Moving a robot starting from a theory of actions

Giuseppe De Giacomo and **Luca Iocchi** and **Daniele Nardi** and **Riccardo Rosati**
Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza"
Via Salaria 113, 00198 Roma, Italy
{degiacomo,iocchi,nardi,rosati}@dis.uniroma1.it

Abstract

The paper describes an approach to reasoning about actions and planning that starting from a logical formalization arrives at the realization of an actual agent, the mobile robot "Tino". The reasoning tools allow the robot to derive plans from the knowledge about the environment and the action specification, while its reactive capabilities allow it to execute its plans in the real world. The formalization is based on the propositional dynamic logics framework, but exploits the correspondence that exists between propositional dynamic logics and description logics, to carefully weaken the logical inference process in order to keep the reasoning tools of the robot both effective and efficient. Such reasoning tools are then implemented by making use of a general knowledge representation system based on description logics, namely the system CLASSIC.

Introduction

In Artificial Intelligence there has always been a great interest in the design of agents that can exhibit different forms of intelligent behavior. A mobile robot can be regarded as an intelligent agent, that is designed both to achieve high-level goals and to be able to promptly react and adjust its behavior based on the information acquired through the sensors. In the recent past, much attention has been directed to the reactive capabilities of agents in general, and mobile robots in particular. Reactive capabilities are often necessary to cope with the uncertainties of the real-world. However, the capabilities of reasoning about actions and high-level planning are important as well. In this respect, we consider two aspects particularly relevant: the integration of the reactive and planned activities, and the possibility of using a general knowledge representation system, where one can design the agent by relying on the general purpose tools provided by the system. In this paper we mostly focus on the latter issue. Indeed we address high-level planning for a mobile robot within the framework of Propositional Dynamic Logics (PDLs), and we rely on a tight correspondence that exists between PDLs and Description Logics (DLs). In this way we are able to relate planning in PDLs with an implementation that uses the knowledge representation system CLASSIC (Borgida *et al.* 1989, Borgida & Patel-Schneider 1994), which is based on DLs. However, we also address the integration of the proposed framework within an actual mobile robot. Indeed, starting from a logical formalization we arrive at the realization of an *Erratic*-based mobile robot, which is equipped with wheels and sonars (Konolige 1995). We named our mobile robot agent "Tino" and demonstrated it at the 1995 Description Logic Workshop.

The natural way to approach the design of a system for plan generation in a knowledge representation system has its roots in the work on deductive planning. The idea is that a plan can be generated by finding an existence proof for the state where the desired goal is satisfied (Green 1969). However, this approach has mostly been considered at a theoretical level, since the computational cost of deriving a plan from a logical specification has always been considered too high. We face this difficulty, that arises in the PDLs-framework as well, by making use of the mentioned correspondence to take advantage of the work on DLs, which has paid a special attention to the trade-off between expressivity and efficiency of reasoning.

The basic elements of our work originate from the proposals in (Rosenschein 1981) of using the PDLs framework for reasoning about actions and deductive planning. In this setting PDLs formulae denote properties of states, and actions (also called programs) denote state transitions from one state to another. The dynamic system itself is described by means of axioms. Two kinds of axioms are introduced, "static axioms" that describe background knowledge, and "dynamic axioms" that describe how the situation changes when an action is performed. In our formalization we closely follow this setting. However, since we want our robot to reason about actions efficiently, we use the correspondence between PDLs and DLs to rephrase the above setting in DLs, so as to exploit the large body of research on the trade-off between expressivity and efficiency of reasoning. Notably, from this research it is known that the typical form of dynamic axioms is problematic wrt efficiency (such axioms are "cyclic" in the DLs terminology). Hence we have reinterpreted dynamic axioms by means of the so-called procedural rules. By relying on the epistemic interpretation of these rules given in (Donini *et al.* 1994) we have been able to define a setting which provides both a novel epistemic representation of dynamic axioms and a weak form of reasoning, which makes the implementation of the deductive planning approach computationally feasible.

The paper is organized as follows. After reviewing the main technical points upon which our work relies, we describe our setting for the representation of dynamic systems and address reasoning in this setting, pointing out the connections with the actual realization in CLASSIC. We finally describe the implementation of "Tino", which includes a simple monitor and a module for the exchange of information with the underlying low-level software.

Preliminaries

In this section we present the correspondence between propositional dynamic logics (PDLs) and description logics (DLs) and the various technical notions upon which our formalization is based.

The correspondence between PDLs and DLs, first pointed out by Schild (Schild 1991), is based on the similarity between the interpretation structures of the two kinds of logics: at the extensional level, states in PDLs correspond to individuals (members of the domain of interpretation) in DLs, whereas state transitions correspond to links between two individuals. At the intensional level, propositions correspond to concepts, and actions correspond to roles. The correspondence is realized through a (one-to-one and onto) mapping from PDLs formulae to DLs concepts, and from PDLs actions to DLs roles. For a detailed presentation of such a mapping and more generally of the correspondence we refer to (Schild 1991, De Giacomo & Lenzerini 1994). For our purposes it suffices to consider DLs concepts and roles as syntactic variants of PDLs formulae and actions respectively.

We present our formalization using directly the notation of DLs, in order to make it easier to relate the formalization to the actual implementation in CLASSIC, which is a DLs based knowledge representation system.

Technically we base our work on a simple DL which is extended with a modal operator interpreted in terms of minimal knowledge as in (Donini *et al.* 1994, Donini, Nardi, & Rosati 1995), and thus denoted as **K**. We call $\mathcal{KDL}$ the epistemic language and simply $\mathcal{DL}$ the language obtained dropping the epistemic operator. The concept expressions of $\mathcal{DL}$ coincide with those of $\mathcal{FL}^-$ (Brachman & Levesque 1984) and are expressible in the system CLASSIC. An epistemic extension of CLASSIC has been proposed in (Becker & Lakemeyer 1994), but it is restricted to answering epistemic queries to a first-order knowledge base.

The syntactic definition of $\mathcal{KDL}$ is as follows (C, D denote concepts, Q denotes a role, A denotes a primitive concept and R a primitive role):

$$\begin{array}{rcll} C, D & \longrightarrow & A \mid & \text{(primitive concept)} \\ & & \top \mid & \text{(top)} \\ & & C \sqcap D \mid & \text{(intersection)} \\ & & \forall Q.C \mid & \text{(universal quantification)} \\ & & \exists Q.\top \mid & \text{(existential quantification)} \\ & & \mathbf{K}C & \text{(epistemic concept)} \\ Q & \longrightarrow & R \mid & \text{(primitive role)} \\ & & \mathbf{K}R & \text{(epistemic role).} \end{array}$$

The semantics of $\mathcal{KDL}$ is based on the Common Domain Assumption (i.e. every interpretation is defined over a fixed domain, called Δ).

An *epistemic interpretation* is a pair $(\mathcal{I}, \mathcal{W})$ where $\mathcal{W}$ is a set of possible worlds, $\mathcal{I}$ is a possible world such that $\mathcal{I} \in \mathcal{W}$, such that the following equations are satisfied:

$$\begin{array}{rcl} \top^{\mathcal{I},\mathcal{W}} & = & \Delta \\ A^{\mathcal{I},\mathcal{W}} & = & A^{\mathcal{I}} \subseteq \Delta \\ (\mathbf{K}C)^{\mathcal{I},\mathcal{W}} & = & \bigcap_{(\mathcal{I},\mathcal{J}) \in \mathcal{W}\times\mathcal{W}} (C^{\mathcal{J},\mathcal{W}}) \\ (C \sqcap D)^{\mathcal{I},\mathcal{W}} & = & C^{\mathcal{I},\mathcal{W}} \cap D^{\mathcal{I},\mathcal{W}} \\ (\forall Q.C)^{\mathcal{I},\mathcal{W}} & = & \{d_1 \in \Delta \mid \\ & & \forall d_2. (d_1, d_2) \in Q^{\mathcal{I},\mathcal{W}} \rightarrow d_2 \in C^{\mathcal{I},\mathcal{W}}\} \\ (\exists Q.\top)^{\mathcal{I},\mathcal{W}} & = & \{d_1 \in \Delta \mid \exists d_2. (d_1, d_2) \in Q^{\mathcal{I},\mathcal{W}}\} \\ R^{\mathcal{I},\mathcal{W}} & = & P^{\mathcal{I}} \subseteq \Delta \times \Delta \\ (\mathbf{K}R)^{\mathcal{I},\mathcal{W}} & = & \bigcap_{(\mathcal{I},\mathcal{J}) \in \mathcal{W}\times\mathcal{W}} (P^{\mathcal{J},\mathcal{W}}). \end{array}$$

Notice that non-epistemic concepts and roles are given essentially the standard semantics of DLs, conversely epistemic sentences are interpreted on the class of Kripke structures where worlds are interpretations, and all worlds are connected to each other, i.e. the accessibility relation is universal.

As usual a $\mathcal{KDL}$-knowledge base Ψ is a pair $\Psi = \langle \mathcal{T}, \mathcal{A} \rangle$, where $\mathcal{T}$, called the *TBox*, is a set of inclusion statements of the form:

$$C \sqsubseteq D$$

where $C, D \in \mathcal{KDL}$, and $\mathcal{A}$ (the *ABox*) of is a set of membership assertions, of the forms:

$$C(a), R(a, b)$$

where $C, R \in \mathcal{KDL}$ and a, b are names of individuals (different names denote different individuals). The truth of inclusion statements and membership assertions in an epistemic interpretation is defined as set inclusion and set membership, respectively.

An *epistemic model* for an $\mathcal{KDL}$-knowledge base Ψ is a a set of interpretations $\mathcal{W}$ such that for each world

$\mathcal{I} \in \mathcal{W}$, every sentence (inclusion or membership assertion) of Ψ is true in the epistemic interpretation $(\mathcal{I}, \mathcal{W})$.

A *preferred model* for Ψ is an epistemic model $\mathcal{W}$ which is a maximal set of worlds, that is no world can be added to $\mathcal{W}$ obtaining another (greater) epistemic model for Ψ.

A $\mathcal{KDL}$-knowledge base Ψ is said to be *satisfiable* if there exists a preferred model for Ψ, *unsatisfiable* otherwise. Ψ logically implies an assertion σ, written $\Psi \models \sigma$, if σ is true in every preferred model for Ψ.

Using the epistemic operator, it is possible to provide a characterization of procedural rules as epistemic sentences of the form:

$$\mathbf{K}C \sqsubseteq D$$

where C, D are concept expressions of $\mathcal{DL}$ and there are no other occurrences of the epistemic operator in the knowledge base. Notice that a knowledge base in which the epistemic operator occurres only in rules of the above form has a unique preferred model. Moreover, reasoning can be performed by constructing a knowledge base including all the consequences of the rules, called *first-order extension*, which can be used for answering queries in place of the epistemic knowledge base (see (Donini *et al.* 1994)). We point out that in our formalization the use of the epistemic operator will be restricted to the sentences expressing dynamic axioms. Such sentences are obtained by exploiting the above notion of procedural rule.

Reasoning about actions

Following a common approach in reasoning about actions, dynamic systems are modeled in terms of state evolutions caused by actions. A *state* is a *complete* description (wrt to some logic/language) of a situation the system can be in. *Actions* cause state transitions, making the system evolve from the current state to the next one.

In principle we could represent the *behavior* of a system (i.e. all its possible evolutions) as a *transition graph* where:

- Each *node* represents a state, and is labeled with the properties that characterize the state.
- Each *arc* represents a state transition, and is labeled by the action that causes the transition.

Note, however, that *complete knowledge* of the behavior of the system is required to build its transition graph, while in general one has only partial knowledge of such behavior. In deductive planning such knowledge is phrased in *axioms* of some logic (e.g. PDLs (Rosenschein 1981, De Giacomo & Lenzerini 1995) or situation calculus (Reiter 1993)). These axioms select a subset of all possible transition graphs, which are similar since they all satisfy the same axioms, but yet different wrt those properties not imposed by the axioms. The actual behavior of the system is indeed denoted by one of the selected graphs, however which one is not known. Hence one has to concentrate on those properties that are true in all the selected graphs, i.e. those properties that are *logically implied* by the axioms.

Following Rosenschein (Rosenschein 1981) two kinds of axioms are distinguished:

- *Static axioms* are used for representing background knowledge, that is invariant with respect to the execution of actions. In other words, static axioms hold in any state and do not depend on actions.
- *Dynamic axioms* are introduced to represent the changes actions bring about. Typically they have the following form[1]:

$$C \Rightarrow [R]D$$

where R is an action, C represents the *preconditions* that must hold in a state, in order for the action R to be executable; D denotes the *postconditions* that are true in the state resulting from the execution of R in a state where preconditions C hold.

In deductive planning one is typically interested in answering the following question: "Is there a sequence of actions that, starting from an initial state leads to a state where a given property (the goal) holds?". Assuming actions to be deterministic this is captured by the following logical implication (here we phrase it in PDLs):

$$\Gamma \models S \Rightarrow \langle \alpha^* \rangle G \qquad (1)$$

where: (i) Γ is the set of both static and dynamic axioms representing the (partial) knowledge on the system; (ii) S is a formula representing the (partial) knowledge on the initial situation; (iii) G is a formula representing the goal, which is, in fact, a (partial) description of the final state one wants to reach; (iv) $\langle \alpha^* \rangle G$ is a formula that expresses that there exists a finite sequence of actions (here α stands for any action) leading to a state where G is satisfied.

From a *constructive proof* of the above logical implication one can extract an actual sequence of actions (a plan) that leads to a atate where the goal is satisfied.

Observe that in this setting one may have a very sparse knowledge on the system (say a few laws (axioms) one knows the system obeys) and yet be able to make several non-trivial inferences. Unfortunately, this generality is paid by a high computational cost (typically PDLs are EXPTIME-complete (Kozen & Tiuryn 1990)).

Instead of referring to the subset of transition graphs determined by unrestricted axioms, we lower the computational cost by recovering the ability of representing the behavior of the system by means of a graph, called in the following *partial transition graph*. The partial transition graph is an incomplete description of the

[1]The validity of $\langle R \rangle \top$ is also assumed in (Rosenschein 1981)

actual transition graph where certain states and transitions may be missing, and the properties of the states present in the graph may be only partially specified.

As we shall see, the notion of partial transition graph is going to be captured by dynamic axioms of a special form involving the epistemic operator, and by the notion of first-order extension (FOE).

Representation of actions in $\mathcal{KDL}$

In our ontology, an agent is, at any given point, in a certain *state* (roughly corresponding to the position of the robot).

Properties of states are represented as concept expressions of $\mathcal{DL}$. This means that a concept expression denotes a property that may hold in an state.

Instead, actions are represented as $\mathcal{DL}$ roles. In fact we distinguish two kinds of roles: *static-roles* which represent the usual notion of role in DLs and can be useful to structuring properties of states; *action-roles* which denote actions (or better state transitions caused by actions) and are used in a special way.

The behavior of the agent is described by means of both static axioms and dynamic axioms. In principle, axioms could be represented as inclusion statements of the form:

$$C \sqsubseteq D$$

where C and D are concepts of $\mathcal{DL}$. Notice that inclusion statements of the above kind represent the most general form of TBox, since concept definitions can be treated as double inclusions. However, reasoning with general inclusions is intractable and restrictions are normally considered. For example, concepts on the left-hand side are typically required to be primitive, and cycles (recursive inclusions), which are especially problematic from the computational point of view, are not allowed (see for example (Buchheit *et al.* 1994)).

Therefore we model static axioms as acyclic inclusion assertions and concept definitions, not involving action-roles. While we model the dynamic axioms, which are inherently cyclic, by making a special use of the epistemic operator provided by $\mathcal{KDL}$ in order to weaken the reasoning capability and thus gain efficiency.

Specifically dynamic axioms are formalized through epistemic sentences of the form:

$$\mathbf{K}C \sqsubseteq \exists \mathbf{K}R.\top \sqcap \forall R.D$$

which, as we shall see, can be intuitively interpreted as: "if C holds in the current state x of the partial transition graph, then there exists an R-successor y in the partial transition graph, and for all the R-successors of x (and hence for y) D holds".

Given an initial state denoted by the individual *init* and a concept S describing our knowledge about the such an initial state. The partial transition graph can be generated as follows. We start from the initial state *init*, which is going to be the first state included in the partial transition graph. On this state certain properties, namely those denoted by S, are known to hold. Hence we can apply the dynamic axioms whose preconditions C hold in *init* – i.e. $C(init)$ must be implied by S plus the static axioms. By applying a dynamic axiom in *init* we get an R-successor s in the partial transition graph, and in the new state s we know that D holds. We can now apply the dynamic axioms whose preconditions hold in s, and so on. Note that if two states of the partial transition graph have exactly the same properties, i.e. the same concepts hold, then we can collapse the two states into one. Therefore, to avoid redundancies, we can assume that in the final partial transition graph all states are in fact distinguished. We shall see that the notion of first-order extension exactly captures this process. Moreover it ensures us that there exists a unique partial transition graph.

Notably the dynamic axioms cannot be used in the reverse direction for contrapositive reasoning. This weakening is essential to lower the computational cost of reasoning in our formalism.

We remark that, although at first sight it may not be apparent, in our formalization actions are deterministic. Indeed the only existential we allow for actions is unqualified ($\exists R.\top$). Hence we do not have the ability to say that something holds in one R-successor state and doesn't in another one.

In our CLASSIC implementation static axioms become CLASSIC inclusion assertions and concept definitions (cycles are not allowed), while dynamic axioms become CLASSIC rules of the form

```
C => (AND (ALL R D) (FILLS R a))
```

where a is an individual denoting the state reached as a result of the action execution. In this way, we identify all successor states resulting by applying a dynamic axiom, which results in a sound reasoning method as long as the following condition holds: *for each action R the preconditions C in dynamic axioms involving R are disjoint from each other.* Indeed, this condition implies that in every state at most one of the preconditions C for each action R is satisfied. Therefore the only concept that holds in a R-successor, obtained by applying the dynamic axiom $\mathbf{K}C \sqsubseteq \exists \mathbf{K}R.\top \sqcap \forall R.D$, is D. Hence, we can identify all such successor states and give an explicit name (using FILLS) to such a successor without compromising the soundness of reasoning. In the next section we shall see that this condition causes the generation of a number of new known individuals which is at most linear in the number of rules, and hence it also keeps the CLASSIC implementation tractable.

Reasoning

We now formulate the planning problem in the $\mathcal{KDL}$ setting. In the following, we use Γ_S to indicate the set of static axioms and Γ_D to indicate the set of dynamic axioms; S stands for the concept describing the properties of the initial state and G stands for the concept corresponding to the goal.

A plan exists for the goal G if there exists a finite sequence of actions which, from the initial state, leads to a state satisfying the goal. This condition can expressed in our setting by a logical implication which is similar to (1), namely:

$$\langle \Gamma_S \cup \Gamma_D, \{S(init)\}\rangle \models ((\exists \mathbf{K}\alpha)^*.\mathbf{K}G)(init) \qquad (2)$$

where $(\exists\mathbf{K}\alpha)^*\mathbf{K}G$ stands for *any* concept expression of the form

$$\exists\mathbf{K}R_1.\exists\mathbf{K}R_2.\ldots.\exists\mathbf{K}R_n.\mathbf{K}G$$

in which $n \geq 0$ and each R_i is an action-role. Intuitively, condition (2) checks for the existence of a state of the partial transition graph in which the goal is satisfied.

It can be shown that condition (2) holds iff the $\mathcal{KDL}$-knowledge base

$$\langle \Gamma_S \cup \Gamma_D \cup \{\top \sqsubseteq \neg\mathbf{K}G\}, \{S(init)\}\rangle \qquad (3)$$

is unsatisfiable. Informally, this is due to the fact that (3) is unsatisfiable iff there exists a named state in every model where G holds. However, the only way for G to hold in every model is either to be valid and in this case it would hold in particular on *init*, or there must be a sequence in the partial transition graph leading to a state where G holds. Hence we can conclude that (3) implies (2). The converse is immediate to verify.

Now, because of its restricted form, the knowledge base $\Sigma = \langle \Gamma_S \cup \Gamma_D, \{S(init)\}\rangle$ is trivially satisfiable. Hence, condition (3) is verified iff for each preferred model $\mathcal{M}$ for Σ, there exists an individual x such that $\mathcal{M} \models G(x)$.

We point out that in general the $\mathcal{KDL}$-knowledge base $\Sigma = \langle \Gamma_S \cup \Gamma_D, \{S(init)\}\rangle$ has many preferred models, which are distinguishable even up to renaming of individuals. Nevertheless, we can still follow the approach defined in (Donini *et al.* 1994) to capture the notion of minimization of the information the agent is able to infer from its initial knowledge. Specifically, we can give a notion of completion of Σ, in terms of a first-order extension of the epistemic knowledge base, based on the observation that a rule should always generate a new individual, unless such an individual is exactly characterized by the same properties of another existing individual.

The first-order extension (FOE) of a knowledge base $\Sigma = \langle \Gamma_S \cup \Gamma_D, \{S(init)\}\rangle$ is thus introduced with the aim of providing a characterization of the knowledge that is shared by all the preferred models of Σ. $FOE(\Sigma)$ is computed by the algorithm reported in Figure 1, in which $POST(\Sigma, R, x)$ denotes the effect of the application of the rules of Σ involving the action-role R to the individual x, i.e. the set of postconditions (concepts) of the rules which are triggered by x, and $CONCEPTS(\Sigma, i)$ denotes the set of concepts verified by the individual i in Σ. Formally,

$$POST(\Sigma, R, x) = \{D_i \mid (\mathbf{K}C_i \sqsubseteq \exists\mathbf{K}R.\top \sqcap \forall R.D_i) \in \Sigma \wedge \Sigma \models C_i(x)\}$$

and

$$CONCEPTS(\Sigma, i) = \{D \mid \Sigma \models D(i)\}$$

It can be shown that for every knowledge base Σ the first-order extension is unique, that is, every order of extraction of the individuals from the set SET-IND produces the same set of assertions, up to renaming of individuals. Moreover, it is easy to see that the algorithm terminates, that is, the condition SET-IND $= \emptyset$ is eventually reached, since the number of created individuals is bounded to the number of different conjunctions of postconditions of the rules, i.e. 2^n, where n is the number of rules in Σ. Finally, the condition $CONCEPTS(\langle \Gamma_S, Assertions\rangle, l) = CONCEPTS(\langle \Gamma_S, Assertions'\rangle, j)$ can be checked by verifying whether $\langle \Gamma_S, Assertions\rangle \models C(l) \equiv \langle \Gamma_S, Assertions'\rangle \models C(j)$, for each concept C obtained as a conjunction of the postconditions of the rules in Γ_D.

Informally, the algorithm, starting from the initial individual *init*, applies to each named individual the rules in the set Γ_D which are triggered by such individual. A new individual is thus generated, unless an individual with the same properties had already been created. This corresponds in the transition graph to the creation of a new node (state) caused by the execution of an action, unless such a node is already present in the graph. In this way the effect of the rules is computed, thus obtaining a sort of "completion" of the knowledge base.

Observe that the above notion of first-order extension captures the intuition underlying partial transition graphs.

Since the first-order extension of Σ represents the information which must hold in *any* preferred model for Σ, up to individual names, we can establish the following property:

Theorem 1 *There exists an individual x such that*

$$FOE(\Sigma) \models G(x) \qquad (4)$$

if and only if, for each preferred model $\mathcal{M}$ for Σ, there exists an individual x such that $\mathcal{M} \models G(x)$.

By the above property, we can solve the planning problem (2) by solving (4) on the first-order knowledge base $FOE(\Sigma)$. Notice that the number of rule applications required to verify condition (4) is in general exponential in the number of rules appearing in Σ, since the number of new individuals generated in the first-order extension is 2^n in the worst case, where n is the number of rules in Σ. Under the assumption that for each action-role R the preconditions C in the rules involving R are disjoint from each other, the number of individuals generated by the FOE algorithm is at most equal to the number of rules, since every application of the same rule generates the same individual.

As shown in the previous section, this condition must hold for the correctness of our CLASSIC implementation. In fact, in the CLASSIC setting, rules are actually used to extend the knowledge base just like in the first-order extension. Since in the CLASSIC implementation we use the **FILLS** construct to simulate the existential quantification on epistemic roles, each individual name must appear in the rules, therefore the cardinality of the set of individuals in the CLASSIC knowledge base is linearly bounded to the number of rules. Consequently, we cannot compute the FOE of a $\mathcal{KDL}$-knowledge base Σ in the general case using CLASSIC.

The mobile robot "Tino"

Our planning methodology has been practically tested on the *Erratic* base (Konolige 1995) and demonstrated at the 1995 Descriptions Logic Workshop. The basic approach to the software architecture of the robot allows one to reach a good balance between reactive behavior and high-level goal achievement.

This is obtained by a fuzzy controller which is responsible of obstacle avoidance and, more generally, of reactive actions, while the robot is trying to achieve a high-level goal such as reaching the next door in the corridor. A critical aspect of the multi-level approach to the robot software architecture is in the exchange of information between the different layers. In our case we need to represent the information about the map into a knowledge base of static axioms and to represent the action descriptions into a set of rules. This is actually achieved by analyzing the low-level representation of the robot and constructing a CLASSIC knowledge base. For example given the map of Figure 2 we obtain the knowledge base of Figure 3. To this end we have implemented a module that takes as input the internal map and generates the knowledge base.

The overall system is constituted by a plan generation module, whose implementation follows the above described setting and a monitor, which handles the communication between the planner and the fuzzy controller.

The planner is activated by adding to the knowledge base the (partial) description of the initial state. CLASSIC rules are thus triggered, and their propagation eventually generates a state where the goal is satisfied. The plan is then extracted by the explanation facility associated with the rules, which allows for an automatic generation of a graph (essentially the partial transition graph) with all the paths from the initial state to the final state. An example of the explanations and their associated graph are given below, in Figure 4 and Figure 5, respectively. The plan to be sent to the robot is then selected by finding the path between the initial state and the final state which is minimal in terms of the number of actions.

The monitor calls the planner, activates the execution of the plan and controls it. This is achieved by combining each action with the reactive behaviors for obstacle avoidance, through the underlying "blending" mechanism (see (Saffiotti, Konolige, & Ruspini 1995)). By setting a time out for the execution of each action the monitor detects the plan failure and provides a justification for the failure such as "door closed". In such a case the system can re-plan by updating the knowledge base and, consequently, the internal representation.

Conclusion

The goal of our work was to exploit the reasoning services offered by a knowledge representation system based on description logics to the task of plan generation.

We have presented a formalization of planning based on Description Logics and described an implementation of the proposed framework in the system CLASSIC. Previous work (Devambu & Litman 1991) aimed at using CLASSIC for reasoning about plans, by introducing a formalism for describing plans, while the present work is focussed on plan generation and relies on the standard features of the system. Our implementation is actually used to plan the actions of the mobile robot *Erratic*, capable of integrating reacting behavior with action execution.

One original contribution of our work is in the idea of using the rule mechanism, and its associated interpretation in terms of minimal knowledge, to weaken the assumptions underlying a first-order formalization of actions. In this way a plan can be obtained by a forward reasoning process, that is weaker than ordinary deduction, but semantically justified and computationally feasible. We are currently addressing other aspects of reasoning about actions, such as the treatment of perception actions and the frame problem.

References

Becker, A., and Lakemeyer, G. 1994. Epistemic queries in CLASSIC. In *Proceedings of the Eighteenth German Annual Conference on Artificial Intelligence (KI-94)*. Springer-Verlag.

Borgida, A., and Patel-Schneider, P. F. 1994. A semantics and complete algorithm for subsumption in the CLASSIC description logic. *Journal of Artificial Intelligence Research* 1:277–308.

Borgida, A.; Brachman, R. J.; McGuinness, D. L.; and Alperin Resnick, L. 1989. CLASSIC: A structural data model for objects. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 59–67.

Brachman, R. J., and Levesque, H. J. 1984. The tractability of subsumption in frame-based description languages. In *Proceedings of the Fourth National*

Conference on Artificial Intelligence (AAAI-84), 34–37.

Buchheit, M.; Donini, F. M.; Nutt, W.; and Schaerf, A. 1994. Terminological systems revisited: Terminology = schema + views. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94)*, 199–204.

De Giacomo, G., and Lenzerini, M. 1994. Boosting the correspondence between description logics and propositional dynamic logics. In *Proceedings of the 12th National Conference on Artificial Intelligence*, 205–212.

De Giacomo, G., and Lenzerini, M. 1995. Enhanced propositional dynamic logic for reasoning about concurrent actions (extended abstract). In *Working notes of the AAAI 1995 Spring Symposium on Extending Theories of Action: Formal and Practical Applications*, 62–67.

Devambu, P., and Litman, D. 1991. Plan-based terminological reasoning. In Allen, J.; Fikes, R.; and Sandewall, E., eds., *Proceedings of the Second International Conference on the Principles of Knowledge Representation and Reasoning (KR-91)*, 128–138. Morgan Kaufmann, Los Altos.

Donini, F. M.; Lenzerini, M.; Nardi, D.; Nutt, W.; and Schaerf, A. 1994. Queries, rules and definitions. In *Foundations of Knowledge Representation and Reasoning*. Springer-Verlag.

Donini, F. M.; Nardi, D.; and Rosati, R. 1995. Non first-order features in concept languages. In *Proceedings of the Fourth Conference of the Italian Association for Artificial Intelligence (AI*IA-95)*, Lecture Notes In Artificial Intelligence. Springer-Verlag.

Green, C. 1969. Theorem proving by resolution as basis for question-answering systems. In *Machine Intelligence*, volume 4. American Elsevier. 183–205.

Konolige, K. 1995. Erratic competes with the big boys. *AAAI Magazine* Summer:61–67.

Kozen, D., and Tiuryn, J. 1990. Logics of programs. In van Leeuwen, J., ed., *Handbook of Theoretical Computer Science*. Elsevier Science Publishers (North-Holland), Amsterdam. 790–840.

Reiter, R. 1993. Proving properties of states in the situation calculus. *Artificial Intelligence* 64:337–351.

Rosenschein, S. 1981. Plan synthesis: a logical approach. In *Proc. of the 8th Int. Joint Conf. on Artificial Intelligence*.

Saffiotti, A.; Konolige, K.; and Ruspini, E. 1995. A multivalued logic approach to integrating planning and control. Technical report, SRI International, Menlo Park, CA.

Schild, K. 1991. A correspondence theory for terminological logics: Preliminary report. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI-91)*, 466–471.

```
ALGORITHM FOE

INPUT: $\Sigma = \langle \Gamma_S \cup \Gamma_D, \{S(init)\} \rangle$
OUTPUT: $FOE(\Sigma)$

begin
  SET-IND = $\{init\}$;
  SET-ALL-IND = $\{init\}$;
  $Assertions = \{S(init)\}$;
  repeat
    ind = choose(SET-IND);
    for each action-role $R$ do
    begin
      j = NEW individual name;
      $Assertions' = Assertions \cup \{R(ind, j)\} \cup$
                  $\{D_i(j) | D_i \in POST(\langle \Gamma_S \cup \Gamma_D, Assertions \rangle, R, ind)\}$;
      if there exists an individual $l \in$ SET-ALL-IND such that
        $CONCEPTS(\langle \Gamma_S, Assertions \rangle, l) = CONCEPTS(\langle \Gamma_S, Assertions' \rangle, j)$
      then $Assertions = Assertions \cup R(ind, l)$
      else
      begin
        $Assertions = Assertions'$;
        SET-IND = SET-IND $\cup \{j\}$
        SET-ALL-IND = SET-ALL-IND $\cup \{j\}$
      end
    end;
    SET-IND = SET-IND $-\{ind\}$
  until SET-IND = $\emptyset$;
  return $\langle \Gamma_S, Assertions \rangle$
end;
```

Figure 1: Algorithm computing $FOE(\Sigma)$

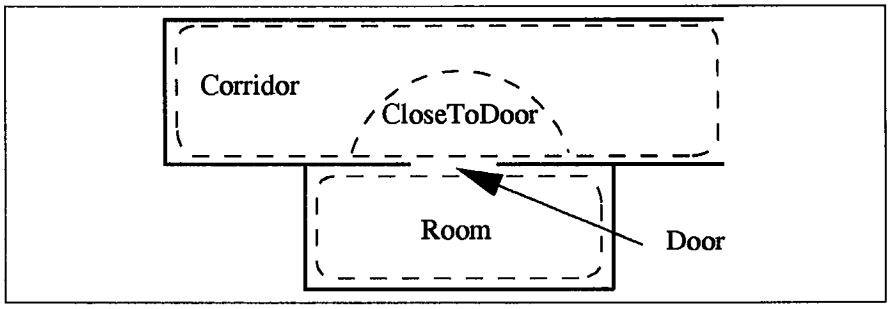

Figure 2: A simple environment

```
; Role definitions
(cl-define-primitive-role 'Followcorr)
(cl-define-primitive-role 'Enter)
(cl-define-primitive-role 'Exit)
(cl-define-primitive-role 'NextDoor :attribute T)
;
; Concept definitions
(cl-define-disjoint-primitive-concept 'Corridor 'classic-thing 'state)
(cl-define-disjoint-primitive-concept 'Room 'classic-thing 'state)
(cl-define-disjoint-primitive-concept 'Door 'classic-thing 'state)
(cl-define-concept 'CloseToDoor '(AND Corridor (FILLS NextDoor xDoor)))
;
; Dynamic axioms
(cl-add-rule 'r1   @CloseToDoor   '(ALL Enter  Room) )
(cl-add-rule 's1   @CloseToDoor   '(FILLS Enter  xRoom) )
(cl-add-rule 'r2   @Corridor   '(ALL Followcorr  CloseToDoor) )
(cl-add-rule 's2   @Corridor   '(FILLS Followcorr  xCloseToDoor) )
(cl-add-rule 'r3   @Room   '(ALL Exit  CloseToDoor) )
(cl-add-rule 's3   @Room   '(FILLS Exit  xCloseToDoor) )
;
; Individuals
(cl-create-ind  'xDoor 'Door)
(cl-ind-add-fillers @xRoom @NextDoor (list @xDoor))
;
; Initial state
(cl-create-ind  'xCorridor  'Corridor)
```

Figure 3: The Classic knowledge base

```
@i{XRoom} ->
  primitives: (@c{Room} @c{CLASSIC-THING})
    @c{Room} : PROPAGATION: from individual: @i{XCloseToDoor};
        through role: @r{Enter}.
@i{XCloseToDoor} ->
  primitives: (@c{CloseToDoor} @c{Corridor} @c{CLASSIC-THING})
    @c{Corridor} : PROPAGATION: from individual: @i{XCorridor};
        through role: @r{FollowCORR}.
        Information propagated:
          @c{CloseToDoor}.
      PROPAGATION: from individual: @i{XRoom};
        through role: @r{Exit}.
        Information propagated:
          @c{CloseToDoor}.
      PROPAGATION: from individual: @i{XCloseToDoor};
        through role: @r{FollowCORR}.
        Information propagated:
          @c{CloseToDoor}.
    @c{CloseToDoor} : PROPAGATION: from individual: @i{XCorridor};
        through role: @r{FollowCORR}.
      PROPAGATION: from individual: @i{XRoom};
        through role: @r{Exit}.
      PROPAGATION: from individual: @i{XCloseToDoor};
        through role: @r{FollowCORR}.
@i{XCorridor} ->
  primitives: (@c{Corridor} @c{CLASSIC-THING})
    @c{Corridor} : TOLD-INFO
```

Figure 4: The Classic explanation

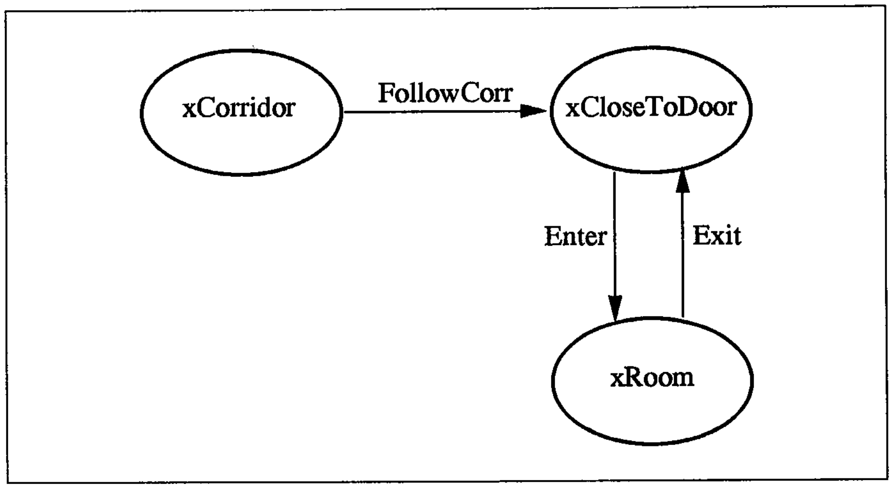

Figure 5: The action graph