symbolically based generalized states when the state space becomes huge as mentioned below.

- **Type of planning scheme.** Reinforcement learning has been usually applied for reactive planning (with some exceptions).

- **Temporal credit assignment problem.** In some tasks, the concept of time delayed reward is very important; how to assign credit to actions whose reward is only known after some time has passed [154].

- **Cooperation among agents.** Some people have focused on the interaction among different agents [159]. He explores the cooperation among agents by sharing instantaneous information (perceptions, actions or rewards), sequences of perception-action- reward, and learned policies.

- **Large state and action spaces** When using reinforcement learning techniques with large state and/or action spaces, an efficiency problem appears: the size of the state-action tables. Current solutions to this problem rely on applying generalization techniques to the states and/or actions. Some systems have used decision trees as in the G-learning algorithm [62], kd-trees (similar to a decision tree) in the variable resolution dynamic programming algorithm [132], the PartiGame algorithm [133], neural networks [118], vector quantization [84], or belief networks [51, 68].

2.2.5. Integrated approaches

Approaches that integrate planning, learning and execution for robotic tasks are:

- Christiansen [63] also addresses the problem of learning operators (task theories) in a robotic domain.

- Other systems for robotic tasks are [42] and [107]. The first one deals with the concept of *permissiveness*, that defines qualitative behavior for the operators. The second one uses Inductive Logic Programming for learning the operators of the domain by doing a transformation from the sensor data into predicate logic. They need some type of prior background knowledge, either a predefined domain theory in the form of initial operators, or external instruction and knowledge on how to perform the transformation.
- Other approaches have only been applied within simulators, such as Lope [92], that allows to learn a model of the environment by performing actions on it, and observing the new states it arrives to.

2.3. Robustness and Adaptivity in Planning

Section proposed by Patrick Fabiani, ONERA, Toulouse

2.3.1. Decision-theoretic planning for tracking problems

Robot tracking problems are a challenging combination of different planning problems that have previously been studied in Robotics [115], but separately. Visibility and collision constraints must be satisfied in the presence of uncertainties in the positions of the pursuer and the target.

Navigation can be based on odometric and landmark techniques, with easily recognizable landmarks like in [31]. Natural landmarks can be used as well [72, 100], but require more sophisticated vision techniques. Landmark-based navigation has been addressed from different points of view in the literature [50, 149, 157, 117]). The principle is simple: if the robot primarily localizes itself relative to landmarks, the planner must guarantee that the robot will see landmarks often enough along its path [90].

The problem of maintaining visibility with the target while avoiding collision with the obstacles is addressed in [116], but position uncertainties are not taken into account. When the target is fully predictable, that is, when its future trajectory is completely known in advance, a dynamic programming approach [45] can be used to compute a trajectory of the pursuer that has minimal length.

As shown in [116], this approach becomes intractable in practice if the target is only partially predictable. Then, the approach can be applied to choose a motion command that aims to maximize the likelihood that the target will remain visible during a short interval of time in the future. The pursuer iterates this computation while tracking the target and updates its motion heading at each iteration. This is essentially the approach taken in [81], where the essential contribution is that the planner takes into account the uncertainties in the pursuer's and target's positions.

Taking position uncertainties into account naturally leads to planning the pursuer's motions in order to take advantage of the landmarks in the workspace to reduce the imprecision of the estimate of the pursuer's position (and consequently that of the target), whenever this does not immediately conflict with keeping the target in the pursuer's field of view. This motivates the embedding of this approach into a game-theoretic framework.

The theory of Games and Decision [150, 120] provides a convenient framework to express dynamic decision problems in the presence of uncertainty. [78, 103] provides an overview of problems of reasoning with uncertainty in Robotics and probabilistic approaches are most often chosen. [114]

presents an application of Hidden Markov Models to robot navigation. Yet, probabilistic inference in Bayesian Networks [140], Markov Decision Problems (MDP) [119] or Partially Observable MDP's (POMDP, [167]) of quite reasonable size may be a task of high computational complexity [53]. [25] presents an application of MDPs to robot navigation and propose state aggregation in order to reduce the computational complexity. [52] is remarkable as proposing to reuse ideas from Graphplan [49] for reachability analysis in solving MDPs in general. The idea in [127, 82, 126] is to try to follow a similar approach and compute efficiently a pruned planning graph in which to perform plan optimization. The problem is to be able to keep track of an optimization criteria in the planning graph, which is not easy in the original Graphplan approach.

2.3.2. Discussion

On the basis of this state of the art, let us now discuss the notions robustness and adaptivity with respect to deliberation, planning and reactivity (not taking into account learning). We consider autonomous systems with perception capabilities in an uncertain and changing environment, in which closed loop acting is generally required. Let us distinguish:

- *Reactive systems*: correspond to closed loop controlled systems and are based on control laws taking sensing inputs and giving command outputs, their behavior is reactive, but this does not mean that they rely upon planning capabilities: navigation with repulsive potential fields based on range sensors, or elastic band control laws based on the detection of a target reference is generally not considered as planning - such methods can be used efficiently provided that one can prove that there will always be a correct behavior in all situations, which requires a robust conception.
- *Adaptive planning*: correspond to closed loop planning or conditional planning: the planner uses information on the environment and the sensing capabilities to produce (on-line or off-line) a conditional plan, or a strategy, that will then be used on- line to obtain an action to perform depending on the sensing inputs or estimated state of the robot; the optimization stage of the plan can be obtained by any technique, including potential fields, elastic bands, or compliant motion methods, the main difference being that the *planner will verify before execution the achievement of a termination condition or the performance of a desired level of service along the plan.*
- A system can be *open-loop* controlled and follow a *robust* plan that works whatever happens in the environment.
- Systems with adaptive planning capabilities have a reactive behavior, whether the plan is produced on-line or off-line. On-line adaptation of the plan, would the be called adaptive replanning to our sense, the difference being that replanning is needed whenever the planner's ”knowledge” about its environment and capabilities has been altered so that the robot does not have a solution for the new ”possible cases”. This is clearly debatable.
- Systems with planning capabilities do not necessarily have a reactive behavior, if the planner is able to produce open-loop plans that work in all situations.

Consequently, a robotics system relies upon planning and deliberation when it is able to optimize/verify on the base of its ”knowledge” about its environment and capabilities and before execution the achievement of a termination condition or the performance of a desired level of service.

According to what is said above, in order to assess the usefulness and eventual need for planning in robotics, let us remark that:

- *reactive systems*, without planning capabilities can be efficient in cases where there reactive behavior is proved to be correct behavior in all possible situations, which requires a *robust conception* and thus generally correspond to special cases (analytical solutions, special hypotheses);
- *non-reactive systems* can work if they have *robust planning capabilities*: an open-loop plan that achieves the goals and performs the desired level of service whatever the uncertainties and the changes in the environment is a **robust plan**. Situations where this is ”always” feasible seem to be sufficiently rare, fortunately for researchers in adaptive planning techniques. Producing a robust plan requires to take into consideration all the possibilities, thus considering them as a set for which there can be a **compact representation** (geometry in navigation, ...). Finding a solution may be hard, yet deterministic planning techniques apply in that context.
- *adaptive planning* can produce more or less robust solutions but generally requires to optimize a sequence of decision over a number of possibly reachable states, or over a space of possibly reachable information states (space of probability distributions over the state space) ; thus *adaptive planning* can be considered as generally more costly, and often considerably more costly than classical deterministic planning techniques.

A robot using planning will not necessarily be more effective than a robot that does not, but robot without planning capabilities can only be designed so that it achieves its missions in special cases where a robust and optimized behavior can be achieved by specific means, under particular special hypotheses. Those hypotheses have to be precisely stated and verified, otherwise the behavior of the robot can appear to be not optimal. Planning techniques extend the variety of missions that robots can achieve and can also allow to prove the optimality or correctness of a plan, or the non-existence of a plan in a wider set of environments. Once again, this discussion *does not take learning capabilities into consideration.*

Autonomous systems need planning capabilities in order to decide on the best sequence of actions and perceptions to perform either to achieve a specified goal in a state space, or to perform the best level of service on a specified planning horizon. Tracking problems correspond to the second category. Yet, most often the problem is eventually solved by planning intermediate goals to achieve, falling though a little bit in the first category.

As a matter of fact, recent work on a problem of tracking a partially predictable target with uncertainties and visibility constraints [81, 80] shows that in uncertain and changing environments, there can be a planning horizon that achieves the best compromise between the robustness of the plan with respect to uncertainties and its adaptivity to the changes in the environment: real time constraints, but also limited perception capabilities can lead to prefer a shorter planning time over a shorter predictive horizon. In that case, planning should consist first in finding the right planning horizon and define the goal to achieve on that horizon from the initial problem definition.

When there are real time constraints in a tracking problem, two time scales are “competing” :

- Δt is the time for the target to move by a distance of Δx ;

- $\delta t(\Delta t)$ is the time for the planner to compute the best moving decision taking into account all possible moves of length $\leq \Delta x$ of the target during time Δt.

For the planner to be *robust*, we need:

$$\delta t(\Delta t) \quad < \quad \Delta t$$

otherwise the plan is produced *"too late"*.

Similarly, when the pursuer's perception capabilities are limited, two time scales are "competing" :

- Δt is the time for the target to move by a distance of Δx ;
- ΔX is the maximum radius of the region of the state space in which the pursuer is *"certain to find the target in one observation"* - ΔX is limited by the perception capabilities of the pursuer.
- ΔT is the time for the target to move by a distance of ΔX ;
- $\delta t(\Delta x)$ is the time for the planner to compute the best moving decision taking into account all possible moves of length $\leq \Delta x$ of the target.

For the planner to be *robust*, we need :

$$\delta t(\Delta X) \quad < \quad \Delta T$$

otherwise by time the plan is produced (*"too late"*) the region that can be reached by the target cannot be covered in one shot by the pursuer: the pursuer is not guaranteed any more to keep track of the target. On the contrary, if $\delta t(\Delta X) \quad < \quad \Delta T$, then the pursuer is guaranteed to keep track of the target.

The planning horizon achieving the best *compromise robustness/adaptivity* would then be h such as:

$$h = \min\{\Delta T,\ \max\{\Delta t/\delta t(\Delta X)\ <\ \Delta T\}\}$$

Producing a robust plan, is closer to a satisfiability problem, like in classical planning. Yet, robots have to deal with situations where the robust solution does not exist and classical planning approaches cannot produce an *"optimized compromise"*, which is the job of decision-theoretic or game-theoretic planning. On the other hand, decision-theoretic planning is computationally complex and probably far less efficient than classical planning techniques when a robust solution exist.

3. Robot planning research at TCU nodes

3.1. GMD

Joachim Hertzberg
http://ais.gmd.de/ARC/

PLANET work at GMD is anchored in the Robot Control Architectures (ARC) team of the Institute of Autonomous Intelligent Systems (AiS).

In ARC, we investigate hybrid robot control architectures that amalgamate reactive components working in close sensor-motor coupling, on the one hand, and deliberative components working on an explicit symbol level, on the other hand. Of particular interest are the flow of information from the sensor-motor components to the symbol level, and the way in which the reasoning components modify physical robot action.

The rationale for this work is this: As a piece of software, a robot control system is special in that it must cope with data in many different grades of granularity (from sensor readings to user-supplied mission data) and yield purposeful action on different time scales (form collision reflexes to optimal long-term mission planning and organization). Accordingly, a robot control program needs a special structure and organization that integrate these incoherent pieces into coherent overall action—it needs a special architecture. However, to quote R. Arkin [1] (p. 207): ”The nature of the boundary between deliberation and reactive execution is not well understood at this time, leading to somewhat arbitrary architectural decisions.”

The vision behind our work is to formulate and demonstrate structure and organization principles for robot controllers as well as a controller design methodology that together allow concrete controllers for concrete robot applications to be designed in a principled way.

We approach the problem from three different angles:

Tools Arguably, a robot controller is best designed in the programming style of concurrent programming; it should be open towards running distributedly on several computers as well as towards controlling a team of physically different robots or mobile/stationary hardware components. The high-level control programming environment as well as generic control architectures must support this style. Based on work about the Flip-Tick Architecture (FTA, [5]), we aim at developing a high control system specification and implementation level, on top of which concrete or generic robot control architectures can be specified.

Case Studies We are building robot controllers for concrete robots in the sense of case studies. These studies are application- driven (like, build the mission-level controller for a sewer robot, [4]) or method-driven (like, build a controller based on Dual Dynamics and classical action planning, [2]). Driven both by application needs and by interest in the respective methods, we are investigating active perception as an instance of a robot control issue [3] that, by definition, comprises reflexive and deliberative aspects, involving low-level, raw sensor data as well as high-level, abstract, symbolic data and tasks, which have to be integrated.

Methodology We are studying control in natural cognitive and/or biological agents, as far as reported in the literature.The aim is to identify innovative control principles or features that can be operationalized on a robot control architecture level, and to provide prototype implementations thereof.

3.2. LAAS, Toulouse

Rachid Alami, Malik Ghallab, Felix Ingrand
http://www.laas.fr/RIA/

The Robotics and AI group at LAAS, is currently working on various research themes related to Robot Action Planning.

3.2.1. Robots that cooperatively enhance their plans

Following our work on multi-robot cooperation, we have developed a general architecture for multi-robot cooperation based on a scheme called "M+ Cooperative task achievement".

This scheme is essentially based on on-line combination of local individual planning and coordinated decision for incremental plan adaptation to the multi-robot context.

Its main originality comes from its ability to allow the robots to detect and treat - in a distributed and cooperative manner - resource conflict situations as well as sources of inefficiency among the robots.

The system has been illustrated through a simulated system, which allows a number of autonomous mobile robots to plan and perform cooperatively a set of servicing tasks in a hospital environment.

3.2.2. Propice-Plan: Toward a Unified Framework for Planning and Execution

PropicePlan [70, 71] is an integrated software designed to investigate the links between planning and plans execution. It implements supervision and execution capabilities, combined with different planning techniques:

- plan synthesis to complement existing operational plans
- anticipation planning to advise the execution for the best option to take when facing choices (by anticipating plans execution), and to forecast problems that may arise due to unforeseen situations.

This system relies on a common language to represent plans, actions, operational procedures and constraints, in order to make transitions between planning and execution activities seamless.

PropicePlan is used in two complex real-world problems: planning and control for autonomous mobile robots, and for the transition phases of a blast furnace.

3.2.3. A non-deterministic planner

We have developed a new planning approach that authorizes an autonomous robot to reason about the inaccuracy of the world description and of its possible evolutions. We represent the uncertainty with the possibility theory; this allows us to distinguish between two types of non-determinism: a non-determinism from insufficient modeling and a non-determinism from uncertainty. Besides, we introduce perception actions as well as a model of the environment dynamics through "contingent events". We have implemented an experimental planner, based on Graphplan search paradigm. This planner is able to produce plans that are robust with respect to contingent events, and whose goal-achieving ability is evaluated a priori. The obtained plans can be conformant or conditional depending on the context and the user requirements.

3.2.4. Integrating Planning in LAAS Architecture

One important research theme related to Robot Action Planning is how to embark planning onboard robots, and how to integrate it with the other components.

LAAS Architecture (see [22]) allows a robot to plan its tasks, taking into account temporal and domain constraints, to perform corresponding actions and to control their execution in real-time, while being reactive to possible events.

It is composed of three levels: a decision level, an execution level and a functional level. The later is composed of modules that embed the functions achieving sensor data processing and effector control. The decision level is goal and event driven, it may have several layers, according to the application; their basic structure is a planner/supervisor pair that enables to integrate deliberation and reaction.

The LAAS architecture relies naturally on several representations, programming paradigms and processing approaches meeting the precise requirements specified for each level. We developed proper tools to meet these specifications and implement each level of the architecture: IxTeT a temporal planner, Propice a procedural system for task refinement and supervision, Kheops for the reactive control of the functional level, and for the specification and integration of modules at that level. Validation of temporal and logical properties of the reactive parts of the system, through these tools, are presented.

3.3. ONERA, Toulouse

Patrick Fabiani, Roger Mampey, Yannick Meiller
`http://www.cert.fr/dcsd/cd/planetnode.html`

Our research group addresses problems of decision and planning for autonomous systems in cooperation with other systems in uncertain and changing environments, which is mostly the case in realistic applications. The group's research activity in planning is more specifically dedicated to aerospace and underwater robots, but inevitably also to ground robots. Another activity of the research group is to develop decision aid tools either for the management of aerospace systems or for the conception and control of large systems (traffic, logistics, production).

Past work in our research group addressed the problem of plan execution control and robot motion planning:

- Studies on planetary rovers led to the development of a plan execution control tool *ProCoSa*, that has been successfully implemented recently on an underwater exploration robot [29].
- Work on the problem of dealing with uncertainties and plan robustness led to address both problems of perception planning in robot navigation [90] or in surveillance applications [83] and the issue of hierarchical motion planning [91].

Our group's general approach is at the crossing of Automatic Control and Artificial Intelligence in the sense that we have been trying to combine a decision-theoretic or game-theoretic framework for decision making, hopefully based on a tractable probabilistic representation of uncertainties on the one hand and geometric and symbolic reasoning tools on the other hand. This was equally true in the automatic surveillance project *Perception* [27] or in the *Tandem* project on a mixed patrol of aircraft including unmanned air vehicles (not to mention multiagent aspects of the problem). This is particularly clear in more recent work on a problem of tracking a partially predictable target with uncertainties and visibility constraints [81].

Lately, the opportunity of using more widely classical decision- theoretic frameworks such as POMDPs in order to solve realistic robot planning problems has been studied [139] [58], yet without really conclusive results. The computational complexity of solving the stochastic dynamic programming equations often makes the problem intractable in most reasonably realistic applications. POMDPs lead to the use of very general probability distributions over the state space without any compact representation. Yet recent work leads us to further investigate in that direction and try to combine compact representations of uncertainty, symbolic or geometric tools and stochastic optimization algorithms.

A first approach is to try and use the specifics of the planning domain and use symbolic reasoning tools in order to reduce the size of the search space. Trying to reuse recent approaches in classical planning, preliminary results concern the use of tokens in a graph planning approach [127, 82, 126] in order to keep track of the optimization criteria and control the level of state space splitting at planning time. Other work addresses the problem of multimodel estimation for decision making on the basis of a compact, but yet not unimodal, probability distribution : an application to robot localization is described in [28].

3.4. Örebro University

Alessandro Saffiotti
`http://www.aass.oru.se/Research/Robots/`

Research related to planning at Örebro University is carried out at the Mobile Robotics laboratory of the Center for Applied Autonomous Sensor Systems (AASS). The objective of this lab is the development of general principles and techniques that allow the autonomous operation of mobile robots in natural, dynamic, and uncertain environments. By "natural" we mean environments that have not been specifically modified to accommodate the robots. We are especially interested in the issue of how to connect abstract reasoning and physical interaction with the environment.

In the context of planning systems, autonomous robotics presents two major challenges. The first one is the need to combine both high-level means-end reasoning, and fast reactions to changing environmental conditions. The second one is the massive presence of uncertainty induced by several heterogeneous sources, including the actions of other agents, incomplete prior information, limited sensing capabilities, and imperfect robot's actions.

The following research lines of our laboratory are related to issues of planning.

- *Integration of behavior-based execution and behavior planning.* We study the integration of means-end reasoning with execution of lower level robot behaviors. The main issues here are: (i) the development of a formal framework to relate execution of composite behaviors to achievement of composite goals; (ii) the definition of B-plans, a formalism for purely reactive plans based on fuzzy logic; and (iii) the study of a model-free form of execution monitoring to be used with B-plans [10, 7].
- *Run-time deliberation in uncertain environments.* We aim at combining results and techniques from the areas of robot navigation and of intelligent agency. In particular, we investigate techniques that allow the robot to decide when to go on with execution, and when to stop and ponder about a new situation. Our initial efforts are directed towards the integration of an existing navigation system based on fuzzy logic with a deliberator based on the so-called BDI model [11, 12].
- *Generating and executing plans under uncertainty.* We investigate the generation and implementation of plans in the presence of several sources of uncertainty. We focus on the impact of uncertainty in representing and reasoning about actions, sensing, and complex plans. We also focus on the problem of detecting unsatisfactory behavior and better opportunities under uncertainty.
- *Anchoring symbols to sensor data.* We are engaged in the theoretical and applied study of the problem of anchoring: the process of creating and maintaining the correspondence between the symbols used by high-level processes to denote objects, and the percept that refer to the same physical objects. In the context of planning, anchoring is needed to ground the symbols used by a planner to describe actions to actual sensor and motion signals. Moreover, possible problems with anchoring (e.g., failure to perceive an object) must be considered by the planner [9, 6].

We are also starting a new research activity on cooperative robotics, where we intend to tackle the issue of planning and executing tasks that require the coordination of perceptual activities between multiple, possibly heterogeneous robots. We use RoboCup as one of the test-beds for this activity.

3.5. Rovira i Virgili University, Tarragona

Miguel Angel Garcia
`http://www.etse.urv.es/recerca/rivi`

The Intelligent Robotics and Computer Vision (IRCV) group at the Rovira i Virgili University is concerned with the development of efficient software solutions to a variety of problems in robotics and computer vision. Three of our working areas are related to robot planning: *Heterogeneous World Modeling*, *Sensor Planning* and *Multiagent-Based Disassembly Planning and Scheduling*. Details about the IRCV group can be found at:

3.5.1. Heterogeneous World Modeling

World models are important components of knowledge-based robotic systems, since they allow the integration of information gathered from different sensory sources in order to obtain a reliable computer representation of the environment in which a robotic system operates. Such a representation is useful for planning the actions of the system, taking into account updated knowledge about its workspace. Actions determined in this way are more likely to perform successfully than if they were generated through pure reactive behaviors that only considered sensory information processed on the fly.

Since robotic systems rely on a wide variety of sensors, a suitable world model must support the coexistence of heterogeneous information. This information may also be synthetically generated, representing a priori knowledge about the environment. Furthermore, a suitable world model must be compatible with knowledge-based systems, which are traditionally based on symbolic representations.

The IRCV group has been working on the definition of a hierarchical world model that supports the integration of heterogeneous information at different levels of abstraction [14, 15]. This model is based on a structured representation that describes the contents of the physical workspace at multiple levels of abstraction through a hierarchy of oriented bounding boxes (OBBs). This hierarchical representation is implemented by means of a frame-based system. The latter provides a high degree of flexibility and facilitates the incorporation of the world model into a larger symbolic, knowledge-based robotic system. The proposed representation supports the integration of global and local heterogeneous information. Global information consists of geometric, scalar and procedural attributes which are either valid or applicable to the contents of each OBB as a whole Local information utilizes adaptive triangular mesh of arbitrary topology and genus to provide a detailed description of the shape of 3D surfaces associated with individual OBBs, as well as scalar data associated with specific points lying on those surfaces.

The IRCV group has also been working on the development of algorithms aimed at the automatic generation of a world model such as the one described above. In particular, we have developed an efficient technique for generating hierarchies of objects from their bounding spheres [19]. We have also been working on the efficient approximation of object surfaces through adaptive triangular mesh [13, 16, 20]. Finally, we have proposed an efficient algorithm to integrate object surfaces acquired from different points of view [21].

3.5.2. Sensor Planning

Determining the locations where a sensor should be placed in order to observe a certain working area is an important problem in robotics, especially when a world model, such as the one described above, is to be built and kept up-to-date. The problem of finding the next position where a sensor must be placed in order to observe the maximum amount of unexplored space, taking into account its previous locations and the amount of space observed so far, is known in the literature as the Next-Best-View problem.

The IRCV group has been working on the development of an efficient solution to the next-best-view problem, considering that the sensing device is a range sensor [17, 18]. A range sensor is a device capable of generating range images. A range image differs from a conventional gray-level image in that every image pixel does not keep a level of light intensity, but a measure of the distance between the sensor and a point on the surface of an object present in the observed scene. The proposed algorithm determines the next location of the sensor through a voting process that takes into account the orientations of the boundaries of the surfaces that have been observed so far. The final aim is the acquisition of all the surfaces that constitute the objects contained in the scene, assuming that the sensor moves over an "observation sphere" that embodies all the objects to be sensed. This voting-based solution is far more efficient that previous solutions based on visibility analysis and optimization techniques.

The research line on sensor planning is a complement of the world modeling research line, since it aims at the automatic acquisition of a world model. Continued work on both lines is going to be carried out in the immediate future after the acquisition of a low-cost range sensor (Triclops) based on stereo vision. This sensor generates both a conventional color image and a range image. Therefore, it allows the recovery of object surfaces along with associated color and texture attributes. This sensor will be mounted on a robotic arm, with the aim of generating automatic world models of complex scenes while the robot wanders inside them.

3.5.3. Multiagent-Based Disassembly Planning and Scheduling

The need for disassembly appears in many industrial activities, including: remanufacturing, maintenance, repairing, recycling and disposal. Determining optimal disassembly sequences is known as disassembly planning. Within disassembly planning, it is possible to distinguish between complete disassembly and selective disassembly. Complete disassembly involves disassembling all the components of a complex object. Selective disassembly involves disassembling a subset of components from the assembly. Selective disassembly is more appropriate for de-manufacturing applications, such as maintenance, repairing and recycling.

The IRCV group is working on the problem of geometric selective disassembly planning, which consists of determining a minimum sequence of components to be extracted in order to remove a predefined number of selected components from a given assembly. Once that sequence is determined, a scheduler must coordinate a multirobot system composed of several articulated arms in order to execute the previous plan and physically carry out the disassembly. We are currently investigating the implementation of a disassembly scheduler based on multiagent technology. Real experiments will be performed upon Lego assemblies by utilizing two CRS A255 articulated robots.

The research line on multiagent-based disassembly planning and scheduling is within the research line on multiagent-based scheduling, with which the IRCV group participates in the Dynamic Scheduling TCU of PLANET.

3.6. Salford University

Alexandra Coddington
`http://www.salford.ac.uk/iti/projects/MACTA/macta.html`

At the University of Salford we are currently working on a project known as MACTA-LAB which started on 1st July 1998 and which is due to finish on 30th June 2000. The MACTA-LAB project continues work undertaken during a previous 3 year project known as MACTA (1st June 1994 - 31st May 1997) which is described in the following paragraph.

During the previous MACTA project a heterogeneous multi-agent system was developed comprising one computer-based software agent (known as a ”Reflective Agent”) and several behavior-based mobile robots (known as ”Behavioral Agents”). The role of the Reflective Agent, which incorporates the planner UCPOP, is to first generate a plan which achieves a set of user-supplied goals. This process involves deciding which of the Behavioral Agents will execute the final plan. The Reflective Agent then translates the plan into a set of scripts which are sent (each script is transmitted as part of a message) to each Behavioral Agent selected to execute that plan. The Behavioral Agents (which are implemented both in software as simulated agents, and in hardware as mobile robots) then execute their scripts and report success or failure. The Behavioral Agents are sensor-driven and operate using a behavioral control architecture which means that under certain sensory conditions a set of behaviors will be activated in order to produce an appropriate emergent behavior.

The current MACTA-LAB project is concerned with applying the multi- agent architecture developed during the preceding MACTA project to the area of automated laboratories. The developing area of ”immobile robots” or ”immobots” suggests that large, distributed, sensor-rich systems (such as automated laboratories, networked building energy systems and space probes) can be viewed as multi- agent systems in which the robots do not move. The aim of the MACTA- LAB project is to produce a software tool to aid the design, development and evaluation of automated laboratories. Reflective Agents (processing symbolic information) and Behavioral Agents (processing non-symbolic information) will be defined in order to support a WHAT-IF? exploration of the consequences of design decisions.

The functional requirements of a water testing laboratory have been analyzed and these requirements are being modeled in terms of a heterogeneous multi-agent ”immobot” application. The multi-agent architecture includes a Reflective Agent which is being extended to incorporate planning, diagnostic and resource allocation capabilities, and a number of Behavioral Agents with reactive, sensor-driven capabilities.

3.6.1. Current systems being developed

We are developing a software tool which will enable the design, development and evaluation of alternative automated laboratory layouts (a prototype, intended for future use as an industrial tool).

The potential applications are automated water testing laboratories.

The MACTA-LAB project has 2 industrial collaborators: UK Robotics, Manchester, and ALcontrol UK, Bradford

3.6.2. The technology used

An extension of the planner SNLP (implemented in Lucid Common Lisp on a SPARCstation LX) is being developed which enables the interleaving of planning and execution as well as reasoning about time and resources.

The mobile robot control architecture is an implementation of a behavioral control architecture using OCCAM. The SPARCstation LX communicates with the robots using an RF link.

3.6.3. Future work

We are interested in 3 main areas.

The interface between planning and execution. Currently, there is no clearly defined interface between planning and execution (i.e. between the data structure used to represent plans which is output by a planner, and the input required by execution agents). In our research we had to implement code which converted plans output by UCPOP into a format which could be used by the execution agents (in our case, mobile robots).

planning for multiple execution agents with differing capabilities Our planner (incorporated within a Reflective Agent) generates plans which are executed by more than one mobile robot which have differing capabilities. This means that firstly the planner has to decide which robot will execute which of the actions in a plan. When assigning robots to actions, where robots have different capabilities, it is important to ensure the robot is capable of executing the action (e.g. there is no point assigning a robot to pick up an object if the robot does not have the capability to pick up objects). We believe that execution agents should be given a special status for the purposes of planning - planners such as UCPOP treat execution agents as any other parameter when instantiating actions.

Integrating Planning and Scheduling In our work the planner must allocate execution agents to actions as part of the planning process. In addition, once an execution agent has been allocated to an action, the planner may need to plan for the agent to be at the location at which that action is to be executed. When planning to achieve water testing laboratory tasks, the planner must reason about time and resources. These require the integration of planning and scheduling.

3.7. University of Bonn

Michael Beetz

The research group RHINO focuses on the design of intelligent systems with special emphasis on planning, image processing, robust state estimation, and autonomous robot control under uncertainty. The group evaluates and analyzes planning techniques and control principles for real time decision making by implementing and testing them on autonomous mobile robots (a RWI B21 mobile robot and an Active Media Pioneer robot). The RWI B21 robot is equipped with a manipulator, stereo color cameras, a ring of sonars and two laser range finders.

3.7.1. Research Areas

Research on robot action planning concentrates on the following areas: structured reactive controllers, planning reactive behavior, autonomous learning of symbolic robot action plans, autonomous robotic agents, and the application of robot action planning techniques to distributed Supply Chain control.

- **Structured Reactive Controllers.** We investigate computational mechanisms that enable autonomous robots to exhibit competent, goal-directed behavior in mixed man-machine environments. In our research we apply *structured reactive controllers (SRCs)* to couple high-level reasoning with continuous low-level control processes. SRCs aim at enabling autonomous robots to accomplish non repetitive sets of complex jobs, which are mostly — but not all —

routine, reliably and without wasting resources. SRCs are collections of concurrent control routines that specify routine activities and can adapt themselves to non-standard situations by means of planning. SRCs execute three kinds of control processes: routine activities that handle standard tasks in standard situations, monitoring processes that detect non-standard situations, and planning processes that adapt, if necessary, routine activities to non-standard situations. *Selected publications:* [39, 41].

- **Planning reactive behavior of autonomous mobile robots.** We consider robot action planning to be the computational process of generating and revising high-level robot control programs based on foresight. Our research goal is to equip autonomous robot controllers with robot action planning capabilities that enable them to perform better than they possibly could without having these capabilities. Planning processes reason about structured reactive plans, plans that specify how the robot is to react to sensory input in order to accomplish their goals and revise plans while they are executed [40]. Our work on robot action planning concentrates on three aspects.

  – Methods for robot action planning (such as Probabilistic, Prediction-based Schedule Debugging); *Selected publications:* [32, 37].
  – Realistic models for symbolically predicting concurrent reactive robot behavior; *Selected publications:* [36, 35]; and
  – Runtime plan adaptation for autonomous robots *Selected publications:* [40, 38]

- **Autonomous learning of symbolic robot action plans.** Autonomous robots, such as robot office couriers, need control routines that support flexible task execution and effective action planning. In our research on autonomous robot learning we develop XFRMLEARN, a system that learns symbolic structured navigation plans. Given a navigation task, XFRMLEARN learns to structure continuous navigation behavior and represents the learned structure as compact and transparent plans. *Selected publications:* [34, 33].

- **Autonomous robotic agents.** Autonomous service robots such as office couriers or museum tourguides have become challenging testbeds for developing and testing computational models of competent agency. In the previous years we have worked on two robotic museums tourguides and an autonomous robot office courier. *Selected publications:* [60, 162, 39].

- **Interactive, planning-based Software agents for Distributed Supply Chain Management.** In this research project we apply techniques from robot action planning, in particular, transformational planning of reactive behavior and reactive execution of concurrent processes to the problem of supply chain management planning and control.

3.7.2. Research Projects and Memberships in Research Networks

Our research group carries out and participates in the following research projects:

- TOURBOT: Interactive Museum Tele-presence through Robotic Avatars; Esprit project IST-1999-12643

- XFRM-Learn: Model- and Diagnosis-based Transformational Learning of Symbolic Plans for Mobile Robots

- MAP-SCM: Interactive, plan-based Software Agents in the Distributed Supply Chain Management

In addition, the research group actively participates in the following European research networks:

- VIRGO: Vision-Based Robot Navigation Research Network
- AgentLink: ESPRIT-funded Network of Excellence for agent-based computing.
- PLANET: European Network of Excellence in AI Planning

3.8. University of Linköping

Patrick Doherty, Patrik Haslum and Jonas Kvarnström

3.8.1. Linköping Node Planning Activities

Research in planning at the Linköping node is pursued in the context of the WITAS Unmanned Aerial Vehicle Project. The WITAS UAV project is centered around the study and design of deliberative/reactive architectures to support autonomous operation of a surveillance aircraft. The main example application is traffic surveillance, including tasks such as finding and tracking individual vehicles, and identifying complex patterns such as erratic driving, U-Turns, overtakes, etc. The main sensor used is an active vision system with both digital video and infrared cameras. For navigation, the use of a differential GPS and inertial navigation system is assumed. Research activity in the project is intended to be pursued using both simulation and actual flight tests.

Work related to planning is currently centered around the following topics.

- Plan Specification and Execution – The current D/R architecture can be described conceptually as a non-hierarchical (3) layered architecture consisting of a layer of deliberative services such as path planning, mission planning, chronicle recognition, a process layer consisting of traditional control loops for sensing and actuation, and a reactive layer which manages dynamic task, or plan, execution.

  In AI, a number of languages for task/plan specification have been developed recently (eg. RAPS, Propice, etc). We've also looked at languages designed for control and real-time programming (eg. Esterel), and are trying to integrate useful ideas from both areas.

- Predictive Mechanisms for Planning – For planning in domains that include independent, or sparsely interacting, agents, knowledge of the expected behavior of such agents is essential. Prediction also provides a way to enhance robustness, flexibility and efficiency of reactive programs, forming a sort of middle ground between purely reactive behavior and planning from first principles.

  As an alternative to approaches based on probabilities, as in Markov processes, we are experimenting with predictive models which make assessments of the normality of an agent's behavior at the level of sequences of events rather than on events themselves.

- Forward Chaining Planners – We have recently developed a forward chaining planner, TALplanner [73], based on ideas proposed by Bacchus and Kabanza. In this paradigm, planning is performed by searching the space of states, where each trajectory of states represents a potential plan. Sequences of states are filtered using domain dependent knowledge about the application. The basis for TALplanner is a highly expressive non monotonic narrative-based temporal logic for reasoning about action and change. Initial results have been shown to be quite promising. A version of TALplanner is implemented and entered in the AIPS'2000 planning competition. An interesting topic being pursued is the integration of the predictive mechanism described above with TALplanner and the application of the resulting module to the WITAS traffic surveillance domain.

- Planning in Incompletely Specified Domains – In many realistic planning applications, the assumption of complete information about world state is not feasible and the CWA can not be used for the concise and efficient representation of negative information. On the other hand there is a broad spectrum of incompleteness assumptions between the CWA and the full open world assumption. Etzioni et al were one of the first to pursue relaxation of the CWA and developed an algorithm for the Localized Closed World Assumption (LCWA) which was integrated with a planner. We have been pursuing a more generic means of representing the LCWA based on the use of circumscription, quantifier elimination, and viewing questions about world state as database queries. Preliminary results show that several of the LCWA approaches are subsumed by the method and one retains sound and complete behavior in addition to polynomial performance of the algorithm.

- Perception Planning – A number of researchers are pursuing perception planning by use of knowledge producing actions and their integration in planning formalisms. A related topic is to plan for the use of sensors when there are several and to take into account resource limitations associated with the use of sensors, and to replan sensor usage during actual use. We are pursuing these topics for our UAV platform. An especially interesting issue is to dynamically modify various vision algorithms associated with the active vision system on-line, based on the current flying context.

3.8.2. Empirical Evaluation of Planning Activities

In the WITAS project, we have set up a quite sophisticated research infrastructure which consists of a simulation platform for the operational environments we plan to fly over and the integration of the D/R architecture with the simulation tool which is used for testing the functionality of the architecture. One interesting feature of our approach is that there will be a real-to-life correlation between the simulation environment and the actual operational environments (OE) we will fly in. We will try to achieve this by using a terrain model for the OE generated via the use of a laser sensor which gathers elevation data for the OE, and the use of actual aerial photos for ground texture. The elevation data is accurate to within 1 decimeter in the x,y,z directions. The terrain model can be used in simulation and as part of the on-board geographical knowledge base used by the UAV, for instance for navigation. We believe this approach has great potential for empirical evaluation of UAVs in simulation as well as in actual fly zones.

The UAV and traffic surveillance domain provides several planning problems, both challenging and quite different from those otherwise considered in planning research. The goal of the UAV is typically to gather certain information, rather than to change the world state. It can sense certain aspects of the environment, but only covering a relatively small part of it at a time. The task has similarities to research on planning for information gathering in so called “softbots”, but is much complicated by the fact that the world is highly dynamic, with many agents acting concurrently with, and regardless of, the UAV.

3.9. University Carlos III of Madrid

Daniel Borrajo

- Problems and projects this node is currently working on Reactive Planning

  Learning Behaviors for Reactive Planning by means of:

  1. Reinforcement Learning in the Robocup
  2. Evolution Strategies to learn weights in Neural Networks
  3. Classifier Systems to learn symbolic rules
  4. Genetic Algorithms to learn fuzzy rules
  5. Integrated architectures for planning, learning, and executing in robots, and in the Robocup

  Learning Operators for Deliberative Planning

  Design of a new control architecture able to represent deliberative planning and reactive planning

  Integration of planning and acting in real applications

- Current systems being developed, their status (prototypes,industrial tools,...)

  Prototypes of reactive control systems

  Prototypes of learning systems (detailed above) applied into real robots

  Simulator of ”Office assistant” performing: transportation search for objects (or individuals) collecting objects surveillance

- When does this node expect to have results?
  For the learning systems: from today to five years
  For the simulator: two years
  For the design of new architecture: three years

- What are the potential applications?

  Adaptation of systems to real environment by means of learning

- What are the industrial collaborations of an academic node, on which applications?

  We do have collaboration with industries, but in relation to other types of planning and learning.

- What is the technology used?

1. Learning Techniques
   Neural Networks Evolutionary Strategies Genetic Algorithms Fuzzy Logic Induction Reinforcement learning
2. Simulators Khepera Sim SimDAI Robosoccer
3. Real Robot
   Khepera (two units)

- Prospective view: other problems worth to be pursued? What are open problems?

  Integration of planning and acting: deliberative versus reactive

  Continuous learning

  Generalization of knowledge learned

  Integration of several learning systems, by sharing common representations

3.10. University of Ulm

Susanne Biundo

Currently, research activities at the Ulm node are devoted to hierarchical planning and to system support in the construction and maintenance of provably consistent domain models.

3.10.1. Hierarchical Planning for Autonomous Systems

The main aim of this project is to develop a hierarchical planning system that integrates deliberative planning, plan execution, and information gathering. The exchange of information between the plan generation and execution components is implemented in such a way that information gathering actions and plans can be parts of the usual domain plans on arbitrary levels of abstraction. Their goal- directed execution is interleaved with planning and the information obtained this way is used to guide the further planning process. In addition execution failures are reported to the planning component, which then initiates a replanning process for the failed task.

The approach relies on the HTN methodology, which it extends towards several directions. The extensions include an increased expressiveness of the underlying planning language. Control structures are introduced and executable programs are integrated as “bodies” of basic HTN methods. The planning strategy to be implemented allows for incremental planning and stepwise refinement and for the exploitation of information gathered during planning. Most importantly, the strategy also enables the continuous integration of new tasks into the current planning process.

Furthermore, a formal, logic-based semantics will be provided for this planning method. Based on this semantics, formal properties of plans like correctness, safety, and feasibility can be formulated and proved. This is in particular important in view of safety- critical applications.

The application potential of the approach is manifold. It includes the control of autonomous systems like robots as well as the control of software agents. It is also suitable for applications in the command and control area.

3.10.2. Systematic Domain Model Construction

In complex and large-scale planning applications, like planning for autonomous systems, the development and maintenance of the underlying domain models is a crucial task. Consequently, system support in constructing these models and in keeping them consistent is essential when aiming at these applications.

To this end, a concept to assist users in the incremental and modular construction of verified models of planning domains has been developed [46, 47]. It is based on a temporal logic representation formalism and considers domain models as formal structures. Well-defined and safe operations for the union, extension, and refinement allow to build complex domain models out of already existing simpler ones. A deductive component will automatically perform the proofs necessary to guarantee both the consistency of single models and the safety of operations on models. A GUI will be used to keep users from the details of the underlying logical formalism.

While the concept has been completely worked out and a proof-of- concept prototype implementation was completed, a new project currently begins to work on a more sophisticated implementation. The prototype of this system will be available within one year.

In addition to the topics addressed above, research interests and activities of the group are centered around formal methods in planning that allow for the specification and proof of formal properties like the correctness of plans and planning systems, the safety and robustness of plans, the consistency of models, and the reliability of systems.

Bibliography

[1] R. Arkin. *Behavior-Based Robotics.* MIT Press, Cambridge, MA, 1998.

[2] J. Hertzberg, H. Jaeger, U. Zimmer, and Ph. Morignot. A framework for plan execution in behavior-based robots. In *Proc. of the 1998 IEEE Int. Symp. on Intell. Control (ISIC-98)*, pages 8–13, Gaithersburg, MD, September 1998.

[3] L. Paletta, E. Rome, and A. Pinz. Visual object detection for autonomous sewer robots. In *IEEE Intl. Conf. Intelligent Robots and Systems (IROS-99)*, pages 1087–1093, October 17-21 1999.

[4] E. Rome, J. Hertzberg, Th. Christaller, F. Kirchner, and U. Licht. Towards autonomous sewer robots: The MAKRO project. *J. Urban Water*, 1:57–70, 1999.

[5] H. Veit and G. Richter. The FTA design paradigm for distributed systems. *J. Future Gener.Comp.Syst.*, (6):727–740, 2000.

[6] S. Coradeschi and A. Saffiotti. Anchoring symbols to sensor data: preliminary report. *Proc. of the 17th AAAI Conf.* Austin, Texas, 2000. pp. 129–135.

[7] A. Saffiotti. Handling uncertainty in control of autonomous robots. In M. Wooldridge and M. Veloso, eds, *Artificial Intelligence Today*, pp. 381–408. LNAI 1600, Springer-Verlag, DE, 1999.

[8] A Saffiotti, M. Boman, P. Buschka, P. Davidsson, S. Johansson, and Z. Wasik. Team Sweden. In P. Stone, T. Balch, and G. Kraetzschmar, eds, *RoboCup 2000.* Springer-Verlag, DE, in press.

[9] A. Saffiotti and K. LeBlanc. Active Perceptual Anchoring of Robot Behavior in a Dynamic Environment. *Proc. of the IEEE Int. Conf. on Robotics and Automation* (ICRA), pp. 3796–3802. San Francisco, CA, 2000.

[10] A. Saffiotti, K. Konolige, and E.H. Ruspini. A multivalued-logic approach to integrating planning and control. *Artificial Intelligence*, 76(1-2):481–526, 1995.

[11] S. Parsons, O. Pettersson, A. Saffiotti, and M. Wooldridge. Robots with the best of intentions. In M. Wooldridge and M. Veloso, eds, *Artificial Intelligence Today*, pp. 329–338. LNAI 1600, Springer-Verlag, DE, 1999.

[12] S. Parsons, O. Pettersson, A. Saffiotti, and M. Wooldridge. Intention reconsideration in theory and practice. *Proc. of the 14th European Conference on Art. Intell.* Berlin, DE, 2000. pp. 378–382.

[13] M. A. Garcia. Fast Approximation of Range Images by Triangular Meshes Generated through Adaptive Randomized Sampling. *Proceedings of IEEE International Conference on Robotics and Automation*, pages 21-27, Nagoya, Japan, May 1995.

[14] M. A. Garcia. A Hierarchical World Model Representation Supporting Heterogeneous Multisensory Integration. *Proceedings of International Conference on Advanced Robotics*, pages 461-471, Sant Feliu de Guixols, Spain, September 1995.

[15] M. A. Garcia and L. Basanez. Heterogeneous Multisensory Integration in Robotics: A Data Representation Approach. *Proceedings of SPIE Conference on Sensor Fusion and Networked Robotics*, vol.2589, pages 142-153, Philadelphia, USA, October 1995.

[16] M. A. Garcia, A. D. Sappa and L. Basanez. Efficient Approximation of Range Images Through Data Dependent Adaptive Triangulations. *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, pages 628-633, San Juan, Puerto Rico, June 1997.

[17] M. A. Garcia, S. Velazquez and A. D. Sappa. A Two-Stage Algorithm for Planning the Next View From Range Images. *Proceedings of the 9th British Machine Vision Conference*, pages 720-729, Southampton, United Kingdom, September 1998.

[18] M. A. Garcia, S. Velazquez, A. D. Sappa and L. Basanez. Autonomous Sensor Planning for 3D Reconstruction of Complex Objects from Range Images. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3085-3090, Leuven, Belgium, May 1998.

[19] M. A. Garcia, A. D. Sappa and L. Basanez. Efficient Generation of Object Hierarchies from 3D Scenes. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1359-1364, Detroit, USA, May 1999.

[20] A. D. Sappa and M. A. Garcia. Modeling Range Images with Bounded Error Triangular Meshes without Optimization. *Accepted for publication in 15th IAPR International Conference on Pattern Recognition*, Barcelona, Spain, September 2000.

[21] A. D. Sappa and M. A. Garcia. Incremental Multiview Integration of Range Images. *Accepted for publication in 15th IAPR International Conference on Pattern Recognition*, Barcelona, Spain, September 2000.

[22] R. Alami, R. Chatila, S. Fleury, M. Ghallab, and F. Ingrand. An Architecture for Autonomy. *International Journal of Robotics Research*, 17(4):315–337, April 1998.

[23] Jose A. Ambros-Ingerson and Sam Steel. Integrating Planning, Execution and Monitoring. In *Proceedings of the 7th National Conference of the American Association on Artificial Intelligence (AAAI-88)*, pages 83–88, Saint Paul, Minnesota, USA, August 1988. AAAI Press/MIT Press.

[24] Y. Arens, C. Y. Chee, C. N. Hsu, and C. A. Knoblock. Retrieving and Integrating Data from Multiple Information Sources. *International Journal of Intelligent and Cooperative Information Systems*, 2(2):127–158, 1993.

[25] O. Aycard, P. Laroche, and F. Charpillet. Mobile robot localization in dynamic environment using places recognition. In *IEEE International Conference on Robotics and Automation.* ICRA'98, 1998.

[26] S. Baluja. Evolution of an artificial neural network based autonomus plan vehicle controller. *IEEE transactions on Systems, Man and Cybernetics*, 26(3):450–463, June 1996.

[27] C. Barrouil, C. Castel, P. Fabiani, R. Mampey, P. Secchi, and C. Tessier. A perception strategy for a surveillance system. In *13th European Conference on Artificial Intelligence*, Brighton, aug 1998. ECAI 98, John Wiley & Sons.

[28] C. Barrouil, J. B. Cavaillé, L. Chaudron, K. Deschinkel, P. Fabiani, J. L. Farges, R. Mampey, Y. Meiller, and C. Tessier. D3i: Décision dynamique distribuée dans l'incertain. T/R 294/99, ONERA/DCSD/CD, dec 1999.

[29] C. Barrouil and J. Lemaire. An integrated navigation system for a long range auv. In *OCEANS'98*. IEEE Oceanic Engineering Society, oct 1998.

[30] Rosy Barruffi and Michela Milano. Interactive constraint satisfaction techniques for information gathering in planning. In Henri Prade, editor, *Proceedings of the 13th European Conference on Artificial Intelligence (ECAI-98)*, pages 514–515, Brighton, UK, 1998. John Wiley & Sons, Chichester, UK.

[31] C. Becker, J. Salas, K. Tokusei, and J.C. Latombe. Reliable navigation using landmarks. In *ICRA*, 1995.

[32] M. Beetz, M. Bennewitz, and H. Grosskreutz. Probabilistic, prediction-based schedule debugging for autonomous robot office couriers. In *Proceedings of the 23rd German Conference on Artificial Intelligence (KI 99), Bonn, Germany*. Springer Verlag, 1999.

[33] M. Beetz and T. Belker. Environment and task adaptation for robotic agents. submitted for publication, 2000.

[34] M. Beetz and T. Belker. Learning structured reactive navigation plans from executing mdp navigation policies. submitted for publication, 2000.

[35] M. Beetz and H. Grosskreutz. Causal models of mobile service robot behavior. In R. Simmons, M. Veloso, and S. Smith, editors, *Fourth International Conference on AI Planning Systems*, pages 163–170, Morgan Kaufmann, 1998.

[36] M. Beetz and H. Grosskreutz. Probabilistic hybrid action models for predicting concurrent percept-driven robot behavior. In *Proceedings of the Fifth International Conference on AI Planning Systems*, Breckenridge, CO, 2000. AAAI Press.

[37] M. Beetz and D. McDermott. Improving robot plans during their execution. In K. Hammond, editor, *Second International Conference on AI Planning Systems*, pages 3–12, Morgan Kaufmann, 1994.

[38] M. Beetz and D. McDermott. Local planning of ongoing activities. In Brian Drabble, editor, *Third International Conference on AI Planning Systems*, pages 19–26, Morgan Kaufmann, 1996.

[39] M. Beetz. Structured reactive controllers — a computational model of everyday activity. In *Proceedings of the Third International Conference on Autonomous Agents*, 1999.

[40] M. Beetz. Runtime plan adaptation in structured reactive controllers. In *Proceedings of the Fourth International Conference on Autonomous Agents*, Barcelona, Es, 2000. ACM Press.

[41] M. Beetz. Structured reactive controllers. *Journal of Autonomous Agents and Multi-Agent Systems*, 2000. to appear.

[42] Scott W. Bennet and Gerald DeJong. Real world robotics: Learning to plan for a robust execution. *Machine Learning*, 23(2/3):121–162, May/June 1996.

[43] Antonio Berlanga, Pedro Isasi-Viñuela, José M. Molina, and Araceli Sanchis. Competitive evolution to find generalized solutions: the arms race perspective. In *Intelligent Engineering Systems*, pages 61–65, Vienna (Austria), September 1998.

[44] Antonio Berlanga, Araceli Sanchis, Pedro Isasi, and José Manuel Molina. Neural networks robot controller trained with evolution strategies. In *Proc. of 1999 Congress on Evolutionary Computation, CEC99*, 1999.

[45] D.P. Bertsekas. *Dynamic programming : deterministic and stochastic models*. Prentice-Hall, 1986.

[46] S. Biundo and W. Stephan. Modeling Planning Domains Systematically. In W. Wahlster, editor, *Proceedings of the 12th European Conference on Artificial Intelligence (ECAI-96)*, pages 599–603. Wiley & Sons, 1996.

[47] S. Biundo and W. Stephan. System Assistance in Structured Domain Model Development. In M.E. Pollack, editor, *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI- 97)*, pages 1240–1245. Morgan Kaufmann, 1997.

[48] Avrim L. Blum and Merrick L. Furst. Fast Planning Through Planning Graph Analysis. In C. S. Mellish, editor, *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 1636–1642. Morgan Kaufmann, 1995.

[49] A. Blum and M. Furst. Fast planning through planning graph analysis. *AI*, (90):281–300, 1997.

[50] B. Bouilly and T. Siméon. A sensor-based motion planner for mobile robot navigation with uncertainty. In Dorst et al. [78].

[51] Craig Boutilier, Richard Dearden, and Moises Goldszmidt. Exploiting structure in policy construction. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 1104– 1111, Montreal, Quebec, Canada, August 1995. Morgan Kaufmann.

[52] C. Boutilier, R.I. Brafman, and C. Geib. Structured reachability analysis for markov decision processes. In *UAI'98*, pages 24–32, Madison, jul. 1998.

[53] G. Boutilier, T. Dean, and S. Hanks. Decision-theoretic planning: Structural assumptions and computational leverage. *Journal of Artificial Intelligence Research*, 11:1–94, 1999.

[54] Valentino Braitenberg. *Vehicles. Experiments in Synthetic Psychology*. MIT Press, Cambridge, MA (USA), 1984.

[55] Rodney A. Brooks. A roboust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, RA- 2(1):14–23, March 1986.

[56] Rodney A. Brooks. Intelligence without Representation. *Artificial Intelligence*, 47:139–159, 1991.

[57] Rodney A. Brooks. Artificial life and real robots. In *Toward a practice of autonomous systems: Proceedings of the Fist European Conference on Artificial Life*. MIT Press, 1992.

[58] L. Bruyère. Approches stochastiques de la représentation des incertitudes pour la planification de déplacements d'un robot mobile en environnement mal connu. DEA Electronique-Optronique Université de Bretagne Occidentale, 1999.

[59] J. Buhmann, W. Burgard, A.B. Cremers, D. Fox, T. Hofmann, F. Schneider, J. Strikos, and S. Thrun. The mobile robot rhino. *AI Magazine*, 16(1), 1995.

[60] W. Burgard, A.B. Cremers, D. Fox, D. Hähnel, G. Lakemeyer, D. Schulz, W. Steiner, and S. Thrun. The interactive museum tour-guide robot. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI'98)*, 1998.

[61] A. Cassandra, L. Kaebling, and M. Littman. Acting optimally in partially observable stochastic domains. In *Proceedings of the American Association of Artificial Intelligence (AAAI-94)*, pages 1023–1028. AAAI Press, 1994.

[62] David Chapman and Leslie P. Kaelbling. Input generalization in delayed reinforcement learning: An algorithm and performance comparisons. *Proceedings of the International Joint Conference on Artificial Intelligence*, 1991.

[63] Allan Christiansen. *Automatic Acquisition of Task Theories for Robotic Manipulation*. PhD thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, March 1992.

[64] D.T. Cliff, P. Husband, and I. Harvey. Explorations in evolutionary robotics. *Adaptive Behaviour*, pages 73–110, 1993.

[65] Jonathan Connell and Sridhar Mahadevan. *Robot Learning*. Kluwer Academic Publishers, Boston, MA (USA), 1993. (eds.).

[66] Jonathan H. Connell. *Minimalist Mobile Robotics: A Colony-style Architecture fo a Mobile Robot*. Academic Press, Cambridge, MA, 1990.

[67] K. Currie and A. Tate. O-Plan: The Open Planning Architecture. *Artificial Intelligence*, 52(1):46– 86, 1991.

[68] Thomas Dean and Robert Givan. Model minimization in markov decision processes. In *Proceedings of the American Association of Artificial Intelligence (AAAI-97)*. AAAI Press, 1997.

[69] J. del R. Millán. Rapid, safe, an incremental learning of navigation strategies. *IEEE transactions on Systems, Man and Cybernetics*, 26(3):408–420, June 1996.

[70] O. Despouys and F. F. Ingrand. Propice-Plan: Toward a Unified Framework for Planning and Execution. In *Proceedings of the Fifth European Conference on Planning*, pages 280–292, Durham, UK, September 8- 10 1999.

[71] O. Despouys and F. F. Ingrand. Une architecture intégrée pour la planification et le contrôle d'exécution. In *Proceedings of Reconnaissance des Formes et Intelligence Artificielle*, volume I, pages 69–78, Paris, France, February 1-3 2000.

[72] M. Devy, R. Chatila, P. Fillatreau, S. Lacroix, and F. Nashashibi. On autonomous navigation in a natural environment. *Robotics and Autonomous Systems*, 16(1):5–16, 1995.

[73] P. Doherty and J. Kvarnström. TALplanner: An empirical investigation of a temporal logic-based forward chaining planner. In Claire Dixon and Michael Fisher, editors, *Proc. 6th International Workshop on Temporal Representation and Reasoning (TIME'99)*, pages 47–54. IEEE Computer Society, 1999.

[74] Marco Dorigo and M. Colombetti. Robot shaping: Developing autonomous agents through learning. *Artificial Intelligence*, 71(2):321–370, 1994.

[75] M. Dorigo and U. Snepf. Genetics based machine learning and behavior based robotics: a new sysnthesis. *IEEE transactions on Systems, Man and Cybernetics*, 23:141–153, 1993.

[76] Marco Dorigo. Message-based bucket brigade: An algorithm for the appointment of credit problem. In Yves Kodratoff, editor, *Machine Learning. European Workshop on Machine Learning*, LNAI 482, pages 235–244. Springer-Verlag, 1991.

[77] Marco Dorigo. Alecsys and the autonomouse: Learning to control a real robot by distributed classifier systems. *Machine Learning*, 19:209–240, 1995.

[78] L. Dorst, M. van Lambalgen, and F. Voorbraak, editors. *Reasoning with Uncertainty in Robotics*. Springer, 1996.

[79] Denise Draper, Steve Hanks, and Daniel S. Weld. Probabilistic Planning with Information Gathering and Contingent Execution. In Hammond, ed., *Proceedings of the 2nd International Conference on Artificial Intelligence Planning Systems (AIPS-94)*, pages 31–36, AAAI Press, 1994.

[80] P. Fabiani, H.-H. González-Baños, J.-C. Latombe, and D. Lin. Tracking a partially predictable object with uncertainty and visibility constraints. working paper, 2000.

[81] P. Fabiani and J.-C. Latombe. Dealing with geometric constraints in game-theoretic planning. In *IJCAI'99*, Stockholm, aug. 1999. Morgan Kaufmann.

[82] P. Fabiani and Y. Meiller. Théorie des jeux et planification pour le dilemme perception- action. In *RFIA'2000*, Paris, F vrier 2000.

[83] P.J. Fabiani. Dynamics of beliefs and strategy of perception. In *12th European Conference on Artificial Intelligence*, pages 8–12, Budapest, aug 1996. ECCAI, John Wiley & Sons.

[84] Fernando Fernández and Daniel Borrajo. Vector quantization applied to reinforcement learning. In Manuela Veloso, editor, *Working notes of the IJCAI'99 Third International Workshop on Robocup*, pages 97– 102, Stockholm, Sweden, July-August 1999. IJCAI Press.

[85] Richard E. Fikes, Peter E. Hart, and Nils J. Nilsson. Learning and executing generalized robot plans. *Artificial Intelligence:251-288. Reprinted in "Tutorial on robotics", Ed. C.S.G. Lee, R.C.Gonzalez et al, IEEE Computer Society Press*, 3:433–470, 1972.

[86] R. Firby. An Investigation into Reactive Planning in Complex Domains. In *Proceedings of the 6th National Conference of the American Association on Artificial Intelligence (AAAI-87)*, pages 202–206, 1987.

[87] D. Floreano and F. Mondada. Evolution of homming navigation in a real mobile robot. *IEEE transactions on Systems, Man and Cybernetics*, 26(3):396–407, June 1996.

[88] N. Franceschini, J.M. Pichon, and C. Blanes. Real time visuomotor control: from flies to robots. In *Proceedings of the fifth International Conference on Advanced Robotics*, pages 91–95, June 1991.

[89] Judy A. Franklin, Tom M. Mitchell, and Sebastian Thrun (eds.). *Recent Advances in Robot Learning*. Kluwer Academic Publishers, 1996.

[90] T. Fualdes and C. Barrouil. Perception planning for an U.G.V. *Robotics and Autonomous Systems*, 11(2), 1993.

[91] F. Garcia and R. Mampey. Mobile robot motion planning by reasoning both at itinerary and path levels. In *Fifth International Conference on Advanced Robotics*, volume 2, june 1991.

[92] Ramón García-Martínez and Daniel Borrajo. An integrated approach of learning, planning, and execution. *Journal of Intelligent and Robotic Systems*, 2000. Accepted for publication.

[93] E. Gat. Integrating Planning and Reacting in a Heterogenous Asynchronous Architecture for Controlling Real-world Mobile Robots. In *Proceedings of the 10th National Conference of the American Association on Artificial Intelligence (AAAI-92)*. AAAI Press, 1992.

[94] P. Gaudiano, E. Zalama, and J. Lopez. An unsupervised neural network for low level control of a wheeled mobile robot: noise resistance stability and hardware implementation. *IEEE transactions on Systems, Man and Cybernetics*, 26(3):485–495, June 1996.

[95] G. De Giacomo, L. Iocchi, D. Nardi, and R. Rosati. Moving a Robot: the KR&R Approach at work. In Luigia Carlucci Aiello, Jon Doyle, and Stuart Shapiro, editors, *Proceedings of the 5th International Conference on Principles of Knowledge Representation and Reasoning (KR-96)*, pages 198–209, San Francisco, November 5–8 1996. Morgan Kaufmann.

[96] G. De Giacomo, L. Iocchi, D. Nardi, and R. Rosati. Planning with sensing for a mobile robot. In S. Steel and R. Alami, editors, *Proceedings of the 4th European Conference on Planning (ECP-97)*, volume 1348 of *Lecture Notes in Artificial Intelligence*, pages 156–168, Toulouse, France, 1997. Springer.

[97] Keith Golden and Daniel Weld. Representing Sensing Actions: The Middle Ground Revisited. In Luigia Carlucci Aiello, Jon Doyle, and Stuart Shapiro, editors, *Proceedings of the 5th International Conference on Principles of Knowledge Representation and Reasoning (KR-96)*, pages 174– 185, San Francisco, November 5–8 1996. Morgan Kaufmann.

[98] Keith Golden. Leap before You Look: Information Gathering in the PUCCINI Planner. In *Proceedings of the 4th International Conference on Artificial Intelligence Planning Systems (AIPS-98)*, pages ??–?? AAAI Press, 1998.

[99] D.H. Graf and W.R. LaLoncle. A neural controller for collision-free movement of general robot manipulators. In *Proceedings of the IEEE second International Conference on Neural Networks*, volume I, pages 77–84. IEEE, 1988.

[100] P. Hébert, S. Betgé-Brezetz, and R. Chatila. Probabilistic map learning: Necessity and difficulties. In Dorst et al. [78], pages 307–321.

[101] Karen Zita Haigh and Manuela M. Veloso. Learning situation-dependent costs: Improving planning from probabilistic robot execution. In *Proceedings of the Second International Conference on Autonomous Agents*, Minneapolis, MN, USA, May 1998.

[102] B. Hayes-Roth. An Architecture for Adaptive Intelligent Systems. *Artificial Intelligence*, 72, 1995.

[103] IJCAI Workshop. *Reasoning with Uncertainty in Robot Navigation*, Stockholm, aug 1999. Morgan Kaufmann. `http://aass.oru.se/Living/RUR99/`.

[104] F. F. Ingrand, R. Chatila, R. Alami, and F. Robert. PRS: A high level supervision and control language for autonomous mobile robots. In *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 43–49, Minneapolis, April 1996.

[105] S. Ishikawa. A method of autonomous mobile robot navigation by using fuzzy control. *Advanced Robotics*, 9(1):29–52, 1995.

[106] Lelie Pack Kaelbling, Michael L. Littman, and Andrew W. Moore. Reinforcement learning: A survey. *International Journal of Artificial Intelligence Research*, pages 237–285, 1996.

[107] Volker Klingspor, Katharina J. Morik, and Anke D. Rieger. Learning concepts from sensor data of a mobile robot. *Machine Learning*, 23(2/3):305–, May/June 1996.

[108] Craig A. Knoblock and Alon Y. Levy. Exploiting Run-Time Information for Efficient Processing of Queries. In *AAAI Spring Symposium on Information Gathering in Heterogeneous, Distributed Environments*, Paolo Alto, CA, 1995.

[109] Craig A. Knoblock. Planning, Executing, Sensing, and Replanning for Information Gathering. In Chris Mellish, editor, *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 1686–1693, San Francisco, 1995. Morgan Kaufmann.

[110] Sven Koenig and Reid Simmons. Passive distance learning for robot navigation. In *Machine Learning: Proceedings of the Thirteenth International Conference (ICML96)*, pages 266–274, 1996.

[111] J. Koza. Evolving emergent wall following robotic behavior using the genetic programming paradigm. In F.J. Varela and P. Bourgine, editors, *Toward a practice of autonomous systems. Proceedings of the First European Conference on Artificial Life*, pages 110–119, Cambridge, MA, 1991. MIT Press and Bradford Books.

[112] J. R. Koza. Evolution of subsumption using genetic programming. In F. J. Varela and P. Bourgine, editors, *Proceedings of the First European Conference on Artificial Life. Towards a Practice of Autonomous Systems*, pages 110–119, Paris, France, 11-13 December 1992. MIT Press.

[113] Nicholas Kushmerick, Steve Hanks, and Daniel S. Weld. An algorithm for probabilistic planning. *Artificial Intelligence*, 76(1-2):239–286, September 1995.

[114] P. Laroche and F. Charpillet. State aggregation for solving markov decision problems : An application to mobile robotics. In *10th IEEE International Conference on Tools with Artificial Intelligence*. ICTAI'98, 1998.

[115] J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, 1991.

[116] S.M. LaValle, H. Gonzalez-Banos, C. Becker, and J.C. Latombe. Motion strategies for maintaining visibility of a moving target. In *ICRA*, 1997.

[117] A. Lazanas and J.C. Latombe. Motion planning with uncertainty: a landmark approach. *Artificial Intelligence*, 76(1-2), 1995.

[118] Long-Ji Lin. Scaling-up reinforcement learning for robot control. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 182–189, Amherst, MA, June 1993. Morgan Kaufman.

[119] M.L. Littman, T. Dean, and L.P. Kaelbling. On the complexity of solving markov decision processes. In *UAI'95*, pages 394–402, Providence, jul. 1995.

[120] R. D. Luce and H. Raiffa. *Games and Decisions*. John Wiley & Sons, New York, 1957.

[121] P. Maes and R. Brooks. Learning to coordinate behaviors. In *Proccedings of the Eighth National Conference on Arfificial Intelligence*, pages 796–802, San Mateo, CA, 1990. Morgan Kaufmann.

[122] S. Mahavedan and J. Connell. Automatic programming of behavior-based robots using reinforcement learning. *Artificial Intelligence*, 55:311–365, 1992.

[123] Maja J. Mataric and Dave Cliff. Challenges in evolving controllers for physical robots. *Journal of Robotics and Autonomous Systems*, 19(1):67–83, October 1996.

[124] Vicente Matellán, José Manuel Molina, and Camino Fernández. Genetic learning of fuzzy reactive controllers. *Robotics and Autonomous Systems*, 25(1- 2):33–41, October 1998.

[125] Drew McDermott. Planning and Acting. *Cognitive Science*, 2(2):78–109, 1978.

[126] Y. Meiller and P. Fabiani. Perception-action dilemma at planning time : Getting the best out of game theory and classical planning. In *18th Workshop of the UK Planning and Scheduling Special Interest Group (PLANSIG99)*, Manchester - UK, dec 1999. University of Salford.

[127] Y. Meiller. Planification de déplacements et de prises d'information pour la poursuite d'objets mobiles en présence d'incertitudes. Rapport d'activité 5/7602.12, ONERA/DCSD - Toulouse, dec 1999.

[128] O. Miglino, H. Hautop, and S. Nolfi. Evolving mobile robots in simulated and real environment. *Artificial Life*, 2:417–434, 1995.

[129] Tom M. Mitchell. *Machine Learning*. McGraw- Hill, 1997.

[130] José M. Molina, Araceli Sanchis, Antonio Berlanga, and Pedro Isasi. An enhanced classifier system for autonomous robot navigation in dynamic environments. *Intelligent Automation and Soft Computing*, 2000. in press.

[131] Francesco Mondada and Edoardo Franzi. Mobile robot miniaturisation: A tool for investigation in control algorithms. In *Proceedings of the Second International Conference on Fuzzy Systems*, San Francisco, USA, 1993.

[132] Andrew W. Moore. Variable resolution dynamic programming: Efficiently learning action maps in multivariate real-valued spaces. *Proceedings in Eighth International Machine Learning Workshop*, 1991.

[133] Andrew W. Moore. The party-game algorithm for variable resolution reinforcement learning in multidimensional state-spaces. In J.D. Cowan, G. Tesauro, and J. Alspector, editors, *Advances in Neural Information Processing Systems*, pages 711–718, San Mateo, CA, 1994. Morgan Kaufmann.

[134] Karen L. Myers. Towards a framework for continuous planning and execution. In *Proceedings of the AAAI Fall Symposium on Distributed Continual Planning*, 1998.

[135] S. Nagata, M. Sekiguchi, and K. Asakawa. Mobile robot control by a structures hierarchical neural network. *IEEE Control Systems Magazine*, pages 69– 76, April 1990.

[136] Nils J. Nilsson. Shakey the robot. Technical report, SRI A.I., April 1984.

[137] Duane Olawsky, Kurt Krebsbach, and Maria Gini. An Analysis of Sensor-Based Task Planning. Technical Report 95-51, Department of Computer Science, University of Minnesota, July 1995.

[138] Duane Olawsky and Maria Gini. Deferred Planning and Sensor Use. In *Innovative Approaches to Planning, Scheduling and Control. Proceedings of the 1990 DARPA Workshop*, pages 166–174. M. Kaufmann, San Mateo, Ca, 1990.

[139] M. Pawlowski. Processus décisionnels markovien partiellement observables (pomdps) - étude de la complexité . DEA Math matiques Appliquées SupAéro-Université Paul Sabatier, 1999.

[140] J. Pearl. *Probabilistic reasoning in intelligent systems : networks of plausible inference*. Morgan Kaufmann, San Mateo, 1988.

[141] E. Pednault. ADL: Exploring the Middle Ground Between STRIPS and the Situation Calculus. In *Proceedings of the 1st International Conference on Principles of Knowledge Representation and Reasoning (KR-89)*, pages 324–332. Morgan Kaufmann, 1989.

[142] B. Pell, E. Gat, R. Keesing, N. Muscettola, and B. Smith. Robust Periodic Planning and Execution for Autonomous Spacecraft. In M. E. Pollack, editor, *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI-97)*, pages 1234–1239. Morgan Kaufmann, 1997.

[143] J. S. Penberthy and Daniel S. Weld. UCPOP A Sound, Complete, Partial Order Planner for ADL. In *Proceedings of the 3rd International Conference on Principles of Knowledge Representation and Reasoning (KR-92)*, pages 103–114, 1992.

[144] M. Peot and D. Smith. Conditional Nonlinear Planning. In James Hendler, editor, *Proceedings of the 1st International Conference on Artificial Intelligence Planning Systems (AIPS-92)*, pages 189–197, College Park, Maryland, June 15–17 1992. Morgan Kaufmann.

[145] Dean A. Pomerleau, Jay Gowdy, and Charles E. Thorpe. Combining artificial neural networks and symbolic processing for autonomous robot guidance. *Engineering Applications of Artificial Intelligence*, 4(4):279– 285, 1991.

[146] Dean Pomerleau. *Neural Network Perception for Mobile Robot Guidance*. Kluwer Academic Publishers, 1993.

[147] Louise Pryor and Gregg Collins. Planning for contingencies: A decision-based approach. *Journal of Artificial Intelligence Research*, 4:287–339, 1996.

[148] A. Saffiotti, K. Konolige, and E. H. Ruspini. A multivalued-logic approach to integrating planning and control. *Artificial Intelligence*, 76(1-2):481–526, 1995.

[149] A. Saffiotti and L.P. Wesley. Perception-based self-localization using fuzzy locations. In Dorst et al. [78], pages 368–385.

[150] L.J. Savage. *The Foundations of Statistics*. Dover, New York, 1972.

[151] Peter Stone and Manuela Veloso. Multiagent systems: A survey from a machine learning perspective. Technical Report CMU-CS-97-193, Computer Science Department, Carnegie Mellon University, 1997.

[152] Peter Stone and Manuela M. Veloso. A layered approach to learning client behaviors in the robocup soccer server. *Applied Artificial Intelligence Journal*, 12:165–188, August 1998.

[153] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning. An Introduction*. The MIT Press, 1998.

[154] Richard Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3(1):9–44, August 1988.

[155] Richard Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Proceedings of the Seventh International Conference on Machine Learning*, pages 216–224, Austin, TX, 1990. Morgan Kaufmann.

[156] Richard S. Sutton. Introduction: The challenge of reinforcement learning. *Machine Learning*, 8(3/4):225–227, May 1992.

[157] H. Takeda, C. Facchinetti, and J.C. Latombe. Planning the motions of a mobile robot in a sensory uncertainty field. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 16(10), 1994.

[158] M. Tambe, W. L. Johnson, R. M. Jones, F. Koss, J. E. Laird, P. S. Rosenbloom, and K. Schwamb. Intelligent Agents for Interactive Simulation Environments. *AI Magazine*, 16(1):15–39, 1995.

[159] Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 330–337, Amherst, MA, June 1993. Morgan Kaufman.

[160] A. Tate. Coordinating the activities of a planner and an execution agent. In G. Rodriguez, editor, *Proc. NASA conf. on space telerobotics*, NASA JPL, 1989. JPL Publications.

[161] Sebastian Thrun, Arno Bücken, Wolfram Burgard, Dieter Fox, Thorsten Frölinghaus, Daniel Hennig, Thomas hofmann, Michael Krell, and Timo Schmidt. Map learning and high- speed navigation in RHINO. In D Kortenkamp, Bonasso R.P., and Murphy R.R., editors, *AI-based Mobile Robots: Case studies of successful robot systems*. MIT Press, 1997.

[162] S. Thrun, M. Bennewitz, W. Burgard, A.B. Cremers, F. Dellaert, D. Fox, D. Haehnel, C. Rosenberg, N. Roy, J. Schulte, and D. Schulz. Minerva: A second generation mobile tour-guide robot. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'99)*, 1999.

[163] Sebastian Thrun. Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, 99(1):21–71, 1998.

[164] C. J. C. H. Watkins and P. Dayan. Technical note: Q-learning. *Machine Learning*, 8(3/4):279–292, May 1992.

[165] Gerhard Weiss and Sendip Sen, editors. *Adaption and Learning in Multi Agent Systems. Proceedings of the IJCAI 95 Workshop*. Springer-Verlag, 1996.

[166] Daniel S. Weld, Corin R.Anderson, and David E. Smith. Extending GRAPHPLAN to Handle Uncertainty & Sensing Actions. In *Proceedings of the 15th National Conference of the American Association on Artificial Intelligence (AAAI-98)*, pages 897–904. AAAI Press, 1998.

[167] D.J. White. *Markov Decision Processes*. John Wiley & Sons, Chichester, 1993.

[168] David Edward Wilkins. *Practical Planning: Extending the AI Planning Paradigm*. Morgan Kaufmann, San Mateo, California, 1988.

[169] B. C. Williams and P. P. Nayak. A Reactive Planner for a Model-based Executive. In M. E. Pollack, editor, *Proceedings of the 15th International Joint Conference on Artificial Intelligence (IJCAI-97)*, pages 1178- -1185. Morgan Kaufmann, 1997.

Part II.

Knowledge Engineering for Planning

Thomas Lee McCluskey

(University of Huddersfield, UK)

with contributions from Ricardo Aler (Universidad Carlos III de Madrid, Spain), Daniel Borrajo (Universidad Carlos III de Madrid, Spain), Patrick Haslum (Linköping University, Sweden), Peter Jarvis (SRI International, USA) and Ulrich Scholz (Darmstadt University, Germany).

4. The Nature of Knowledge Engineering

Knowledge Engineering (KE) in AI Planning is the process that deals with the acquisition, validation and maintenance of planning domain models, and the selection and optimization of appropriate planning machinery to work on them. Hence, knowledge engineering processes support the planning process - they comprise all of the off-line, knowledge-based aspects of planning that are to do with the application being built.

The main characteristic of a domain model is that it is possible for an agent to use one to make rational deductions about the domain it represents. In particular to planning, it is commonly assumed that a model contains a declarative description of a domain and that the model's most important component is a set of action descriptions. The conventional wisdom is that a declarative model should be developed to a large extent independently of the planning engine and any other software that will form the rest of the application. This tends to ease the process of validating the domain model. It also gives the stakeholders and developers more flexibility in the use of their product and lessens the investment risk; an independent domain model may be used with a range of general planning engines, and may be used in many other ways not necessarily connected to planning.

Acquisition of a domain model may involve a prolonged analysis of the application domain by knowledge engineers, using structured interviews with stakeholders, system manuals, existing models, software documentation etc. A list of questions should be asked, as shown below in (1) and (2). Hertzberg in section 3 of reference [32] declares a similar list of questions in his discussion of the characteristics of application domains.

(1) Fundamental environmental assumptions: Is the problem one of scheduling or planning, or a combination of the two? Can actions be modeled deterministically? Do they have to be modeled with duration? Do we have to model the resource they consume? Can we assume the planner has complete knowledge of the world? ...

(2) Pragmatic environmental assumptions: What are the kind of plans required? What is the relation between planning and execution? What are the the optimization criteria for plans? ...

Once environmental assumptions are known, the process of domain modeling itself can begin, leading to the kinds of knowledge in (3) and (4).

(3) Domain structure and dynamics: The identification of the relevant objects and object classes in the domain, their properties and relations (predicates), and constraints among these predicates. Dynamic knowledge is dependent on (1), but typically involves specifying how these predicates change truth value as actions are executed.

(4) Domain heuristics: General approximate rules and the general applicability of techniques that have been found to help in plan generation, where their use leads to a speed-up in planning and/or an increase in the quality of the output plans.

The type of application is determined by the requirements that fall into categories (1) and (2). The results of this early analysis is fundamental to or implicit in the domain modeling language used to capture the information in (3) and (4), as well as critical in the choice of planning algorithm.

Validation of a model is the process that promotes its quality in terms of internal and external criteria by the identification and removal of errors in the model. Internal criteria includes properties such as syntactic correctness and logical consistency; in general these properties can be proved formally and are not problematic. External criteria includes properties such as accuracy, correctness and completeness. Given that the sources of the model will not often be a mathematical object, these properties can never be proved correct (in the same sense that a requirements specification can never be proved correct). Note the distinction between validation of a domain model and validation of a planning system. The former supports the latter, and occurs at a much earlier stage in system development.

4.1. The Agents Involved in the Knowledge Engineering Process

There are several kinds of roles that have to be filled in the technical side of the knowledge engineering process: planning experts, domain engineers, domain experts, software and HCI experts, and end users. Often a person may have more than one role; a researcher in academia may have to fill every role. Assuming there is a distinguished formal system within which the domain model is being encoded, it seems necessary to have several kinds of "interface languages" to this formalism to suit participants in different roles. You would not want, for example, a user to help in the static validation of a domain model by reading through a complex logic formalism.

4.2. The Bigger Picture

Knowledge acquisition for knowledge-based systems and requirements engineering in software engineering are very active areas in Computer Science, and are associated with a growing body of literature, methods and techniques. They are related areas in a number of respects. In particular, one can consider that the results of knowledge acquisition and requirements engineering both involve some kind of domain model [64]. Using our general definition above, KE for planning may be seen as a special case of these general areas. Hence it may prove useful to derive methods and adapt tools from these areas, although there are peculiarities of planning that clearly distinguish engineering planning knowledge from them:

- The knowledge elicited in planning is largely knowledge about actions and how objects are effected by actions. This knowledge has to be adequate in content (and ultimately in form) to allow efficient automated reasoning and plan construction. In contrast, knowledge elicited about processes/actions in traditional software engineering tends to be done with the purpose of helping in the analysis and understanding of a system, and to be used in the forming of a specification of a new system.
- The ultimate use of the planning domain model is to be part of a system involved in the "synthetic" task of plan construction. This makes it very specific in the world of KBS, where many successful systems are, in contrast, aimed at solving diagnostic or classification problems.

Despite the difference of purpose, adopting tools and techniques from these general areas seems likely to be a good strategy. For example, the insights gained from the use and development of methods such as KADS [2], and the use of requirements modeling languages and methods in software engineering (e.g. [29]) need to be used when developing KE methods for planning applications.

4.3. Some Planning Projects which have exploited KE tools and techniques

The following is a list of planning projects that have addressed or are addressing the issues of KE and/or knowledge representation:

- O-Plan, SPAN and I-N-OVA: Edinburgh/ARPI projects that tackle plan representation and KB planning (e.g. see [65])
- The Planform project, aimed to create a simple knowledge engineering tools environment (e.g. see [1])
- The SIPE-2 system, featuring advanced HCI tools (e.g. see [52])
- The multimission Vicar Planner applied to automated image processing
- The Remote Agent Experiment, the control of NASA's Deep Space 1 by an AI planner.

Experience in applied work in planning suggests that the outputs of the KE process, such as "activity descriptions" are a vital core concept for many types of reasoning beyond planning. KE tools, methods and languages need to be suited to the much wider role of capturing knowledge of activity whether or not planning is involved in guiding and directing those activities towards purposeful outcomes.

4.4. Actions for the Planning Community

An overall finding of this project has been the general lack of knowledge in the (European) planning community of knowledge engineering issues. This may well originate from the algorithm-bias that planning and planning researchers seem to exhibit!

Hence a first step in our action plan would be to perform a review of the knowledge and requirements engineering literature and create a catalogue of tools and techniques that may be relevant to the AI Planning area. Important questions that such a review must answer include:

1. Are these tools actually in use outside the laboratory?
2. How have these tools been evaluated?
3. What lessons can the Planning community learn from the methodology for research and development in related areas?

Hertzberg in the last section of reference [32] declares that there is a lack of a "vocabulary for describing the characteristics of domains, plans . . . ". In the context of Knowledge Engineering, the pursuit of such a classification system and/or vocabulary seems worthy of action.

5. Related Areas

In this section we consider research areas which contribute to knowledge engineering for planning.

5.1. Formal Methods in Software Engineering

Formal Methods in software engineering covers (1) the capture and analysis of a (formal) specification of software within a structured formal language, and (2) the refinement of a formal specification into an efficient implementation. The formal specification language has to be appropriate to the application at hand, be sufficiently abstract to allow formal reasoning, and be well supported with a tools environment. The primary concerns in formal methods is to show that the initial specification is internally consistent and externally valid, and to prove that that the derived implementation is correct with respect to the specification. These processes are meant to improve the quality of the software process and product, as they are aimed at the early identification and removal of bugs. Superficially at least, there is a strong similarity between formal specification languages and planning languages. Both kinds of languages are designed to allow engineers represent actions precisely and declaratively. Take VDM-SL (the Vienna Development Method's Specification Language [35]) and a STRIPS-language. Both are based around the notion of a state, allow the developer to create operators, and in both cases those operators are defined using pre- and postconditions. Further, they are both based on the assumptions of closed world, default persistence and instantaneous operator execution. VDM encourages the creation of state invariants for validity and documentation purposes; state invariants are also used in some planning languages, but the main rationale here seems to be plan generation speed-up.

The difference between those using formal specification to describe systems and those using a planning language to model a planning domain, is that in the former case the specification is used as a blueprint for design, whereas in the latter case the specification is used as input to a planner to be reasoned with in order to construct plans to achieve goals. States in languages such as VDM-SL and Z [57] are built up from mathematical data types such as sets, mappings, sequences etc. With the exception of work in deductive planning [9], much of the work carried out in planning research assumes little or no structure to types (predicates are often assumed to be "function free").

We can learn from the development of formal methods. One particularly successful use of a formal system in AI planning and scheduling is in the use of Petri Nets (e.g. [50]). It is agreed that, in general, the take up of Formal Methods in mainstream software development has been problematic, mainly for the reason that the formalisms are not understandable to most stakeholders in the system being developed. In the same way, we would not want to let our users look at pieces of ADL or scrutinize HTN operators! One of the approaches being pursued in formal methods is the use of "methods integration" [4], that is using an informal, graphical front-end method to allow non-mathematicians to develop the specification. Languages such as Z and methods such as OMT have been fused this way [4]. After initial informal development, tools translate the diagrammatic language to a formal specification language. A formalist then fills in more details of the specification (the informal language invariably leads to underspecified action schemata) and the result is (ideally) mapped back to the friendly front end for further validation.

For an introduction to the algebraic and model-based formal specification one can consult reference [66]. This reference also describes how a Tweak-like planner can be be represented in VDM-SL, how its plan-space operations can be specified using VDM operations, and how the whole specification can be rigorously translated into a logic program.

5.2. Domain Analysis

With present planning technology, finding heuristics and domain control knowledge to improve planning efficiency and plan quality is an important aspect of the knowledge acquisition process (*cf.* section 7). In the problem domains there tends to be a rich structure "hidden" in the domain description and some rules follow naturally from this structure, given the planning algorithm. Instead of letting a domain expert formulate this knowledge or let a machine learning system discover it (*cf.* next section), it is often possible to find it automatically. In short, "if a machine can do something for you, don't do it yourself!"

Domain analysis (DA) is the automatic pre-processing of planning domains to extract knowledge about the domain and problem. DA is different from machine learning in that nothing is learned: The knowledge found is always a logical consequence of the domain and problem description. Some DA techniques depend on the planning problem instance, while others are independent of the problem and just need a stable and fully-known domain description.

DA techniques can aid planning and knowledge engineering for planning in several ways:

Planning speed-up and plan quality improvement: This has been the focus of research in domain analysis so far. It can be done either "off-line", i.e. only once for each domain, or "on-line", i.e. for every problem instance, depending on the DA technique.

Model validation: Static analysis can aid in the internal validation of a domain description, e.g. to find state invariants for user inspection, and to analyze the effects applying of hand-coded control knowledge.

Matching planning technology with domains: Domain analysis deals with structural features of planning domains and problems, and thus it can help in choosing the right planner and/or the right control knowledge for this planner in an automatic way.

There is a large variety of domain analysis techniques described in the literature, but most of them are integrated with a specific planning system and are not available as separate modules. A reason for this is that domain analysis does not directly produce control knowledge: It is only in combination with knowledge of the planning algorithm that domain knowledge can be effectively used for control.

TIM [22, 23] finds types and state invariants, as do DISCOPLAN [25], c-Constraints [63] and others [59]. RIFO [53] removes irrelevant facts and operator instantiations, while RSA [62] and RedOp [31] find different types of constraints on what action sequences are necessary or relevant for solving a given problem. Detection of symmetry [23, 18] and goal ordering [40] can also speed up planning.

Some DA techniques find knowledge which is useful in combination with particular planning algorithms, e.g. STATIC [20] or Alpine [39], others are helpful for a class of them, e.g. TOP [62] for total-order planners. Several planners have some analysis preprocessing step, e.g. the graph construction with mutexes in Graphplan [11] or precomputation of heuristics [58, 12].

The extraction of properties such as types and invariants can be a way to automatically characterize planning domains and problems [44]. It can also be used to detect subproblems, e.g. a TSP or shortest-path problem, embedded in a planning problem [45].

5.2.1. Action for the Planning Community

According to the points above, the planning community should

- Decouple domain analysis techniques from planning engines, so as to enable arbitrarily combining different DA techniques and planning engines. For this to be possible, the domain description formalism that is input to the planning engine must support expression of domain knowledge.
- Analyze hand-coded control knowledge in use and design automatic tools to find as much of that knowledge type as possible.

5.3. Machine Learning (ML)

In the AI research literature, there are many systems that use "learning" tools in the development of a planner, or within the planning product itself. This is an attractive area from the symbolic ML point of view, as planning involves the acquisition of knowledge and high level cognitive skills. Planning programs can then be used as the performance components in evaluating learning techniques. ML techniques, on the other hand, can help planning in several ways:

Knowledge acquisition: ML techniques could be used to remove the bottleneck of eliciting knowledge in much the same way as ML techniques have been used in front-ends for expert systems. For example, it may be more useful to induce the symbolic specification of complex actions than trying to get an expert to describe them in formal terms. For instance, in references [14, 28] experimentation is used in the domain to extract domain knowledge, while in [68] a domain expert agent uses observation with the same purpose, and in [24] domain actions are learned from an autonomous agent moving around in a robotic domain.

Model validation: Complex models contain errors, for example errors of incorrectness and incompleteness. If classified training data is available then, for example, a theory revision tool could be used to help identify and remove errors.

Learning to improve the quality of plans finding plans of good or optimal quality is very important. Here "quality" is any metric applied to a plan, e.g. economic cost of applying the plan, time required to execute it, number of operators in the plan, etc. Examples of such planners are QUALITY [54], Hamlet [13], Scope [19].

5.3.1. Exploiting ML for Planning Speed-up

Perhaps the most popular application of ML to planning is in plan generation speed-up, where learning concerns itself with improving planning efficiency (decreasing resources in time and space). Acquiring heuristics for any knowledge-based task is as difficult as acquiring the model's dynamics and structure. In the case of planning, to hand-code heuristics the user would need to know how the planning engine works to be able to define correctly those heuristics. Therefore, ML techniques have the potential to overcome this knowledge acquisition bottleneck by automatically learning those heuristics, with the possibility of presenting them to the user to refine or validate them. There have been many approaches, for example:

Macro-operator learning: Macro-operators are sequences of planning operators that have been compiled into a form that can be easily retrieved to form a partial solution to a planning problem. Early work concentrated on macros to help in general problem solving [41, 56]. In the FM system [48] macros were used to make problem solving in STRIPS-robots domains highly efficient.

Analogy/Case Based Reasoning: This approach uses planning problems that have already been solved to guide the solution of similar problems e.g. Chef [30], Analogy [67], CAPlan/CbC [51, 7].

Learning sequences of subgoals: It is also possible to learn sequences of subgoals as stepping stones in the planning process. For example SteppingStone [60], EAS [61] (which is also CBR based).

Learning heuristics: Planning is usually seen as a search process, therefore it is possible to use heuristics to guide this process. The main kind of systems proposed are Pre-processing systems and Trace-based systems.

Trace based "deductive" systems are similar to DA techniques in that they use the domain description, but in addition rely on a few traces of the planner after solving planning problems to create control knowledge for that domain. Most of them use EBL e.g. Prodigy-EBL [49], ULS [16], for total order planners [8], and SNLP+EBL [37] for partial order planners. There are also hybrid systems, like DERSNLP+EBL [34] that combines CBR and EBL. FM [47] used a deductive technique to create the hypothesis space for the conditions of a heuristic that was further refined using induction as more traces became available.

Trace-based systems can be purely inductive, relying mostly on planning traces to build control knowledge. They generalize from the specific traces to build planning control knowledge, examples are given in references [49, 21, 43, 42]. Multi-strategy systems combine deductive and inductive systems to overcome the limitations of both kind of approaches (e.g. FM [46], AxA-EBL [17], Hamlet [13], SCOPE [19], EvoCK [3]).

5.3.2. Conclusions

ML is as relevant for KE in planning as it is for KE in other fields: ML-based tools have the potential to reduce the knowledge acquisition bottleneck. For instance, if learning and/or analysis is used, not so much knowledge for planning efficiency has to be acquired.

However, it is true that not enough effort has been done to apply ML techniques to real-size planning problems. This does not mean ML is not relevant, but that a lot of work needs to be done in that area. In particular, a lot has to be done in studying the interaction between a ML system and a planning expert/user/knowledge engineer (for instance, to validate knowledge automatically acquired). Also, there are problems with the learning techniques themselves. For instance, EBL generally requires correct, complete and tractable domain theories and has to be used carefully so as not to actually degrade system performance (the "utility problem"). With inductive learning, the choice and order of training examples may be critical to the success of the learning system [38].

6. Knowledge Engineering Support Tools and Environments

6.1. Tool Support

Naturally, the amount and coverage of tool support required depends on the size and nature of the application. Many researchers seem to use nothing more than basic syntax checkers in support of their model building process. They then "debug" their model exclusively through dynamic testing. For an application requiring a larger domain model, this approach appears at best inefficient.

For knowledge acquisition, tools are needed to support interviews with domain experts and encoding of their knowledge. Learning tools (see above) that for example induce operator descriptions from traces of actions may also be used. A further possibility is to provide an interface to the model's formal language that allows a domain expert to directly encode planning knowledge. This could well be more efficient than having a "middle-man", and may preserve in the user the feeling of control and ownership of the problem.

Developing domain models in isolation from a planning engine means that we can split the process of into user inspection, static validation, and dynamic validation (animation).

In user inspection, an appropriate tool would be one that maps back and forth between a user-friendly, diagrammatic language and the domain model language itself. This kind of tool would be similar to a graphical front-end to a formal specification language.

Tools for static validation essentially perform domain analysis: They reason with the model to check that it is self-consistent, to check that it is complete (in a restricted sense) and to output consequences of the model that might be useful in for user inspection. For example, tools may check that an operator is consistent (it never inputs a valid state and outputs an invalid state); reason with operators and output state invariants to be visually checked by a user; or output necessary goal orderings, to check for impossible goal combinations and help in dynamic testing.

Dynamic validation entails acquiring a set of test cases and associated (optimal) plans and using these to test the functioning of the planner as well as the validity of the model.

Both the O-Plan and SIPE projects have developed tools that help in the knowledge engineering process. With O-Plan we have the "Common Process Method" [55, 65]. With SIPE we have the Act Editor [52]. Both are essentially sophisticated directed graph presentation tools, i.e. they let the developer examine the nodes in a HTN operator and the temporal constraints between them. Two deficiencies of the O-Plan/SIPE visual tools is that (1) they provide no visualization or checking for the state based model that the user builds around the operators. There is no definition of what properties the operators available for refining a task must hold nor the transitions that a domain object can pass through or the states that it can exist in. (2) There has been no evaluation of these tools to determine if they help domain model development and to what extent.

6.2. Actions for the Planning Community

Following on from (1) and (2) above:

- Carry out research studies to evaluate the effectiveness of currently available visualization tools, and derive lessons that can be used in the design of future tools.
- Aim to design visual tools that draw on the whole of the domain model (objects, object hierarchies, invariants, states, as well as actions) and help in checking self consistency, accuracy and error removal.

6.3. A KE Support Environment

After having discussed the nature of KE, and the kinds of tools likely to be found in a KE environment, we now use Figure 6.1 to show the kind of architecture that integrates these artifacts. This "idealised planning KE environment" was inspired by the Planform project proposal [1], although of course it could be changed to other topologies. It may be, for example, that the interface tools to the domain model could be assumed to be interfacing to all aspects of the system, in which case "Planning Application" would be at the core of the environment.

Note that we concentrate here on the knowledge-based aspects of the environment; as the application of planning technology will generally be a complex task, it seems necessary that issues of configuration management, version control etc must also be addressed. These and other vital concerns relating specifically to software engineering and project management will not be considered further, as they lie outside the "knowledge-based" concerns.

Users, Managers and Domain Engineers may want to add and adjust knowledge, apply measurements to the model, inspect the model, animate the model (using a simple planner) or explore the truth of properties of the model.

Existing data and models may form a substantial part of the planning system, and will need a customized acquisition tool to convert it into the internal format.

The results of previous analysis and feasibility studies may prove useful - hence information from known system models and from the project's requirements specification will influence the development of the planner.

Existing plans are a database of the kind of plans expected from the final planner. Hence they can be used in may ways: For initial knowledge acquisition, as a source of heuristics for the planner, and as validation of the final planner.

So that knowledge is not "hand coded" into the planning architecture, and to avoid problems in maintenance of the system, it seems sensible to keep separate algorithmic, heuristic and fundamental domain knowledge until these can be fused into a planning application. Ideally, this fusion will be performed by a compilation tool. This tool would, using aspects of the domain, evaluate to find out which planning technology was most appropriate and from this build up a final planning application.

After an initial knowledge acquisition and static validation phase, the compiler can be used to produce an application on which traditional dynamic testing can begin.

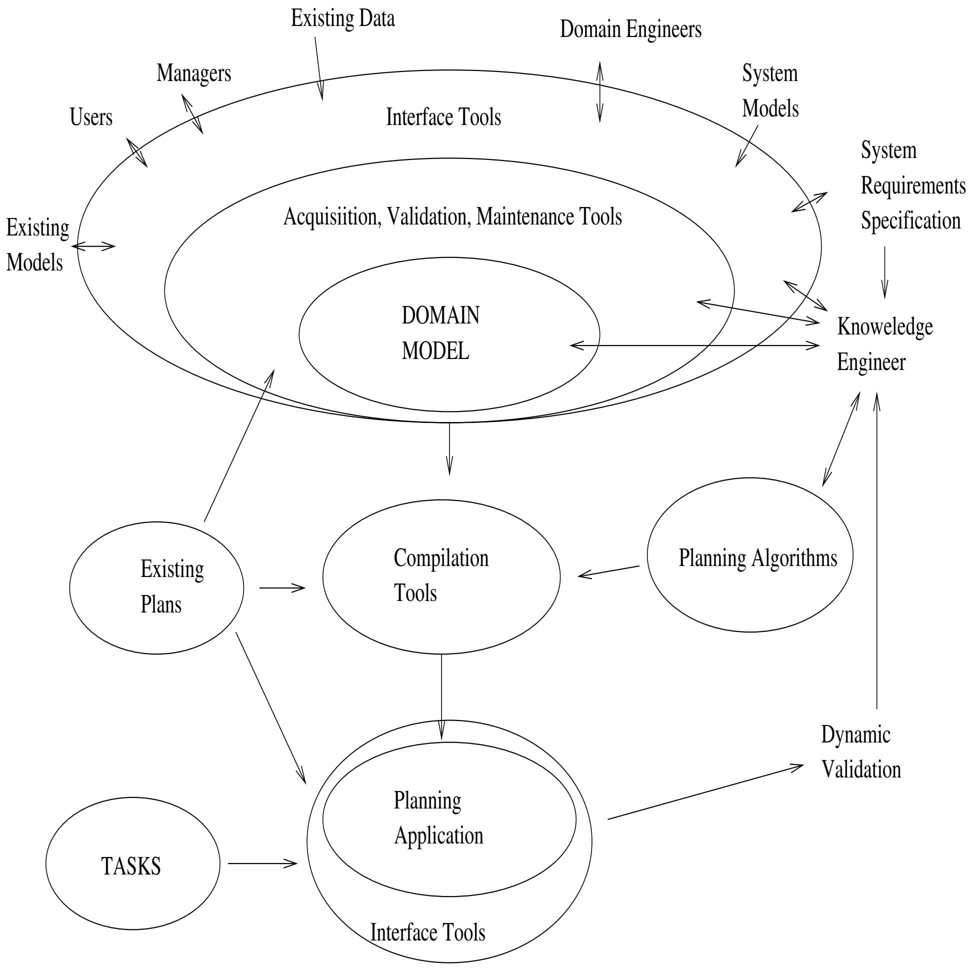

Figure 6.1.: An Idealised Planning Knowledge Engineering Environment

7. Knowledge Requirements

What kinds of knowledge must be captured in order to build a planning application? How can we provide better formalisms for capturing and expressing the required knowledge?

A domain modeling language should:

- Be structured. It should provide mechanisms that allow complex actions, complex states and complex objects to be broken down into manageable and maintainable units. For example, the dynamic state of a planning application could be broken down into the dynamic state associated with each object. On this structure can then be hung ways of checking the model for internal consistency and completeness.
- Be associated with a method. This will give a set of ordered steps to be carried out in order to capture the domain model, thus guiding the knowledge engineer throughout the process. It could contain activities such as modeling of state changes and the discharging of proof obligations.
- Support the operational aspects of the model. The language's framework should include a set of properties and metrics which can be evaluated to assess a model's operationality and likely efficiency. It should be possible predict whether the model can be translated to an efficient application.
- Be tool supported. These tools will support the steps in the method and, using the structure of the model language, be able to provide powerful support for statically validating, analyzing and operationalizing the model.
- Be expressive and customizable. The language needs to be generally applicable, yet customizable in some sense so that it "fits" well with applications. Since there is a whole range of assumptions involved in planning which may or may not hold in an application (e.g. to do with uncertainty, resources, closed world) it may be that the modeling language will have "variants" to deal with different assumptions.
- Have a clear syntax and semantics. As a basis for the other aspects above, and for the analysis of models encoded in the language, it should be possible to map models to a "meaning" within some well-known formal system such as a modal logic.

It seems that no modeling language fulfills all these criteria. Languages that have been developed from the point of view of knowledge acquisition include DDL.1 [15], TF [65] and OCL [1]. The most commonly used domain description language, PDDL [27], fulfills only the last. PDDL, however, has the potential to be easily extensible and at the same time widely accepted and used within the planning community.

7.1. Actions for the Planning Community

Evaluation of knowledge engineering methods: How do we evaluate and benchmark knowledge engineering methods? It appears very expensive in terms of time and effort to carry out case studies, and even then we only have the results of the case. It appears that the only other way is to compare a method to existing methods in similar domains, or base a KE for planning method on an existing one e.g. a planning-oriented form of KADS. It could be argued that the general problem of method evaluation tends to make researchers avoid looking into KE as it is harder to publish (given the problems of evaluation of the research).

Human Planning Knowledge: How should the problem of encoding human planning knowledge be dealt with? What are the benefits and trade-offs? The benefits are that if the system can draw on human control knowledge in a specific domain then that distilled knowledge can be used to great effect in planning. However, it is also a brittle solver that requires a lot of effort to build and maintain. In a sentence, it might be dirty but if you want your problem solved in today's technology then you better encode control knowledge. To quote Ginsberg's example, flights are scheduled hence we should plan them before the taxi ride to the airport. The domain independent heuristic in this case is "plan the most constrained things first". How do we put this intuition into practice?

Acquisition of control knowledge: How can search control knowledge be captured, while hiding the details of a planning system's implementation? A possibility is to capture and formalize "deeper" knowledge about the domain, and leave it to a planning expert how to make use of this knowledge in the context of a specific planning algorithm. Another way is trying to understand the relation between structural features of problems and properties of planning techniques on the one hand and the corresponding control knowledge on the other. This can automate the choice of relevant control knowledge.

How can domain and/or control knowledge be formalized and represented in a clean way? Can we find a way of better way of linking domain specific control knowledge to domain independent control knowledge?

8. Taxonomy of Planning Methods

Consider an experienced systems analyst or software engineer who is working on a system that would be enhanced by the ability to reason about actions or to plan. In this section we consider how AI planning methods should be indexed so that a person in this position can readily identify applicable methods or, recognizing that this is a research area, the groups that might most readily develop the required technology. We start by outlining work that has addressed this problem in some way and conclude by suggesting how future work should progress.

8.1. Existing Taxonomies

Refinement Planning Kambhampati [36] aims to provide a unified overview of plan synthesis strategies. He constructs a "refinement planning" strategy and then shows how the majority of planning methods are a special case of this method. While this work has enabled planning researchers to identify subtle distinctions between planning methods it is not intended to be exploited by people outside the planning community. The benefits of the framework to our stereotypical systems analyst or software engineer are not immediately obvious.

The Agent, Task, and World Triangle Aylet and Jones [5] describe a three point framework for categorizing planning methods. We summarize each in turn.

Agent Dimension: A planning problem may demand the control of a single agent or multiple agents. The degree of cooperation in the multi-agent case my also vary. For example, two robots carrying a tray must be closely coordinated while two robots working on completely independent task will not demand much coordination. We must also consider the abstraction level at which a planner must work down too. In the case of robots detailed planning possible to the actuator level may be necessary while with human agents typically higher level instructions can be given.

Task Dimension: Does the problem demand one or several tasks to be planned for together and what is the degree of interaction between those tasks. Also is it sufficient to specify only the goal state of a problem or must intermediate conditions such as safety constraints also be expressed.

World Dimension: How many objects and relations will be required in the conceptualization of the problem. Can the world be considered static and changing only under the actions of the agent(s) for which we are planning or will it change independently or even intentionally attempt to disrupt our plans. Will the planner be able to assume complete and accurate information about the world or will the agent have a degree of uncertainty in its beliefs and the need to plan to acquire information.

While Aylet and Jones provide an appealing taxonomy which intuitively might be of use to our analyst or engineer, they only place their own work within it. It might be beneficial if further work considered positioning a range of planning methods on these axis.

8.2. Problem Solving Methods

Benjamins et al. [6] extend the work on problem solving methods in other areas of AI to the planning area. There framework provides a list of the basic knowledge roles used in planning methods, basic methods for composing planning strategies, and a set of features designed to enable our analyst or engineer to ascertain which methods are applicable to a given problem. While the work is well founded formally, it is arguably still too far removed from our analyst or engineer to be of use. It would be useful to consider combining this work with that of Aylet and Jones to create a well founded taxonomy which is expressed in terms accessible to a systems analyst or software engineer.

8.3. Suggestions for further work

Work in line with that of Kambhampati is not designed to assist lay people in understanding or applying AI planning technology. While such provides excellent insights for AI planning researchers more work is needed in line with Aylet and Jones and Benjamins et al. In the short term it would be useful for a organization such a PLANET to produce a taxonomy of the planning methods under development by its members in a framework akin to that given by Aylet and Jones. This would be a major step towards the sort of guidance our analyst or engineer require. It might also identify gaps in the coverage of planning methods that could motivate further AI planning research.

8.4. Knowledge Engineering Requirements for Applied Technical Areas in PLANET

We need to summarize the real world problem areas dealt with in PLANET in terms of Knowledge Engineering. For example in WorkFlow Management there are two key contributions from KE in AI planning.

First, existing workflow systems take a model of a business process and then use that to co-ordinate the execution of an instance. If one views a business process description as a HTN operator and a business process instance as a instantiation of such an operator (i.e. a plan) then the KE work can help as follows: We have worked out ways of representing time, resource, and state knowledge. If we then develop methods for modeling this knowledge then these should help business process modeling too. This kind of modeling may well be of use to standards bodies or commercial consortia such as the Workflow Management Coalition.

Second, at the moment business process descriptions are fixed (universal plans). AI planning can help in synthesizing a process instance that takes into account the situation and purpose of its execution. If we had a good method for capturing process knowledge and for reasoning with it then the workflow community would (a) use such a method and (b) provide a market for the actual use of AI planning technology. From the workflow TCU we can learn about the requirements of action representation.

An example from document versioning: Basically this requires a planner to handle the creation of new objects which is tricky with current planning technology - especially when quantification is required. However, this is a pretty simple everyday requirement.

9. Summary of Problems

The roadmap points out some general problems to be overcome. These include:

1. There is a general lack of experience in the planning community (especially in Europe) of knowledge engineering concepts and methods, and we have much to learn and techniques to import from related knowledge engineering work.
2. The evaluation of KE tools and methods is problematic, and tends to be harder than the evaluation of, say, a planning algorithm. This is seen as a barrier to future research as researchers find it harder to publish in the planning literature.
3. We need to learn how to characterize domains, and hence build up knowledge about how to match up planning technology with domain models. An example of work in this direction, based on statistical analysis, may be found in [33].
4. KE involves a group of stakeholders with differing backgrounds and knowledge of computing. In particular, it includes consideration of Human Factors
5. Planning research has a history of association with toy problems where KE issues don't count that much.

9.1. Summary of Actions

The problems lead to some general actions that the community should carry out. These are summarized in Figure 9.1.

1. Survey KA / RE / FM areas for tools, techniques, and methods that may be relevant to planning.
2. Distill experience and induce general methods from the experience of previous applications of planning technology. We can start by studying KE in the application TCUs in PLANET.
3. Attempt to build basic, integrated engineering environments for building planning applications. This may also involve promoting standards and/or formalisms that enable integration of planning engines and support tools developed separately.
4. Find a research methodology that allows efficient evaluation of work in KE to allow a route to publication and hence attract more research effort.
5. Build on previous work to create a planner taxonomy usable by knowledge engineers.

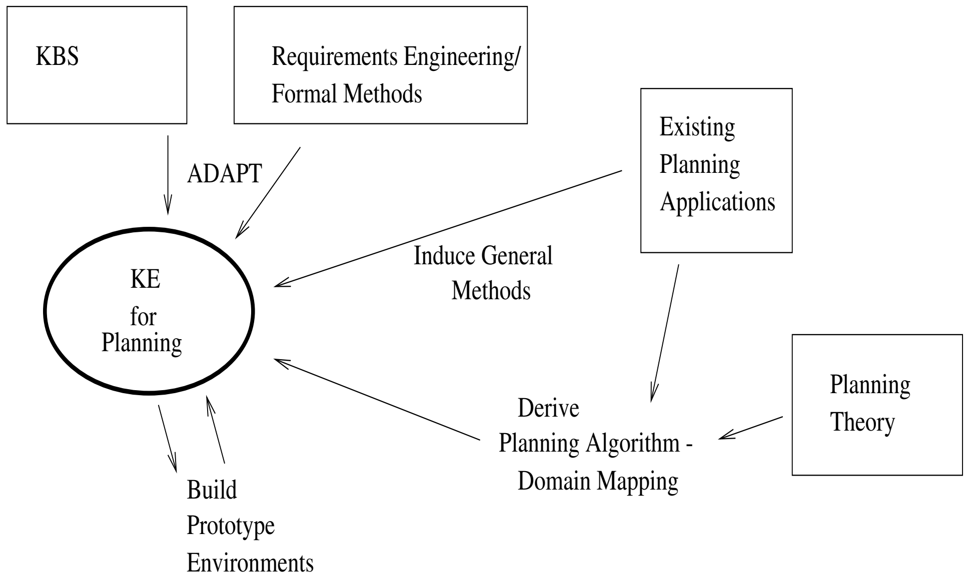

Figure 9.1.: Roadmap Summary

Bibliography

[1] Planform: An open environment for building planners. http://helios.hud.ac.uk/planform.

[2] M. Aben, J. Balder, and F. van Harmelen. Support for the formalisation and validation of KADS expertise models. Technical Report KADS-II/M2/UvA/DM2.6a/1.0, ESPRIT, 1994.

[3] Ricardo Aler, Daniel Borrajo, and Pedro Isasi. Genetic programming and deductive-inductive learning: A multistrategy approach. In Jude Shavlik, editor, *International Conference on Machine Learning*, pages 10–18, Madison, Wisconsin, July 1998.

[4] P. M. Allen and L. T. Semmens. Integration of structured methods and formal notations. In *International Conference on Information Systems Development*, Bled, Slovenia, September 1994.

[5] R. Aylet and A. Jones. Planner and domain: Domain configuration for a task planner. *International Journal of Expert Systems*, 9(2):279–316, 1994.

[6] R. Benjamins, L. Barros, and A. Valente. Constructing planners through problem-solving methods. In B. Gaines and M. Musen, editors, *Proceedings of the Tenth Knowledge Acquisition for Knowledge-Based Systems Workshop*, 1996. http://www.swi.psy.uva.nl/usr/richard/html/kaw96-plan/doc.html.

[7] Ralph Bergmann and Wolfgang Wilke. On the role of abstraction in case-based reasoning. In *European Workshop on Advances in Case-Based Reasoning*, volume 1168 of *LNAI*, pages 14–16, Berlin, November 1996. Springer.

[8] Neeraj Bhatnagar and Jack Mostow. On-line learning from search failures. *Machine Learning*, 1(15):69–117, 1994.

[9] S. Biundo and W. Stephan. Modeling planning domains systematically. In *Proceedings of the European Conference on Artificial Intelligence (ECAI)*, 1996.

[10] Susanne Biundo and Maria Fox, editors. *Proceedings of the European Conference on Planning*. Springer, September 1999.

[11] Avrim L. Blum and Merrick L. Furst. Fast planning through planning graph analysis. *Artificial Intelligence*, 90:281–300, 1997.

[12] Blai Bonet and Hector Geffner. Planning as heuristic search: New results. In *[10]*, pages 359–371, 1999.

[13] Daniel Borrajo and Manuela Veloso. Lazy incremental learning of control knowledge for efficiently obtaining quality plans. *AI Review Journal. Special Issue on Lazy Learning*, 11(1-5):371–405, February 1997.

[14] Jaime Carbonell and Yolanda Gil. Learning by experimentation: The operator refinement method. In R. Michalski and Y. Kodratoff, editors, *Machine Learning: An Artificial Intelligence Approach*, volume III, pages 191–213. Morgan Kaufmann, Palo Alto, CA, 1990.

[15] A. Cesta and A. Oddi. DDL.1: A formal description of a constraint representation language for physical domains. In *[26]*. 1996.

[16] M. P. Chase, M. Zweben, R. L. Piazza, J.D . Burger, P. P. Maglio, and H. Hirsh. Approximating learned search control knowledge. In *International Workshop on Machine Learning*, pages 218–220, 1989.

[17] W. W. Cohen. Learning approximate control rules of high utility. In *International Conference on Machine Learning*, pages 268–276, Austin, TX, 1990. Morgan Kaufmann.

[18] J. M. Crawford, M. Ginsberg, E. Luks, and A. Roy. Symmetry breaking predicates for search problems. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning (KR)*, 1996.

[19] Tara A. Estlin and Raymond J. Mooney. Learning to improve both efficiency and quality of planning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1227–1233, San Francisco, August 1997. Morgan Kaufmann.

[20] Oren Etzioni. Acquiring search-control knowledge via static analysis. *Artificial Intelligence*, 62(2):265–301, 1993.

[21] Oren Etzioni and Steven Minton. Why EBL produces overly-specific knowledge: A critique of the Prodigy approaches. In *International Conference on Machine Learning*, pages 137–143, Aberdeen, Scotland, 1992. Morgan Kaufmann.

[22] Maria Fox and Derek Long. The automatic inference of state invariants in TIM. *Journal of AI Research*, 9:367–421, 1998.

[23] Maria Fox and Derek Long. The detection and exploitation of symmetry in planning domains. Technical Report tr 1/99, Durham University, 1999.

[24] Ramn Garca-Martnez and Daniel Borrajo. An integrated approach of learning, planning, and execution. *IEEE Transactions on Systems, Man and Cybernetics*, 1998.

[25] Alfonso Gerevini and Lenhard Schubert. Inferring state constraints for domain-independent planning. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, pages 905–912, 1998.

[26] M. Ghallab and A. Milani, editors. *New Directions in AI Planning*. IOS Press (Amsterdam), 1996. Proceedings of the 3rd European Workshop on Planning (EWSP95), Assisi, Italy, September 27-29, 1995).

[27] Malik Ghallab, Adele Howe, Craig A. Knoblock, Drew McDermott, Ashwin Ram, Manuela Veloso, Daniel Weld, and David Wikins. PDDL - the planning domain definition language. Technical Report CVC TR-98-003/DCS TR-1165, Yale Center for Computitational Vision and Control, 1998.

[28] Yolanda Gil. Acquiring domain knowledge for planning by experimentation. Technical Report CMU-CS-92-175, School of Computer Science, Carnegie Mellon University, August 1992. PhD thesis.

[29] S. Greenspan, J. Mylopoulos, and A. Borgida. On formal requirements modeling languages: RML revisited. In *International Conference on Software Engineering*. IEEE Computer Science Press, 1994.

[30] Kristian J. Hammond. Chef: a model of case-based planning. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, 1986.

[31] Patrik Haslum and Peter Jonsson. Planning with reduced operator sets. In Steve Chien, Subbarao Kambhampati, and Craig A. Knoblock, editors, *Proceedings of the Conference on Artificial Intelligence Planning & Scheduling*, pages 150–158, April 2000.

[32] J. Hertzberg. On building a planning toolbox. In *[26]*. 1996.

[33] Adele E. Howe, Eric Dahlman, Christopher Hansen, Michael Scheetz, and Annelise von Mayrhauser. Exploiting competitive planner performance. In *[10]*, pages 60–72, 1999.

[34] L. Ihrig and S. Kambhampati. An explanation-based approach to improve retrieval in case-based planning. In *[26]*, pages 395–406. 1996.

[35] C. B. Jones. *Systematic Software Development using VDM*. Prentice-Hall, 2 edition, 1990.

[36] S. Kambhampati. Refinement planning as a unifying framework for plan synthesis. *AI magazine*, 18(2), Summer 1997.

[37] Suresh Katukam and Subbarao Kambhampati. Learning explanation-based search control rules for partial order planning. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, volume 1, pages 582–587, Menlo Park, CA, USA, July 31–August 4 1994. AAAI Press.

[38] Vera Kettnaker. Training of a case based reasoning planner with dynamically generated subproblems. Master's thesis, University of Kaiserslautern, January 1994.

[39] C. A. Knoblock. Automatically generating abstractions for planning. *Artificial Intelligence*, 68:243 – 302, 1994.

[40] Jana Koehler and Jörg Hoffmann. On reasonable and forced goal orderings and their use in an agenda-driven planning algorithm. *Journal of Artificial Intelligence Research*, 12:338–386, 2000.

[41] Richard E. Korf. Macro-operators: A weak method for learning. *Artificial Intelligence*, 26:35–77, 1985.

[42] C. Leckie and I. Zukerman. Learning search control rules for planning: An inductive approach. In *International Workshop on Machine Learning*, pages 422–426, Evanston, IL, 1991. Morgan Kaufmann.

[43] Christopher Leckie. Being correct is not enough - the role of domain knowledge in the utility problem. In *Australian Joint Conference on AI*, November 1993.

[44] Derek Long and Maria Fox. Automatic synthesis and use of generic types in planning. tr 3/99, Durham University, 1999.

[45] Derek Long and Maria Fox. Extracting route-planning: First steps in automatic problem decomposition. In *AIPS Workshop on Analysing and Exploiting Domain Knowledge for Efficient Planning*, 2000.

[46] T. L. McCluskey. Combining weak learning heuristics in general problem solvers. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*. Morgan Kaufman, August 1987.

[47] T. L. McCluskey. *Experience-driven Heuristic Acquisition in General Problem Solvers*. PhD thesis, The City University, London, 1988.

[48] T. L. McCluskey and J. M. Porteous. Engineering and compiling planning domain models to promote validity and efficiency. *Artificial Intelligence*, 95:1–65, 1997.

[49] Steven Minton. *Learning Effective Search Control Knowledge: An Explanation-Based Approach*. Kluwer Academic Publishers, Boston, MA, 1988.

[50] A. R. Moro, H. Yu, and G. Kelleher. Applying new search methodologies for scheduling FMS using petri nets. In *Proceedings of the 17th UK PlanSIG*, University of Huddersfield, UK, September 1998.

[51] Héctor Munoz-Avila and Jochem Huellen. Retrieving cases in structured domains by using goal dependencies. In Manuela Veloso and Agnar Aamodt, editors, *International Conference on Case-Based Reasoning Research and Development*, volume 1010 of *LNAI*, pages 23–26, Berlin, October 1995. Springer Verlag.

[52] K.L. Myers and D.E. Wilkins. *The Act-Editor User's Guide: A Manual for Version 2.2*. SRI International Artificial Intelligence Center, Menlo Park, CA, September 1997.

[53] Bernhard Nebel, Yannis Dimopoulos, and Jana Koehler. Ignoring irrelevant facts and operators in plan generation. In *[10]*, pages 338–350, 1997.

[54] M. Alicia Pérez. *Learning Search Control Knowledge to Improve Plan Quality*. PhD thesis, School of Computer Science, Carnegie Mellon University, July 1995. Also available as Technical Report CMU-CS-95-175.

[55] S. Polyak. A common process methodology for engineering process domains. In *Proceedings of the Systems Engineering for Business Process Change (SEBPC) workshop, SMBPI: Systems Modelling for Business Process Improvement*, University of Ulster, March 1999.

[56] Bruce W. Porter and Dennis F. Kibler. Experimental goal regression: A method for learning problem-solving hueristics. *Machine Learning*, 1(3):249–285, 1986.

[57] B. Potter, J. Sinclair, and D. Till. *An Introduction to Formal Specification and Z*. Prentice-Hall, 1991.

[58] Ioannis Refanidis and Ioannis Vlahavas. GRT: A domain independent heuristic for STRIPS worlds based on greedy regression tables. In *[10]*, pages 346–358, 1999.

[59] Jussi Rintanen. An iterative algorithm for synthesizing invariants. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, 2000.

[60] David Ruby and Dennis Kibler. Learning subgoal sequences in planning. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 609–614, San Mateo, CA, 1989. Morgan Kaufmann.

[61] David Ruby and Dennis Kibler. Learning episodes for op5ization. In *International Conference on Machine Learning*, pages 379–384, Aberdeen, Scotland, 1992. Morgan Kaufmann.

[62] Ulrich Scholz. Action constraints for planning. In *[10]*, pages 148–160, 1999.

[63] Ulrich Scholz. Extracting state constraints from PDDL-like planning domains. In *Proceedings of the Workshop at AIPS on Analyzing and Exploiting Domain Knowledge for Efficient Planning*, pages 43–48, April 2000.

[64] L. G. Shaw and B. R. Gaines. Requirements acquisition. *Software Engineering Journal*, 11:149–165, 1996.

[65] A. Tate, S. Polyak, and P. Jarvis. TF method: An initial framework for modelling and analysing planning domains. In *AIPS-98 workshop on "Knowledge Engineering and Acquisition for Planning: Bridging Theory and Practice"*, Carnegie-Mellon University, June 1998. AAAI Technical Report WS-98-03.

[66] J. G. Turner and T. L. McCluskey. *The Construction of Formal Specifications: An Introduction to the Model-Based and Algebraic Approaches*. McGraw-Hill series in Software Engineering. McGraw-Hill, 1994.

[67] Manuela M. Veloso. Flexible strategy learning: Analogical replay of problem solving episodes. In *Proceedings of the AAAI National Conference on Artificial Intelligence*, pages 595–600, Seattle, WA, August 1994.

[68] Xuemei Wang. Learning planning operators by observation and practice. In *Proceedings of the Conference on Artificial Intelligence Planning Systems*, pages 335–340, Chicago, IL, June 1994. AAAI Press, CA.

Part III.

Intelligent Manufacturing

Ruth Aylett

(Centre for Virtual Environments, University of Salford, UK)

with contributions from.

10. Introduction and overview

This document evaluates the fields of AI Planning and Scheduling as they apply or could be applied to Intelligent Manufacturing Systems. By manufacturing we mean the industries concerned with the production of physical goods whether by discrete or continuous process and whether for intermediate or end users. Thus cars, food, chemicals, machinery and electronics are all included among others. All phases of manufacturing are to be considered from design of plant and products, to operations to plant decommissioning, not only within individual enterprises but also between them, as in supply-chain management and the concept of the virtual enterprise. This requires an engagement with manufacturing concepts such as just-in-time (JIT), concurrent engineering, flexible manufacturing and holonics.

By Intelligent Manufacturing we mean the application of AI and Knowledge-based technologies in general to manufacturing problems as just described. This not only includes a large number of technologies outside the scope of PLANET – such as machine learning, expert systems, data mining and neural networks – but it appears that these same technologies have so far proved more popular than AI Planning and even AI Scheduling in such applications. Looking at the reason for the relative neglect of the PLANET technologies and for ways to overcome this neglect is one theme of this document.

We also distinguish the scope of this document from that being produced by the PLANET TCU on Workflow. The distinction is a little artificial since any business activity can be considered in terms of workflow including all those related to manufacturing. Nevertheless, for the purposes of this document, we see the concerns of the Workflow TCU as operating with processes in which information constitutes the main flow and people or software are the main executors of actions, while those of PIMS operate with processes in which physical material is the main ingredient and machines or plant form a major execution resource. The headings used by the 1998 workshop organised by the American Association for AI Special Interest Group in Manufacturing (SIGMAN) [1] provide a good guide. These were:

- Intelligent product and process design
- Production planning and scheduling
- Robotics, sensors and control
- Systems engineering, learning and architectures

This list also illustrates the potential overlap with the Robotic Planning (cf. chapter I) and the On-line Scheduling TCUs (cf. chapter V). The distinction to be made here is again one of the specific versus the general – this document will only encompass robotic planning and on-line scheduling as part of the manufacturing process rather than as general problems.

First we will discuss the state-of-the-art, both in Intelligent Manufacturing and in AI Planning and Scheduling. Next we will examine the requirements of the manufacturing domain for AI Planning and Scheduling. These requirements will then be mapped onto the current state-of-the-art in the technologies in order to assess what applications can currently be targeted and which require further research developments. These research developments will then be discussed and the time scales and resources required to achieve them will be analysed. We will also consider the organisational and social obstacles to these developments and how such obstacles might be tackled. This document concludes with overall recommendations.

11. The current state of the art

In this section we will consider the state-of-the-art both in the field of Intelligent Manufacturing and in Planning and Scheduling. It should be said at the outset that one of the issues is that these are two almost completely separate communities consisting of different individuals and focusing on different sets of conferences, workshops and other activity as well as mostly publishing in different journals. Intelligent Manufacturing can be considered as a subfield of those interested in applying IT in general to manufacturing, while AI Planning and Scheduling is related to Computer Science. In other words, PIMS must work in a multi-disciplinary fashion in order to create dialogue and synergy between the two communities, making the work understandable and appealing enough for both (see [4]). Work already exists showing that researchers can find a way forward [5, 6, 7].

A further separation arises from the fact that similar terms are used to mean rather different things – thus planning in the AI sense does not correspond to planning in the manufacturing sense. Consider the example of production planning. Where AI Planning is an abstract approach to generating sequences of actions to meet goals from a defined initial state, production planning often starts with templates containing sequences of actions and allocates resources to them, making it closer to what in AI would be called scheduling. Some production planning is at least as much concerned with computational geometry, as for example, generating the machining sequences needed to produce particular parts.

There is a closer match in the use of the term scheduling, but the term is typically used in the AI community to cover a wider range of activities than those covered by the term in the manufacturing community, as for example rostering and timetabling as well as much of logistics.

11.1. Intelligent manufacturing

11.1.1. Themes

There appear to be several major themes in this area reflecting the overall changes in manufacturing: integration, robustness, flexibility and ability to make rapid changes.

Integration

Integration ñ of processes within an organisation belonging to different stages of the production life-cycle, and between organisations, as in supply chain management (SCM) – is a major theme. Rather than managing one factory or one organisationís set of factories, the idea of supply chain management is to cover all the factories supplying inputs to a particular production process. This has emerged as a consequence of ideas such as Just-In-Time – the elimination of stock holding by producing inputs at the time when they are needed – which can only work properly with this type of integration. Supply chain management is probably best thought if as operating at a strategic level of ordering and delivery processes and may involve inter-company processes such as service-level agreements. Software packages for supply chain management are available from companies like Manugistics and i2, but they are currently very expensive and up to now only affordable by very large companies.

ERP (Enterprise Resource Planning) is a similar extension of MRP (Materials Resource Planning) from immediate factory floor requirements into a longer view of resources within the whole enterprise. Supply chain management and ERP while superficially covering similar ground in fact appear to operate at different levels in the organisation, to involve different groups of people and to be only loosely integrated with each other and with what actually happens on the factory-floor operations level. Being large systems with wide scope over different groups of people and processes they are difficult to implement and even harder to change once they have been implemented.

The latest buzz word in this domain appears to be Advanced Planning and Scheduling (APS). This seems to be an attempt to incorporate better technology as stand-alone 'point' solutions. APS is aimed on the one hand at smaller companies, who have not been able to afford an all-encompassing package, or on the other at integratable add-ons to existing ERP or SCM packages for larger organisations who feel their big systems need updating but do not want to reimplement the whole thing. While the introduction of AI Planning and Scheduling into the large integrated packages is problematic, influencing organisations producing APS solutions appears more feasible.

Note that integration supposes bringing together people doing different tasks within the organisation as well as automation using big software packages. For example, one might supply more information about operational constraints to designers of processes, or of design rationale and alternative possible designs to operations people in order to integrate design and operations more closely [25].

Robustness

Robustness is also an important theme in intelligent manufacturing. In the real-world, machines break, humans go sick, materials may be faulty. Errors must be diagnosed rapidly and accurately and the effects of problems and outages must be contained and handled. Replanning and rescheduling important isssues, and here links with the On-line Scheduling TCU in PLANET are very clear.

However planning for the most obvious contingencies in advance, rather than after the event, is also an important field. It is linked to the assessment of risk – what is most likely to go wrong – and to how to detect these contingencies during the execution of the plan whether by means of sensors or by any other information gathering action. One of the more important issues here is that the execution of the plan can not be broken after the ocurrence of such contingencies. Therefore there must be some capability of including pre-planned responses to these contingencies such that the execution of the plan never breaks but adapts to them.

Changes in the approach to maintenance are important to the theme of robustness. A former approach was that of scheduled maintenance, in which equipment was taken out of service and maintained at pre-determined intervals. However this is both wasteful of resources – equipment might be taken out of operation when it was functioning perfectly and not in need of maintenance – and high-risk , since equipment might break down before it was due to be maintained. A more current approach is therefore to use sensors and diagnostic processing to detect problems before they become serious and to schedule maintennance when it is required.

There are thus links to the theme of flexibility discussed below. An organisation which is flexible in the the face of changes in external demand may be able to use some of the same methods to respond to internal change through the use of rapid plant reconfiguration and on-line planning and scheduling.

Flexibility

A third manufacturing theme is flexibility in the face of customer requirements. In the classic manufacturing life-cycle, planning in the AI sense of producing an abstract sequence of actions is kept to a minimum, as it requires skilled personnel to carry it out and has a big impact on what happens on the factory floor. A factory that produces large runs of identical components, or a process plant that produces bulk supplies of the same chemical, are essentially pursuing one plan for long periods of time (maybe selected from a stable library of plans for a known set of products), with the scheduling of resources to support it on a day-to-day basis.

In this environment, while scheduling is carried out very often, and therefore may be seen as an activity that should be supported by computer-based tools, planning is carried out less often. In the past it was often linked to the introduction of a new product, new manufacturing process or new equipment. The engineers carrying out this planning often saw it as a one-off containing a large element of judgement and skill, and they did not see the need for extensive computer support.

However this environment is changing very rapidly. The basic idea of supply-chain management is that production should be determined by "pull" from the market, not by "push" from the production process. This supports minimal stock holding along the supply chain since manufacturing is then always responding to current demand rather than producing for stock. However, a consequence of the "pull" approach to manufacturing is that variation in customer requirements is fed back into production to a far higher degree. No longer can you only have a car in any colour as long as it is black, or the standard widget a factory produces. Large numbers of goods can now be configured on demand, leading to a much greater planning as well as scheduling requirement. The Advanced Planning and Scheduling (APS) developments referred to above can be seen as a response to this need for flexibility.

Responsiveness to change

A final theme, related to flexibility, is proactive responsiveness to change. The idea here is that a company ought to be "nimble", that is when it spots a new opportunity, it should be able to organise itself to meet it as quickly as possible. This means that, for example, it should be possible to have an idea for a new product and turn that idea into manufacture and marketing much more quickly as well as much more frequently than was the case in the past. The requirement for computer support for more skilled and creative activities springs from the idea that these can then be carried out more rapidly, as well as from the idea that more alternatives can ñ and must – be explored. Planning and scheduling support form an important component of exploring the consequences of a new idea as well as of putting the new idea into practice much more rapidly.

11.1.2. AI/KBS Technology in Manufacturing Applications

If one consults the proceedings of relevant conferences (for example [1, 8, 9, 24]) one finds a variety of problems tackled with a variety of techniques. Among these are some planning and scheduling applications related to different manufacturing problems, but these are approached very much from the viewpoint of simple heuristic search problems, and thus, seen as very simplistic AI problems. This reflects a general lack of knowledge by workers in the Intelligent Manufacturing community of generative planning technology and a tendency therefore to "reinvent the wheel" for these problems. For example, the history of work in the generation of operating procedures for chemical plant, discussed in [5], shows that chemical engineering researchers attacked it with state-space search and other simple techniques, finally arriving at a linear planner only in 1991. Because of the size and complexity of some of the search spaces considered, some workers in intelligent manufacturing have applied techniques such as genetic algorithms or constraint propagation, but have not drawn on the contributions of generative planning to the search of such spaces.

In the same way, much of the work on the machining of complex artefacts, which has been tackled by a large number of groups, uses other approaches than generative planning, such as case-based planning [21], constraint-based reasoning [26] or selection from plan libraries.

In the recent period, multiagent approaches to manufacturing problems have become more popular, especially as they map well onto distributed control systems. A particular intelligent manufacturing slant on multi-agents is provided by the concept of 'holons' [28, 29], a term coined by Arthur Koestler while analysing hierarchies and stable intermediate forms in living organisms and social organisations. He observed that although it is easy to identify subwholes or parts, 'wholes' and 'parts' in an absolute sense do not exist. He therefore proposed the word holon to describe the hybrid nature of subwholes/parts in real-life systems; holons simultaneously are self-contained wholes to their subordinated parts, and dependent parts when seen from the inverse direction.

One might think of holons as agents in the AI sense, though closer to Brooksian behavioural agents than the logic based agents, say, of BDI architectures, since reactiveness and adaption are often emphasised. For example, a holonic approach to route planning might allow a route to be determined dynamically as each holon interacts with the next one along, rather than by a global routing process. Holons are inherently hierarchical but the hierarchy is usually not a fixed one. A substantial amount of work is being carried out in applying this concept to a variety of applications such as scheduling [30], machine controllers [31] and Computer-Aided Process Planning (CAPP) [32]. It is a concept onto which one could map distributed planning and scheduling in the PLANET sense and represents an area in which the two communities could make contact.

11.2. State of the art in planning and scheduling

In this section we will consider the state of the art in planning and scheduling with the emphasis on what technology is being used for applications rather than on theoretical developments which are as yet to be applied. We begin by an informal definition of what is meant by the terms AI Planning and AI Scheduling, since as, noted above, they do not always correspond to the meanings accepted in the manufacturing community.

Planning: the automatic or semi-automatic construction of a sequence of actions such that executing the actions is intended to move the state of the real world from some initial state to a final state in which certain goals have been achieved.

This sequence is typically produced in partial order, that is with only essential ordering relations between the actions, so that actions not so ordered appear in pseudo-parallel and can be executed in any order while still achieving the desired goals. However some models do explicitly represent true parallelism between actions.

Scheduling: in the pure case, the organisation of a known sequence of actions or set of sequences along a time-line such that execution is carried out efficiently or possibly optimally. By extension, the allocation of a set of resources to such sequences of actions so that a set of efficiency or optimality conditions are met.

Scheduling can therefore be seen as selecting among the various action sequences implicit in a partial-order plan in order to find the one that meets efficiency or optimality conditions and filling in all the resourcing detail to the point at which each action can be executed.

The two definitions here reflect the division of the community itself into those concerned with planning and those concerned with scheduling – however as we shall see below this division is to some extent an artificial one.

11.3. What applications have been attempted and with what success?

It has to be said immediately that there are few example generative planning applications in manufacturing, though quite a number of scheduling ones. The domain with the greatest number of 'live' planning applications is Space: the European Space Agency have Optimum AIV [16], and an Astronaut training application, NASA have the Remote Agent, RAX, mounted on the Deep Space Probe last May [3], VICAR [14] - planning vision processing for scientists - and control of Deep Space Telecomms facilities [19]. While as seen below NASA researchers have drawn some interesting general lessons from these applications, the applications themselves are not something one could use as examples with a manufacturing audience.

Both SIPE-2 and OPLAN have been applied to a number of large-scale military logistics problems in the US, and AI Planning and Scheduling technology (mainly the latter) was famously applied in the DART [15] crisis action planning system used in the Gulf War which was said to have paid back all of 30 years funding of AI in the US in a matter of months. Again, these are not in general relevant to the needs of manufacturing though work using AI scheduling technology and constraint logic in particular has been applied to commercial logistics planning. SIPE-2 has also been applied to production planning for a brewery in Australia [34]. It generated a daily plan (master schedule) for two eight-hour shifts on each of six production lines, while the plan scheduled dozens of orders (for possibly hundreds of products) with approximately 20 separate product runs (with their corresponding needs for different raw materials).

Some promising work has been carried out in the last few years on plant control applications, whether at the level of plant operating procedures [5] or at the lower level of generating control sequences [6]. This work is at the proof-of-concept stage but it is clear that it could be taken through to industrial demonstrators and eventual systems if it were attractive to end-users in manufacturing. Planning has also been incorporated into a project investigating Wastewater plant control [23], though here reactive planning technology has been used as a variation on the idea of a plan library.

An interesting use of a planner to support diagnosis rather than plan generation was demonstrated in the TIGER project [17, 36] in which the temporal-logic based planner IxTeT worked out expected sequences of actions for gas turbines in order to catch errors if they did not in fact occur. However it is noticeably that this has not as yet been incorporated into the commercial version of the TIGER software, possibly because it was seen as a complex and high-risk component.

As indicated above, a lot of work has been carried out into the matching of complex parts, but almost entirely using case-based planning or plan libraries. an exception to this however is the work on manufacturability being carried out in the US IMACS project [38] which incorporates planning technology alongside feature extraction and other intelligent subsystems. The same group at Maryland have applied some of these ideas to process planning for the production of microwave modules.

One should note that the Agents Community are involved in a parallel activity to that of PLANET in assessing the fit of their technology to manufacturing [37]. While their preoccupation is at this stage with mapping problems onto a set of distributed agents and with negotiation and communication protocols, it is hard to avoid incorporating some form of planning into agents with any substantial functionality, and the possible convergence of interests should certainly be explored.

11.4. What lessons have been learned from current work?

One of the issues in examining current work is that it does not always consider the general lessons of the particular application being attempted. We present a tentative list therefore, with discussion following it indicating where each point was raised so far as this has been established.

a) Life-cycle costs must compete with traditional solutions

b) Knowledge must be acquired painlessly and quickly

c) Currently, configuring a planner to a new domain can be very time-consuming and require a planning expert

d) Most users want interactivity and mixed initiative. They like to retain the user in the decision-making loop and are very reluctant to allow planning software to automate decision-making.

e) Users want validation and ability to assess correctness of plans up front - hence the popularity of plan libraries

f) Users want results in the formalisms they are used to, rather than having to learn new ones

g) Integration with conventional software may be required to provide a complete solution

h) Users often express a wish for optimisation of plans in terms of overall resource allocation and for the ability to assess plan quality numerically or via an actually scheduled plan.

Issues a)-c) are very closely linked. The issue a) of competing with existing life-cycle costs has been raised very strongly by Steve Chien of NASA [19]. This point links in very closely with the work of the PLANET Knowledge Acquisition TCU, since the areas identified as potentially costly in comparison with conventional solutions were precisely those of modelling the domain – issue b) – and validating and verifying the knowledge acquired. The NASA work involved the development of tools for validation and verification, but such tools are not widely available for AI Planning systems. The same point has been noted by other, later work, see for example [5] where substantial effort was put into developing tools for acquiring planning knowledge in the domain of chemical plant operation in order to combat issue c).

Issues d) and e) have both been raised by workers who have gone for plan libraries, and more recently case-based planning [11, 21], rather than the generative planning approaches of PLANET. The advantage of both these approaches is that it is easy to involve the user in the selection process – though note that plan adaption is more problematic unless plan representations are used that prevent users from producing incoherent or invalid plans. A library of plans or of planning cases can be developed off-line and validated and verified, giving the user more confidence than they necessarily feel in a generative planning mechanism [20]. Similar points have been made with respect to users outside manufacturing, for example in military logistics [22], where point e) has also been raised.

Points f) and g) are also closely linked, since the formalisms that users are used to are often those produced by conventional software packages. A key part of the acceptability of the live application discussed in [16] – OPTIMUM AIV, used for organising the layout of rocket bays by the European Space Agency – was its integration with the Artemis scheduling application. It is argued strongly in [6] that the MACHINE application for generating plant control sequences will only be acceptable to control engineers if its output can be represented as the Petri nets or Grafcet diagrams they are accustomed to.

Point h) is a tricky one since optimisation in real-world domains is often difficult in practice with "satisficing" – producing an acceptable outcome – frequently more realistic. However if resources are not allocated to a plan it is often hard for users to see how its quality might actually be measured.

12. Areas of Application

12.1. What applications should we tackle?

Much industrial activity can be characterised by a very abstract sequence. First a problem to be solved is formulated – the business opportunity. An analysis and design activity results, in which an artefact – a process, a factory, a product – is designed to meet certain criteria. The end result of this activity can be thought of as largely declarative in the knowledge-engineering sense, consisting of a CAD model or other specification. Design is a much investigated activity, as discussed above, but even where it is routine rather than heavily creative has more in common with configuration or constraint satisfaction than with planning or scheduling, though the constraint technologies often used in scheduling can of course also be applied to it.

However a design must then be decomposed into the sequence of operations which will allow artefacts to be constructed which meet it. In planning terminology, a design defines the end state to be achieved by a sequence of actions. Thus a plant has to be built, a process has to be operated, an artefact has to be manufactured and assembled. All these activities require the right actions to be executed in the right order.

In the same way, when a problem occurs in a process, diagnosis establishes what fault exists. In the same way as a design, a diagnosis is a declarative statement, in this case of the start state which is to be changed by a sequence of actions back into the desired repaired state. Again, the right actions must be carried out in the right order.

Finally, we should add the need to plan/schedule for contingencies. Since the real world is not always predictable, plans must be built taking this into account. This may involve the production of extra or alternative plans, or it may involve plans adapting their execution to possible contingencies or changes during such execution using information from sensors, or from the interaction with human controllers.

We briefly consider some of the manufacturing life-cycle and consider where planning applications could be located.

12.2. Design

Design must take into account basic aspects of functionality and engineering constraints. However increasingly, it is being realised that good design should also take into account operational constraints. For a component this is manufacturability, for a process plant operability, in construction, buildability. In all cases the designer needs to know that the design can be executed, while the organisation as a whole needs to know something about the life-cycle costs of doing so. Designers often lack the specialised knowledge needed to make this assessment unaided - clearly there are a whole number of planning applications which could make this knowledge available and assist the designer in producing a more effective and practical design. This very much fits into the integration theme discussed above.

12.3. Process planning

In component manufacture, this can be defined as the act of preparing detailed operating instructions that transform an engineering design to a finished part. This is clearly a planning problem, but must also draw on knowledge of computational geometry as well as the real-world characteristics of tools and ,materials. Many specific applications have been developed in this area using a KBS approach, what planning technology can offer is a much more generic system, which, supported by the appropriate libraries and knowledge sources, could be easily adapted to new components and new production processes. In this way production of new components could be carried out much more rapidly and flexibly.

Process planning may also involve organising the workcell for production. Here configuration is a primary concern, but planning is then required to produce the desired configuration - in miniature yet another example of the link of planning to a design activity.

12.4. Production planning

We here define production planning as the more day-to-day activity of meeting the requirements of producing the right goods at the right time and price, with the right quality. While this is largely a scheduling activity, there is a clear interaction between planning production, and the basic process plan on which it is based. Investigation carried out for this document suggested that just as there is a gap between the concerns of the designer and the process planner, so there is a gap between those of the process planner and the production engineer. Process engineers normally propose a single process plan in which characteristics such as robustness and scope are implicit rather than explicit. A planning system is capable of producing alternative plans, in which the characteristics can be much more explicit, allowing production engineers to tailor the actions taken to the particular situation on the factory floor.

12.5. Operations and Process Control

There is a very clear relationship between planning and operations and control. In operations, humans carry out defined procedures, while in control, machine controllers do the same thing at a much lower level. In both cases, considerable effort up front goes into making sure that procedures and control sequences are correct and comprehensive. AI planning clearly has a potentially major role to play in this area [5, 6] which has so far been little explored.

The other major operational area is that of maintenance and problem-solving. Maintenance is changing quite rapidly in most industries from a routinely scheduled activity to one carried out when a problem is detected, increasing the role of automatic diagnosis systems. As argued above, where one has diagnosis, one also has repair, and this is essentially a planning task. For many maintenance activities, it is just a case of making sure a suitably comprehensive set of procedures have been developed in advance, but in some cases – process plant for example – this is non-trivial to achieve and coverage is far from complete. Here an AI planner has an obvious role to play.

This is also the case where problems may be signalled by complex signals of alarms. A diagnostic system is used to filter these for the root problem, but having done this, again, a course of action is required. It is impossible in complex plant to plan for every combination of faults in advance, while allowing human operators who may be under considerable stress to react without support may be potentially quite dangerous. A planning system could be sued here to generate good advice in dealing with a problem, with a particular strength in showing the causal links in the actions proposed and the assumptions made in proposing them.

13. The way forward

Here we will consider what the PLANET technologies have to offer, what new development and research is needed to increase the number of manufacturing applications, and what sort of applications look promising.

13.1. What does planning/scheduling technology have to offer?

Any business activity that involves deciding what actions to carry out and in what sequence can be thought of as one to which AI Planning can be applied. AI Planning has a number of potential advantages.

a) Flexibility and responsiveness Once the necessary knowledge base has been created for a particular domain, an AI planner can typically produce new plans very quickly and with very little user effort. This makes it useful for exploring 'what if' scenarios and contingencies which are often too expensive to consider manually. It also makes it easier to replan if execution hits errors.

b) An intelligent interface A hierarchical planner has the effect of increasing the level of abstraction at which a human user can operate since it can allow problems to be posed as high-level goals and work out the detailed implications. This makes planning quicker and also less prone to error as low-level interactions between actions are dealt with automatically.

c) Ability to maintain correctness Plan representations demonstrate how all the causal links match up to produce a correct plan. Such representations therefore provide a basis for user modification of plans while preventing the production of incoherent or invalid plans as a result.

d) Makes assumptions explicit In common with other KBS technologies, the explicit representation of knowledge in a planner can make it clear what is implicitly assumed in a problem that is solved manually. Thus it has a role to play in knowledge management and knowledge-sharing.

13.2. What new developments are needed?

Manufacturing as a domain has a number of important characteristics from the point of view of AI Planning:

- Execution is key - planning and scheduling are only relevant insofar as they improve execution
- Thus the changeableness of the real-world must always be considered. All decisions may need to be revised
- There are often many execution agents
- Getting accurate current information about the state of the whole system is often hard
- Sensor data may be important
- Costs and efficient use of resources are always important

This suggests some of the areas in which further research effort is needed.

a) Execution and replanning There is scattered work in this area, but no real consensus on how planning and execution are related and how replanning should be tackled if a plan fails during execution. Much planning research ignores the issue of how actions in the plan will be executed and does not consider whether execution resources should be represented or reasoned about during planning. In manufacturing, scheduling is required before a plan can be executed, so that ways of bringing planning and scheduling together are a vital research topic for the domain. Planners such as SIPE-2 and OPLAN have combined reasoning about resources with classic planning, as has much of the NASA work (see for example [18] on combining planning and scheduling), but this technology is not perceived as being generally available to the research community.

Work in the Robot Planning TCU is of great relevance here as planning for robots naturally does have to confront these issues. Research in robot planning has considered the interleaving of planning and execution for example, as well as the issue of planning sensor actions as a way of gaining information for planning. A major theme in robot planning is also the integration of predictive and reactive planning as a way of making activity more resilient in the face of a changing environment.

b) User-friendly tools There is so far very little work in this area. While NASA developed some in-house tools for validating and verifying domain models, such tools are not generally available to the research community, never mind to potential users in manufacturing. This is in contrast to the situation in Knowledge-Based Systems in general, where such tools have been extensively developed for many years.

Here, work in the Knowledge Acquisition TCU is of great relevance, since the costs of adapting a planner to a particular domain are currently very high and an expert in both planning and the domain are usually needed. This is a major barrier to the use of the technology outside the home community.

c) An easy entry-level Expert systems became very widely applied at least partly because of the existence of small, low-function expert system shells, initially arising from the wide dissemination of the MYCIN story and the production of the shell E-MYCIN. Shells enabled enthusiasts in industrial domains to experiment at low cost and risk and to gain an appreciation of what the technology was good for. AI Planning has no such equivalent, since although AI planners are now available over the web they are non-trivial to use. Research is needed to establish whether a simple planning shell is feasible and what it should look like.

d) Integration There is little work investigating integration issues either with existing conventional tools or with other AI and computer science techniques. For example, planning machining sequences requires interaction between a planner and computational geometry system. Planning repair actions after a failure requires interaction between a planner and a diagnosis system. Little investigation seems to have been carried out into general mechanisms for supporting this type of interaction, leaving groups who need it to work out ad hoc techniques for particular domains.

It is clear that just as the APS systems mentioned above have been built as bolt-ons to existing manufacturing software, so the PLANET technologies must be integrated with the conventional software already in use if they are to be taken up. This is another area in which there is common ground with the Agents Community, since integration with legacy software is an issue researched there.

The planning community also appears to have split into separate camps concerned with generative and with case-based planning. Yet a generative planner is an obvious way of producing material for a case base, while planner representations, as argued above, provide necessary constraints and safe-guards where users are allowed to adapt plans from a case-base.

e) Mixed initiative systems Little work is carried out in the AI Planning community into interactive or mixed initiative planning. While individual projects have seen the need to support interaction ([5, 20, 35]), the only well-known project to concentrate on this issue was TRAINS [33] which in recent years has seemed to focus more on natural language aspects than interactive planning ones. Research is needed to establish what the general choices are for interaction with a planner and how a true mixed-initiative system can be built.

f) Common ontologies and libraries Work on ontologies has been carried out in the ARPA Planning Initiative, and to some extent in the development of the domain description language PDDL. However much more work has been carried out by groups in other areas of KBSs and some of this has been used to inform the development of knowledge acquisition tools for particular domains. There is as yet nothing in AI Planning that corresponds for example to the KADS components developed for diagnosis systems.

14. Recommended actions

14.1. Research direction actions

There is some perceived bias in the AI Planning Community towards rewarding the development of new planning algorithms rather than some of the areas raised in section above. The AIPS Planning Competition in its present form, for example, encourages the development of fast algorithms but not the production of interactive planners or of replanning capabilities. The most attractive areas for research, judging from the volume of papers submitted in the recent period, is Graphplan and DecisonTheoretic Planning, both very much development of algorithms. If we agree that other areas should also be developed, then ways of encouraging researchers to do so should be sought.

Some possible actions include the development of some standard problems which require the desired research developments (these need not be in manufacturing domains at all); the formulation of outline projects for national or European funding bodies which researchers can develop according to their particular interests and concerted attempts to publicise what has been done in these areas via journal special issues or conference workshops.

14.2. Community bridge-building activities

It is clear that linking the AI Planning Community to those working in intelligent manufacturing is vital if the PLANET technologies are to be applied to manufacturing problems. As argued above, there is very little awareness of these technologies outside their home community, and, it should be added, there is generally very little appreciation of manufacturing issues in the PLANET community. Pushing the technology directly to end-users is a possibility, but working more closely with people who are already familiar with manufacturing seems a quicker and more efficient way of educating both communities.

Links have already been made with the ICIMS EU network, and their members have been circulated with a view to joining PLANET. They have also agreed to fund members who wish to attend PLANET activities. PLANET therefore has every reason to approach members of the ICIMS network in order to try to involve them. This contact ought also to work in the opposite direction though, so that PLANET should encourage its members to participate in ICIMS activities. An obvious way of increasing contact and the exchange of knowledge would be to explore joint events - workshops, tutorials, etc.

The other community with which more contact would benefit both sides is that of Agents. initial contact has been made with the Product Design and Manufacturing subgroup of FIPA, whose discussions on applying their technology to manufacturing parallel very closely those of PIMS. As can be seen in [37], many of the manufacturing domains of interest to the Agent community are the same as those to which PLANET technology can be applied. Discussion on the formulation of some common benchmark applications would help to bring researchers together and prevent each community having to absolutely master the technology of the other.

14.3. Unifying applications and problems

It is not possible in this document to arrive at a set of unifying problems or applications in manufacturing since this really requires some of the community bridge-building activities of the previous section. It might well be posed as a possible outcome of a workshop or series of workshops . What is required is a set of problems and backing materials which are reasonably realistic, in the sense of not excluding key features of the domain, but are of a tractable scale for researchers to practise on. Some of the areas discussed above could be examined for such unifying applications.

15. Conclusions

This Roadmap has attempted to evaluate the obstacles which have impeded the penetration of PLANET technologies into manufacturing applications as well as to examine the ways in which those obstacles might be eroded. PIMS has always been an embryonic group, but this document suggests that there is much promise in the manufacturing area for AI Planning and Scheduling, and that with the right approach, a much more thriving community of interest can be built to the benefit of all.

Bibliography

[1] Proceedings, AI and Manufacturing Research Planning Workshop: State of the Art and State of the Practice. Editor George F. Luger, AAAI Press, 1998.

[2] C. Knoblock. AI Planning systems in the real world. *IEEE Expert*, pages 4-12, 1996.

[3] N. Muscettola and P. Pandurang Nayak and B. Pell and B. C. Williams. Remote agent: to boldly go where no AI system has gone before. *Artificial Intelligence*, 103(1): 5–48, 1998.

[4] D. Nau and S. K. Gupta and W. C. Regli. AI planning versus manufacturing-operation planning: A case study. *Proceedings of the IJCAI-95*, pages 1670–1676, 1995.

[5] R. Aylett and J. Soutter and G. Petley and P. Chung. AI Planning in a Chemical Plant Domain. *Proceedings of the European Conference on Artificial Intelligence 1998*, 1998.

[6] L. Castillo and J. Fdez-Olivares and A. Gonzalez. Automatic generation of control sequences for manufacturing systems based on nonlinear planning techniques. *Artificial Intelligence in Engineering* 4(1): 15–30, 2000.

[7] I. Klein and P. Jonsson and C. Backstrom. Efficient planning for a miniature assembly line. *Artificial Intelligence in Engineering*, 13(1): 69–81, 1998.

[8] Workshop on European scientific and industrial collaboration on promoting advances technologies in manufacturing - WESIC'98, eds. J. Amat et. al, University of Girona, 1998.

[9] 7th IEEE International Conference on Emerging Technologies and Factory Automation, eds J. M. Fuertes, IEEE Press, 1999.

[10] Y. Descotte and J. C. Latombe. GARI: A problem solver that plans how to machine mechanical parts. *Proceedings of the IJCAI 1981*, pages 766–72, 1981.

[11] H. Chang and W. Lu and X. Liu. Intelligent case retrieval and modification for machining process planning of axisymetric parts. In: [1], pages 55–62, AAAI Press, 1998.

[12] W. Shen, D. H. Norrie. Knowledge and Information Systems, 1(2) pages 129–156, 1999.

[13] D. E. Wilkins, M. des Jardins. A call for knowledge-based planning, 1999.

[14] S. Chien. Using AI Planning Techniques to Automatically Generate Image Processing Procedures: A Preliminary Report. *Proceedings of the AIPS94*, Chicago, IL, June 1994, pages 219-224, 1994.

[15] S. E. Cross and E. Walker. DART: Applying Knowledge-Based Planning and Scheduling to Crisis Action Planning. In: M. Zweben and M. S. Fox, editors, *Intelligent Scheduling*, Morgan Kaufmann, 1994.

[16] M. Aarup and M. M. Arentoft and Y. Parrod and I. Stokes and H. Vadon and J. Stager. Optimum-AIV: A Knowledge-Based Planning and Scheduling System for Spacecraft AIV. In: M. Zweben and M. S. Fox, editors, *Intelligent Scheduling*, Morgan Kaufmann, pages 451-469, 1994.

[17] L. Trave-Massuyes and R. Milne. Gas-Turbine Condition Monitoring Using Qualitative Model-Based Diagnosis. *IEEE Expert: Intelliegent Systems and their Applications*, May/June 1997, pages 22–31, IEEE Computer Society, 1997.

[18] N. Muscettola. HSTS: Integrating Planning and Scheduling. In: M. Zweben and M. S. Fox, editors, *Intelligent Scheduling*, Morgan Kaufmann, pages 169–212, 1994

[19] S. A. Chien and R. W. Hill and X. Wang and T. Estlin and K. V. Fayyad and H. B. Mortenson. Why Real-world Planning is Difficult: a Tale of Two Applications. In: M. Ghallab and A. Milani, editors, *New Directions in AI Planning*, IOS Press, Washington DC, pages 287–298, 1996.

[20] X. Li and S. Kambhampati and J. Shah. An Interactive Approach for Satisfying Process Plan Generation. In: [1], pages 97–103, 1998.

[21] M. Marefat and J. Britanik. Case-based Planning for Three-Dimesional Machined Components. Chapter 7 of J. M. Usher and J. Roy and H. Parsaei, editors, *Integrated Product and Process Development: Methods, Tools and Technologies*, pages 185–225, John Wiley & Sons, 1998.

[22] Austin Tate, Brian Drabble and Jeff Dalton. O-Plan: A Knowledge-Based Planner and its Application to Logistics. In Austin Tate, editor, *Advanced Planning Technology, The Technological Achievements of the ARPA/Rome Laboratory Planning Initiative*, AAAI Press, Menlo Park, California, May 1996.

[23] M. Sanchez-Marre and U. Cortes and J. Lafuente and I. R-Roda and M. Pcha. DAI-DEPUR: an integrated distributed architecture for wastewater treatment plants supervision. *AI in Engineering*, 10:275–285, 1996.

[24] Intelligent Manufacturing Systems Workshop, Leuven September 22-24, 1999.

[25] W. Nederbragt and R. Allen and S. Feng and S. Kaing and R. Sriram and Y. Zhang. The NIST Design/Process Planning Integration Project. In [1], pages 135–139, AAAI press, 1998.

[26] C. Hayes and D. Gaines. Constraint-Based Algorithm for Reasoning About the Shape Producing Capabilities of Cutting Tools. In *Machined Parts, ASME Design Technical Conference*, Sacramento, CA, Sept. 1997.

[27] C. Hayes. P3: A Process Planner for Manufacturability Analysis. *IEEE Transactions on Robotics and Automation, Special Issue on Assembly and Task Planning*, 12(2):220–234, April, 1996.

[28] H. van Brussel. Holonic Manufacturing Systems, the vision matching the problem. In *Proceedings of the First European Conference on Holonic Manufacturing Systems*, Hannover, 1994.

[29] Hendrik van Brussel and Jo Wyns and Paul Valckenaers and Luc Bongaerts and Patrick Peeters. Reference Architecture for Holonic Manufacturing Systems: PROSA. In *Computers in Industry, Special Issue on Intelligent Manufacturing Systems*, 37(3):255–276, 1998.

[30] Luc Bongaerts and Paul Valckenaers and Hendrik Van Brussel and Patrick Peeters. Schedule Execution in Holonic Manufacturing Systems. In *Proceedings of the 29th CIRP International Seminar on Manufacturing Systems*, May 11-13, 1997, Osaka University, Japan, pages 209–215, 1997.

[31] P. I. Tanaya and J. Detand and J. P. Kruth. Holonic Machine Controller – a study and implementation of holonic behaviour to current NC controller. In *Co-operation in Manufacturing : CIM at work*, Conference Preprints, pages 375–396, Kaatsheuvel, The Netherlands, Aug 28-30, 1995. (Accepted for publication in journal *Computers in Industry*).

[32] J. Kempenaers and J. Pinte and J. Detand and J. P. Kruth. A collaborative process planning and scheduling system.In *Proceedings of the 1994 ASME - International Computers in Engineering Conference*, Volume 1, pages 221–226, Minneapolis, 1994.

[33] George Ferguson, James Allen, and Brad Miller. TRAINS-95: Towards a Mixed-Initiative Planning Assistant. In *Proceedings of the Third Conference on Artificial Intelligence Planning Systems (AIPS-96)*, Edinburgh, Scotland, 29-31 May, pages 70–77, 1996.

[34] D. E. Wilkins. Can AI planners solve practical problems? *Computational Intelligence*, 6(4):232–246, 1990.

[35] Austin Tate, Jeff Dalton and John Levine. Generation of Multiple Qualitatively Different Plan Options. In *Proceedings of AIPS-98*, Pittsburgh, June, 1998.

[36] R. Milne and C. Nicol and M. Ghallab and L. Trave Massuyes and K. Bousson and C. Dousson and J. Quevedo and J. Aguilar Martin and A. Guasch. TIGER: real-time situation assessment of dynamic systems. Rapport LAAS No94445, *Intelligent Systems Engineering*, 3(3):103–124, Autumn, 1994.

[37] Weiming Shen and Douglas H. Norrie. Agent-Based Systems for Intelligent Manufacturing: A State-of-the-Art Survey. `http://sern.ucalgary.ca/CAG/publications/survey-abm.htm`, 1999.

[38] IMACS: A System for Computer-Aided Manufacturability Analysis. University of Maryland, http://www.cs.umd.edu/projects/imacs/imacs.html

[39] J. Meyer and M. Ball and J. Baras and A. Chowdhury and E. Lin and D. Nau and R. Rajamani and V. Trichur. Process Planning in Microwave Module Production. In [1], pages 120–126, 1998.

Part IV.

Workflow Management

Paul Kearney

(British Telecom Laboratories, UK)

with contributions from.

16. Introduction

This is the first version of an R&D Road Map for AI Planning and Scheduling (AI P&S) applied to business process management (BPM) produced by the Workflow Management Technical Coordination Unit (TCU) of PLANET. PLANET is the European Network of Excellence on AI P&S. The purpose of a Road Map is to coordinate R&D by establishing end-user requirements on short medium and long time scales and proposing research and technology transfer goals and activities that will enable the requirements to be satisfied. The current version is only a first step towards such a Road Map, which in any case should be a living document updated regularly.

BPM and AI P&S are two disciplines with many parallels, but which have largely been pursued by disjoint communities. A necessary precursor to producing a Road Map is to align the two disciplines so that specialists in each can understand each other. One of the main achievements to date has been to develop an understanding of how the 'world view', vocabulary, challenges, etc. of Business Process / Workflow Management relate to AI Planning and Scheduling. This has been possible because of the active participation of a number of workflow and process management experts from end-user organisations and consultancy companies.

Workflow and its role in business process management

A business process is the chain of activities involved in delivering a product or service to a customer (within or outside the organisation). Designing business processes is a knowledge-intensive human activity supported by software modelling and simulation tools, and is closely tied in with matters such as business policy and enterprise organisation and culture. An instance of a business process created, for example, to deliver a particular service to a particular customer is analogous to a plan in AI. In BPM terminology, however, a plan also includes allocation of resources (e.g. workers) and target start and completion times. In some application domains, for example military logistics, instantiating a process may be complicated, and AI planning techniques are being applied successfully in such areas. In the typical business application, however, it basically involves selecting from a set of templates. The main technical challenges arise because an organisation is a distributed system that executes many process instances concurrently in an uncertain environment. Furthermore, failures and other exceptions occur frequently, and re-planning must be integrated with execution.

A workflow management system (WfMS) automates the coordination of activities and transfer of documents within a business process. It delivers the work to the 'in-tray' of the appropriate software component or human worker or team according to pre-defined rules (a process or workflow definition). Note that current WfMS do not (generally) perform planning, scheduling or resource allocation. Any such considerations must be built in to the process definition or else handled by the productive resources owning the in-trays. Specifying this low level process or workflow definition is again primarily a human design activity performed with the assistance of software tools (often specific to the WfMS).

Requirements

Requirements have been classified as short, medium and long term as follows:

Short term: address short-comings in current-generation process management software. The most important items in this category are: integration of temporal reasoning and resource allocation/management algorithms into workflow management software; and incorporation of a planning capability to enable a WfMS to modify the process instance automatically during execution, to cope with failure, changed objectives, and other exceptions.

medium term: Current generation workflow software handles high volume routine processes, typically involving low-skill workers. The medium term requirements concern extending this support to high-skill knowledge workers. This may involve, for example, building process awareness into software tools.

long term: More radical (e.g. adaptive self-organising) approaches addressing the need for organisations to function in a business environment that is increasingly uncertain and subject to change.

The Road Map report

The report expands on the comparison of the two disciplines and the requirements. It then looks at ways in which the requirements could be met by existing and future results from AI P&S and related disciplines. The discussion is divided into a number of themes. The first theme deals with human issues. A feature of business processes is that much of actual work is performed by people. There is a tendency in BPM to pursue automation and to treat human actors in the process as if they were machines. This is often counter-productive resulting in de-motivation and a failure to utilise human qualities. The following chapter looks at software infrastructure. It is important to appreciate that for AI P&S techniques to be applied in practice they need to be integrated with / interfaced to commercial software packages. This chapter looks at issues such as reference architectures and interface standards. Chapters covering life-cycle oriented technical themes then follow: business / process modelling and knowledge engineering; planning, scheduling and resourcing; enactment/execution and monitoring; adaptation, optimisation and metrics. Finally, there is a chapter summarising conclusions and recommendations for future actions.

Conclusions and recommendations

AI planning and scheduling and business process management are complementary disciplines with much to gain from collaboration. The ability to invoke AI components flexibly and dynamically from within a workflow framework would considerably enhance business productivity and give the European software industry a competitive advantage. The Workflow Management TCU has a valuable continuing role to play in bringing together researchers, software developers and end-users from the two communities and promoting joint work between them. The active participation of workflow and process management experts from end-user organisations and consultancy companies in discussion with planning researchers has enabled considerable progress to be made on the R&D Road Map. The TCU must make every effort to involve more end-user representatives from a spectrum of industries. A number of commercial software vendors are registered on the TCU mailing list but have not as yet participated actively. It is important bring such organisations fully into the fold. The concluding section of the report makes some proposals for future activities.

17. Introduction

This is the first version of an R&D Road Map for AI Planning and Scheduling (AI P&S) applied to business process management (BPM) produced by the Workflow Management Technical Coordination Unit (TCU) of PLANET. PLANET is the European Network of Excellence on AI P&S. The purpose of a Road Map is to coordinate R&D by establishing end-user requirements on short medium and long time scales and proposing research and technology transfer goals and activities that will enable the requirements to be satisfied. The current version is only a first step towards such a Road Map, which in any case should be a living document updated regularly.

BPM and AI P&S are two disciplines with many parallels, but which have largely been pursued by disjoint communities. As has been said of the UK and USA, the two communities are separated by a common language, with terms such as *planning* being used in both with different meanings. A necessary precursor to producing a Road Map is to align the two disciplines so that specialists in each can understand each other. One of the major successes of the TCU to date has been to bring AI P&S researchers into contact with BPM specialists and so further mutual understanding. This progress is documented in the next chapter, which gives and overview of BPM and how it aligns with AI P&S. This is followed by an outline of requirements on short, medium and long timescales as perceived by BPM end-users.

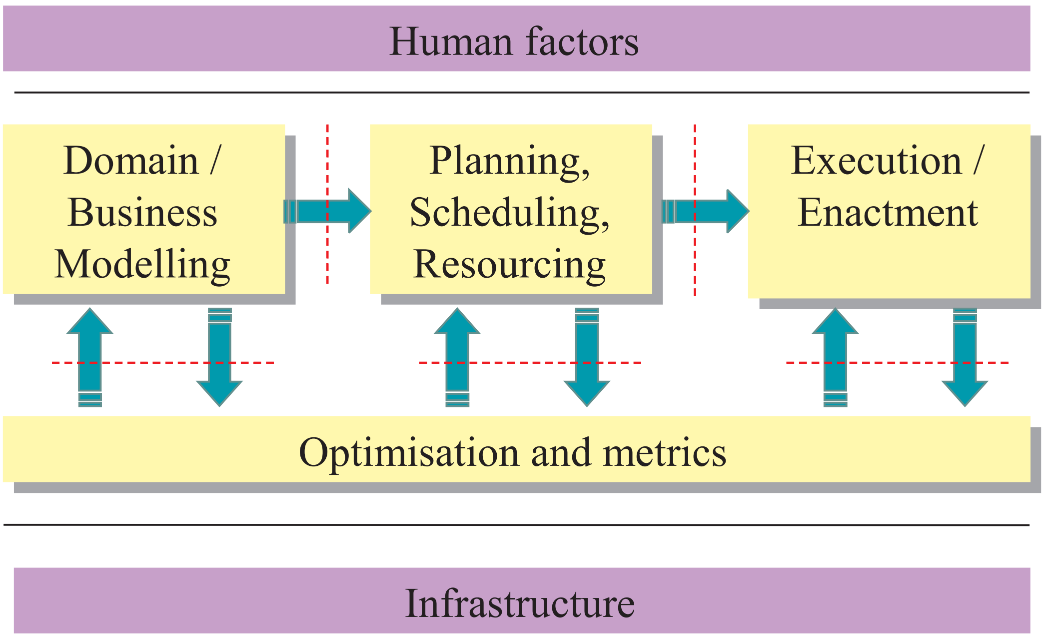

Figure 17.1.: Themes of discussion.

The remainder of the document expands on these requirements and looks at ways in which they could be met by existing and future results from AI P&S and related disciplines. The discussion is divided into a number of themes (see Figure 17.1). Each themed chapter includes sections on the state of the art (including trends and current projects), research goals and open issues, and recommended actions. The first theme deals with human issues. A feature of business processes is that much of actual work is performed by people. There is a tendency in BPM to pursue automation and to treat human actors in the process as if they were machines. This is often counter-productive resulting in de-motivation and a failure to utilise human qualities. This chapter explores these issues and examines how they might be addressed. The following chapter looks at software infrastructure. It is important to appreciate that for AI P&S techniques to be applied in practice they need to be integrated with / interfaced to commercial software packages. This chapter looks at issues such as reference architectures and interface standards. A common understanding of architecture would also facilitate collaborative research and demonstrations. Chapters covering life-cycle oriented technical themes then follow:

- business / process modelling and knowledge engineering;
- planning, scheduling and resourcing;
- enactment/execution and monitoring;
- adaptation, optimisation and metrics.

Finally, there is a chapter summarising conclusions and recommendations for future actions.

18. Overview chapter

This chapter gives an overview of the ‘standard model’ of process management and attempts to position AI Planning and Scheduling within this context.

18.1. Process management and workflow

A core business process is the end-end chain of activities involved in delivering a product or service to a customer. Note that customers may be internal to the organisations. ‘End-end’ signifies that a business process starts with initial contact with the customer and runs through to completion of the contract (including billing and payment). In fact, since the customer’s satisfaction with one service influences requirements for future services, a business process is best seen as a closed loop. In addition to core business processes, there are management processes (including processes concerned with designing the core processes) and support processes that facilitate the other types of process.

The set of business processes for an organisation comprises the organisation’s working practices. Organisations differ in how explicitly the processes are defined, and in the form they are represented. In some cases the processes are implicit, in others they are recorded in textual codes of practice, in others they are documented in (semi-) formal representations and/or software modelling tools. A set of business processes is highly analogous to a set of stored plan templates or a hierarchical task network (HTN) and could readily be represented in this way (this will be explained further later).

Business process management can be presented as having the following aspects:

Process modelling: This involves designing, modelling, evaluating, modifying, optimising, etc. the organisation’s processes[1]. For each basic product or service the organisation offers to its customers, the activities involved, the relationships between them, resource requirements, etc. must be defined. It is basically a human activity, though supported by computer-based tools to record and display the process model, run simulations, etc. Design decisions are made based on experience and analogy to previous designs. Choices are tied closely to other aspects of the enterprise and business environment such as: the nature of the business, organisational structure (of the enterprise), enterprise culture, legacy infrastructure, etc. Although process design is often presented as happening top-down, the practical constraints imposed by the current state of the enterprise mean that there is a strong bottom-up element. Note that design of the process and activities typically go hand-in-hand, so that although the analogy between process and AI plan is strong, the analogy between the activities of process design and classical AI planning is much weaker.

[1] Jacobson (1995) splits BPR into “model existing system”, “envisioning”, “design new system” and “implement new system”.

Process planning (elaboration, resourcing and scheduling): A process definition is basically a template. This phase involves identifying the appropriate template to use, elaborating and filling in an instance of that template in sufficient detail for it to be executed. The first step is normally to gather information from the customer on the service required. This allows the tree of possible processes to be pruned considerably, but a number of alternative branches may still remain. The next step is to produce a schedule based on a target end date required by the customer, dependencies between tasks, and knowledge about how long tasks take (e.g. typical and minimum times). If it is not possible to achieve the target end date, negotiation with the customer takes place. Then, the people and other resources required for each task are identified and 'reserved' for the appropriate time slots ('resourcing' or 'provisioning'). The resourcing and scheduling problems are coupled by virtue of finite capacity and/or non-sharable resources. If the required resources are not available, then the earlier steps must be revisited. In process management the result is referred to as the plan, and the process of producing it as planning, which is a source of confusion as the usage is different fom that in AI[2]. Note that further detail will often be decided at execution time, and the balance between design time and execution time decisions varies considerably. Again, these activities are often performed by people assisted by relatively dumb software tools. The nature of the tools and the form of the output depend on context. For example, MS Project (or the equivalent from another supplier) could be used to create a 'production plan' to be carried out by a human organisation. Alternatively a proprietary tool could be used to generate a process description for enactment by a workflow engine. Note that in each case, the tool and the representation is often different from those used earlier in the modelling phase.

Enactment / execution: The production plan is carried out, with detail being elaborated during execution. Note that the boundary between planning and enactment is context dependent. Process planning is essentially the first part of enactment of a core process. Furthermore, at the start of enactment proper, the plan may still contain alternative branches that are pruned as information is gathered and decisions made during enactment. Almost always, execution is distributed, with different production resources, computer programs, or people carrying out the constituent activities. The activities have to be coordinated to ensure correct sequencing and also to ensure that compatible variants of the activities are performed. Coordination takes place via mechanisms such as: events, transfer of documents, existence checks on documents, etc. A **workflow management system** uses information contained in a low-level process plan definition to route work items to the appropriate production resource and provide the necessary coordination signals. Note that workflow systems (generally) do not plan work, and workflow also assumes resources will be available. Production resources will be involved in enacting multiple processes and instances of the same process in a time-sharing manner. A production resource (or rather a component encapsulating one or more resources) sees the processes in which it participates as a queue of work items (or tasks) waiting to be acted upon. Depending on how the system is organised it may simply work on the next task whose pre-conditions are satisfied, or it may have rules for prioritising tasks. Either way, different processes can interfere with each other due to the finite capacity of a shared production resource.

Monitoring: As execution proceeds, information on progress (e.g. notification of completion of tasks, delays and other problems) is fed up to a management function. This compares actual progress with the production plan. Minor differences between the plan and actual progress may simply require

[2] I will try to use the terms production plan and AI plan to avoid ambiguity.

updating of the plan (for example with slightly different commencement times for tasks). These changes need to be propagated to the resources executing the plan. More significant differences may require the planned activities to be altered during execution. This may include some back-tracking, for example to remove some item of equipment that was installed following the earlier plan, but is now no longer required. More drastic problems may require all the effects of the plan to be undone and a new plan created. The monitoring function may try to anticipate future problems and modify the plan in advance to avoid the problems. This is sometimes known as jeopardy management.

18.2. AI planning and scheduling

AI planning and scheduling (AIP&S) is concerned with determining a sequence of actions that when executed by one or more agents with the world in some initial state satisfying given conditions, results in world state satisfying given goal conditions. A process is a description of activity. A plan is a description of activity for a given objective - it is an instantiated process. AI planning provides two main techniques. In STRIPS style planning the operators consist of individual activities. A planner combines these for given objective to form a plan. However, HTN planning domain descriptions are essentially process descriptions. They let you specify parameterised descriptions of processes that are can be automatically assembled and instantiated to form a plan for a given objective.

Classically, this overall problem is divided into a number of stages:

Modelling (or knowledge engineering) : this concerns finding the right way to represent the world and the problem so that planning and scheduling may be performed. Classically, this representation consists of a definition of some space of states that the world and its constituents may be in, and a set of primitive operators that can be applied to cause (constituents of) the world to change states.

Planning : this concerns finding one or more sequences of actions[3] that should cause the world to change from the initial state to a state satisfying the goal conditions. The ordering of these sequences is not necessarily completely determined. Planning is concerned with logical dependency of actions in the sequence, e.g. that if action A is necessary to bring about the preconditions for action B, then A is performed before B. Planning may be performed bottom up by chaining together actions until the gap between initial and final states is spanned (e.g. STRIPS). Alternatively in may be performed top-down by recursively refining generic plans until they are expressed entirely in terms of executable actions (e.g. HTN).

Scheduling : A plan expresses the orderings of actions that should be able to bring about the goal. Scheduling determines which of the orderings of actions consistent with the plan will actually be used. Often, this choice is based on some form of efficiency measure, for example overall time taken to execute.

Execution : Execution as such has not been a major concern of AIP&S. However, particular branches of AIP&S *are* concerned with execution-time issues. For example, it is recognised that an action does not always achieve its intended result. Thus monitoring must take place to compare anticipated events with actual ones, and if deviation is significant plan repair is initiated. If deviation is excessive or repair impossible, then the plan is abandoned and a new plan

[3]Let us say that an action is an operator applied to specific objects. in the world.

generated. At the extreme end of the scale are so-called re-active planners in which planning and scheduling take place at execution time, with planning, scheduling and monitoring actions interleaved with the goal-achieving actions.

18.3. A comparison

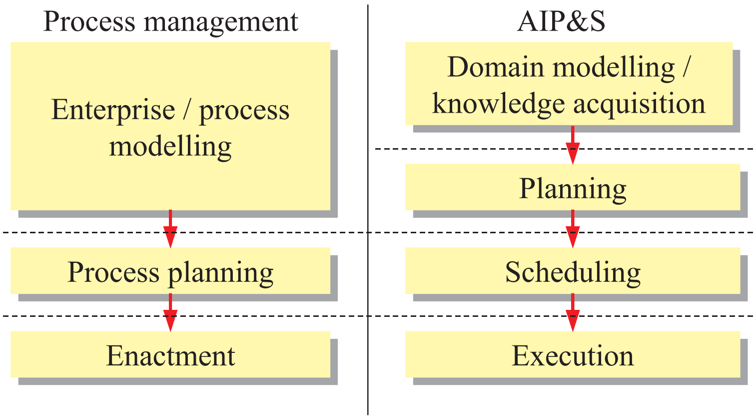

Figure 18.1.: A comparison of process management and AIP&S.

It is clear from the above descriptions that process management and AIP&S address similar issues, and there are many parallels between the two disciplines. Figure 18.1 compares the two at a coarse level, aligning phases that are roughly equivalent. Note, however, there is no direct equivalent to AI planning on the process management side – although a process (model) is approximately equivalent to an AI plan, it is generated by people supported by software drawing and modelling tools rather than by an analogue of AI planning.

Note also that both business processes and AI plans can occur on multiple levels. Thus process management can itself be seen as an enactment of a meta process; enactment of strategic processes may involve definition of tactical processes and so on. Similarly execution of strategic AI plans may involve planning at a tactical level. In consequence, it is important to consider applications of AIP&S within the activities taking place during enactment a business process. For example, an early step in a process may involve detailed planning or scheduling of activities occurring later in the process.

Despite the similarities, there are also significant differences:

- terminology – the word ‘plan’ itself has a different meaning in the two disciplines;
- most of the design-time (as opposed to execution-time) activities in process management are performed by people assisted by relatively dumb software tools. In contrast the emphasis in AIP&S is on producing intelligent software that can perform planning and scheduling largely automatically, with occasional assistance from a person;
- AIP&S representations tend to be mathematically formal and semantically precise, though this often means they are difficult (for a non-expert) to understand. In process management, the opposite is true: the representations are domain-oriented and easy to understand, though the semantics are often somewhat vague.

- Languages for defining processes as input to workflow engines are basically scripting languages for coordination of activities and are at a lower level than AI plan languages.
- classical AI planning techniques focus on difficult combinatorial problems – many different combinations of operators and states are possible, only a few of which constitute viable plans. In process management, activities are fairly specific to processes, and there is much less scope for combining them in different ways to form different processes.

These differences present opportunities for synergy as well as barriers to be overcome.

19. Requirements

19.1. Current state of the art in workflow and process management

The idea of process management is still fairly new. In the past, an enterprise's processes were implicit in its organisational structure and culture. Departmental procedures and practices would be known within the department, but no individual had a clear end-end view of a process. A similar statement could be made about the software systems that support the enterprise's operations. These were, and still are, often large monolithic applications in which the business processes are implicit. Consequently they are difficult to change and tend to tie the organisation in to the processes encoded in the software. However, the importance to the day to day operation of the enterprise and the expense and disruption involved in replacing them mean that many of these so-called legacy systems are still in active use.

The current trend in both enterprises and their operational support software is to represent the business processes in an explicit and distinct manner. As a result it is easier to study how to improve a process and also easier to implement the improvement. In the case of the software, the need for modifiability and software re-use has led to a component-based philosophy. Instead of monolithic applications, functionality is encapsulated in re-usable modules that can be combined in different ways to construct new 'virtual' applications rapidly. One way to view workflow management systems is as the architectural glue that links the components together to form the application. At least in theory, the process definition can be changed independently of the components, and functionally equivalent components substituted without changing the process definition. Note that often these components do not *replace* the legacy applications. Rather the components use them as servers in providing their functionality.

Of course, much of the work in a business process is performed by people. A workflow management system treats people in much the same way as the computation components. Typically an interface is provided that presents the user with an in-tray and out-tray of work items. This interface encapsulates the user in a similar manner to that in which the component interface encapsulates the software functionality. This approach is suitable for partial automation of well-understood routine processes.

Sometimes groupware software systems (such as Lotus Notes and Microsoft Exchange) are described as workflow systems. These primarily provide a messaging and information sharing environment that can be used by participants in business processes. However facilities such as document routing scripts and forms can be used to define workflows to some degree.

Industry is currently in transition from the old-style monolithic support applications and paper based office processes to workflow-based systems (for a picture of the current state of progress in BT see [14]). Legacy applications certainly will not disappear overnight. Rather, components and workflow systems will gradually diminish their role. The legacy software may never disappear entirely, however, especially where the products / services and associated processes are relatively mature.

AI scheduling (and to a much lesser extent planning) techniques have certainly been used in special purpose business support applications. See for example the use of on-line scheduling techniques in BT's Work Manager application (`http://innovate.bt.com/showcase/work_scheduling/index.htm`). However there has been little or no influence by AIP&S on process management as a discipline or on the methods and tools through which it is applied. Similarities (for example between plan and process description languages) are due more to convergent evolution than to direct influence.

19.2. Enumeration of requirements

There now follow descriptions of areas in which current business process management is recognised to be deficient. The list is not exhaustive, and we invite proposals for additions to the list. The requirements are into short, medium and long timescale categories. The short term requirements concern ways in which current practice and tools can be improved. The medium term requirements concern extension of workflow-related support into classes of process and user that are not catered for by current workflow systems. The long term requirements concern the need for a more radical re-think of how enterprises and their software support infrastructure are organised.

19.2.1. Short term:

The following are seen as short-comings in current-generation process management software. They are presented in approximate order of importance, though the first two are of comparable ranking.

Integration of scheduling and resource allocation/management algorithms into workflow management software. Current workflow management software automates the flow of work items between work queues according to pre-determined rules. It does not deal with allocation of resources to tasks or take resource availability into account in prioritising or scheduling the work. (high importance)

Re-planning. There is a requirement for incorporating an ability to modify the process instance automatically during execution, to cope with failure, changed objectives, and other exceptions. This could be done by altering the process instance plan being executed (inserting and deleting steps) or by creating and executing an ancillary plan containing the additional process steps. (high importance)

Generation of workflow definitions from high-level process models. Process modelling tools work with relatively high level process definitions, whereas workflow management systems require low level definitions. Current generation tools do not do a good job of bridging this gap. Tools are required that automatically generate low-level definitions that can be input directly to workflow management systems. The ability to do this in reverse is also desirable. (medium importance)

The ability to feed data captured in the workflow engine back into the modelling and simulation tool to improve modelling at that level. Workflow engines capture a great deal of data in the course of enacting process instances. This contains useful information latent within it, but it is rare that data mining techniques are used on it. (low importance)

19.2.2. Medium term:

Process support for intermediate level and knowledge workers: As argued in [15][1], current workflow and groupware systems 'pick the low-hanging fruit', that is they automate that which is easy to automate – enactment of routine processes and providing information-sharing and communication services. The tasks performed within these processes are routine also, and are performed by relatively low skill workers. There is very little process-related support available for high skill (professional) knowledge workers or workers at intermediate skill levels[2]. 'Process aware' and 'knowledge aware' support to enhance the effectiveness of intermediate and high-value knowledge workers is required, but much more difficult to achieve.

Empowerment of users: (see also [16][3]) current workflow management systems are suitable for routine processes and demand uniformity from users, effectively expecting them to behave like machines. This makes poor use of human abilities, even in the case of low-skill workers, and can have a de-motivating effect. Informing people about the context of their work is a necessary short-term requirement, which should eventually lead to systems that encourage and support initiative, but would require workflow systems to be enriched with a semantic knowledge the processes they enact. Future systems need to assist people in achieving their potential in their roles, which means encouraging initiative and adapting to human diversity rather than enforcing regimentation.

Visual representations of the current status of a process instance, so that workers within a process can see how their activities fit into the "big picture" (Zuboff). This is actually intermediate between the 'short term' requirements, which might be satisfied by incremental additions to current generation workflow, and the 'medium term' ones which require a change in philosophy.

19.2.3. Long term:

The following are factors driving process management development in the long term. Mostly they concern the need for organisations to function in a business environment that is increasingly uncertain and subject to change.

Flexibility: One of the main drivers in process management is the need to be able to get new products and services to market quickly. This means that an enterprise and its supporting infrastructure must be capable of enacting a wide variety of processes, with the actual set of processes active at a given time being easily changed.

[1] `http://www.labs.bt.com/projects/planet/Discussion/Requirements-Vlong-v1.doc`
[2] in `http://www.labs.bt.com/projects/planet/Discussion/Requirements-Vlong-v1.doc` these three categories of workers are described as back office workers, sales workers and knowledge workers.
[3] `http://www.labs.bt.com/projects/planet/Discussion/Users-NikM.txt`

Evolvability: No matter how flexible an enterprise is in the short term it will have to change in the longer term in response to changing markets, technology, etc. Change takes time, however, and the enterprise must continue to operate. The enterprise needs to be capable of gradual evolutionary change to avoid the current problems with legacy systems recurring in the future.

Adaptiveness: Currently, enterprises and their business processes are seen as basically static, but subject to occasional discrete changes such as re-organisation or introduction of a new product and/or process. However, the frequency of change is increasing. In the future organisational models will be needed in which continuous change is the normal state of affairs. Such models most incorporate processes that 'sense' the drivers for change (e.g. increasing demand for a product) and cause appropriate changes to the enterprise model. Enterprise software infrastructure will need to support such dynamic organisational models.

Decentralised management: A management paradigm shift is currently under way motivated by the need for flexibility, evolvability and adaptiveness. This is variously described as a move from **centralised** to **decentralised** management, from management **push** to **market pull**, and from **plan and build** to **sense and respond**. This involves devolution of decision-making responsibility from central management to autonomous local units. The behaviour of the enterprise as a whole is then the cumulative result of local decisions. The role of higher management is then one of defining performance metrics and other incentives by which local managers make decisions, and also of providing means by which the autonomous units can interact constructively. Workflow management systems are very much tied into a plan and build management style. A new approach to software infrastructure is required to support decentralised management. An agent-based approach seems well suited in this respect. Use of an agent-based approach does not in itself guarantee the benefits sought from decentralisation, however. A better understanding of how to apply agents and agent-based approaches to achieve the benefits is still required.

Dynamically changing organisations of the future will involve forming opportunistic organisational structures and dynamic supply chains. Theoretical work in Virtual Organising and "switching" is related to planning approaches (Mowshovitz, Venkatraman and Henderson) and can be used to build the workflow systems of the future.

19.3. Recommended actions

- The potential for benefit from applying existing AI P&S techniques to short term requirements should be explored through case studies of large-scale real problems conducted jointly with domain experts and process management experts. It is apparent that process management problems can be posed as planning and scheduling problems. However, for the techniques to be adopted, it must be shown that they result in a significant benefit compared to current practices in the context of realistic business processes. The techniques must also integrate with other components of enterprise software, and be usable by the typical software or business process engineer. PLANET cannot perform such case studies, but it should facilitate and encourage them. It should also publicise the results.

- Generally, links with process management and software engineering communities need to be strengthened by means of interdisciplinary events and other measures.
- To encourage case studies and increase awareness of process management challenges within the planning community, PLANET should collect examples of process management problem domains. The BT OSS Job Management domain[4] is a good example, but others are needed from different industries to supplement it.

[4] http://www.labs.bt.com/projects/planet/RoadMap/OSSs.pdf

20. Road Map themes

The rest of the document looks more closely at state of the art (including trends and current projects), research goals and open issues, and recommended actions within the themes shown in Figure 17.1. The dashed lines in this figure indicate interfaces / models or documents shared by the boxes. The boxes in the middle correspond to the main divisions shown in Figure 18.1. Some further explanation is needed where the two halves of Figure 18.1 do not align, however. Basically, activities in which people design or elicit models (possibly supported by software tools) are included in the modelling theme. Planning (as understood in AIP&S) is included with scheduling and resourcing. Note that planning, scheduling and resourcing can be performed during as part of the enactment of processes as well as off-line.

The optimisation and metrics theme recognises the need to measure attributes of the various models and provide feedback to improve desirable qualities. Note that the feedback can be to the same box, e.g. measurements of the attributes of a process model can be used to optimise the process model. Larger-scale optimisation is also important, however. For example, measurements during execution of a process can be used both to adjust the theoretical production plan to better reflect reality, and to provide information to help improve it. These improvements can be fed back into the executing process, and so on.

The remaining two boxes represent 'orthogonal issues': infrastructure and human factors. An important infrastructure issue is the establishment of a reference architecture for a highly-modular 'AI-enabled' process management system, covering both design-time (off-line) and execution-time systems. The human factors box reflects the need to take into account (throughout process management) of the special characteristics of the people involved in performing the processes, and to form a symbiotic relationship between 'man and machine'.

These two 'orthogonal themes' are covered first.

21. Human issues

Section proposed by Nikolay Mehandjiev, University of Manchester

21.1. Introduction

Human issues can be divided into two main categories:

- involvement of people in the process being managed
- support for people performing process management tasks (including process design)

To avoid overlap with the issues covered by the the Domain Modelling section of this document and by the Knowledge Engineering TCU of PLANET, this section focuses on the first set of issues.

21.2. Current state of the art

Most business processes require people to perform at least some of the tasks. Workflow management systems tend to view people and software resources in the same way: as means of carrying out process steps. At the present state of the art, they can be applied only to well defined, routine processes, introducing even more regimentation into dull, boring jobs. All too often, software is seen as a means of decreasing costs through automation and standardisation rather than as a means of enhancing value and quality of he product/service and hence customer satisfaction. It is also used to monitor productivity, thereby increasing pressure on workers further. Unfortunately the quantities measured tend to be those that are *easy* to measure (number of calls handled in a call centre, etc) rather than true measures of value contributed.

- AI planning too has traditionally been concerned with automation of processes. The goal has been to build intelligent machines, i.e. to enable machines to perform activities that currently only people/animals can do. Little attention has been paid to amplifying human abilities, though there is a body of work in the are of mixed-initiative planners.

Computer-supported cooperative work (Beaudouin-Lafon, 1998) was a very active field in the early nineties, though activity seems to have died down recently, re-focusing on work within virtual teams. A stream of work within CSCW focuses on Tailorable Workflow systems (Kahler *et al* 2000), where workers can modify run-time functionality of workflow via preferences. Industrial psychologists work in the area of job design, , which focuses on maximising the motivational characteristics that people experience in their jobs (Oldham 1996), and have formulated factors that improve job satisfaction, for example autonomy, variety and responsibility. Technology such as workflow has the potential to both simplify and enrich the nature of work. Control over various aspects of work such as timing and method is thus considered crucial in workflow-type environments, with higher control leading to better productivity and work attitudes (Jackson et al 1993).

21.3. Research issues

Aim: Understanding of how to achieve a synergistic, symbiotic relationship between human workers and managers and software systems within business processes.

- How to involve users in controlling their coordination support and workflow planning systems? A decision to empower users so that they can control workflow systems requires appropriate interfaces and methods to make this user control possible. These interfaces and methods have to be carefully designed to take into account the expected variety of user backgrounds and programming skills. The academic fields of End User Development (Nardi 1993) and Visual Programming (Burnett 1995) have researched these issues in the general case of software programming and control. An interesting research direction would be to see how the general findings in these areas map onto the specific domain of workflow planning.
- How to take into account human issues when at the process modelling and definition stage of workflow planning systems? Two approaches that are used in mainstream software development are Participatory Design (Kuhn and Muller 1993), which aims to involve user representatives in the design of new software, and input from industrial psychology such as job satisfaction factors.
- What is the optimal balance between users and software during the different stages of workflow planning and scheduling?
- If planning techniques are to be employed successfully new means of visualisation and explanation need to be developed to reflect the combinatorial nature of AI planning.

21.4. Recommended actions

- To run a set of trans-disciplinary workshops, which discuss the relationship between systems providing user-control of workflow and contributions from the areas of Participator Design, Industrial Psychology, Visual Programming and End User Development. To summarise the workshop findings in a report.
- To develop a prototype demonstrator to test the feasibility of user control at different stages of the planning-driven workflow development and enactment.

22. Infrastructure

Section proposed by Nicola Matino, Centro Ricerche Fiat

22.1. Introduction

This section concerns the need to establish computational infrastructure to facilitate research, development and end-use planning and scheduling technology for process management applications. This infrastructure can be divided into:

- an open, layered architecture and interface definitions that allow independently-developed modules to be combined in a flexible way,
- the software modules that work within this architecture.

PLANET's role should not be to favour one technology over another, or to enter the debate over free software, open-source software and proprietary software. Rather, it should aim to ensure that research, transfer and exploitation can be conducted effectively. Wherever possible, compliance with existing official and de facto standards and interoperability with existing solutions should be encouraged.

22.2. Current state of the art

- the WfMC[1] has established a reference architecture based on 5 interfaces between workflow engines and other classes of associated software, and is working on detailed definitions of these interfaces. Conformance to the WfMC standards by the major software developers is mixed, however.
- a 3 / 4 layer enterprise architecture has now become standard practice in industry, with the layers consisting of user client software, back-end applications, and one or two layers of so-called middleware. The BT 4-layer architecture is described in Jim Hutton's paper on the PLANET Workflow web pages[2] While this is specific to BT it is probably a typical state-of-the-art solution.
- there are various competing standards for middleware components including: CORBA (OMG), COM, etc. (Microsoft), java-based solutions (Sun Microsystems and others)

[1] see `http://www.wfmc.org/`
[2] see `http://www.labs.bt.com/projects/planet/Discussion/JimHutton-bttjPrePrint.doc`

- there are a number of emerging standards for interoperation of software agent from FIPA, DARPA (KQML/KIF), OMG, etc.

22.3. Requirements

A reference architecture and interface (language / api) covering modelling, build-time and execution is required to enable highly modular approach to research and application systems. A modular approach is necessary to facilitate:

- re-use of software and avoid wasteful duplication of effort.
- synergy between the work of research groups developing complementary technologies
- exploitation of research results as add-on modules to 'standard' software.
- a steady flow of incremental enhancements form research into application.

The architecture must be compatible with the WfMC ref architecture and API and other standards

22.4. Current research trends and active projects

It may be argued that software technology has developed by reducing the amount that a computational entity needs to know at runtime in order to be able to interoperate with other entities. Components have a more tightly defined interface than objects and provide interoperation primitives in the form of events that pass highly informative objects to receiving entities. Software agents that use communication languages based on speech acts are sometimes presented as being the logical next extension of this trend. A number of collaborative and individual research projects have been and continue to be conducted in the application of agents to workflow management. The collaborative projects include: ADEPT, ENTERPRISE, TBPM (UK collaborative projects), EURESCOM Project P815[3].

22.5. Open issues

It is important for PLANET to remain agnostic over middleware technology (CORBA vs. COM vs Java Beans, Jini, ...) until matters resolve themselves in the real world. However, to facilitate research collaboration, it is necessary to pick one of these technologies on which to base a common research platform architecture.

22.6. Research goals

Development of planning and scheduling servers that can be accessed by software components in the same way as back-end application software.

[3] `http://www.eurescom.de/Public/Projects/P800-series/P815/p815.htm`

22.7. Recommended actions

- draw up reference architecture
- agree interface standards for research collaborations within network
- working group to establish interface standards for research cooperation

23. Domain / business modelling

Section proposed by Daniel Borrajo, University Carlos III of Madrid

23.1. Introduction

This section is concerned with methods, tools, languages, etc. used to model businesses and other application domains. In business process management, the purpose is to design, improve or define more precisely the enterprise and its processes. In AI P&S, the purpose is to find a way of modelling the problem domain that enables planning and scheduling techniques to be performed effectively. Knowledge engineering (KE) for planning is being addressed by the Knowledge Engineering TCU, and the reader is referred to KE roadmap for general information on KE for Planning a discussion of issues of specific relevance to business process management is given below. These purposes are entirely compatible, and there are many similarities between the representation languages used. In business process management the object is normally to define a set of processes, whereas in AI planning it is more a space of plans/processes that is defined (in terms of operators and/or task networks). With the growing requirement for flexibility and adaptability, modelling for process management is likely to move closer to the 'planning' model in the future, however. There is also a possible difference in the stage of development at which between business modelling and KE are applied. Business modelling is performed when establishing requirements, whereas KE is performed during early stages of software development.

Where workflow management systems are used, there are two distinct stages of modelling. The first is elicitation/documentation or design of the enterprise model (including business processes, resources and organisational structure). This may be documented, analysed, and simulated using high level modelling tools such as ARIS. Representation formalisms and storage formats tend to be proprietary. The second step is to produce a lower level model suitable for execution by a workflow engine. Unfortunately, while there is work by the WfMC towards an official vendor-neutral process definition language, this has not been taken up by software vendors so far. Different workflow products differ considerably in the style and syntax of input required. It is generally oriented to the requirements of automated execution of the flow of documents and control between task-performing resources. It says little or nothing about the nature of the tasks that are linked in this way. Although some high-level modelling tools do claim to generate workflow definitions for specific workflow engines, this capability is generally felt to be inadequate at present. Often engineers must write the workflow definition using a modelling tool associated with the workflow engine, using the high level model as a reference. It is hoped that tools from AI planning may help bridge this gap.

If planning is to be used for BPR problems, the first step would be to think at a high level of what inputs of a planner correspond to the knowledge that BPR tools use, as well as what output of the planner corresponds to what knowledge on BPR tools. At a high level, one could establish the following relation:

Inputs of a planner

Domain theory: usually composed of a set of operators in STRIPS-like language (described in terms of pre- and post-conditions). Each BPR domain (e.g. the accounting domain in an organisation) can be defined in terms of a set of activities (here, the terminology can vary and use other words as tasks, or, even, processes) that are performed by organisation agents (either human or software). Therefore, there is a strong relation between operators in planning and activities in BPR, but it is not clear yet how to go from an activity based representation (agents responsible of a task, resources to be used, time that it takes to perform it) to an operator based representation (pre- and post-conditions, and, in some cases, other issues such as time constraints).

Problem: in planning, problems are described in terms of an initial state and a set of goals. They represent particular instances of situations for which one would like to have a solution. For BPR, a problem might be described as a process that has to be designed (modelled) for a particular task to be performed within the organisation. For instance, modelling the purchasing of an organisation, or the process of installing a new telephone line at a given address.

Initial state: in planning, one has to specify the starting situation of the posed problem. In the case of the BPR domain one would have to represent all knowledge that the organisation has about itself and can be used for the modelling of a specific process within the organisation. For instance, the hierarchical and/or functional representation of the organisation, the resources that it can use in its processes, or the documents that are generated within the organisation and travel around, being filled in, or filed, etc.

Goals: they describe in planning what one would like to be true at the end of the solution of the problem; that is, a set of assertions that have be true in a final state. In the case of BPR, this might be represented by the business goal of the organisation with respect to that process. For instance, a purchase has to be done, having in mind a set of time or cost constraints.

Output of the planners: usually AI planners generate a plan or set of plans. A plan can be seen as a sequence of operator applications that can lead from the initial state to a state in which the goals are reached. In the case of BPR, most processes are sequences of activities, adding conditional branches. Therefore, one would have to work on the generation of conditional plans, if a “typical” BPR model wants to be built.

23.2. Current state of the art

The state of the art is characterised by the large number and diversity of representation languages and modelling tools available. This may be indicative of the importance of the topic and that a definitive solution is still a long way off.

23.2.1. Process management

There are many different modelling tools. Often they are expensive and need skilled personnel, though there is considerable variation. A common criticism is that the modelling tools do not produce output in a format that is acceptable to process definition tools. Examples include: iThink[1] (relatively simple to use, systems-oriented), ARIS[2], ARENA (both complex, include simulation tools), ProSim/ProCap[3]

Standard process description languages include: IDEFn[4], PIF[5], EPIF, PSL[6], WPDL[7], CPR(Core Plan Representation)[8], SPAR()[9] There are many other proprietary languages associated with particular tools.

23.2.2. AI Planning

Plan description/modelling languages include: ADL (action description language), PDDL (Planning Description Domain Language[10], a planning interlingua), TF (O-Plan[11]), the domain description language of IxTeT, HSTS-DDL[12], Biundo-Stephan papers[13], theory of action formalisms.

The knowledge engineering TCU discussed requirements for a planning domain description language at the September 1999 meeting in Durham. PDDL was seen to be deficient in that it is not equipped with a methodology or language structure that helped the planning domain modeller. STRIPS/PDDL was likened to a 'low level' language - theoretically expressive but not pragmatically expressive enough. Also, the underlying STRIPS-assumptions were thought to restrict the usefulness of the language.

23.2.3. Ontologies

Work on enterprise ontologies has been conducted by

- University of Toronto (Fox et al, TOVE project, etc.)
- AIAI (Enterprise and TBPM projects)

A lot of work on ontologies has been sponsored by DARPA, e.g. the knowledge sharing effort[14], though it is not clear whether enterprise modelling has been addressed specifically. (See also the proceedings of the AAAI 94 workshop on AI and business process re-engineering).

[1] http://www.hps-inc.com/bus_solu/ithink/ithink.htm
[2] http://www.iwi.uni-sb.de/teaching/ARIS/aris-i/aris-e-i/index.htm
[3] see e.g. http://www.idef.com/
[4] see e.g. http://www.idef.com/
[5] http://ccs.mit.edu/pif1.html
[6] http://www.mel.nist.gov/psl/
[7] see http://www.wfmc.org/
[8] http://projects.teknowledge.com/CPR2/
[9] http://www.aiai.ed.ac.uk/~arpi/spar/
[10] ftp://ftp.cs.yale.edu/pub/mcdermott/software/pddl.tar.gz
[11] http://www.aiai.ed.ac.uk/~oplan/
[12] http://www.cs.cmu.edu/afs/cs/project/ozone/www/PCP/hsts.html
[13] http://www.informatik.uni-ulm.de/ki/Biundo/publications/publications.html
[14] http://ksl-web.stanford.edu/knowledge-sharing/README.html

MIT's Process Handbook is an evolving repository of business process knowledge. It is also available in 'shell' form for organisations to populate with their own knowledge[15].

23.2.4. Software/knowledge engineering

Software engineering modelling languages and knowledge representation techniques also cover similar ground. The Unified Modelling Language (UML)[16] has become dominant in object oriented software engineering and is increasingly being used in business modelling too. Recent extensions improve its usefulness in modelling processes. The CoRE (Controlled Requirements Expression) method originally developed by British Aerospace (Military Aircraft) and Systems Designers has been used by AIAI in conjunction with TF/O-Plan[17] and is also the basis of the COGSYS EnCore tool for requirements engineering. The best-known knowledge engineering method is CommonKADS[18].

23.3. Requirements

- easy to use tools combining modelling and simulation capabilities,
- standardisation of representations and tools to aid interoperability and minimise requirements for re-training when changing to a new tool,
- ability to output to / integration with process definition tools

23.4. Current research trends and active projects

Relevant active / recent European projects include:

- GLOBE – Esprit (SEMA is partner)
- ESPRIT 4 projects – BPR
- COMPETE (integrated approach to HR-oriented aspects of problem)

There is a mutual benefit in liaising with the knowledge engineering TCU on this topic.

23.5. Open issues

- What is the best way to synthesise business process management, AIP&S and ontology modelling languages?
- Domain experts have stated a requirement for new simulation tools. Can we clarify the requirement given that simulation tools do exist? Why are existing tools not good enough? Price? Ease of use?

[15] `http://ccs.mit.edu/CCSWP198/`
[16] `http://www.rational.com/uml/index.jtmpl`
[17] see Tate et al paper in the proceedings of the knowledge acquisition TCU workshop in Salford, April 1999
[18] `http://www.commonkads.uva.nl/`

23.6. Research goals

A modelling language for enterprises and their activities that:

- domain experts are comfortable with: they can understand it and write in it
- has a rigorous semantics
- has textual and graphical representations
- is executable (for simulation purposes)
- is mathematically formal so that the (static and dynamic) properties of models can be analysed
- planning techniques can be applied to.

A means of creating and managing a library of processes / activities. It should include means of verification / analysis of redundancy

23.7. Recommended actions

- define a taxonomy of description languages
- Write report documenting an agreed comparison & classification of PM, AIP&S and Ontology modelling languages
- Write a comparative survey of existing software tools.

24. Planning, scheduling and resourcing

Section proposed by Amedeo Cesta, IP-CNR, Rome, Italy

24.1. Introduction

- apply planning and scheduling techniques where needed, i.e. where combinatorics are significant.
- support user, not just automate.

24.2. Current state of the art

- many planning techniques and software, need to be applied by expert on case-by-case basis.
- some commercial scheduling tools (e.g. ILOG Schedule). Not entirely general purpose. Need to be customised for new applications. need to be used by experts (in the tool).
- some successful industrial applications of scheduling.

24.3. Requirements

- integration of scheduling into process management (and execution) tools.

24.4. Current research trends and active projects

- integrated planning and scheduling. (Ixtet, HSTS)
- model checking
- SATplan, Graphplan
- generating schedules that guarantee certain behaviours at run-time, i.e. that are robust against limited changes in the environment
- constraint based approach to scheduling – liaise with on-line scheduling TCU
- mixed initiative planning

- Projects: Nasa JPL projects (manned Mars missions), DARPA projects
- decision-theoretic planning (survey article in JAIR 1^{st} issue of 99) – see also execution
- ...

24.5. Open issues

- how best to combine human capabilities with planning and scheduling:
- automate boring work
- get software to work out the combinatorics and present results for person to use.
- mixed initiative planning
- keep human in control

24.6. Research goals

- planner that is easy for domain experts to use. Could have limited capability.
- ...

24.7. Recommended actions

- Define graduated reference problems

25. Enactment / execution

Section proposed by Daniel Borrajo, University Carlos III of Madrid

25.1. Introduction

This section describes the issues related to the application of a pre-defined process model (or plan) in the real environment. If we are dealing with workflow tools, enactment deals with the application of a process model in the target organisation. In case we are dealing with planning, execution deals with the application of plans to achieve the proposed goals. There are many common aspects that emerge from the comparison between the enactment of workflow processes and the execution of plans, such as tasks like monitoring, control, exception handling, adaptation, or interaction with the environment.

25.2. Current state of the art

Most of the current work on execution of plans belongs to the field of robot planning, given that most work on planning has been devoted to developing faster or more powerful planners, and there have been very little applications of planning techniques within organisations. Within this field, there has been work done in the following issues related to execution:

Conditional planning: since in many domains it is very hard to think "a priori" about all possible outcomes of the actions in the environment, conditional planning generates plans with branches. When executing a plan, every time alternatives are found, the current state of execution is consulted and one branch is selected. Another type of domains which require a similar treatment are information gathering tasks, in which in order to select an action or another, some data has to be gathered.

Decision-theoretic planning: in many domains, the world is uncertain and/or non deterministic (execution of the same action does not always arrive to the same state). In those domains, it is needed to explicitly reason about probabilities of fluents to be true, as well as actions to cause certain effects. (cite survey article in JAIR 1^{st} issue of 99, as well as Jim Blythe paper on AI Magazine, Buridan, Weaver, . . .) – see also under planning

Reactive planning: most work in robotic tasks deals with two types of planning: deliberative and reactive. Deliberative planning is used to generate high level descriptions of sequences (or sets of sequences) of actions to be applied, without consideration of the actual details of the plans. When execution begins, control is assigned to a reactive component that decides what to do in the real environment. It selects the next action to be performed according to the current state of the system and the desired goals. In many cases, the reasoning is as simple as a pre-defined algorithm, and in other cases, it performs a very narrow local search to decide what to do next. Usually, the reactive behaviour has been learnt by using many different techniques, although in some systems it is an "ad-hoc" procedure built from scratch. (references to work on reactive behaviours, PRS, RAP, learning, reinforcement learning)

25.3. Requirements

In order to apply planning-execution techniques to enactment of workflow models, there is a need to consider the following set of requirements:

Techniques for monitoring execution: given the complexity of many organisation processes, it is very important to be able to continuously look at the enactment of the processes. Monitoring should report on tasks that are being delayed, aborted, or resumed. Sometimes the identification of any of these issues is very difficult. For instance, it is very common that people forget to notify the computer (monitoring software) the completion of tasks, or changes in tasks development. Also, in most situations, people tend to delay the execution of tasks as much as possible, allowing very few time for reaction.

Techniques for exception handling: related to the previous requirement, once a problem is encountered, there is a need of defining procedures for recovery of the flaw. There have been some work in the field of planning with respect to re-planning, and recovery from failures that could be of some help to workflow failures. From the point of view of workflow systems, exception handling is usually performed by "ad-hoc" procedures that applied given that a problem is detected. In contrast, by allowing a declarative representation of operators, the system might be able to reason about possible failures and how to solve them.

25.4. Open issues, research goals, and recommended actions

Analysing the current research trends and the state of the art on the execution of plans, the following is a set of subjects that are pending to be solved:

How to combine user preferences: usually a process that is being enacted is composed of many tasks to be performed by people with different roles and different qualifications. Assigning a task to a human is usually performed having in mind the set of roles that s/he is able to perform. However, there are other aspects that are worth considering such as user competence (even if a person is able to perform a given task, in what type of tasks is s/he really good at?), or user preferences (what type of tasks does s/he really enjoy performing?). Also, given that some tasks have to be performed by a group, this arises issues such as how to arrange the most productive group. Some of this issues are strongly related to the currently very intensive field of knowledge management.

This generates the goal of developing theories that define and reason about user models with respect to the assignment of tasks

Flexible working with overall plan: in most cases, detailing all possible aspects of the tasks can cause users to loose interest on their work. It is a better policy to provide some level of freedom for the execution of tasks, allowing decisions to be made by the human when executing a task. This is analogous to the integration of deliberative and reactive planning in robot tasks. A given degree of reactivity, allows the robot to be better prepared to cope with uncertainty and non determinism. A given degree of freedom, also allows the user to be prepared for uncertainty and non determinism, but also influences his/her way of looking at the work. The issue is how to combine the overall plan with the specific interests of the humans that have to carry out the plan steps.

The associated goal would be the definition of models that allow to generate processes at various levels of detail, and interleave the execution of a high level plan with a somewhat reactive component.

Can plan repair or re-planning techniques be helpful in exception handling / jeopardy management? As mentioned before, there has been some work done from the planning perspective with respect to handling plan failures during execution. It is not clear how this work should help and/or influence the workflow jeopardy or failures.

It would be needed to study the sets of possible failures that can occur within the enactment of a process, and the set of repair procedures for those failures.

How to provide personalised view of process (visualisation of big picture): another of the features that users find very important when performing a task of a process is knowing issues such as: why am I doing this?, where does this document come from?, or who should read this document afterwards? All of them deal with the problem of giving the users the ability to inspect at a certain level of detail the connections between the activity they are performing and the overall picture of the whole set of processes of the organisation.

A research goal in this respect would be the description of variable visualisation techniques for parts of processes and the relationships among the processes of an organisation, having in mind security issues.

How to combine and interleave plans for multiple humans (agents): if a distributed plan has been generated, the execution of that plan should monitor the interactions of the plans for each agent and combine the executions in the most effective way. Also, it should solve problems arising from the failure in an agent plan that has connection with other agents plans.

The definition of a protocol of communication and negotiation between agents plans, execution of plans, failures, and repair methods would be needed.

26. Adaptation, optimisation and metrics

Section proposed by Daniel Borrajo, University Carlos III of Madrid

26.1. Introduction

Proposed by Paul Kearney

In this chapter, we will discuss an increasingly important aspect of workflow enhancement: how processes can be optimised/adapted according to design or enhancement problems. We will also discuss about the metrics considered for changing the processes. In general, there are two places in the application of workflow technology to organisation processes in which changes to the processes are involved:

Design phase: when designing a given set of processes, the user might want to obtain an optimal process model according to a set of metrics and constraints. Usually, time and cost have been the only metrics considered for optimisation. Also, optimisation has been mainly a manual process, helped by the use of (sophisticated) simulation and analysis tools.

Enhancement phase: when a process is being enhanced, many mismatches (might) occur between the designed process and its actual implementation. The role of adaptive workflow would be to feed the design and/or enhancement with those mismatches in order to optimise/adapt the process to the real situations.

Following the analogy between the process of applying planning technology and workflow technology that appears in chapter 18.3, there are several aspects that workflow and planning have (or not) in common with respect to optimisation:

Design phase: the goal of both tasks (planning and workflow enhancement) is to obtain a process (plan) to be enhanced (executed) in the 'real' world. However, while workflow has always considered optimisation (of time and cost) as a part of its design phase, it has not always been the case for planning. In the case of planning, the main emphasis is on satisfying a goal, rather than on finding an optimal plan. This is mainly so, due to the already inherent complexity of finding 'a' plan. When plan quality is considered, it has been mainly computed as 'plan length', instead of using any user defined metric.

There have been though several approaches that try to plan for optimal solutions (cite some). In most cases, the approach has been to learn control knowledge to guide the planner towards 'good' solutions.

Enhancement phase: the second main goal (should we say the first one) of both tasks is to enhance (execute) the designed process (plan). Here, we also find some differences between workflow and planning. Workflow enhancement is currently very widely done, so most organisations that have been (re-)designing their processes are following them. However, very few applications of planning systems have been built and used. Therefore, from the optimisation/adaptation point of view, there are many more lessons to be learned from workflow applications than from planning applications. Since optimisation/adaptation coming from the enhancement (execution) needs to know what types of failures can occur within the execution of a process (plan), we might have more information coming from workflow.

Listing all possible metrics is an infinite task. However, there are some that have been considered in many applications:

Cost: measured by whatever means. Currently, ABC analysis is commonly carried out within business processes.

Time: usually measured as time steps of the process.

Quality: e.g. defect rate in a product, delays and dropped packets in a network.

Value of the end-product: e.g adding an extra processing stage may increase the value of the end product more than it increases the cost.

Flexibility: the ability to change processes quickly is important.

Processes that are highly optimised w.r.t. cost or time may well be inflexible.

Robustness: the probability of success of the processes.

A related issue is the use metrics to motivate and assess the performance of people. Inappropriate metrics can have the opposite effect to that intended. For example if targets are perceived as impossible, then people will ignore them. Thus if a target is made more demanding it may in fact decrease performance. Similarly, taking a call centre as an example, an ‘obvious’ performance metric is number of calls handled per day. However, this encourages staff to keep calls short, which may mean that poor answers are given leading to more calls. This improves apparent productivity, but customer satisfaction goes down. The ‘correct’ productivity metric must take into account whether the caller was satisfied, but this is more difficult to measure.

In the next sections, we discuss issues related to optimisation with respect to: open questions; research results; barriers to technology transfer; and software and application requirements.

26.2. Current state of the art

The following is a set of results that might be used to approach the open questions of previous section:

- There are all types of mature optimisation techniques coming from AI and operations research such as: heuristic search; genetic algorithms; or linear/dynamic programming.
- There have been some approaches on planning for better solutions (Ambite and Knoblock) and learning to plan for better solutions (SteppingStone, Quality, or Hamlet)

26.3. Requirements

Here, we discuss what the workflow tools and applications should have in order to allow optimisation:

Integration with process design and enhancement tools: optimisation and adaptation procedures should be integrated on one hand with process modelling techniques (for obtaining good models), and, on the other, with process enactment tools (for adapting the models according to actual enactment of the processes)

Interaction with the user: an important aspect of the tools consists on allowing the user to interact with the optimisation and adaptation procedures so that s/he is able to direct towards process models that comply with user expectations

Use-definable metrics and optimisation parameters: the user should be able to provide in a given language descriptions on how metrics should be computed, as well as parameters for controlling how optimisation and adaptation should be performed

26.4. Open issues, research goals, and recommended actions

The set of open questions with respect to optimisation/adaptation and metrics are:

- Do workflow applications really need metrics different than time and cost? If we are going to define tools for performing adaptation/optimisation according to user defined metrics, we should first make sure that users will need different types of metrics. A possible recommended action would be to survey in some organisations about this aspect.
- What language should we use to provide those metrics to the system? We should study what are good languages for describing those metrics, so that potential users of the tools are able to easily define metrics by themselves
- If multiple agents are used, how should their respective metrics be combined/negotiated? Should it be left to execution time or should it be worked out before execution starts?
- What is the set of possible failures of a process (plan)? Although this question also appears in the section on execution, within this section, it refers to the generation of plans that are optimised according to, for instance, less probability of failure
- How should workflow enhancement influence optimisation/adaptation? This issue is related to the plan repair techniques in the execution section
- Where should design/enhancement optimisation knowledge come from? There might be three different types of sources: experts on a given domain (they usually know what models are wrong and why, what resources should be assigned to what task, ...); experts on BPR or workflow enactment (usually they work on consultancy firms and provide advice on how different organisations implement their processes); learning from past executions of the workflow or from the history of the processes execution in the organisation.

- Are there experts on resolving failures of execution, or anticipating problems? This issue is related to the previous one. Usually, in big organisations there is people in charge of this task that could be of great help.

- Can the systems recognise a ‘good’ solution? Or how do we define procedures for computing how good a model is?

- How should the interaction with the user be integrated when optimising? Optimising a process might result in a less intelligible process, so an analysis on what is preferred.

When trying to apply optimisation to process design/enhancement, the following is a list of possible and actual problems:

- The user might not know/distinguish when s/he needs optimisation.

- How does the user describe optimisation and metrics knowledge?

27. Summary and conclusions

This document has presented the first issue of the PLANET R&D Road Map for AI Planning and Scheduling applied to Workflow Management. In an applied discipline such as this, a Road Map must not only identify research challenges, but also match them to current and projected end-user requirements. It must also consider the process by which the results are incorporated into the tools of the trade of the end-users and application developers. Furthermore, necessary preconditions for successful application of the results must be taken into account. This version is an important step towards a coherent strategy, but is not itself the definitive answer. The Road Map needs to be a living document that is developed and updated and regular intervals.

27.1. Main achievements

One of the main achievements to date has been to develop an understanding of how the 'world view', vocabulary, challenges, etc. of Business Process / Workflow Management relate to AI Planning and Scheduling. This has been possible because of the active participation of a small number of workflow and process management experts from end-user organisations and consultancy companies. The site visit to BT to gather information on existing (non-AI) software applications was also extremely valuable in this regard. The TCU must make every effort to involve more end-user representatives (not just researchers, but problem owners) from a spectrum of industries. A number of commercial software vendors are registered on the TCU mailing list but have not as yet participated actively. It is important for such organisations to become actively involved. For planning techniques to be of practical use they must be integrated with, or must interface to, commercial workflow management systems (WfMS) and other related software.

Requirements have been classified as short, medium and long term as follows:

Short term: address short-comings in current-generation process management software. The most important items in this category are: integration of scheduling and resource allocation/management algorithms into workflow management software; and incorporation of a planning capability to enable a WfMS to modify the process instance automatically during execution, to cope with failure, changed objectives, and other exceptions.

Medium term: Current generation workflow software handles high volume routine processes, typically involving low-skill workers. The medium term requirements concern extending this support to high-skill knowledge workers. This may involve, for example, building process awareness into software tools.

Long term: More radical (e.g. adaptive self-organising) approaches addressing the need for organisations to function in a business environment that is increasingly uncertain and subject to change.

This document has also made a start on identifying planning techniques and research goals that address these requirements. In addition to the application of planning and scheduling algorithms we discuss: advantages to be gained from using AI plan representations for processes, ideas from plan execution (especially in uncertain environments), and work on adaptation optimisation and metrics. Further work remains to be done, however, to identify specific research goals and projects.

Two further topics are also discussed: human issues and infrastructure. It is important to remember that much of the work in a business process is performed by people. Often technology is seen primarily as a means of cutting costs through automation rather than enhancing value by enabling people to work more effectively. The result of treating people like machines is often demotivation, high staff turnover, loss of productivity, etc. In addition, human qualities are under-utilised. There is a danger that must be guarded against that planning and scheduling techniques may make this situation worse rather than better. The discussion of infrastructure mainly focuses on the need for a reference architecture and interface standards to AI-based software tools to be integrated with each other, with conventional process management software, and with the general enterprise infrastructure.

27.2. Main recommendations

The TCU is playing a useful role in closing the gap between industry and academic research. However further work is needed:

- to make researchers aware of the real challenges and constraints of the workflow domain;
- to make application and tool developers aware of what AI planning and scheduling research has to offer;
- to address practical issues of integrating planning and scheduling technology into suites of application software, and of making the techniques usable by typical software engineers, analysts, etc.
- to form a consensus on medium and long term research goals.

The Road Map should be seen as a living document and be extended and updated regularly.

Specific recommendations on how to do this now follow:

- active membership of the TCU needs to broadened to include more end-user organisations from a broad spectrum of industries, and developers of commercial business process software tools. To achieve this, we must show what these organisations will gain from participation;
- we should build up a library of descriptions of application scenarios. The site visit to BT led to very useful insight into the challenges faced by real workflow applications and the approaches currently used to address them. These are documented in the visit report. However, we need similar examples form other companies and other industries.

- there is potential scope for use of developments of existing planning, scheduling and plan representation techniques to address short term requirements. This needs to be confirmed through case studies of real business problems to clarify the benefits to be gained, practical feasibility, etc. Note that to be of practical utility, the techniques need to be usable by an average software or business process engineer, and to be integrated into the wider business process toolset of the organisation. The TCU should play a role in promoting and facilitating such projects.

- we should hold workshops focussed on key open issues or research challenges identified in the Road Map. One suitable topic for such a workshop is *Dynamic re-planning and resource allocation in a distributed execution environment*. The BT site visit report can be used to provide a context.

- Site visits to gather information on application requirements need to be followed with specific proposals on how the problems can be addressed. These proposals could identify solutions based on current planning results or longer-term research or development targets.

- The TCU should not itself propose *projects*, but should identify topics for projects and encourage its members to conduct them. The TCU can also play a match-making role in bringing together compatible partners.

Bibliography

[1] An overview of workflow management: form process modelling to workflow automation, Georgakopolous, Hornich and Sheth, Distributed and Parallel Databases 3, 119-153 (1995).

[2] Functionality and Limitations of current Workflow Management Systems, Alonso, Agrawal, Abbadi and Mohan

[3] Workflow Management Systems: an AI perspective, Myers and Berry, Technical Report Artificial Intelligence Center, SRI International, 333 Ravenswood Ave., Menlo Park, CA 94025, Jan 1999. December 23rd 1998, `http://www.ai.sri.com/~berry/publications/prc-sri.ps`

[4] Jacobson, Ivar et. al., Object Advantage: Business Process Re-engineering with Object Technology, Addison-Wesley, 1995 (Note: a textbook for BPR. Only to be included if bibliography covers this type of publications, and /or if we expand the coverage of the initial stages of BPR. (NM))

[5] Shoshana Zuboff. In the Age of the Smart Machine: The Future of Work and Power, Basic Books, New York, NY, 1988 ISBN: 0-46503211-7

[6] Abbe Mowshowitz. Virtual Organization: A Vision of Management in the Information Age, in The Information Society, 10 (4), 1994

[7] Michel Beaudouin-Lafon, Computer Supported Co-operative Work (CSCW), John Wiley and Sons Ltd 1998, ISBN 047196736X

[8] G. R. Oldham. Job design. In C. L. Cooper and I. T. Robertson, editors, International Review of Industrial and Organisational Psychology. Wiley, 1996

[9] P. R. Jackson et al. New measures of job control, cognitive demand and production responsibility. *Journal of Applied Psychology*, 78:753–762, 1993

[10] B Nardi. *A Small Matter of Programming: Perspectives on End User Computing*. MIT Press, Cambridge, Massachusetts, USA, 1993

[11] M. Burnett and D. W. McIntyre. Guest editors' introduction: Visual programming. *IEEE Computer*, 28(3):14, March 1995

[12] Kuhn, S. and M.Muller (1993, June). Participatory Design - introduction. Communications of the ACM 36(4), 24-28. Editorial.

[13] Helge Kahler, Oliver Stiemerling, Volker Wulf and Anders Mørch, Computer Supported Cooperative Work: The Journal of Collaborative Computing, special issue on "Tailorable Systems and Cooperative Work", 9(1), Kluwer Academic Publishers, 2000

[14] Hutton99, unknown reference.

[15] Long, unknown reference.

[16] Mehandjiev, unknown reference.

Appendix A: Ideas arising from BT OSS meeting

The following are ideas arising from the meeting of 7th October 1999 in London with the BT OSS specialists. Some points identify requirements, others identify AI techniques that might be useful in addressing problems typified by the BT OSS applications. Most of them need further study to evaluate benefits and penalties compared to the techniques currently employed:

Representation and reasoning:

- conditional plans (choice points represented explicitly)
- declarative way of representing conditions (deductive databases)
- Actions can be performed during execution (cancellation, etc.). Language to describe?
- Evaluation of benefit (and ‘cost’) of formal language use.
- Errors in date entered -> need to handle uncertainty, plus formal reasoning could help detection / correction

Planning / plan repair / re-planning

- use of case-based techniques for plan template selection?
- automatic changing of process (adaptation of process / plan repair) – switch process mid way through
- Many repair methods in CSS. Can we do anything to help? Effects modelling and execution.

Scheduling / temporal reasoning

- considering resources when scheduling

Optimisation:

- use of a multi-valued optimisation function

Modelling / process design:

- modelling: library of processes / activities– creation and management. incl. verification / analysis of redundancy
- dependency between process templates: changing one process requires change to others. Minimise and manage dependencies
- continuously receiving new restrictions on how processes behave: legislative, business rules, restrictions due current state of market, market practices. Have defined language within SHAMASH – Daniel will check whether demo will be possible.
- Requirement for simulation tools – can we clarify requirements given that simulation tools do exist? – why are existing tools not good enough? Price? ease of use?

Other

- graphical visualisation both for 'controller' and for 'worker' within a process (definitely needed, but can planning results help here?)
- mixed initiative systems
- generation of explanations from plans
- produce timing estimates using bottleneck estimates, etc.
- multiple orders for same customer – coordinating site visits, etc. – could also consider external supplier
- prioritisation of jobs in queues (system vs. manager vs user)
- anticipating jeopardy and failures (datamining, historical info)
- inter-company processes – integration
- maintenance of code. Can AI add anything?
- general need for integration with lots of software systems

Possible follow-up actions:

- Possibility of focused tutorial as a kind of return meeting.

Part V.

On-Line Scheduling

Simon de Givry

(Thales LCR, Orsay, France)

with contributions from Gérard Verfaillie (ONERA, Toulouse, France) and David Lesaint (British Telecom, Paris, France).

Introduction

On-line scheduling involves a number of research and development aspects, PLANET aims to concentrate on them. These include, among others, the solution of dynamic constraint-based scheduling problems, multi-criteria scheduling as well as the development of anytime constraint solvers. On-line scheduling is relevant for many applications involving both time and cumulative aspects. Currently, various academic and industrial nodes of PLANET are intensively working on both the application and research aspects of on-line scheduling. Therefore, close collaboration in this area will cause a significant enhancement in the development and transfer of the technology.

The aim of this roadmap is to analyse on-line scheduling in order to extract its main characteristics and to give an overview of the current techniques to answer them. This document is organized as follows:

- An overview of on-line scheduling (chapter 28),
- A description of several on-line scheduling applications (chapter 29),
- An analysis of the main characteristics of on-line scheduling (chapter 30),
- An overview of the current techniques to answer the numerous questions raised by each issue and the difficulties which remain ; all these questions are assembled in a table (cf. p.191) where a degree of maturity is associated to the current techniques for answering a question (chapter 31)

28. Overview

We use the term on-line to refer to operational constraints. An on-line system must interact with the outside world in a continuous and reactive way. Thus, on-line scheduling means bridging the gap between theoretical scheduling and real-world scheduling. In real-world problems, the notion of scheduling is not clearly differentiated from the notion of planning and implies reasoning about actions, states, resources and time globally. Moreover, real-world problems are dynamic, uncertain and often unpredictable.

The objective of scheduling is either to bring the controlled system into a given state or to perform a given level of service permanently. In either case, the classic features of a solution provided by scheduling are its feasibility and its quality. A new feature of any solution is its adequacy to the context as it is when the solution is provided. The faster the solution is produced, the higher its adequacy is. As finding feasible and high quality solutions may take a long time, there is an obvious contradiction between solution feasibility and quality on the one hand and solution adequacy on the other hand. We think that this contradiction is the main problem of on-line scheduling as compared to off-line scheduling and that it must be addressed.

There are different software architectures for on-line systems, such as blackboards or systems with reactive and deliberative components. Although the corresponding techniques are quite mature, it is still difficult to have a clear separation and interaction definition between the reactive and deliberative components.

When scheduling actions, it is necessary to specify on which temporal horizon one wants to schedule, i.e., which set of possible actions and which temporal windows for them one considers. This horizon may be fixed either statically or dynamically. In the latter case, it is difficult to determine a good trade-off between computation time and horizon length. In multi-level scheduling, a problem can be modeled by a set of nested horizons, with a scheduling granularity associated with each of them and increasing with size. The interactions between scheduling levels has to be defined. Instead of high levels making high-level decisions that are imposed on lower levels, these lower levels may continue to use as advice the solutions produced by higher levels. However, they must be able at any moment to make decisions by themselves, when the adequacy of these solutions has clearly become too bad. See hierarchical planning, blackboard systems and mixed-initiative problem solving.

Once scheduling horizons have been fixed, we have to decide when a new schedule should be computed. This will influence the solution adequacy. The scheduler should recompute a new schedule when new conditions invalidate the current schedule, or when the end of the current horizon is closed. An invalidating condition can be a new event (e.g. a new activity, a resource failure, a new criterion requested by the operational user) or a new estimation from sensors (e.g. an updated temporal constraint). Rescheduling can be either event driven or periodic. For detecting and repairing an invalidating schedule, we need tools for solution analysis, conflict explanation and relaxation proposal. There are many links with research on solving over-constrained problems.

A major operational constraint is temporal constraint on reasoning. We can distinguish three kinds of temporal constraint: (i) the temporal deadline is known, the system must deliver a good solution before the end of a given time contract, (ii) the temporal deadline is unknown, the system can be interrupted and asked for a solution at any time, the scheduler should perform well on average during a specified temporal interval, (iii) the temporal deadline is unknown, the system chooses the right moment to deliver its solution. The first case (i) implies using a mechanism for algorithm selection depending on time contracts. For tree search methods, it is important to estimate the number of nodes developed by the search. It is difficult to have a reliable estimation, especially in an optimisation context. The second case (ii) implies using iterative methods, which progressively increase the size of their exploration at each iteration, from a greedy search to a complete search. The last case (iii) implies using a meta-reasoning mechanism based on the notion of utility. Cases (ii) and (iii) are very challenging.

Most of the time, the scheduler has to solve a dynamic problem, in the sense that the new problem to be solved is different from the previous one (the scheduling horizon has been shifted, some of the previously scheduled actions have been executed, changes may have to be considered). However, it is not so different. The scheduler can benefit from a learning mechanism based on solution reuse, on reasoning reuse (set of no-goods), or on both. Special interest has been devoted to incremental constraint retraction mechanisms. Sometimes, problem changes are difficult to identify.

Robustness is an important property of solutions. A solution is said to be robust if it resists changes. A related property is flexibility. A solution is said to be flexible if it is easy to repair, in the case of problem changes. We need a model of the possible changes in order to produce change-proof or flexible solutions. Models of uncertainty have been studied in mathematical research, for example in Decision Theory, Utility Theory and Game Theory. The Markov Decision Process extended to problems with constraints is a promising research direction.

Solution stability is important in all applications where changes in schedule are costly or difficult to execute. A new solution is stable with respect to the previous one if there are few changes between the two solutions. But it must be noted that a contradiction appears between solution quality and stability. Several algorithms are currently being developed and experimented to produce stable solutions.

Sometimes, the problem or its solving method are distributed. The reason may be geographical (different places), organisational (different companies) or simply because it is easier to solve (problem decomposition). If we want to introduce either cooperation or conflict resolution, distributed decision-making protocols have to be proposed. See multi-agent scheduling techniques.

A difficult problem remains: how to validate an on-line scheduling system ? And how to measure the solution adequacy ? We can compare the on-line scheduler with an optimal off-line scheduler, as is done in research about on-line algorithm.

In conclusion, we think that on-line scheduling requires further work on theory, algorithmic and experiments. It is a real challenge to obtain efficient generic tools for on-line scheduling, comparable with problem-dependent approaches.

On-line scheduling is a combination of several Artificial Intelligence research areas, e.g. Operations Research, Constraint Satisfaction Problems, Constraint Logic Programming, Planning, and realtime-oriented research areas, e.g. Hard-Real-Time Systems, Anytime problem solving. Other related research areas should be employed: Case-Based Reasoning, Machine Learning, Decision Theory, Utility Theory, Game Theory, Risk Management, Markov Decision Process, Reinforcement Learning, On-Line Algorithms, Multi-criteria techniques ...

The novelty of on-line scheduling compared to off-line scheduling resides mostly in new solution properties, such as its adequacy, its robustness, its stability. Such properties can be in contradiction with its quality. Dealing with this issue is the main challenge of on-line scheduling.

29. Examples of industrial applications

29.1. Dynamic Workforce Scheduling for British Telecom

Simply stated, field workforce scheduling is about sending the right employee to the right customer at the right place at the right time with the right equipment - at any time and in any operational environment. For BT, which employs over 50,000 field engineers, workforce scheduling is critical, and its ability to provide high quality service while achieving maximum productivity and low operational costs is vital to the company's success and competitiveness.

29.1.1. The Problem

Many factors contribute to the complexity of the problem. First, skill requirements vary immensely. There exists three distinct operational areas - business and residential customer access, national business communications, and core network - which are the responsibility of three separate BT divisions. Second, the workforce being dispersed all over the UK has to be managed in a decentralized way by autonomous centres (called domains). For example, the 15,000-person workforce in the customer access division consists of 175 separate groups that operate within nonoverlapping geographical domains.

Whatever the domain or the division, scheduling decisions are all subject to a complex set of requirements and objectives. These requirements reflect standard allocation constraints and corporate rules, and they determine the feasibility and acceptability of work assignments. Formally, BT's workforce scheduling problem may be defined as a complex variant of the vehicle routing problem featuring multiple vehicles, multiple depots, compatibility constraints, time constraints, operational constraints, synchronisation constraints, and conflicting quality objectives.

Constructing feasible and good-quality work schedules under these conditions is hard in itself but the problem is further compounded by the inherent instability of the environment. Indeed much schedule information is uncertain, imprecise or incomplete: BT and its customers may request, cancel, or amend orders unpredictably, engineer availability is subject to last-minute changes and estimates of task duration and travel time cannot be totally accurate, work controllers may modify provisional work assignments and review business objectives at any time; and the environment itself (weather, traffic conditions) is unpredictable.

Therefore, a scheduler must continuously incorporate new data to generate valid work assignments but it must also minimize the impact of these data on the current work schedule. Indeed, anyone interacting with the work-allocation system (e.g. a resource manager) must be provided with schedule information that is as stable as possible over time.

29.1.2. An On-Line Scheduler

In 1996, the Intelligent Systems Research group of BT has designed an on-line scheduler (OLS) based on a combination of heuristic search and constraint-based reasoning to solve BT's field workforce scheduling problem. OLS has been based on two principles:

- coupling loosely an on-line allocator and a predictive scheduler to preserve responsiveness while benefiting from global optimization (Fig. 29.1), and
- using a uniform constraint optimization approach to provide a generic and efficient underlying computational model.

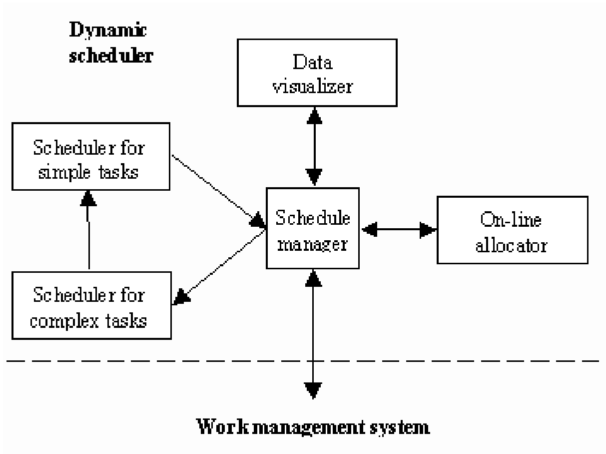

Figure 29.1.: The on-line scheduler (dynamic scheduler) consists of a schedule manager that stores and communicates the provisional schedule to the work-management system, a scheduler for complex tasks and a scheduler for simple tasks that periodically produce the long-term schedule, an on-line allocator that adjusts the schedule before dispatch and a schedule and data visualizer (a desktop computer) field resource controllers and managers use to access problem and schedule information.

A tree search algorithm and simulated annealing algorithm form the predictive component of OLS. They are run every five to fifteen minutes to generate a provisional schedule (which typically covers two days), which is then fed to the on-line allocator. The on-line allocator dispatches work assignments to engineers. It is triggered by external events, for example, tasks requested by on-line engineers, or tasks that require immediate attention before they become failed business targets. By default, it simply forwards the assignments prescribed in the provisional schedule to the engineers, but it may modify these assignments if unforeseen events have caused infeasibility or suboptimality.

Ideally, one would tackle instability by capturing and reasoning about elements of change using such techniques as stochastic programming. However, it is very difficult to forecast the evolution of BT's workforce scheduling environments accurately. Although we have identified general patterns (for example, on customer calls), no forecasting model is accurate enough to be exploited during scheduling.

The allocator's decision making is rule based, and the scope of its reasoning is limited to a subset of the resources and tasks so that it can deliver answers within a few seconds. The contents of the answer itself is kept to a strict minimum: e.g. on-line technicians are just given their next assignments, not a 2-days plan. The objective here is to make sure that scheduling decisions will be as synchronous as possible with the real world at dispatch time. Reducing the scope of the reasoning and restricting the form of the answer are two methods amongst others to achieve this objective.

29.1.3. Present and Future

Rolled out in 1997 and reaching 20,000 engineers in 1998, OLS with Work Manager is saving BT $150 million a year on operational costs. When deployed over the targeted workforce of 40,000 people, the system will save an estimated $250 million a year. Overall, OLS has reduced BT's operational costs, improved the integrity, quality, and consistency of work allocation for the whole workforce; improved customer satisfaction; enhanced resource control; and offered BT the potential of simplifying its work-management organization.

Technically, OLS has demonstrated that integrating heuristic search and constraint programming is a successful approach for tackling (static) large-scale industrial vehicle routing problems. Experimentally, the mixed scheduling strategy implemented by OLS - short-term reactive scheduling coupled with long-term periodic scheduling - has proved superior to a purely reactive strategy on this problem.

However, many questions remain unanswered, to give but a few: How often should we reschedule? What should be the scope of reasoning of the allocator and the predictive scheduler? Could we make the two components really collaborate? Could we use other scheduling components: e.g. schedule analysis, schedule repair? What does optimization mean in the context of rescheduling? etc.

These questions apply to other classes of on-line scheduling problems and simply underline the difficulty researchers and practitioners have to understand and define what the on-line scheduling problem is about from a computational viewpoint. This, in turn, explains the present absence of tools and methodology, not to mention theory, that address on-line scheduling problems satisfactorily.

29.2. On-Line Scheduling at Thales

We present briefly three examples of on-line scheduling systems, currently developed at Thales LCR (ex Thomson-CSF).

29.2.1. Defence weapon scheduling

The first system is a defence weapon system. It regularly plans ripostes to attacks. The system is autonomous and cyclic. The period is very short. A plan, that is a solution to a scheduling and resource allocation problem, is computed at each cycle. More precisely, the problem consists in associating some missiles to the current detected enemy tracks, while respecting the physical constraints of the weapon system (like limitations to the resource allocation). There is not always a solution with all the tracks well engaged, so, the optimization criterion is a function of the missile to track association.

The operational constraint specific to this on-line system is that the solving method should end before the end of each period and should always return a solution of sufficient quality. It means that we must develop a specific algorithm, doing an incomplete search and using the available computation time the best way. Such an algorithm is called an anytime algorithm. It dynamically adapts its search, taking into account the difficulty of the instance to solve, the deadline and the evolution of the quality.

29.2.2. Airline scheduling

The second system is an interactive tool to solve the airline scheduling problem. One needs to schedule the time of arrival of each flight, taking into account all the constraints. These constraints are induced by the structure of the approach area (number of runways, number of arrival procedures), the characteristics of aircraft (weight, wake vortex, landing distances, fuel capacity), the estimated landing time and the airlines' and airport authorities' strategies. The objective function is to maximise the number of flights that will be cleared to land within a time period and to minimise the delays imposed on certain flights.

This is an on-line problem because the environment is highly dynamic and unpredictable : the flow of aircraft entering the terminal area is continuous and unforeseen events should be taken into account (unavailable runways, weather conditions, ...). The system must maintain a feasible schedule and continuously adapt it to the evolution of the situation.

29.2.3. Optimal mapping for signal processing on parallel architectures

The third system is an interactive tool to help the partitioning and scheduling of a signal processing application on a parallel computer. A solution for this system is to map data on the processors and schedule the tasks on each processor. The dimensioning of the parallel computer is also a part of the solution. There are several criteria to be optimized, like minimizing the computer cost or minimizing the signal processing latency. These criteria may be contradictory. The user can control the search by assigning some variables, or by forbidding some values or partial assignment. Then, the system must solve again the problem, modified by the user. Moreover, the user builds a solution by specifying which criterion is to be optimized and how much time is available.

Thus, it concerns multi-criteria optimization algorithms on dynamic problems. Therefore one needs methods which, starting from the previous solution and the previous reasoning, allow a new solution to be rapidly found, as close as possible to the previous one.

30. What is new in on-line scheduling ?

Introduction

This text is the result of a two day brainstorming about on-line scheduling between Gérard Verfaillie (ONERA, Toulouse, France), Simon de Givry (Thales LCR, Orsay, France), and David Lesaint (British Telecom, Paris, France). This brainstorming aimed at a better characterization of what is new with on-line scheduling, when compared to a classical off-line scheduling. The underlying idea was that is useless to start with difficult technical developments before the nature of the real problem to solve has been well defined.

30.1. Preliminaries about scheduling

30.1.1. Scheduling problem definition

Usually, in the *Artificial Intelligence* community, *planning* is defined as the reasoning task that aims at choosing a set of totally or partially ordered actions that allow the controlled system to reach a given state. On the other hand, in the *Operations Research* community, *scheduling* is defined as the reasoning task that aims at choosing resources, either total or partial order between actions, either temporal windows or precise dates, that allow the set of actions defined by the planning task to be achieved.

In other words, planning implies reasoning about actions and system states, and scheduling implies reasoning about actions, resources, and time.

But it has been observed for many years that this usual distinction between planning and scheduling is not really relevant when dealing with real problems, that implies globally reasoning about actions, states, resources, and time. So, although we use the term *scheduling* in this text, the reader can replace it by *planning and scheduling*.

30.1.2. Two kinds of task to perform

When looking at real problems, one can see that there exist two main kinds of task to perform, and thus two main kinds of context in which scheduling is involved.

With the first kind of task, the objective is to bring the controlled system in a given *state*. When this state has been reached, the task is finished. To land a spacecraft on the surface of a planet or to hit a given target for a military aircraft or missile are two examples of this kind of context.

With the second kind of task, the objective is to perform permanently a given level of *service*. The task is permanent. It is never finished. To manage either an Earth observation satellite or a space telescope, to manage either air traffic or air crews inside an aircraft company are some of the many examples of this kind of context.

30.1.3. Solution feasibility and quality

Whatever the kind of task to perform, the main features of a solution provided by scheduling are its *feasibility* and its *quality* :

- a solution is said feasible if it meets all the physical and technological constraints ;
- a solution has a sufficient level of quality if it meets all the hard user requirements, concerning either the state to reach or the service to perform; its exact level of quality depends on the way it satisfies all the user criteria (or soft user requirements); note that the frontier between hard and soft user requirements, that is between constraints to satisfy and criteria to optimize depends on the way users express their objectives : for example, they can require a spacecraft to land, either within a given area, or as close as possible to a given point at the surface of the target planet; they can require, either all the clients of level 1 to be satisfied, or the maximum level of all the unsatisfied clients to be minimized.

30.2. Nature of the on-line scheduling problem

30.2.1. On-line versus off-line scheduling

Usually, the term *off-line scheduling* is used when action execution and scheduling are not connected : actions are scheduled and then executed as they have been scheduled.

Conversely, the term *on-line scheduling* is used when action execution and scheduling are strongly connected: actions are scheduled but, as soon as they are executed, execution results may modify the scheduling problem to solve.

In the Automatic Control community, the term *off-line scheduling* is often replaced by the term *open loop scheduling*, and the term *on-line scheduling* by the term *closed loop scheduling*. See figure 30.1 and 30.2 for a graphical representation of both types of scheduling contexts.

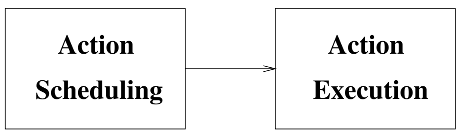

Figure 30.1.: Off-line or open loop scheduling.

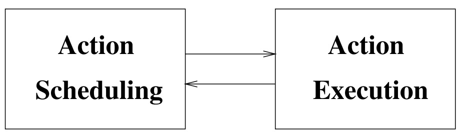

Figure 30.2.: On-line or closed loop scheduling.

30.2.2. Uncertainty, dynamicity, and unpredictability

When people speak of the on-line scheduling context, they often use the following sentences in different forms : *The level of uncertainty is high. The environment is highly dynamic and unpredictable*.

In fact, when they use the term *dynamicity*, they refer to the stream of events coming from the controlled system itself, from its environment, from its users, or from other systems.

When they use the term *uncertainty*, they want to refer to the uncertainty at a given time about this stream in the future : uncertainty about system failures, about action results ...

When they use the term *unpredictability*, they want to specify that neither probabilistic knowledge about this stream in the future, nor any knowledge of this type is available. It is often the case with events coming either from users, or from other systems within a multi-agent system context, for which there exists no predictive model.

30.2.3. Reactivity versus scheduling

In this kind of context, a first question arises : *Why to schedule when everything changes at any time ? Would it not be better to adopt a reactive behavior, as it is proposed in the* Markov Decision Process *(*MDP *[70]) and* Reinforcement Learning *[84] approaches ?*.

We think there are at least three kinds of situations where scheduling may be preferred to a reactive behavior :

- the level of dynamicity and uncertainty is not too high; so scheduling remains meaningful ;
- system users want to be sure or almost sure, either to reach the goal state, or to perform the required level of service; thus, they want to look behind the next action : for example, delaying an action may allow another one to be performed ;
- operational constraints require to anticipate and thus to schedule; it is for example the case when execution cannot follow decision immediately : either decision needs time to be sent to the system (it is the case for example with an interplanetary spacecraft), or execution needs time to be prepared (human beings have to be informed and trained, components have to be designed, ordered, assembled or carried); it is also the case when only limited temporal windows are available to control the system (it is the case for example with non geostationary satellites).

30.2.4. Solution adequacy

In the dynamic context of on-line scheduling, a third feature of any solution, in addition to its feasibility and its quality, is its *adequacy* to the context as it is when the solution is provided.

If producing a feasible and high quality solution on the basis of a given context at time t takes a lot of time, and if this context has completely changed when the solution is provided at time t', then this solution is of no interest to control the system : its adequacy is null.

As any reasoning, even the simplest one, takes a minimum time Δt, this adequacy can never be completely guaranteed. It is only guaranteed within time Δt. At any time t, scheduling can anticipate the state of the system at time $t + \Delta t$, choose a solution that fits this state, and send it to the control system at $t + \Delta t$. But nothing prevents it against any unforeseen change in the system state between t and $t + \Delta t$.

30.2.5. Solution feasibility and quality versus solution adequacy

As finding feasible and high quality solutions may take a large amount of time, there is an immediate contradiction between solution feasibility and quality, on the one hand, and solution adequacy, on the other hand.

If it is difficult to find feasible and minimum quality solutions, that is solutions that satisfy all the physical and technological constraints and all the minimum user requirements, it may be impossible to obtain a given level of solution adequacy in a very dynamic context, without sufficient computing resources. That implies that it may be impossible to control the system correctly. It is certain that kind of situation should be detected either statically or dynamically.

Given a dynamic context, a given amount of computing resources, a given reasoning complexity, the higher the quality of the produced solutions, the longer the time to produce them, and the lower their adequacy.

We think that the presence of this contradiction is the main novelty of on-line scheduling with regard to off-line scheduling and that dealing with it is the main challenge of on-line scheduling.

30.3. Architectural choices

To go further, it may be useful to precise some usual architectural choices.

30.3.1. Usual architecture

Firstly, we make the assumption of a high level control system which is composed of two components : a *reactive* and a *deliberative* components.

The function of the reactive component is :

- to organize the normal life of the system to control;
- to react to events coming from the system itself, its environment, its users, and other systems;
- to call and to control the deliberative component, when necessary.

The function of the deliberative component is to make and deliver decisions (actions to perform, resource allocations, action schedules, starting dates . . .), when called by the reactive component.

To be more precise, it is the role of the reactive component to impose temporal constraints on the deliberative component. It is not its role to make decisions for which the deliberative component has been designed.

Figure 30.3 shows such an architecture.

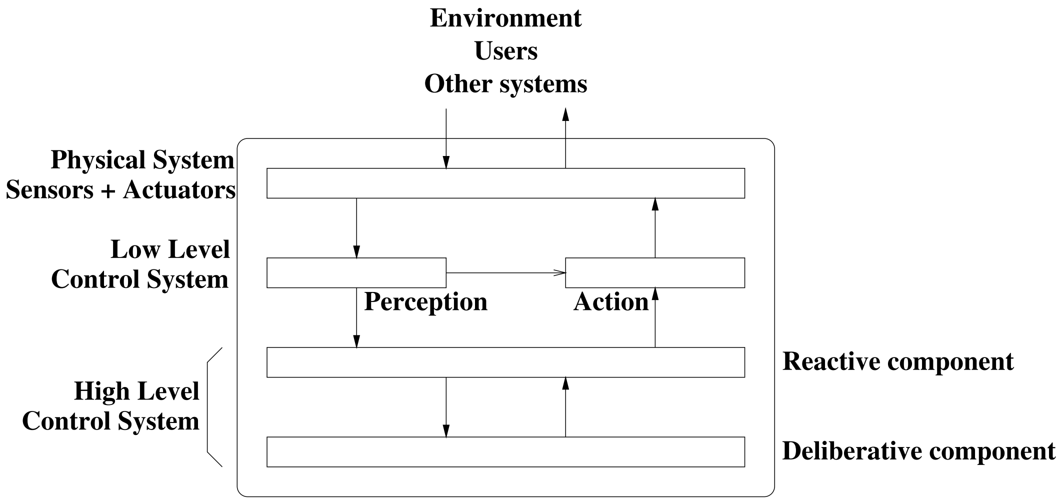

Figure 30.3.: Classical architecture of a on-line control system.

30.3.2. Scheduling horizons

When scheduling actions, it is necessary to specify on which temporal horizon one wants to schedule, that is which set of possible actions and which temporal windows for them one considers.

This horizon may be fixed either statically or dynamically. In the latter case, its length may be fixed, either by the reactive component when calling the deliberative component, or by the deliberative component itself.

It may be either simple or multiple. A multiple horizon is a set of nested horizons, with a scheduling granularity associated with each of them and increasing with size. For example, within an one hour horizon, dates are specified to within one minute. Within an one day horizon, they are specified to within one hour. And, within an one month horizon, they are specified to within one day. One speaks of a *multi-level scheduling*.

Moreover, a specific reasoning task may be associated with each horizon. For example, in production management, resource assignment is usually decided on long term horizons, action scheduling on short term horizons, and action starting dates on very short term horizons.

30.3.3. Scheduling frequencies

Once scheduling horizons have been fixed, it remains to decide at which frequency a new scheduling is called.

At worst, it has to be called just before the end $t + h$ of the current horizon, at $t + h - d$ to take into account a maximum time d for reasoning, and applied at $t + h$ (see figure 30.4). But, it may be decided to call it more frequently (see figure 30.5).

30.3.4. Interactions between scheduling levels

If we make the assumption that the deliberative level is composed of at least two levels of scheduling : a normal level and another one whose horizon is restricted to the next action, we have now to define

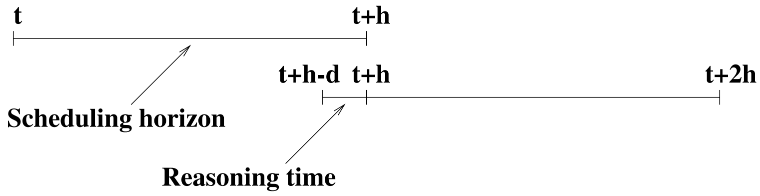

Figure 30.4.: New scheduling call at the end of the current horizon.

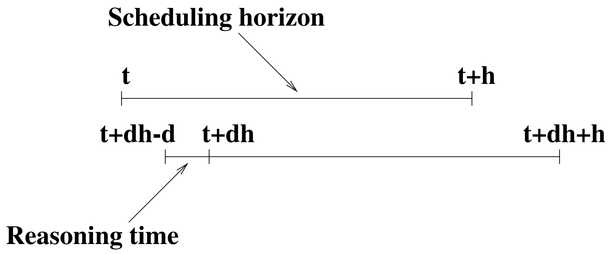

Figure 30.5.: New scheduling call before the end of the current horizon.

the relations between the different scheduling levels.

With classical off-line scheduling, these relations are clear : high levels make high level decisions that are imposed on lower levels.

With on-line scheduling, they are not so clear because of the contradiction between, on the one hand, the feasibility and the quality of the solutions and, on the other hand, their adequacy. Reasoning on larger horizons allows solution feasibility and quality to be increased. As with off-line scheduling, lower levels would have to follow the solutions produced by higher levels. But producing these solutions may take a lot of time and their adequacy may become low when important changes occur. So, lower levels may continue to use as advice the solutions produced by higher levels, but they must be able at any moment to make decisions by themselves, when the adequacy of these solutions has clearly become too bad.

30.4. Reasoning task features

30.4.1. Temporal constraints on reasoning

Temporal constraints on the reasoning of the deliberative component may be imposed either by the reactive component, or by the deliberative component itself.

In the first case, one usually distinguishes between *anytime* and *contractual* contexts :

- in anytime contexts, the time at which a solution will have to be provided is unknown; a solution may be asked for at any time;
- in contractual contexts, this time is known.

In the second case, it is the deliberative component which manages itself the tradeoff between reasoning time and solution quality, using a kind of meta-reasoning about its own reasoning capabilities :

on the one hand, going on reasoning may increase solution quality; on the other hand, as time passes, scheduling opportunities may disappear, changes may occur, and thus solution quality and adequacy may decrease.

30.4.2. Solution robustness

Scheduling on a given horizon implies to reason about three kinds of *uncertainty* :

- uncertainty about the events that may occur between the current reasoning time and the beginning of the considered horizon and between the beginning and the end of this horizon, coming from the system itself, the environment, the users, or other systems;
- uncertainty about the observations delivered by the perception tasks;
- uncertainty about the effects of the scheduled actions.

As far as knowledge is available concerning these events, observations, and effects, a scheduling that is as *robust* as possible facing them will be searched for.

30.4.3. Problem dynamicity

When regular schedulings are planned, as in the context of figure 30.5, or when important changes occur and impose to build a new schedule before the current one expires, one obtains what is often referred to as a *dynamic* problem, in the sense that the new problem to solve is different from the previous one (the scheduling horizon has been shifted, some of the previously scheduled actions have been executed, changes may have to be considered), but not so different.

In this case, it may be interesting to exploit the proximity between both problems to produce more quickly a feasible and high quality solution for the new problem, and to maintain *stability* between successive solutions.

30.4.4. Solution stability

This latter requirement is important in all the applications where changes in the schedule are costly or difficult to execute. It is the case when execution must have been prepared for a long time : human beings have to be informed and trained, components have to be designed, ordered, assembled or carried.

But it must be noted that a contradiction appears between solution quality and stability : for example, if we systematically favor solution stability, solution quality may decrease as changes occur and differences between the current problem and the first one regularly grow.

30.4.5. Distributed reasoning

When there are several reasoning and decision centers, the other decision centers can always be seen by a given decision center as a part of its environnement. But, if we want to introduce either cooperation, or conflict resolution, distributed decision making protocols have certainly to be proposed.

30.5. Conclusion

As a conclusion, we have to emphasize :

- that the main difference between off-line and on-line scheduling is certainly that a third criterion appears in addition to the classical feasibility and quality criteria for assessing a solution : its *adequacy* to the current situation ;
- that, because reasoning takes time, complete adequacy can never be guaranteed and a *contradiction* appears between feasibility and quality, on the one hand, and adequacy, on the other hand.

To sum up, one could say that *on-line scheduling is the art of reasoning in time about a complex object*, composed of a system state, a set of user objectives, a set of possible actions, a set of predictive models . . . , *which evolves with time*.

31. Current techniques and their limits

Introduction

This document follows the structure of the previous chapter 30. We outline current techniques to answer the numerous questions raised by each point and the difficulties which remain. A list of questions and answers deal with each point. The first section describes the architectural choices. A new subject on validation is added. The second section concerns the reasoning task features. All these questions are assembled in a table in the last section where a degree of maturity is associated to the current techniques for answering a question.

31.1. Architectural choices

31.1.1. System architecture

Q1: How to organize the various functionalities of on-line systems ?

Several ways of organizing a system have been proposed:

- the blackboard approach [21, 50] specifies a programming paradigm for on-line systems ; this approach involves maintaining a single consistency solution of reference and to analyse the solutions produced by several algorithms ;
- the classical dichotomy proposed by [46] between transformational and reactive systems ; an on-line scheduling system is composed of two layers: (a) a set of deliberative tasks that make and deliver decisions, and (b) a reactive part that handles inputs and controls the system ; layer (a) contains non-deterministic reasoning functions ; unlike layer (b) which contains deterministic algorithms.

Many theories and techniques are able to describe and implement the reactive layer, such as Automaton Theory, Decision Tree, Petri Nets, Rule-Based system [38] and Synchronous Programming [14, 59, 6, 3]. These techniques are mature and already used in industrial contexts. There are some tools for verifying formal properties of automata, like Model Checker [71, 66].

Q2: How to distinguish between reactive and deliberative parts ?

The distinction between deterministic and non-deterministic algorithms is difficult. In the case of a limited reasoning time, non-deterministic algorithms are very incomplete. They become deterministic, like greedy search algorithms. When temporal deadlines are deduced by the reactive layer when observing the external environment, these deductions may require algorithms with reasoning capacities.

Q3: How do we limit the size of pending problems ?

The reactive part generates a list of problems that must be solved by the deliberative part.

Q4: How to guarantee that temporal deadlines will be met ?

See section 31.2.1 “Temporal constraints on reasoning”.

31.1.2. Scheduling horizons

Q5: How to select the right scheduling horizon ?

The horizon can be imposed to some extent. For example, there can be a commit horizon where the schedule is fixed ahead of time for communications or other business reasons. Here, operational constraints imply that the scheduling horizon must be higher than the commit horizon. It was one of the arguments for preferring scheduling over reactive behaviour (cf. Reactivity versus scheduling).

The horizon is limited by the level of dynamicity and uncertainty. Beyond a certain limit, the schedule is not reliable. Its adequacy is zero.

Q6: What is the right trade-off between scheduling horizon and computation time ?

The horizon has a direct impact on solution quality and computation time. A wider horizon will take more time to compute and will produce better solution quality.

We can tune computation time by varying the horizon length in order to respect operational constraints like temporal deadlines. This tuning can be done dynamically either by the control part or by the reasoning part itself. The first case implies using either case-based reasoning [7, 69] or more complex anytime algorithm controls based on performance profiles [96]. The second case implies using contract algorithms which are able to adapt their complexity by varying their search parameters (like the horizon).

Q7: What is the increase in the cost of the schedule if we are very reactive ?

It is important to analyse the impact on the solution quality by varying the horizon length. A similar analysis is done by varying the degree of dynamicity. This analysis can be theoretical (see Online algorithms in section 31.1.5), or it can be pragmatic.

A pragmatic approach has been studied in the case of dynamic vehicle routing problems. [55] analyses the loss of solution quality as the degree of dynamicity increases.

Q8: How to combine different scheduling horizons with different scheduling granularities ?

Q9: How to select the right abstraction detail ?

Another aspect of scheduling horizon is the fact that you can represent the same problem by several modelisations with different horizons and different granularities. This multi-level scheduling approach could use abstraction techniques [56, 83] to reduce the complexity of high levels with large horizons. The interaction between scheduling levels is discussed in section 31.1.4.

31.1.3. Scheduling frequencies

Q10: How often do we reschedule ?

The scheduler should recompute a new schedule when new conditions invalidate the current schedule, or when the end of the current horizon is closed. An invalidating condition can be a new event (e.g. a new activity, a resource failure, a new criterion requested by the operational user) or a new estimation from sensors (e.g. an updated temporal constraint).

At worst, rescheduling has to be called just before the end of the current horizon. So the solution adequacy could become very bad if new conditions occur during the current schedule. At best, rescheduling is called at each new condition.

In practice, rescheduling can be either periodical (at each period, less or equal to the scheduling horizon) or event driven (at each new event). If you know exactly the current problem state and how it will change as time passes, until a new event occurs, then event driven rescheduling is the best strategy. Otherwise, if you cannot forecast problem changes as time passes, or you can only forecast them for a short period, especially if the current problem state is based on predictions, then the best strategy is to reschedule periodically.

Q11: What is the impact of scheduling frequency ?

The frequency of rescheduling has some important effects. The advantages of more reactive scheduling (with high frequency) are a better quality of service (user requests are quickly taken into account), a better robustness to uncertainty, the ability to continuously update and improve the schedule, the avoidance of getting stuck in local minima. The drawback is a shorter time for the solving method and, therefore, a possible decrease in terms of quality of the schedule.

Q12: How to filter problem changes ?

In the case of event driven rescheduling, a way of reducing the rescheduling frequency consists in being more phlegmatic about external events. It can be done by filtering and aggregating the events.

31.1.4. Interactions between scheduling levels

Q13: How do the dispatcher and the scheduler components cooperate ?

A classical on-line scheduling system is composed of a dispatcher and a scheduler. The dispatcher is an on-line allocator which contains and maintains, in a reactive way, a consistent solution used as a reference for a very short scheduling horizon. It executes the part of the current schedule corresponding to the current time. It reacts immediately to new external events by modifying the schedule (without optimisation). The scheduler is an optimisation function which produces a long-term schedule.

There may be interferences between the dispatcher and the scheduler, because both are modifying the schedule. For example, the dispatcher fixes a part of the schedule, by executing it as time passes.

Q14: What kind of interaction is there between the scheduling levels ?

For high scheduling levels, their scheduling horizon is wide so their computation time is long. Their solution quality should be good, but their solution adequacy may be bad. In order to obtain good solution adequacies, it is important that lower levels are able to take into account new problem changes directly. The schedules found by higher levels will therefore be used by lower levels as advice and not as orders. This implies that lower levels are able to repair an inconsistent schedule. See appendix A “Absence of solution”.

Potential techniques are: hierarchical planning, blackboard systems [21], mixed-initiative problem solving [79, 50].

31.1.5. Validation

Q15: How can an on-line scheduling system be validated ?

The correctness of a system can be checked by a formal specification and verification approach. Alternatively, it can be validated by a large number of experimental tests. In the second case, the system will run several randomly generated problems.

Q16: How to measure the quality of an on-line scheduling algorithm ?

The quality of the solutions produced by an on-line scheduling system is mainly assessed off-line. First, we can compare the solving method with benchmarks solved by other methods. Next, we can compare the on-line dynamic scheduler with a static scheduler: what is the cost of the schedule if we have all the data in advance compared to responding to problem changes as they come in ?

Online algorithms

An online algorithm [80, 4] receives the input incrementally, one piece at a time. In response to each input portion, the algorithm must generate output, without knowing future input. In a competitive analysis, an online algorithm A is compared to an optimal off-line algorithm OPT. An optimal off-line algorithm knows the entire input sequence in advance and can process it optimally. Given an input sequence I, let $CA(I)$ and $COPT(I)$ denote the costs incurred by A and OPT in processing I. Algorithm A is called c-competitive if there exists a constant a such that $CA(I) <= c*COPT(I)+a$, for all input sequences I. An analogous definition can be given for online maximisation problems. We note that a competitive algorithm must perform well on all input sequences.

The online algorithm approach compares two algorithms using either a minimum horizon (for algorithm A) or a maximum one (for algorithm OPT). This approach could be adapted to produce a theoretical comparison of the impact of different algorithms with different horizon lengths.

31.2. Reasoning task features

31.2.1. Temporal constraints on reasoning

Temporal constraints on reasoning are particular examples of operational constraints. Other examples are limited computation resources, limited memory spaces, continuous usage without failure, language restrictions, operating-system restrictions, etc. We have distinguished three kinds of temporal constraint.

- *Case 1.- Q17: How to obtain an algorithm which produces a good solution with respect to the deadline ?*

  In the first case, the temporal deadline is known, the system must deliver a solution before the end of a given time contract. The system must select and tune its solving method depending on the time contract. This can be done automatically or by some user choices.

  [58, 72, 81] dynamically adjust the depth of a look-ahead search, inserted in a Best First Search method. [18, 91] dynamically adjust the convergence speed parameter of the Iterative Approximating method. Contract search algorithms are being investigated at Thales LCR.

These algorithms adapt their search complexity dynamically with respect to the time contract, by using domain-dependent knowledge [43]. [5, 62] select the right algorithm off-line by doing experiments with domain-dependent benchmark problems. [13] automatically constructs hybrid algorithms using a learning procedure.

Q18: How to estimate the size of a search tree ?

Q19: What is the overhead of the estimation method compared to the search method ?

Q20: How to adapt the estimation method to optimisation ?

Q21: Can we trust the estimation result ? What is its variance ? Can we use a confidence interval ?

For tree search methods, it is important to estimate the number of nodes developed by the search. This estimation gives a way of choosing the right algorithm which will end just before the deadline.

An empirical estimation of the size of a search tree is given in Knuth's sampling method [57], by iteratively generating random paths in the search tree. Knuth's method has been improved in different ways by [68] and [17]. Initially applied to satisfaction problems, it has been adapted to optimisation problems in [61].

For local search methods, it may be easier to adjust their computation time to the given time contract. For example, the simulated annealing algorithm has two main tuning parameters: a temperature decrease scheme and a number of tries per level of temperature. Tuning these parameters will produce simulated annealing which will end just before the deadline, at a low-temperature level where better quality solutions may be found.

- *Case 2.- Q22: How to obtain an algorithm which performs well at any time ?*

  In the second case, the temporal deadline is unknown, the system can be interrupted and asked for a solution at any time. The algorithm should perform well on average during a specified temporal interval. [73] shows that a simple general construction can produce an interruptible algorithm from any given contract algorithm with only a small constant penalty.

  Iterative methods and local search methods are good candidates. For example, Limited Discrepancy Search [48, 47] increases the size of its exploration progressively at each iteration, from a greedy search to a complete search.

- *Case 3.- Q23: How to choose between continuing search efforts and delivering the current solution ?*

  In the third case, the temporal deadline is unknown, the system chooses the right moment to deliver its solution. This meta-reasoning capacity is based on the notion of utility [90, 36, 51]. The utility of a solution as time passes is a combination of a performance profile and a time cost function. [45] uses a utility function to monitor problem solving algorithms.

  Unanswered questions are:

  Q24: Can we have reliable performance profiles (with little variation) ?

  Q25: How to select the right performance profile for a particular problem instance at run-time ?

  Q26: How to build the time cost function ?

  (related to the notion of scheduling opportunities)

Q27: How to measure the quality of a solution at run-time (normalised quality) ?

(see the approach on anytime bounding of the optimum [25, 12, 24])

Q28: What is the minimal computation time required to get a first solution ?

This makes a distinction between a mandatory portion (to obtain a first solution) and an optional portion (to obtain an optimal solution), as proposed in the imprecise computation model [82]. In general, for satisfaction problems the time needed to find a feasible assignment increases exponentially with problem size. For scheduling problems however, it is often possible to postpone a task if there are inconsistencies with it. So, a greedy search algorithm will find a first schedule rapidly. See methods such as [2, 16].

Q29: What is an acceptable partial solution for my problem ?

If there is a temporal limit (e.g. a time window) for a conflicted task, this task will simply be discarded from the schedule. In this case, only a partial solution will be delivered.

Q30: What are the ways of producing a partial solution rapidly ?

One way consists in simplifying the problem. For example, some hard constraints are changed into soft constraints. Then a local search method will search for solutions which minimise the soft constraint violations (see A.2).

Q31: Have we forgotten to look at the problem? (complexity peak)

We are sometimes too pragmatic in our efforts to solve a problem and field a useful system. We often don't have the time to reflect and investigate just why the techniques used delivered a solution.

Recent research shows that there is a great similarity between the physical and computational world. NP-complete problems have much in common with physical systems with many states (such as ferro-magnetic structures, spin glasses, and the like). This has brought together statistical physicists, mathematicians and computer scientists in their quest for a better understanding of the nature of NP-completeness and just what makes a problem hard.

Current state of the art (see the recent Topical Conference on NP-hardness and Phase Transitions at `http://www.cs.strath.ac.uk/~pat/cv/triesteProgramme`) addresses only academic problems such as satisfiability, graph colouring, number partition,...

Yet this research has identified startling phenomena, such as the computational complexity peak coinciding with the solubility phase transition, theories to predict this, and the phenomenon of emergence of "backbones" within problems during search.

What has not yet been looked at is the occurrence of these phenomena in real problems, and how this will influence our choice of solution technique.

31.2.2. Problem dynamicity

Q32: How to reuse knowledge about the past in order to solve the current problem more quickly ?

Theoretically, problem changes can be viewed as a set of constraint addings and removals (unary constraints included). This is the core of Dynamic Constraint Satisfaction Problems (DCSPs) [28]. Both constraint maintenance systems and solution reuse-based algorithms are affected by constraint addings and removals. Removing constraints will affect the constraint maintenance system, but not solution reuse (previous solutions remain valid but can be sub-optimal). Adding new constraints implies classical propagation for the constraint maintenance system, and previous solution may become inconsistent.

Note that it can be hard to assess problem changes. For example, a radar produces a list of tracks at each cycle. It is difficult to match tracks from one cycle to another.

Q33: How to build dynamic constraint maintenance systems ?

The problem is to dynamically maintain a certain local consistency level (most of the time, the arc-consistency level) inside the constraint store. When some new problem changes occur, the system must be restored to a consistent state, independent of the deleting constraints, and then propagate the effects of the new constraints. The restoring process is called constraint retraction.

Two different approaches implement constraint retraction:

- "justification-based method that would compute and explicitly keep the dependencies between constraints resulting from consistency checking and domain reduction in a TMS [29] or ATMS style [26]; this has been developed in the frameworks of Constraint Satisfaction Problems (CSPs) [67, 27, 53], of temporal networks [15] and of Constraint Logic Programming (CLP) [20, 32]; the drawback of these methods is that they slow down the constraint maintenance system while computing the dependencies and are therefore costly when no retraction is used" [40];
- constraint graph-based method that determines the relevant subgraph to modify upon constraint retraction [65, 41]; no dependency computed during propagation; this has been developed in clp(FD) [19, 40, 41].

Q34: How to implement constraint retraction for global constraints ?

In both approaches, it is still a challenge to depropagate a global constraint incrementally after having enlarged the domain of some variables.

Constraint retraction is also very useful for:

- tree-search methods using intelligent search (for keeping part of an assignment during back-jumping or dynamic backtracking);
- local search methods (for moving from one complete assignment to another);
- conflict explanations (for extracting conflicts).

Constraint retraction does not exist inside current commercial constraint programming tools.

Q35: How to reuse solution and reasoning for efficiency purposes ?

Solution and reasoning reuse has been studied for DCSPs. The existing approaches can be classified into three groups [89]:

- heuristic approaches, which use any previously consistent assignment (complete or not) as a value heuristic [49];

- local repair approaches, which start from any previously consistent assignment (complete or not) and repair it, using a sequence of local modifications [63, 88];
- constraint recording approaches, which record any kind of constraint (called a no-good) which can be deduced in the framework of a CSP and its justification, in order to reuse it in the framework of any new CSP which includes this justification [78].

Solution and reasoning reuse can be combined [49, 87]. [89] compares six different complete search algorithms for dynamic CSPs. The dynamic version of Dynamic Backtracking has good overall behaviour.

In case of optimisation problems, the notion of no-good, initially used in satisfaction problems, needs to be extended. See [23, 22] for the Valued Constraint Satisfaction Problem (VCSP) framework.

In satisfaction or optimisation contexts, some difficulties still remain:

Q36: How to avoid memory overflow with no-good recording methods ?

Q37: How to apply no-good recording methods within incomplete tree search methods ?

Other learning approaches are machine learning [1] and case-based reasoning. Case-based reasoning helps on-line scheduling by using previous experiences (reasoning decisions) and by combining solutions (re-using several solved smaller CSPs) [69]. The retrieval of cases is based on the definition of context that are characterised by constraints and optimality criteria [7].

31.2.3. Solution robustness

Q38: How to produce a robust solution to face uncertainty about events, observations and effects ?

A practical approach is to compute flexible solutions. A solution is said to be flexible if it is easy to repair in case of problem changes. A simple way of getting flexible schedules consists in inserting slack between tasks.

Instead of computing a schedule with all the variables assigned, more flexibility can be gained by sequencing all the tasks for each resource without assigning task start and end times. This flexible schedule is conflict free: all the disjunctive constraints are removed by adding some precedence constraints. We are left with a precedence constraint graph where it is easy to change a start or end time and to recompute a complete solution.

Moreover, the notion of temporal flexibility may be used as a heuristic for making flexible schedules [16].

Q39: What is the right tradeoff between robustness and quality ?

Robust schedules can be sub-optimal. Robustness and quality are two criteria which have to be optimised.

Q40: Do we have a model of uncertainty about events, observations and effects ?

In the Constraint Satisfaction Problem framework, different models and associated algorithms have been studied:

- Fuzzy CSPs [30, 31], where constraints define a certainty level for each forbidden tuple; a minimisation algorithm looks for solutions which minimise the maximum level of forbidden assignments;
- Possibilistic CSPs [76], where a certainty level is associated to each constraint; a minimisation algorithm looks for solutions which minimise the maximum level of unsatisfied constraints;
- Probabilistic CSPs [33], where a probability is associated to each constraint; a minimisation algorithm looks for the most promising solutions;
- Mixed CSPs [34, 35], where a solution is valid whatever the new state of the world;
- Recurrent Dynamic CSPs [92, 93], where heuristic repair methods are experimented for finding solutions that are more likely to remain valid after changes that temporarily alter the set of valid assignments.

Models of uncertainty are also studied in mathematical research, such as Decision Theory [54], Utility Theory [90, 36] and Game Theory [90].

[85, 42] distinguishes two probabilistic models:

- the probabilistic information about the occurrence of an unforeseen event is available and the probability distribution is stationary, i.e. non-dependent on time; this is the case for the above CSP frameworks;
- the probability distribution is non-stationary; only approximations can tackle this case; such approximations consist in a reduction to a stationary case, the use of randomly generated probability distributions with best and worst case analysis, or a transformation into forecast data by detecting trends and seasonal variations in past data.

Risk management deals with the uncertainty of the occurrence of an unforeseen event and the impact on the decision making process if the unforeseen event occurs [85].

Q41: How to extend Markov Decision Process to problems with constraints ?

Markov Decision Process (MDP) [70] proposes a rational way of dealing with uncertainty. It defines an expected global gain criterion by aggregating gains for doing actions (e.g. scheduling a task) and realisation probabilities (e.g. probability that a task will be a success). A Dynamic Programming algorithm is used in the MDP framework to compute optimal policies. Probabilities are learnt, in fact approximated, either off-line from simulations, or on-line from the observation of the system behaviour [84].

31.2.4. Solution stability

Q42: Is solution stability really mandatory ?

If there are people involved in executing or controlling the schedule process, stability of the successive solutions may be required. Instead of producing stable schedules, we may apply a least commitment strategy, e.g. do nothing until it is mandatory. It is not always necessary to give a long-term schedule to operational users. This is the case for BT technicians. Only a very short-term schedule is transmitted to these operational users, e.g. giving them the next customer to visit. On the other hand, operational managers need stable schedules.

Note that solution robustness is a more proactive approach than solution stability [92].

Q43: What is the stability criterion to minimise ?

A distance or metric of changes has to be established in order to compare two solutions. Possible metrics are: (a) a weighted sum of time shifts, (b) a count of activities shifted in time, (c) a sum of changes of positions of activities on a resource. Note that cases (a) and (b) imply that exact start and end times of activities are computed, not only orders between activities on resources.

Q44: How do we combine the stability criterion with the optimisation criterion ?

A simple method is to set an upper limit for one criterion and to minimise the other criterion. If the optimisation criterion is the makespan, minimising solution changes will also minimise the makespan. In [52], the upper limit of the makespan is the previous makespan value plus the duration of problem changes (e.g. duration of machine break-down, amount of increase of release dates). Otherwise, optimisation should minimise both criteria. See multi-criteria optimisation techniques [74, 37].

Q45: What are the possible techniques to get stable solutions ?

In the framework of Constraint Satisfaction Problems, solving methods based on solution reuse can produce stable solutions [49, 63, 88].

In Constraint Programming, [52] describes a method called preference-based search which focuses search on parts of the search space with smaller changes. This yields a good strategy in a limited time context. The method combines heuristics for value selection and a refutation mechanism for cutting bad nodes.

[75] uses Linear Programming techniques to minimise solution changes expressed by the metric (a). A Linear Programming solver is applied to a sub-set of constraints having total unimodularity which produces super-optimal complete assignments in polynomial time, with integer values for variables. These assignments, called probes, drive and scope backtrack search on resource constraints in partially consistent assignments. If a probe satisfies all resource constraints, then this is a new and better solution for a branch and bound search.

31.2.5. Distributed reasoning

Sometimes, the problem or its solving method are distributed. The reason may be geographical (different places), organisational (different companies) or simply because it is easier to solve (problem decomposition).

Q46: How to manage conflict resolution between several reasoning and decision centres ?

Conflicts between different on-line control system units are managed by a high-level control of these systems. We can distinguish between two kinds of cooperation:

- The first one causes conflicts between reactive components. At this level conflicts are about the goal of the system. Resolving them can change the way the reactive parts continue to manage the system. As an example of conflict, two weapons systems can plan to hit the same missile.

- The second one brings conflicts directly to deliberative components. The highest levels can detect conflicts of goals and even conflicts occurring in sub-goal calculations. For example, for hitting distinct targets, two weapons systems can try to use the same resource.

Several protocols have been designed to solve distributed problems. Some of them use an extension of the constraint satisfaction paradigm. This extension unifies search spaces by exchanging information on their own problem resolution. This information allows detection of conflicts which are usually solved according to a partial order between agents [95, 11, 44].

There are difficulties such as:

Q47: How can possible conflicting local goals be combined in an overall objective ?

Q48: Can we trust information coming from different companies (the veracity problem) ?

The distributivity can force a reactive approach (without optimisation), in order to reduce communications.

Multiagent scheduling

In a multiagent system, the solution of an on-line scheduling problem emerges as a consequence of the interaction among a set of agents. Two main approaches can be followed: first, splitting the problem into several agents that locally solve the problem and globally negotiate the integrated solution; and second, assigning a resource (or type of resource) to an agent which is in charge of providing the service to other agents [10, 60, 94, 64].

31.3. Basic roadmap

All the questions of both sections 31.1 and 31.2 are assembled in two tables, one about architectural choices and the other about reasoning task features. A degree of maturity is associated to the current techniques for answering a question. This degree is related to a level of research and development efforts for obtaining a mature technique. We distinguish between three coarse levels of effort :

- short-term means an effort of 1 or 2 years ;
- mid-term means an effort of 2 to 5 years ;
- long-term means an effort of 5 to 10 years.

Ref.	**Questions about Architectural choices**	Level **Short**	**Mid**	**Long**
	System architecture			
1, 31.1.1	How to organise the various functionalities of on-line systems ?	*		
2, 31.1.1	How to distinguish between reactive and deliberative parts ?			*
3, 31.1.1	How do we limit the size of the pending problems ?		*	
4, 31.1.1	How to guarantee that temporal deadlines will be meet ?		*	
	Scheduling horizons			
5, 31.1.2	How to select the right scheduling horizon ?		*	
6, 31.1.2	What is the right trade-off between scheduling horizon and computation time ?			*
7, 31.1.2	What is the increase in the cost of the schedule if we are very reactive ?	*		
8, 31.1.2	How to combine scheduling horizons with scheduling granularities ?			*
9, 31.1.2	How to select the right abstraction detail ?			*
	Scheduling frequencies			
10, 31.1.3	How often do we reschedule ?		*	
11, 31.1.3	What is the impact of scheduling frequency ?		*	
12, 31.1.3	How to filter problem changes ?	*		
	Interactions between scheduling levels			
13, 31.1.4	How do the dispatcher and the scheduler components cooperate ?	*		
14, 31.1.4	What kind of interaction is there between the scheduling levels ?		*	
	Validation			
15, 31.1.5	How can an on-line scheduling system be validated ?			*
16, 31.1.5	How to measure the quality of an on-line scheduling algorithm ?		*	

Ref.	Questions about Reasoning task features	Level: Short	Mid	Long
	Temporal constraints on reasoning			
17, 31.2.1	How to obtain an algorithm which produces a good solution wrt. the deadline ?		*	
18, 31.2.1	How to estimate the size of a search tree ?	*		
19, 31.2.1	What is the overhead of the estimation method compared to the search method ?	*		
20, 31.2.1	How to adapt estimation method to optimisation ?		*	
21, 31.2.1	Can we trust the estimation result ? ...		*	
22, 31.2.1	How to obtain an algorithm which performs well at any time ?			*
23, 31.2.1	How to choose between continuing search efforts and delivering the current solution ?			*
24, 31.2.1	Can we have reliable performance profiles (with small variation) ?			*
25, 31.2.1	How to select the right performance profile for a particular problem instance ?			*
26, 31.2.1	How to build the time cost function ?		*	
27, 31.2.1	How to measure the quality of a solution at run-time (normalised quality) ?		*	
28, 31.2.1	What is the minimal computation time required to get a first solution ?	*		
29, 31.2.1	What is an acceptable partial solution for my problem ?	*		
30, 31.2.1	What are the ways of producing a partial solution rapidly ?	*		
31, 31.2.1	Have we forgotten to look at the problem? (complexity peak)			*
	Problem dynamicity			
32, 31.2.2	How to reuse knowledge about the past in order to solve the problem more quickly ?		*	
33, 31.2.2	How to build dynamic constraint maintenance systems ?	*		
34, 31.2.2	How to implement constraint retraction for global constraints ?		*	
35, 31.2.2	How to reuse solution and reasoning for efficiency purpose ?		*	
36, 31.2.2	How to avoid memory overflow with nogood recording methods ?	*		
37, 31.2.2	How to apply nogood recording methods within incomplete tree search methods ?			*
	Solution robustness			
38, 31.2.3	How to produce a robust solution to face uncertainty ?			*
39, 31.2.3	What is the right tradeoff between robustness and quality ?		*	
40, 31.2.3	Do we have a model of uncertainty about events, observations and effects ?		*	
41, 31.2.3	How to extend Markov Decision Process to problems with constraints ?		*	
	Solution stability			
42, 31.2.4	Is solution stability really mandatory ?	*		
43, 31.2.4	What is the stability criterion to minimise ?	*		
44, 31.2.4	How do we combine the stability criterion with the optimisation criterion ?		*	
45, 31.2.4	What are the possible techniques to get stable solutions ?	*		
	Distributed reasoning			
46, 31.2.5	How to manage conflict resolution between several reasoning and decision centers ?		*	
47, 31.2.5	How can possible conflicting local goals be combined in an overall objective ?		*	
48, 31.2.5	Can we trust information coming from different companies (the veracity problem) ?			*

A. Absence of solution

A.1. Analysis of inconsistencies

On-line systems need tools for solution analysis, in order to detect inconsistencies in the current schedule or parts of the schedule that could be improved.

The system should explain what the conflicts are (e.g. a subset of tasks that cannot be scheduled within their time windows). A conflict can be represented as an assignment to a set of variables that either violates a constraint or cannot be extended to a solution. Such an assignment is called a no-good.

The no-goods can be used by a repair function. This function looks for a solution by relaxing the problem (e.g. adding resources, enlarging time windows, deleting some tasks). The relaxation is applied to the constraints violated by the no-goods.

Potential techniques are: conflict explanation [86] and relaxation proposal [11].

A.2. Over-constrained problems

When no solution exits, one should try to relax the problem by removing or weakening some constraints. The process of finding a good relaxation can be transformed into optimisation problem solving, if the user associates cost values to constraints.

A constraint optimisation algorithm tries to find a solution which minimises the cost of the unsatisfied constraints. There are several ways of combining the cost of unsatisfied constraints, depending on the agregation operator.

Potential techniques are: Partial Constraint Satisfaction Problem [39], Valued Constraint Satisfaction Problem (VCSP) [77], Semiring Constraint Satisfaction Problem (SCSP) [8, 9], Best-First Search with constraint retraction [53].

B. On-Line Scheduling TCU members

- - Node: Thales LCR
  - contact: Simon de Givry
  - participants: Simon de Givry, Jean Jourdan, Juliette Mattioli
  - e-mails: {simon.degivry,jean.jourdan,juliette.mattioli}@thalesgroup.com
  - web page: http://www.lcr.thomson-csf.com/projects/planet/ols-tcu.html
  - related web pages: http://planet.dfki.de/
- - Node: University of Glasgow, University of Strathclyde
  - contact: Patrick Prosser
  - participants: Patrick Prosser, Toby Walsh, Ian P. Gent, Iain Buchanan, Kostas Stergiou
  - e-mails: pat@dcs.gla.ac.uk,{tw,ipg,iain,ks}@cs.strath.ac.uk
  - web page: http://www.cs.strath.ac.uk/˜apes/
  - related web pages: http://www.cs.strath.ac.uk/biography/pat/
- - Node: BT Laboratories
  - contact: Paul Kearney
  - participants: Paul Kearney, David Lesaint, Jon Spragg, Christos Voudouris, Raphael Dorne
  - e-mails: {paul.kearney,lesaind,spraggj,chrisv,dorner}@info.bt.co.uk
  - web page: http://www.labs.bt.com/projects/ibsr/index.htm
  - related web pages: http://innovate.bt.com/showcase/work_scheduling/index.htm
- - Node: ONERA-CERT
  - contact: Gerard Verfaillie
  - participants: Gerard Verfaillie, Lionel Lobjois
  - e-mails: {verfaillie,lobjois}@cert.fr
  - web page: http://www.cert.fr/fr/dcsd/CD/CDPUB/THEMES/oc-english.html
  - related web pages:

- - Node: IP-CNR
  - contact: Angelo Oddi and Amedeo Cesta
  - participants: Amedeo Cesta, Angelo Oddi, Angelo Susi
  - e-mails: `{amedeo,oddi,susi}@pscs2.irmkant.rm.cnr.it`
  - web page: `http://pscs2.irmkant.rm.cnr.it/users/amedeo/www/home.html`
  - related web pages: `http://www.deis.unibo.it/Events/cp-ai-or99.html`
- - Node: SINTEF Applied Mathematics
  - contact: Geir Hasle
  - participants: Dag Kjenstad, Mouhssine Bouzoubaa
  - e-mails: `{dag.kjenstad,Mouhssine.Bouzoubaa}@math.sintef.no`
  - web page:
  - related web pages:
- - Node: Salford University
  - contact: Ruth Aylett
  - participant: Gary Petley
  - e-mail: `itgjp@angmar.iti.salford.ac.uk`
  - web page:
  - related web pages:
- - Node: Universidad Politecnica de Valencia
  - contact: Federico Barber
  - participants: Federico Barber, Antonio Garrido Tejero, Laura Sebastia
  - e-mail: `fbarber@dsic.upv.es`, `agarridot@dsic.upv.es`
  - web page: `http://www.dsic.upv.es/users/ia/gps/index.html`
  - related web pages:
- - Node: European Space Agency (ESA / ESTEC)
  - contact: Eric Bornschlegl
  - e-mail: `ebornsch@estec.esa.nl`
  - web page:
  - related web pages:
- - Node: University of Brescia
  - contact: Alfonso Gerevini
  - e-mail: `gerevini@ing.unibs.it`
  - web page:
  - related web pages:

- - Node: IC-Parc
  - contact: Barry Richards
  - participant: Hani El-Sakkout
  - e-mail: Hani.El-Sakkout@parc-technologies.com, hhe@icparc.ic.ac.uk
  - web page: http://www-icparc.doc.ic.ac.uk
  - related web pages:
- - Node: Universitat Rovira i Virgili
  - contact: Beatriz López
  - e-mail: blopez@etse.urv.es
  - web page: http://www.etse.urv.es/~blopez
  - related web pages: http://www.etse.urv.es/recerca/banzai/, http://www.etse.urv.es/recerca/rivi/

Bibliography

[1] *Machine Learning*. McGraw-Hill, 1997.

[2] Oddi A. and Smith S.F. Stochastic procedures for generating feasible schedules. In *Proc. of AAAI-97*, pages 27–31, Providence, RI, 1997.

[3] Martin Adelantado, Frédéric Boniol, and Simon de Givry. Saturne: a reactive - anytime programming model for intelligent embedded real-time systems. In *3rd IEEE Workshop on Parallel and Distributed Real-Time Systems*, Santa Barbara, California, April 24-26 1995.

[4] S. Albers and S. Leonardi. Online algorithms. Working document, accepted for publication on the electronic version of "ACM Computer Surveys", 1999.

[5] J. Allen and S. Minton. Selecting the right heuristic algorithm: Runtime performance predictors. In *Proceedings of the Canadian AI Conference*, 1996.

[6] G. Berry and G. Gonthier. The Esterel synchronous programming language: design, semantics, implementation. *Science of Computer Programming*, 19(2):87–152, 1992.

[7] Tarner Bilgic and Mark. S. Fox. *Artificial Intelligence in Design*, chapter Constraint-Based Retrieval of Engineering Design Cases: Context as constraints, pages 269–288. Kluwer Academic Publishers, 1996.

[8] Stefano Bistarelli, Ugo Montanari, and Francesca Rossi. Constraint solving over semirings. In *Proc. of IJCAI-95*, pages 624–630, Montréal, Canada, 1995.

[9] Stefano Bistarelli, Ugo Montanari, Francesca Rossi, Thomas Schiex, Gérard Verfaillie, and Hélène Fargier. Semiring-Based CSPs and Valued CSPs: Frameworks, Properties and Comparison. *Constraints*, 4(3):199–240, 1999.

[10] C. Boutilier, M. Goldszmidt, and B. Sabata. Sequential auctions for the allocation of resources with complementarities. In *Proc. of IJCAI-99*, pages 527–534, Stockholm, Sweden, 1999.

[11] P. Burke and P. Prosser. A distributed asynchronous system for predictive and reactive scheduling. *The International Journal for Artificial Intelligence in Engineering*, 6(3):106–124, 1991.

[12] Bertrand Cabon, Simon de Givry, and Gérard Verfaillie. Anytime Lower Bounds for Constraint Optimization Problems. In *Proc. of CP-98*, pages 117–131, Pisa, Italy, October 26-30 1998.

[13] Y. Caseau, F. Laburthe, and G. Silverstein. A meta-heuristic factory for vehicle routing problems, meta-programming for meta-heuristics. In *Proc. of CP-99*, pages 144–158, Alexandria, Virginia, 1999.

[14] P. Caspi, N. Halbwachs, P. Pilaud, and P. Raymond. The synchronous dataflow programming language Lustre. *Proceedings of IEEE, Another Look at Real-time programming*, 79(9):1305–1319, September 1991.

[15] R. Cervoni, A. Cesta, and A. Oddi. Managing dynamic temporal constraint networks. In *Proceedings of the Second Int. Conf. on Artificial Intelligence Planning Systems (AIPS 94)*, pages 13–18, Chicago, IL, 1994.

[16] A. Cesta, A. Oddi, and S.F. Smith. An iterative sampling procedure for resource constrained project scheduling with time windows. In *Proc. of IJCAI-99*, pages 1022–1029, Stockholm, Sweden, 1999.

[17] P. Chen. Heuristic Sampling : a Method for Predicting the Performance of Tree Searching Programs. *SIAM Journal on Computing*, 21(2):295–315, 1992.

[18] Lon-Chan Chu. *Algorithms for Combinatorial Optimization in Real Time and their Automated Refinements by Genetics-Based Learning*. PhD thesis, Department of Electrical and Computer Engineering, University of Illinois, Urbana-Champaign, IL, 1994.

[19] P. Codognet and D. Diaz. Compiling constraints in clp(fd). *Journal of Logic Programming*, 1996.

[20] P. Codognet, F. Fages, and T. Sola. *CLP: Selected Research*, chapter A meta-level compiler for CLP(FD) and its combination with IB. MIT Press, 1993.

[21] D.D. Corkill. Blackboard systems. *AI Expert*, 6(9):40–47, 1991.

[22] Pierre Dago. *Extension d'algorithmes dans le cadre des Problèmes de Satisfaction de Contraintes valués. Application aux systèmes satellitaires d'observation de la Terre*. PhD thesis, ENSAE, Toulouse, France, 1997.

[23] Pierre Dago and Gérard Verfaillie. Nogood Recording for Valued Constraint Satisfaction Problems. In *Proc. of ICTAI-96*, Toulouse, France, 1996.

[24] Simon de Givry. *Algorithmes d'optimisation sous contraintes étudiés dans un cadre temps réel*. PhD thesis, École Nationale Supérieure de l'Aéronautique et de l'Espace, Toulouse, France, 1998.

[25] Simon de Givry, Gérard Verfaillie, and Thomas Schiex. Bounding the Optimum of Constraint Optimization Problems. In *Proc. of CP-97*, pages 405–419, Schloss Hagenberg, Austria, October 29 - November 1 1997.

[26] J. de Kleer. An assumption-based truth maintenance system. *Artificial Intelligence*, 28, 1986.

[27] Romuald Debruyne. Dnac-6. Technical Report 94-054, LIRMM, Montpellier, 1994.

[28] R. Dechter and A. Dechter. Belief Maintenance in Dynamic Constraint Networks. In *Proc. of AAAI-88*, pages 37–42, St. Paul, MN, 1988.

[29] J. Doyle. A truth maintenance system. *Artificial Intelligence*, 12, 1979.

[30] D. Dubois, H. Fargier, and H. Prade. The Calculus of Fuzzy Restrictions as a Basis for Flexible Constraint Satisfaction. In *Proc. of the 2nd IEEE Conference on Fuzzy Sets*, pages 1131–1136, San Francisco, CA, 1993.

[31] D. Dubois, H. Fargier, and H. Prade. *Fuzzy Sets, Neural Networks and Soft Computing*, chapter Propagation and satisfaction of fuzzy constraints, pages 166–187. Kluwer Acad., 1993.

[32] F. Fages, J. Fowler, and T. Sola. A reactive clp scheme. In *Proc. of ICLP'95*, Tokyo, 1995.

[33] H. Fargier and J. Lang. Uncertainty in constraint satisfaction problems: a probabilistic approach. In *Symbolic and Quantitative Approaches to Reasoning and Uncertainty, EC-SQARU'93*, pages 97–104, 1993.

[34] Hélène Fargier, Jérôme Lang, Roger Martin-Clouaire, and Thomas Schiex. Mixed Constraint Satisfaction : a Framework for Decision Problems under Uncertainty. In *Proc. of UAI95 (Uncertainty in Artificial Intelligence)*, Montréal, Canada, 1995.

[35] Hélène Fargier, Jérôme Lang, and Thomas Schiex. Mixed constraint satisfaction : a framework for decision problems under incomplete knowledge. In *Proc. of AAAI-96*, pages 175–180, Portland, OR, 1996.

[36] P. Fishburn. Subjective expected utility: A review of normative theories. *Theory and Decision*, 13:139–199, 1981.

[37] J.C. Fodor and M. Roubens. *Fuzzy Preference Modelling and Multi-Criteria Decision Aid*. Kluwer Academic Publisher, 1994.

[38] Charles L. Forgy. Rete: A fast algorithm for the many pattern/many object pattern match problem. *Artificial Intelligence*, 19(1):17–37, 1982.

[39] Eugene C. Freuder and Richard J. Wallace. Partial constraint satisfaction. *Artificial Intelligence*, 58:21–70, 1992.

[40] Y. Georget, P. Codognet, and F. Rossi. Implementing constraint retraction for finite domains. In *Proc. of ASIAN'97*, Katmandu, Nepal, 1997.

[41] Y. Georget, P. Codognet, and F. Rossi. Constraint retraction in clp(fd): Formal framework and performance results. *Constraints*, 4(1):5–42, 1999.

[42] Carmen Gervet. Ongoing research in chic-2: Decision making under uncertainty. In *CP'97 ERCIM/Compulog workshop on constraints*, Schloss Hagenberg, Austria, 1997.

[43] Simon de Givry, Pierre Savéant, and Jean Jourdan. Optimisation combinatoire en temps limité : Depth first branch and bound adaptatif. In *Proc. of JFPLC-99*, Lyon, France, 1999.

[44] Y. Hamadi, C. Bessière, and J. Quinqueton. Backtracking in distributed constraint networks. In H. Prade, editor, *ECAI*, pages 219–223, Chichester, Aug 1998. John Wiley and Sons.

[45] Eric A. Hansen and Shlomo Zilberstein. Monitoring the progress of anytime problem-solving. In *Proc. of AAAI-96*, Portland, OR, 1996.

[46] 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.

[47] William D. Harvey. *NONSYSTEMATIC BACKTRACKING SEARCH*. PhD thesis, Stanford University, March 1995.

[48] William D. Harvey and Matthew L. Ginsberg. Limited discrepancy search. In *Proc. of IJCAI-95*, pages 607–613, Montréal, Canada, 1995.

[49] P. Van Hentenryck and T.L. Provost. Incremental search in constraint logic programming. *New Generation Computing*, 9:257–275, 1991.

[50] D.W. Hildum, N.M. Sadeh, T.J. Laliberty, J. McA'Nulty, S.F. Smith, and D. Kjenstad. A blackboard-based architecture for supporting mixed-initiative management of integrated problem-solving tasks. In *Proceedings of the 9-th Conference Innovative Applications of Artificial Intelligence (IAAI-97)*, 1997.

[51] Eric Horvitz. *Computation and action under bounded resources*. PhD thesis, Department of Computer Science and Medecine, Stanford University, California, 1990.

[52] U. Junker and W. Nuijten. Preference-based search for minimizing changes in rescheduling problems. In *IJCAI'99 workshop on Scheduling and Planning meet Real-time Monitoring in a Dynamic and Uncertain World*, pages 39–45, Stockholm, Sweden, 1999.

[53] Narendra Jussien. *Relaxation de contraintes pour les problèmes dynamiques*. PhD thesis, Université de Rennes 1, Ecole des Mines de Nantes, Octobre 1997.

[54] R.L. Keeney and H. Raiffa. *Decision with Multiple Objectives*. Wiley, New York, 1976.

[55] P. Kilby, P. Prosser, and P. Shaw. Dynamic vrps: A study of scenarios. Technical report, University of Strathclyde, 1998.

[56] Craig A. Knoblock. Learning abstraction hierarchies for problem solving. In *Proc. of AAAI-90*, volume 2, pages 923–928, Boston, MA, 1990.

[57] D. Knuth. Estimating the Efficiency of Backtrack Programs. *Mathematics of Computation*, 29(129):121–136, 1975.

[58] Richard E. Korf. Real-time heuristic search. *Artificial Intelligence*, 42:189–211, 1990.

[59] P. Le Guernic, T. Gauthier, M. Le Borgne, and C. Le Maire. Programming real time application with Signal. *Proceedings of IEEE, Another Look at Real-time programming*, 79(9):1321–1336, September 1991.

[60] J. Liu. Collective problem solving through coordination in a society of reactive agents. Technical report, The Robotics Institute, Carnegie Mellon University, 1994.

[61] L. Lobjois and M. Lemaître. Branch and Bound Algorithm Selection by Performance Prediction. In *Proc. of AAAI-98*, pages 353–358, Madison, WI, 1998.

[62] S. Minton. Automatically configuring constraint satisfaction programs : A case study. *Constraints*, 1(1):7–43, 1996.

[63] S. Minton, M. Johnston, A. Philips, and P. Laird. Minimizing Conflicts: a Heuristic Repair Method for Constraint Satisfaction and Scheduling Problems. *Artificial Intelligence*, 58:160–205, 1992.

[64] M.X.Weng and H.Ren. A review of multi-agent scheduling. Technical report, Department of Industrial and Management Systems Engineering, University of South Florida.

[65] B. Neveu and P. Berlandier. Maintaining arc consistency through constraint retraction. In IEEE Press, editor, *Proc. of TAI'94*, 1994.

[66] F. Pagani, C. Seguin, P. Siron, and V. Wiels. Verification experiments on a large fault-tolerant distributed system. In *Models and Proofs, 2nd AMAST Workshop on Real-Time Systems*, Bordeaux, France, June 14-16 1995.

[67] P. Prosser, C. Conway, and C. Muller. A constraint maintenance system for the distributed allocation problem. *Intelligent Systems Engineering*, 1992.

[68] P. Purdom. Tree Size by Partial Backtracking. *SIAM Journal on Computing*, 7(4):481–491, 1978.

[69] Lisa Purvis. Dynamic constraint satisfaction using case-based reasoning techniques. In *Constraint Programming'97 workshop on Dynamic Constraint Satisfaction*, 1997.

[70] M. Puterman. *Markov Decision Processes, Discrete Stochastic Dynamic Programming*. John Wiley and Sons, 1994.

[71] J.P. Queille and J. Sifakis. Specifications and verification of concurrent systems in cesar. In *International Symposium on Programming,*, LNCS 137, pages 337–351, 1982.

[72] Stuart Russel and Eric Wefald. *Do the RIGHT Thing. Studing in limited rationality*, chapter 5, pages 119–157. MIT Press, 1991.

[73] S.J. Russell and S. Zilberstein. Composing real-time systems. In *Proc. of IJCAI-91*, pages 212–217, Sidney, Australia, 1991.

[74] M. Sakawa. *Fuzzy Sets and Interactive Multiobjective Optimization*. Plenum Press, 1993.

[75] H. El Sakkout, Thomas Richards, and M. Wallace. Minimal perturbation in dynamic scheduling. In *Proc. of ECAI-98*, Brighton, England, 1998.

[76] Thomas Schiex. Possibilistic constraint satisfaction problems or "how to handle soft constraints ?". In *Proceedings of the 8th International Conference on Uncertainty in Artificial Intelligence*, pages 269–275, Stanford, 1992.

[77] Thomas Schiex, Hélène Fargier, and Gérard Verfaillie. Valued constraint satisfaction problems: Hard and easy problems. In *Proc. of IJCAI-95*, pages 631–637, Montréal, Canada, 1995.

[78] Thomas Schiex and Gérard Verfaillie. Nogood Recording for Static and Dynamic Constraint Satisfaction Problem. *International Journal of Artificial Intelligence Tools*, 3(2):187–207, 1994.

[79] Smith S.F., O. Lassila, and M. Becker. *Configurable, Mixed-initiative Systems for Planning and Scheduling*. AAAI Press, Menlo Park, advanced planning technology edition, 1996.

[80] Jiri Sgall. *Lecture Notes in Comput. Sci. 1442*, chapter On-line scheduling In Online Algorithms: The State of the Art, pages 196–231. Springer-Verlag, 1998. `http://www.math.cas.cz/~sgall/papers.html`.

[81] Shashi Shekhar and Babak Hamidzadeh. Sarts: A dependable real time search algorithm. *IEEE Trans. on Knowledge and Data Engineering*, 1997.

[82] Wei-Kuan Shih and Jane W.S. Liu. On-line scheduling of imprecise computations to minimize error. In *IEEE Real-Time Systems Symposium*, Los Alamitos, California, 1992.

[83] J. Strosnider and C. Paul. A Structured View of Real-Time Problem Solving. *Artificial Intelligence*, 15(2):45–66, 1994.

[84] R. Sutton and A. Barto. *Reinforcement learning*. MIT Press, Cambridge, Massachusetts, 1998.

[85] Hamdy A. Taha. *Operations Research An Introduction*, chapter 3-9, pages 58–99 303–345. Prentice Hall International Editions, 1995.

[86] G. Verfaillie and L. Lobjois. Problèmes incohérents: expliquer l'incohérence, restaurer la cohérence. In *Proc. of JNPC-99*, pages 111–120, Lyon, France, 1999.

[87] Gérard Verfaillie and Thomas Schiex. Dynamic backtracking for dynamic constraint satisfaction problems. In *Proc. of the ECAI94 Workshop on "Constraint Satisfaction Issues Raised by Practical Applications"*, Amsterdam, The Netherlands, 1994.

[88] Gérard Verfaillie and Thomas Schiex. Solution reuse in dynamic constraint satisfaction problems. In *Proc. of AAAI-94*, pages 307–312, Seattle, WA, 1994.

[89] Gérard Verfaillie and Thomas Schiex. Maintien de solution dans les problèmes dynamiques de satisfaction de contraintes: bilan de quelques approches. In *Revue d'Intelligence Artificielle*, volume 9(3), pages 269–309. 1995.

[90] J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior*. Princeton University Press, 1947.

[91] Benjamin W. Wah and Lon-Chan Chu. Tcgd: A time-constrained approximate guided depth-first search algorithm. *Intl. Journal on Artificial Intelligence Tools*, 1997.

[92] R.J. Wallace and E.C. Freuder. Stable solutions for dynamic constraint satisfaction problems. In *Proc. of CP-98*, pages 447–461, Pisa, Italy, 1998.

[93] R.J. Wallace and E.C. Freuder. Representing and coping with recurrent change in dynamic constraint satisfaction problems. In *CP'99 workshop on modelling and solving soft constraints*, Alexandria, Virginia, 1999.

[94] M. Wooldridge, S. Bussmann, and M. Klosterberg. Production sequencing as negotiation. In *Proc. Int. Conf. on Practical Application of Agents and Multi-Agent Systems*, London, April 1996.

[95] M. Yokoo, T. Ishida, and K. Kubawara. Distributed constraint satisfaction for DAI problems. In M. N. Huhns, editor, *Proc. of the 10th International Workshop on Distributed Artificial Intelligence*, chapter 9. 1990.

[96] Shlomo Zilberstein. Using Anytime Algorithms in Intelligent Systems. *AI Magazine*, 17(3):73–83, 1996.