SFB 637 Autonomous Logistics

Lorenz, M.; Ober-Blöbaum, C.; Herzog, O.: Planning for Autonomous Decision-Making in a Logistic Scenario. In: Proceedings of the 21st European Conference on Modelling and Simulation. 2007

PLANNING FOR AUTONOMOUS DECISION-MAKING IN A LOGISTIC SCENARIO

Martin Lorenz, Christian Ober-Blöbaum, and Otthein Herzog
TZI - Intelligent Systems and CRC 637
University of Bremen
Am Fallturm 1, 28359 Bremen, Germany
{mlo|cob|herzog}@tzi.de

KEYWORDS

Planning, Scheduling, Agents, Simulation, Logistics.

ABSTRACT

Decision making in a real-world domain like logistics is challenging for an autonomous technical system like a software agent. In this paper the problem of planning in such an environment is addressed. Classical planning and probabilistic criteria-directed scheduling components are tied together by a meta-level control and supplemented by a sophisticated world model and a risk management module to form a plan-based decision support system for autonomous control of logistic entities. The system is designed to be integrated in a multi-agent based simulation for evaluation and will later be used to support autonoumous decision making in real-world logistic domains.

I. INTRODUCTION

The dynamic nature of modern transport networks increases the complexity of decision-making in today's logistics. An exact or even heuristics-based solution for global optimisation becomes almost impossible [**?**]. Since the distribution of planning and decision-making to autonomous components is a widely accepted promising solution to handle complex problems [**?**], we consider it an appropriate set-up also for logistic systems.

Autonomous logistic processes—as they are proposed in the CRC 637, "Autonomous Logistic Processes – a Paradigm Shift and its Limitations"[1]—aim at managing logistic entities in a highly distributed way by transferring decision-making competencies to the logistic entities, e.g. in transportation, transshipping facilities, means of transport, or even single freight items, represented by autonomous software systems. These entities coordinate in dynamic, transorganisational, and even competitive environments to perform the processes depending on their respective goals and abilities.

A logistics system based on the above principles allows the transfer of more decision competence from the logistics service provider to autonomous representatives of the logistics system user. For autonomous logistic processes, logistic entities and services may be represented as software agents interacting with each other to build up the logistic process. For the time being the entities will be integrated in a multi-agent based simulation platform which has been developed within rhe CRC 637.

A. Planning for Autonomous Logistic Processes

Logistics planning and monitoring poses a technically challenging problem. The various types of planning activities include packing, scheduling, and route planning with several constraints.

The problem is highly distributed, yet interconnected (as the organisations are autonomous). Dynamic situations enter the system, e.g., in the form of changed logistics requests. The simplest plans involve interaction between the components that are distributed in the logistic networks.

In our approach the whole logistic system consisting of packets, vessels, routes, storage facilities, etc. is modelled as a multi-agent system. Each entity is capable of local planning, deciding, and optimising and can furthermore negotiate with other entities. Because the complexity of the local problem is by orders of magnitude lower than a global optimisation approach we are able to model the logistic problem at a much more fine grained level. As such we can give entities quite some reasoning power for themselves. This includes the ability to identify and assess possible risks such as being late (and having to pay a penalty), to rot or thaw, or to be damaged by improper handling.

Therefore, planning for a logistic domain is a real-time problem with relaxed timing constraints of hours or even days rather than seconds. Domain actions like loading, delivering, or monitoring goods do not utilise the full computational power of the agents underlying hardware. Although deliberating can be done concurrently to acting it is constrained by on-board computational resources, which may be very limited regarding computing power and memory.

In this paper we concentrate on the planning facilities of individual agents as this is one of the core components of the local intelligence assumed for autonomous logistic entities. In the remainder we will first discuss different approaches for planning in a real-

[1] see `http://www.sfb637.uni-bremen.de/home.html?&L=2`

world domain like logistics (sec. **??**). Section **??** gives insight into the architecture and needed components we propose for a suitable planning framework. Finally we discuss the current status and further directions in the sections **??** and **??**.

II. PLANNING IN REAL-WORLD SCENARIOS

Since there exist numerous approaches for all classes of planning problems discussing all of them in detail would go beyond the scope of this paper. Therefore, we will briefly describe some important general classes of planning approaches. Due to the fact that the properties of the domain depicted in the introduction result in a hard planning problem, many planning systems belonging to the introduced classes can be ruled out in advance.

A. Classical Planning

Classical planning based on theorem proving dates back to the 1960's. The widely known STRIPS system was introduced in [**?**]. STRIPS-like planners can cope with "fully observable, deterministic, finite, static, and discrete" environments [**?**]. All these restrictions are not acceptable for our domain.

Furthermore, *time* is in no way represented here since the STRIPS-like representations are based on situation calculus [**?**]. Therefore, essential parts of the logistic domain are left out in a scenario description for a STRIPS-like planning system, e.g., earliest start times for actions required by the date of allocation of manufactured commodities or deadlines for achieving a goal which are necessary to define the arrival window for transported goods.

Even though numerous approaches exist that address one or the other restriction, none of them can remove all of them. Nevertheless, some approaches of the class of plan-space planners which is a subclass of classical planners will become relevant again in section **??** as a part of the proposed system.

B. Decision Theoretic Approaches

This class of approaches can handle uncertainty, observations, the concept of utility, and in some extensions also partial observability [**?**].

Much work has been done on Markov Decision Processes (MDPs) and extensions thereof that are able to cope with partial observability [**?**]. But still the algorithms for solving these problems do not scale well in large domains: computing an optimal solution for a Partially Observable Markov Decision Process (POMDP) is a NEXP-TIME-complete problem, exponential in the size of the state space as well as in the size of the horizon [**?**]. Therefore, an optimal POMDP-based solution of the problem becomes unfeasible for our domain (cf., [**?**], [**?**]).

One approach for handling the complexity of POMDPs is the use of dynamic decision networks as an underlying representation for large POMDPs [**?**]. This reduces the complexity and simplyfies the representation of the problem. Unfortunately the size of our domain still leaves too much computational complexity, which cannot be handled by the representation in a satisfactory manner and therefore renders this approach unfeasible (cf., [**?**], [**?**]). Just to give one illustrating example: in an experiment described in [**?**] computing a solution for a POMDP with 35 state variables takes about 1600 seconds. As the overall complexity of this approach is in $EXP(N)$—where N is the number of state variables—and our domain requires an order of magnitude more states we'd expect computation times calculable in month rather than seconds.

C. Soft Real-Time Computation

The main aspect of systems in the class of real-time computation is that timing constraints are given for performing certain operations and achieving goals. In the case of soft real-time systems the constraints are not hard which means that a missed deadline reduces the utility but does not prevent the system from reaching a goal [**?**].

Several approaches and systems have been proposed and implemented to address these problems but still include major shortcomings. Among them are: CIRCA, which lacks the probabilistic effects of actions in multiple dimensions and a complex model of ressources [**?**]; PRS and its successor UMPRS, which work in a more reactive fashion and are not able to predict future behaviour of the system [**?**], [**?**]; RA and its successor IDEA do not model uncertainty and do not use heuristics [**?**], [**?**]; 3T does not model uncertainty in the results of actions in a quantitative way [**?**].

In the remainder of this section we will present two existing state-of-the-art approaches to soft real-time computing, which address certain problems that are also present in our domain. The applicability of these approaches to our domain is discussed.

C1. SRTA

The *Soft Real-time Architecture* is described in [**?**]. It is designed to address the complexity of acting in a real-world environment that features properties like autonomy and distributed control, indeterminism, temporal constraints, shared resources, and an incomplete, inconsistent world view.

The system is meant to be a component of an intelligent agent and is responsible for supporting a high-level domain-dependent reasoner by planning, scheduling, acting and observing.

Fig. 1. SRTA's system structure (from [**?**, p. 4])

The structure of the presented system is depicted in figure **??**. A central component is the *Design-To-Criteria Planner* (DTC, 2) that accepts goals formulated as a hierarchical task network encoded in

the *TAEMS* language [**?**] and produces several plans achieving the given goals with different characteristics ranked by the preferences of a high-level reasoning layer.

The plan generated by the DTC planner is evaluated in the *Partial Order Scheduler* (3), which adds causal links to plans, tries to parallelise as much as possible, and uses a *Resource Modeller* (4) and a *Schedule Merging* (5) component to integrate the new plan with currently active ones.

The *Parallel Execution Module* (6) and the *Conflict Resolution* (7) component trigger the execution, monitor the performance, report the results, or react on unexpected behaviour of actions. The reactions, e.g., in case of a delayed action reach from only checking if the delay violates any constraints of the current schedule, over minor modifications of the schedule that still allow for achieving all current goals, to reporting the problem to a higher level.

The system is described as being capable of planning, scheduling and acting in a real-word scenario in soft real-time. Since it forms a subcomponent of an intelligent agent there are some aspects outside its scope, e.g., domain-specific problem solving, and communicating and negotiating with other agents. Furthermore it has to be noticed, that the aim of this approach is not to find an optimal but a satisficing solution.

A central component of SRTA, the DTC planner, is analysed in detail in [**?**]. The authors discuss inconsistencies and semantic issues regarding the input language TAEMS and the scheduling process itself. Additionally, they present their own implementation of the planner (VIE-CDS) taking into account their comments on the DTC planner.

C2. Meta-level Control

The systems described up to now reason about how to act in their environment but do not consider how to trade off the use of their computational resources between deliberation and performing domain actions.

[**?**] take into account that the meta-level actions like sensing, planning and communicating take time, which could also be used for executing domain actions. This problem, which occurs only in real-time domains, is addressed by a learning, decision-theoretic meta-planning process that dynamically adepts policies for making decisions on the meta level.

It is shown, that the approach is superior to others, which do not reason about the computational resources that are consumed during meta-level reasoning in order to optimise acting in real-time environments.

III. A FRAMEWORK FOR AUTONOMOUS PLAN-BASED DECISION-MAKING

In section **??** we described different approaches to planning for complex real-world environments. Soft real-time computation and the systems built therein seem to be the most promising ones for the logistic domain since there we have to deal with a soft real-time problem. Howsoever, the problem in a logistic domain is slightly different because the timing constraints are not in an order of magnitude of seconds but of hours or even days. Additionally, the actions an intelligent agent performs in the domain do not utilise the full computational power of the system, so that communication, negotiation and deliberation can be done concurrently to actions like delivering goods. This is the main motivation of this work since it allows the system to optimise its strategy while domain actions are performed. These properties of our domain imply that neither the satisficing approach of SRTA nor the Meta-Level Control approach are well suited for our problem since they address difficulties that are not present in the domain of logistics.

Nevertheless, the approach we present here will incorporate Hierarchical Task Networks for planning and scheduling as well as Partial Order Planning for establishing causal links—which is a well suited combination for addressing complex real-world problems [**?**].

A. System Structure

The system described in the following is depicted in figure **??**. It is based upon the VIE-CDS scheduler which is an implementation of the *Design-To-Criteria* planner described in section **??**. Besides this core component which is responsible for dealing with timing constraints, uncertainty and resources we incorporate a domain problem solver that produces the hierarchical TAEMS task structures needed by VIE-CDS. A meta-level control component inspired by [**?**] but yet taking another approach is responsible for coordinating planning and scheduling processes, triggering and monitoring domain actions and dealing with the dynamic, uncertain and only partially observable nature of the environment. The planning framework described here is meant to be one component of the decision procedure of an intelligent agent, which employs risk management as one major decision criteria (cf., [**?**]). Therefore the meta-level control utilises an external risk-management component for schedule ranking. In the following we will describe the main components of the system in detail.

Fig. 2. Structure of the proposed system

A1. Domain Problem Solver

The domain problem solver (2) can be seen as a component that cares about the more fine-grained details that have to be considered for solving problems in a certain domain. Plan-space planners are of use here since they produce plans that are inherently flexible, contain an explicit causality structure and are open for several extensions needed for our domain [**?**]. Since uncertainty, timing constraints and

resource usage is taken care of by VIE-CDS the problem solver is a straight-forward planner that mainly creates alternatives for reaching a goal and produces a causal structure captured in a partially ordered plan that can be translated into a TAEMS task hierarchy (3). The task hierarchy explicitly models alternatives that have been implicit in the set of partially ordered plans. Thus, by building a causal structure the domain problem solver prepares the problem on domain level for being solved by a general criteria-directed scheduler.

A2. VIE-CDS: A Criteria-Directed Scheduler

As described in section **??** this component (4) takes a hierarchical task structure and various preferences for controlling the scheduling process as input and produces a list of schedules ranked by a utility value given the preferences. Additionally, the preferences allow for controlling the duration as well as the quality of the planned process.

The input for this component—the TAEMS task structures—does not encode conditions and effects of actions in a STRIPS-like manner. It rather models the problem on a more abstract level in a hierarchical way. This implies that alternatives for reaching a goal must have been generated before which in our case is done by the problem solver described in section **??**. This allows for the scheduling component to focus on finding a schedule that best fits the given criteria. The search is guided by several heuristics that exploit the hierarchical structure of the problem definition and respect the preferences in terms of duration, costs, quality and certainty of the evaluated schedules [**?**].

A3. Risk Management

When generating executable plans for an autonomous entity in a real-world domain it is crucial to incorporate some sort of risk deliberation in order to increase the reliability of autonomous decisions (cf. [**?**]).

For the purpose of this paper we assume the risk management component (5) as an oracle capable of assessing rankings to plans and world states. It takes a world model, which can be either the current state of the agents beliefs or a predicted possible world generated by the planner, and the plan to evaluate as arguments, and returns a risk annotation for the plan given the world.

More details on the risk management and it's interplay with other components of the surrounding agent framework can be found in [**?**], [**?**], and [**?**].

A4. Meta-Level Control

This is the central component that coordinates the activities of the agent by invoking the different subcomponents described above. Its main task is to decide, which goals to accept and how to achieve them. In order to do this, it experimentally adds new goals to the current ones and evaluates the expected impact on the utility by generating schedules with the problem solver and the scheduler, and reviewing them using the risk management component. Since the risk management is able to identify which knowledge is crucial for lowering risks and therefore raising the chances for a higher general utility, this enables the meta level to add information gathering actions that were not taken into account during the planning and scheduling process. After several iterations of those *what-if* scenarios the expected utility converges and the most promising goals with the related task structures are selected and added to the already active goals.

This process of selecting goals can be done all the time while the system waits for domain actions to finish or new events to occur. Since these actions can take several hours in a logistic domain and new goals may occur only rarely there will also be slack time that can be used to create contingent plans for the event of an action failing to produce the desired outcome.

Compared to SRTA described in section **??** preventing the costly use of the DTC scheduler (VIE-CDS) has a low priority in our system which is due to the more relaxed timing constraints in the logistic domain. This is the reason why the components of SRTA that reduce the computational complexity at the expense of optimality are left out in our architecture.

Reasoning about the meta-level control as described in section **??** seems not to be necessary since the process of deliberation does not reduce the resources available for performing domain actions.

B. Expected Characteristics

The system is designed to have a wider focus than SRTA since it includes high-level reasoning while SRTA is a more low-level subcomponent [**?**]. At the same time our system tries to use all spare time to optimise its actions and does not stop when a satisficing solution is found. In change to the more optimal solutions that are expected to be produced, the computational complexity will be higher, too. Nevertheless, these characteristics will be suitable to the logistic domain with its more relaxed timing requirements.

Additionally the meta-controlling component (1) more explicitly addresses the partially observable nature of the environment by trying to identify the critical information needed to consolidate the certainty measurements of the considered schedules. This reasoning allows for adding information gathering actions that reduce the risks that arise from incomplete knowledge.

IV. CURRENT STATUS AND FINDINGS

At the current point of our research the domain problem solver component (2) and the criteria-directed scheduling component (4) are fully integrated into the system depicted in figure **??**. It is already apparent that the combination of a domain problem

solver and a criteria-directed scheduler can produce viable schedules in a complex domain like logistics in reasonable computation time. Translation of generated plans into a TAEMS task hierarchy allows for optimisations regarding durations, costs, quality, and certainty—all of which are affecting the performance of the system. Since the full integration into the surrounding agent architecture is still outstanding a complete evaluation could not yet be conducted. However the schedules generated by our system span enough variety to allow a ranking of different plans based on risk management.

For the evaluation of our and other approaches the system is integrated into the deliberation process of software agents representing logistic entities. The agents are living within a simulation platform based on a multi-agent framework that is capable of running autonomous agents within a simulated probabilistic environment of real-world complexity (cf. [**?**]). A number of different scenarios can be used to directly compare the performance of our approach with existing and forthcoming implementations for autonomous logistics.

V. FURTHER RESEARCH

After having shown that our approach gives viable results in principle, we need to incorporate the specialised components like risk management and goal selection.

The next step to take is the integration of the risk management component (5). This will allow us to use our system within the deliberation cycle of an autonomous agent and enable the evaluation in a contention against simple, yet faster, deliberation strategies. This will show us in how far the general utility is increased by the more complex reasoning process. In order to achieve comparability the performance of a prototypical implementation of a straight-forward approach has to be measured within our domain. Relevant logistic keymeasures have been identified for comprehensive evaluation scenarios within the CRC 637.

One task still open in our work is the reasoning on the selection of goals within the meta-level of our system. Possible strategies have to be worked out and implemented in the months to come.

Another important point not yet addressed explicitly, is the resulting performance of the system concerning the general utility of a group of agents, i.e., a transport-provider company in terms of a logistic scenario. Consideration of this aspect involves dealing with the coordination of actions within a group of agents and the possible concurrency of action execution. This is currently not the main focus of our work but future extensions of our system regarding this issue are possible since the underlying representation of the problem in the TAEMS language can model commitments made by and to other agents.

Like already emphasised in the introduction the computational resources of the system may be bounded. Therefore, it has to be further evaluated in how far the already modular structure of the system can be exploited to source out complex computations like planning, scheduling, and risk management.

ACKNOWLEDGEMENT

We want to thank our colleague Jan D. Gehrke for his work on the agent simulation platform in which our implementation is founded.

This research was supported by the German Research Foundation (DFG) as part of the Collaborative Research Centre 637 “Autonomous Logistic Processes – A Paradigm Shift and its Limitations”.

AUTHOR BIOGRAPHIES

Martin Lorenz received his diploma degree in computer science from Technical University of Vienna. He graduated in 1999 with a thesis on ’Signing with the Computer - What User Interface Design can Learn from the Visual Language of Deaf People’. After two non-scientific affiliations from 2000 to 2003 he joined the artificial intelligence group at the University of Bremen in fall 2003 as a Research Assistant. He is now affiliated to the CRC 637. He currently aims at his PhD on autonomous decision making and risk management in rational agents. His Web-page can be found at `http://www.tzi.de/~mlo/`.

Christian Ober-Blöbaum joined the Collaborative Research Center 637 “Autonomous Logistic Processes – A Paradigm Shift and its Limitations” as a student researcher in summer 2004. He is currently working on his diploma thesis on planning and scheduling for autonomous agents in logistics. His Web-page can be found at `http://www.tzi.de/~cob/`.

Otthein Herzog is head of the Artificial Intelligence Research Group, which he built up when he was appointed as a full professor to the Computer Science Department of the University of Bremen in 1993. Before he joined academia, he worked in industrial software development with IBM for 16 years, where he collected extensive know-how in numerous international software product development and AI research projects. Dr. Herzog is the director of the Center for Computing Technologies (TZI) at the University of Bremen and director of the CRC 637. In addition he is co-founder and director of the Mobile Research Center (MRC) at the University of Bremen. His Web-page can be found at `http://www.tzi.de/~herzog/`.