Qualitative and Probabilistic Uncertainty in Reasoning about Actions with Sensing

Luca Iocchi **Thomas Lukasiewicz**[*] **Daniele Nardi** **Riccardo Rosati**

Dipartimento di Informatica e Sistemistica, Università di Roma “La Sapienza”
Via Salaria 113, I-00198 Rome, Italy
{iocchi, lukasiewicz, nardi, rosati}@dis.uniroma1.it

Abstract

We present the description logic PN-$\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$ for reasoning about actions with sensing under qualitative and probabilistic uncertainty, which is an extension of the description logic $\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$ by actions with nondeterministic and probabilistic effects. We define a formal semantics of PN-$\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$ in terms of deterministic, nondeterministic, and probabilistic transitions between epistemic states, which are sets of possible states of the world. We introduce the notions of a conditional plan and its goodness under qualitative and probabilistic uncertainty. We then formulate the problem of conditional planning in this framework, and we present an algorithm for solving it. This algorithm is based on a reduction to reasoning in description logics, and is shown to be sound and complete in the sense that it generates all optimal plans. We also describe an application in a robotic-soccer scenario.

Introduction

In reasoning about actions for mobile robots in real-world environments, one of the most crucial problems that we have to face is uncertainty, both about the initial situation of the robot’s world and about the results of the actions taken by the robot. One way of adding uncertainty to reasoning about actions is based on qualitative models in which all possible alternatives are equally considered. Another way is based on quantitative models where we have a probability distribution on the set of possible alternatives, and thus can numerically distinguish between possible alternatives.

Well-known first-order formalisms for reasoning about actions such as the situation calculus (Reiter 2001) allow for expressing qualitative uncertainty about the initial situation and the effects of actions through disjunctive knowledge. Similarly, recent formalisms for reasoning about actions that are inspired by the action language $\mathcal{A}$ (Gelfond & Lifschitz 1993), such as the action language $\mathcal{C}+$ (Giunchiglia *et al.* 2004) and the planning language $\mathcal{K}$ (Eiter *et al.* 2003), allow for qualitative uncertainty in the form of incomplete initial states and nondeterministic effects of actions.

There are a number of formalisms for probabilistic reasoning about actions. In particular, Bacchus, Halpern, & Levesque (1999) propose a probabilistic generalization of the situation calculus, which is based on first-order logics of probability, and which allows to reason about an agent’s probabilistic degrees of belief and how these beliefs change when actions are executed. Poole’s independent choice logic (1997; 1998) is based on acyclic logic programs under different “choices”. Each choice along with the acyclic logic program produces a first-order model. By placing a probability distribution over the different choices, one then obtains a distribution over the set of first-order models. Mateus *et al.* (2001; 2002) allow for describing the uncertain effects of an action by discrete, continuous, and mixed probability distributions, and focus especially on probabilistic temporal projection and belief update. Finzi & Pirri (2001) add probabilities to the situation calculus to quantify and compare the safety of different sequences of actions. Boutilier, Dean, & Hanks (2001) introduce first-order Markov decision processes (MDPs) that are formulated in a probabilistic generalization of the situation calculus, and present a dynamic programming approach for solving them. A companion paper (Boutilier *et al.* 2000) presents a generalization of Golog, called DTGolog, that combines robot programming in Golog with decision-theoretic planning in MDPs. Großkreutz & Lakemeyer (2002; 2001) propose a probabilistic generalization of Golog, called pGolog, especially for probabilistic projection and belief update. A probabilistic extension of the action language $\mathcal{A}$ is given by Baral, Tran, & Tuan (2002), which aims especially at an elaboration-tolerant representation of MDPs and at formulating observation assimilation and counterfactual reasoning.

Even though there is extensive work on reasoning about actions under qualitative and probabilistic uncertainty separately, there is only little work that orthogonally combines qualitative and probabilistic uncertainty in a uniform framework for reasoning about actions. One important such approach is due to Halpern & Tuttle (1993), which combines nondeterminism and probabilistic uncertainty in a game-theoretic framework. Halpern & Tuttle argue in particular that “some choices in a distributed system must be viewed as inherently nondeterministic (or, perhaps better, nonprobabilistic), and that it is inappropriate, both philosophically and pragmatically, to model probabilistically what is inherently nondeterministic”. This underlines the strong need for explicitly modeling qualitative uncertainty in addition to probabilistic uncertainty in reasoning about actions.

[*]Alternate address: Institut für Informationssysteme, Technische Universität Wien, Favoritenstraße 9-11, A-1040 Vienna, Austria; e-mail: lukasiewicz@kr.tuwien.ac.at.

The main idea behind this paper is to orthogonally combine qualitative and probabilistic uncertainty in a uniform framework for reasoning about actions. This idea is already pursued in a companion paper (Eiter & Lukasiewicz 2003) that presents the language $\mathrm{P}\mathcal{C}+$ for probabilistic reasoning about actions, which is a generalization of $\mathcal{C}+$ that allows for expressing probabilistic and nondeterministic effects of actions as well as probabilistic and qualitative uncertainty about the initial situation of the world. A formal semantics of $\mathrm{P}\mathcal{C}+$ is defined in terms of probabilistic transitions between sets of states. Using a concept of a history and its belief state, it is then shown how the problems of prediction, postdiction, and (unconditional) planning under qualitative and probabilistic uncertainty can be formulated in $\mathrm{P}\mathcal{C}+$.

The present paper continues this important line of research. Its main aim is to develop a formalism that additionally allows for *sensing in reasoning about actions under qualitative and probabilistic uncertainty*, and thus to formulate the problem of *conditional planning under qualitative and probabilistic uncertainty*, and to elaborate an algorithm for solving it. The base formalism that we use in this paper is the description logic $\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$ (Iocchi, Nardi, & Rosati 2000), which is a fragment of the autoepistemic description logic $\mathcal{ALCK}_{\mathcal{NF}}$ (Donini, Nardi, & Rosati 2002). It allows for specifying an initial epistemic state and transitions between epistemic states, where an epistemic state represents the set of all alternatives that an agent considers possible in the world, and thus already expresses some form of qualitative uncertainty in reasoning about actions. It has been successfully implemented and used for a robotic soccer team (Iocchi, Nardi, & Rosati 2000).

In this paper, we present an extension of $\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$ by actions with nondeterministic and probabilistic effects. The main contributions can be summarized as follows:

- We present the description logic PN-$\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$ for reasoning about actions with sensing under qualitative and probabilistic uncertainty, which is an extension of $\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$ (Iocchi, Nardi, & Rosati 2000) by actions with nondeterministic and probabilistic effects. As a central feature, it orthogonally combines in a single framework qualitative as well as probabilistic uncertainty about the effects of actions. It also allows for some qualitative uncertainty in epistemic states.
- We define a formal semantics of PN-$\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$ by interpreting an action description in PN-$\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$ as a system of deterministic, nondeterministic, and probabilistic transitions between epistemic states, which are sets of possible states of the world. Here, *probabilistic transitions* are like in partially observable Markov decision processes (POMDPs) (Kaelbling, Littman, & Cassandra 1998), but they are between epistemic states and so *sets of states* rather than *single states*.
- We formulate the problem of conditional planning under qualitative and probabilistic uncertainty in PN-$\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$. In particular, we define the notion of a conditional plan. Based on the concept of a belief tree, we then formulate the goodness of a conditional plan with respect to a goal and an initial observation. In the extended report (Iocchi *et al.* 2003), we also give a compact representation of belief trees and prove its correctness in implementing belief trees.
- We present an algorithm for conditional planning under qualitative and probabilistic uncertainty in PN-$\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$, and we prove in particular that the algorithm is sound and complete in the sense that it generates the set of all optimal conditional plans. It is a significant extension of a previous algorithm for conditional planning in $\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$.
- We describe a formulation of a robotic-soccer scenario in PN-$\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$, which gives evidence of the usefulness of our formalism in realistic applications.

$\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$

We now recall a subset of the description logic $\mathcal{ALCK}_{\mathcal{NF}}$ (Donini, Nardi, & Rosati 2002) (see Appendix A), called $\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$, which we use as a formalism for representing dynamic systems (Iocchi, Nardi, & Rosati 2000). From the semantic viewpoint, the main idea behind this framework is the interpretation of the dynamic system specification at the agent's knowledge level, through the notion of an *epistemic state*, which is a set of possible states of the world. An epistemic state encodes what the agent knows about the world, in contrast to what is true in the world, and planning in the presence of sensing is obtained by modeling the dynamics of the agent's epistemic state, rather than the dynamics of the world. Thus, the notion of an epistemic state allows for expressing a form of qualitative uncertainty in reasoning about actions. The representation of dynamic systems in $\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$ is based on the following correspondences: (i) roles and concepts encode actions and properties of the world, respectively, and (ii) the *epistemic operators* $\mathbf{K}$ and $\mathbf{A}$ are used to encode the epistemic state of an agent.

Syntax

We first recall the syntax of initial state and action descriptions in $\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$. They allow for modeling what an agent knows about the properties of the world and how this knowledge changes through the execution of actions. Properties (resp., actions) are encoded by concepts (resp., roles), and the dynamics of the world by inclusion axioms.

We assume a nonempty finite set $\mathcal{A}$ of atomic concepts, called *fluents*, which are divided into *static* and *dynamic* fluents. We use $\bot$ and $\top$ to denote the constants false and true, respectively. A *fluent literal* is either a fluent A or its negation $\neg A$. A *fluent conjunction* is of the form $L_1 \sqcap \cdots \sqcap L_n$, where $L_1, \ldots, L_n$ are fluent literals and $n \geqslant 1$. The set of *fluent formulas* is the closure of $\mathcal{A} \cup \{\bot, \top\}$ under the Boolean operators $\neg$, $\sqcap$, and $\sqcup$ (that is, if ϕ and ψ are fluent formulas, then also $\neg\phi$, $\phi \sqcap \psi$, and $\phi \sqcup \psi$). We assume a set $\mathcal{R}$ of atomic roles, called *actions*, which are divided into *effect* and *sensing actions*.

A *precondition axiom* is of the form $\mathbf{K}\phi \sqsubseteq \exists \mathbf{K}\alpha.\top$ (abbreviated as **executable** α **if** ϕ), where ϕ is a fluent formula, and α is an action. Informally, α is executable in every state that satisfies ϕ. If $\phi = \top$, then α is always executable.

A *conditional effect axiom* is of the form $\mathbf{K}\phi \sqsubseteq \forall \alpha.\psi$ (abbreviated as **caused** ψ **after** α **when** ϕ, and as **caused** ψ **after** α, when $\phi = \top$), where ϕ is a fluent formula, ψ is a fluent conjunction, and α is an action. Informally, if the current state satisfies ϕ, then executing α has the direct effect ψ. Note that indirect effects may be formulated through domain constraint axioms, which are introduced below.

A *sensing effect axiom* is of the form $\top \sqsubseteq \mathbf{K}(\forall\alpha.\omega) \sqcup \mathbf{K}(\forall\alpha.\neg\omega)$ (abbreviated as **caused to know** ω **or** $\neg\omega$ **after** α), where ω is a fluent conjunction, and α is a sensing action. Informally, after executing the sensing action α, the agent knows that ω is either true or false. That is, sensing actions modify the epistemic state of the agent without affecting the state of the world (Levesque 1996). Note that we assume that sensing actions have only two outcomes, but our approach can be easily generalized to $k > 2$ outcomes.

A *default frame axiom* is of the form $\mathbf{K}\phi \sqsubseteq \forall\mathbf{K}\alpha.\mathbf{A}\neg\phi \sqcup \mathbf{K}\phi$ (abbreviated as **inertial** ϕ **after** α), where ϕ is a fluent conjunction, and α is an action. Informally, if ϕ holds in the current state, then ϕ holds also after the execution of α, if it is consistent with the effects of α.

A *domain constraint axiom* is of the form $\phi \sqsubseteq \psi$ (abbreviated as **caused** ψ **if** ϕ), where ϕ and ψ are fluent formulas. It represents background knowledge, which is invariant relative to the execution of actions.

An *initial state description* ϕ_I is a fluent formula. An *action description* KB is a finite set of precondition, conditional effect, sensing effect, default frame, and domain constraint axioms. A *goal description* ψ_G is a fluent formula.

Semantics

We next recall the semantics of initial state descriptions ϕ_I and action descriptions KB. Informally, ϕ_I represents a set of possible states, while KB encodes a system of state transitions between sets of possible states (a directed graph, where state sets serve as nodes, and the outgoing arrows of each node are labeled with pairwise distinct actions).

We first define states, epistemic states (e-states), the executability of an action in an e-state, and the successor e-state after executing an action in an e-state. In the sequel, let KB be an action description.

A *state* s is a truth assignment to the fluents in $\mathcal{A}$. Every state s is associated with the fluent formula ϕ_s, which is the conjunction of all A (resp., $\neg A$) such that $A \in \mathcal{A}$ and $s(A)$ is $\mathbf{true}$ (resp., **false**). An *epistemic state* (or *e-state*) is a set of states S where $KB \not\models \phi_s \sqsubseteq \bot$ for all $s \in S$. Every e-state S is associated with the fluent formula $\phi_S = \bigsqcup\{\phi_s \mid s \in S\}$, where $\phi_S = \bot$, if $S = \emptyset$.

An action α is *executable* in an e-state S iff $KB \models \mathbf{K}\phi_S \sqsubseteq \exists\mathbf{K}\alpha.\top$. In this case, the *successor e-state* of S under an effect action α, denoted $\Phi(S,\alpha)$, is the smallest e-state S' such that $KB \models \mathbf{K}\phi_S \sqsubseteq \forall\alpha.\phi_{S'}$. The *successor e-state* of S under a sensing action α with outcome $\sigma \in \{\omega, \neg\omega\}$, denoted $\Phi(S,\alpha_\sigma)$, is the smallest e-state S' such that $KB \models \mathbf{K}\phi_S \sqsubseteq \forall\alpha.\phi_{S'}$ and $KB \models \phi_{S'} \sqsubseteq \sigma$. Intuitively, $\phi_{S'}$ encodes the direct effects of the action, the indirect effects due to the domain constraints, and the propagation of inertial properties.

We now define the semantics of initial state and action descriptions as follows. An action description KB encodes the directed graph $G_{KB} = (N, E)$, where N is the set of all e-states, and E contains $S \to S'$ labeled with an effect action α (resp., sensing action α with outcome $\sigma \in \{\omega, \neg\omega\}$) iff (i) α is executable in S and (ii) $S' = \Phi(S,\alpha)$ (resp., $S' = \Phi(S,\alpha_\sigma)$). An initial state description ϕ_I encodes the e-state $S = S_{\phi_I}$ where $KB \models \phi_I \equiv \phi_S$.

PN-$\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$

In this section, we define the novel description logic PN-$\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$, which extends $\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$ by actions with nondeterministic and probabilistic effects.

Syntax

We divide the set of all effect actions (without sensing actions) into *deterministic*, *nondeterministic*, and *probabilistic* actions. Note that deterministic actions are a special case of nondeterministic and probabilistic actions, however, nondeterministic actions are not a special case of probabilistic actions. In order to encode different effects of a nondeterministic or probabilistic action α, we specify a set of different contexts for α and the effects of α under each such context. If α is probabilistic, then we also specify a probability distribution over the contexts for α.

We first introduce *dynamic context formulas*, which associate with such α a set of contexts and eventually also a probability distribution over its set of contexts. A *nondeterministic* (resp., *probabilistic*) *dynamic context formula* for a nondeterministic (resp., probabilistic) action α is of form

$$V(\alpha)=(v_1,\ldots,v_n) \quad \text{(resp., } V(\alpha)=(v_1{:}p_1,\ldots,v_n{:}p_n)\text{)},$$

where (i) $V(\alpha)$ is the *context variable* for α, which may take on values from $\mathrm{dom}(V(\alpha)) = \{v_1,\ldots,v_n\}$, and (ii) $n \geqslant 1$. Every v_i is called a *context* of α, and is associated with a new atomic concept $V^{v_i}_{\alpha}$ (called *context concept*). If α is probabilistic, then $p_1,\ldots,p_n > 0$, $p_1 + \cdots + p_n = 1$, and every context v_i of α has the probability $Pr_\alpha(v_i) = p_i$.

The effects of the nondeterministic (resp., probabilistic) action α in the context v_i of α can now be encoded by conditional effect axioms of the form $\mathbf{K}(\phi \sqcap V^{v_i}_{\alpha}) \sqsubseteq \forall\alpha.\psi_i$, where ϕ is a fluent formula, and ψ_i is a fluent conjunction. Informally, "if $V(\alpha) = v_i$ and ϕ is known, then executing α has the effect ψ_i". A dynamic context formula for α as above along with the axioms $\mathbf{K}(\phi \sqcap V^{v_i}_{\alpha}) \sqsubseteq \forall\alpha.\psi_i$, $i \in \{1,\ldots,n\}$, is also abbreviated as **caused** $\psi_1,\ldots,\psi_n$ **after** α **when** ϕ (resp., **caused** $\psi_1{:}\ p_1,\ldots,\psi_n{:}\ p_n$ **after** α **when** ϕ), where we omit "**when** ϕ" when $\phi = \top$.

An *extended action description* $D = (KB, C)$ consists of an action description KB in $\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$ and a finite set C of dynamic context formulas, exactly one for each nondeterministic or probabilistic action in KB.

Semantics

We define the semantics of an extended action description $D = (KB, C)$ through a system of deterministic, nondeterministic, and probabilistic transitions between e-states. To this end, we add to the transition system of KB a mapping that assigns to each pair (S, α) of a current e-state S and a nondeterministic (resp., probabilistic) action α executable in S, a set of (resp., a probability distribution on a set of) successor e-states after executing α.

In the sequel, let $D = (KB, C)$ be an extended action description. For every nondeterministic or probabilistic action α in D, we define $KB_\alpha = \{\top \sqsubseteq \bigsqcup_{v \in \mathrm{dom}(V(\alpha))} \mathbf{K}V^{v}_{\alpha}\} \cup \{\mathbf{K}V^{u}_{\alpha} \sqsubseteq \neg V^{v}_{\alpha} \mid u, v \in \mathrm{dom}(V(\alpha)),\ u \neq v\}$. Informally, KB_α encodes that the V^{v}_{α}'s are exhaustive and pairwise disjoint,

executable $\mathsf{gotoball}$ **if** $\mathsf{ba}\sqcap\neg\mathsf{bm}$
executable $\mathsf{bodykick}$ **if** cb
executable $\mathsf{straightkick}$ **if** $\mathsf{cb}\sqcap\mathsf{fa}$
executable $\mathsf{sidekick}$ **if** $\mathsf{cb}\sqcap\neg\mathsf{fa}$
executable $\mathsf{aligntoball}$ **if** bm
executable $\mathsf{openlegs}$ **if** bm
executable $\mathsf{sensealignedtoball}$ **if** bm
caused gs **after** $\mathsf{openlegs}$ **when** ab
caused to know cb **or** $\neg\mathsf{cb}$ **after** $\mathsf{senseballclose}$
caused to know fa **or** $\neg\mathsf{fa}$ **after** $\mathsf{sensefreeahead}$
caused to know ab **or** $\neg\mathsf{ab}$ **after** $\mathsf{sensealignedtoball}$
inertial l **after** α (for every fluent literal l and action α)
caused ba **if** cb
caused $\mathsf{gs}, \neg\mathsf{gs}$ **after** $\mathsf{openlegs}$
caused $\mathsf{cb}{:}0.8, \neg\mathsf{ba}{:}0.1, \neg\mathsf{cb}{:}0.1$ **after** $\mathsf{gotoball}$
caused $\neg\mathsf{ba}\sqcap\neg\mathsf{ip}{:}0.1, \neg\mathsf{ba}\sqcap\mathsf{ip}{:}0.5, \neg\mathsf{ip}{:}0.1, \top{:}0.3$ **after** $\mathsf{bodykick}$
caused $\neg\mathsf{ba}{:}0.9, \top{:}0.1$ **after** $\mathsf{straightkick}$
caused $\neg\mathsf{ba}{:}0.7, \top{:}0.3$ **after** $\mathsf{sidekick}$
caused $\mathsf{ab}{:}0.7, \neg\mathsf{ab}{:}0.3$ **after** $\mathsf{aligntoball}$

Figure 1: Extended Action Description

which corresponds to selecting one context of α in every e-state. The notions of states, e-states, and the executability of actions in e-states are defined as above. In particular, a state is a truth assignment to the fluents without the context concepts. If a nondeterministic or probabilistic action α is executable in an e-state S, then the *successor e-state* of S after executing α in its context v, denoted $\Phi_v(S,\alpha)$, is the smallest e-state S' with $KB \cup KB_\alpha \models \mathbf{K}(\phi_S \sqcap V^v_\alpha) \sqsubseteq \forall\alpha.\phi_{S'}$.

Let S be an e-state. For every nondeterministic action α executable in S, the *set of successor e-states* of S under α is defined as $F_\alpha(S) = \{\Phi_v(S,\alpha) \mid v \in \mathrm{dom}(V(\alpha))\}$. Intuitively, executing α in S nondeterministically leads to some $S' \in F_\alpha(S)$. For every probabilistic action α executable in S, the *probability distribution on the successor e-states* of S under α, denoted $Pr_\alpha(\,\cdot\,|S)$, is defined by $Pr_\alpha(S'|S) = \sum_{v \in \mathrm{dom}(V(\alpha)),\, S' = \Phi_v(S,\alpha)} Pr_\alpha(v)$. Intuitively, executing α in S leads to $S' = \Phi_v(S,\alpha)$, $v \in \mathrm{dom}(V(\alpha))$, with the probability $Pr_\alpha(S'|S)$.

We say D is *consistent* iff $\emptyset \notin F_\alpha(S)$ (resp., $Pr_\alpha(\emptyset|S) = 0$) for every nondeterministic (resp., probabilistic) action α and every e-state S where α is executable. In the rest of this paper, we implicitly assume that every extended action description $D = (KB, C)$ is consistent.

Example 1 We describe the actions of a goalkeeper in robotic soccer, specifically in the RoboCup Four-Legged League. The extended action description is shown in Fig. 1. It includes the fluents cb (the robot is close to the ball), ba (the ball is in the penalty area), fa (the space ahead the goalkeeper is free), ip (the goalkeeper is in the correct position), bm (the ball is moving towards its own goal), ab (the goalkeeper is aligned with the direction of the ball), and gs (the goal has been saved). The actions are $\mathsf{gotoball}$ (a movement towards the ball, which possibly touches the ball and moves it outside the penalty area), $\mathsf{bodykick}$, $\mathsf{straightkick}$, and $\mathsf{sidekick}$ (three different kinds of kicks with different capabilities), $\mathsf{openlegs}$ (a position for intercepting a ball kicked towards its own goal), $\mathsf{aligntoball}$ (a movement for aligning to the direction of the ball moving towards its own goal), and several sensing actions for some of the fluents.

Note that the action $\mathsf{openlegs}$ has both the deterministic effect that the goalkeeper is able to save the goal when it is aligned to the ball direction, as well as nondeterministic effects, which encode a possible capability of saving the goal even when the alignment is not known. In addition, if the robot is assumed to be always in its own area, then the axiom **caused** ba **if** cb allows for defining indirect effects of actions (e.g., in several actions, the effect $\neg\mathsf{ba}$ indirectly implies $\neg\mathsf{cb}$). Finally, all the fluents are inertial here. □

Conditional Plans

The conditional planning problem (see also the section on related work below) under qualitative and probabilistic uncertainty in our framework can be formulated as follows. Given an extended action description, an initial state description ϕ_I, and a goal description ψ_G, compute the best conditional plan to achieve ψ_G from ϕ_I. In this section, we first define conditional plans in our framework. Using the notion of a belief tree, we then define the goodness of a conditional plan for achieving ψ_G from ϕ_I.

Conditional Plans

A conditional plan is a binary directed tree (i.e., a directed acyclic graph (DAG) in which every node has exactly one parent, except for the *root*, which has no parents; nodes without children are *leaves*), where each arrow represents an action, and each branching expresses the two outcomes of a sensing action, which can thus be used to select the proper actions. Formally, a *conditional plan* Π is either (i) the *empty conditional plan*, denoted λ, or (ii) of form $\alpha\Pi'$, or (iii) of form $\beta; \mathbf{if}\ \omega\ \mathbf{then}\ \{\Pi_\omega\}\ \mathbf{else}\ \{\Pi_{\neg\omega}\}$, where α is an effect action, β is a sensing action of outcomes ω and $\neg\omega$, and Π', Π_ω, and $\Pi_{\neg\omega}$ are conditional plans. We abbreviate "$\pi; \lambda$" by "π", and omit "$\mathbf{else}\ \{\Pi_{\neg\omega}\}$" when $\Pi_{\neg\omega} = \lambda$.

Example 2 We use the domain of Example 1 for defining two planning problems as follows. The first is specified by the initial situation $\phi_I = \mathsf{ba}\sqcap\mathsf{ip}\sqcap\neg\mathsf{bm}$, where the robot is in its standard position, the ball is in its own area, and it is not moving, and the goal $\psi_G = \neg\mathsf{ba}\sqcap\mathsf{ip}$, which requires the robot to kick away the ball and to remain in its position. Some conditional plans for this first problem are:

$$\begin{aligned}\Pi_1 &= \mathsf{gotoball}; \mathsf{bodykick}\\ \Pi_2 &= \mathsf{gotoball}; \mathsf{sensefreeahead};\\ &\quad\ \mathbf{if}\ \mathsf{fa}\ \mathbf{then}\ \{\mathsf{straightkick}\}\ \mathbf{else}\ \{\mathsf{sidekick}\}\,.\end{aligned}$$

The second problem is specified by the initial situation $\phi_I = \mathsf{bm}$, where the ball is moving, and the goal $\psi_G = \mathsf{gs}$, where the goal has been saved. Some conditional plans for this second problem are:

$$\begin{aligned}\Omega_1 &= \mathsf{openlegs} \qquad \Omega_2 = \mathsf{aligntoball}; \mathsf{openlegs}\\ \Omega_3 &= \mathsf{sensealignedtoball}; \mathbf{if}\ \mathsf{ab}\ \mathbf{then}\ \{\mathsf{openlegs}\}\\ &\quad\ \mathbf{else}\ \{\mathsf{aligntoball}; \mathsf{openlegs}\}\,.\ \square\end{aligned}$$

We next define histories h for a conditional plan Π, which are paths of actions from the root to some node in Π and are used to address actions in Π. A *history* h for Π is either (i) the *empty history*, denoted ε, or (ii) of form $\alpha h'$, if $\Pi = \alpha\Pi'$ and h' is a history for Π', or (iii) of form $\alpha_\sigma h_\sigma$, $\sigma \in \{\omega, \neg\omega\}$, if $\Pi = \alpha; \mathbf{if}\ \omega\ \mathbf{then}\ \{\Pi_\omega\}\ \mathbf{else}\ \{\Pi_{\neg\omega}\}$ and h_σ is a history for Π_σ. The *action length* of h is the number of occurrences of effect actions in h. We inductively define $\Pi.h$ as follows: (i) If $h = \varepsilon$ and $\Pi = \alpha R$, then $\Pi.h = \alpha$. (ii) If $h = \alpha h'$ and $\Pi = \alpha\Pi'$, then $\Pi.h = \Pi'.h'$. (iii) If $h = \alpha_\sigma h_\sigma$, $\sigma \in \{\omega, \neg\omega\}$, and $\Pi = \alpha; \mathbf{if}\ \omega\ \mathbf{then}\ \{\Pi_\omega\}\ \mathbf{else}\ \{\Pi_{\neg\omega}\}$, then $\Pi.h = \Pi_\sigma.h_\sigma$.

Belief Trees

We now define belief trees for conditional plans Π under initial observations ϕ. Intuitively, a belief tree is a directed tree over e-states as nodes. Every arrow (eventually with an associated probability) represents a transition, and every branching represents the different possible effects (resp., outcomes) of some effect (resp., sensing) action. Formally, the *belief tree* $T = (G, Pr)$ for Π under ϕ, denoted $T_{\phi G}$, consists of a directed tree $G = (V, E)$, where the nodes are pairs (h, S) of a history h for Π and an e-state S, and a mapping $Pr\colon E \to [0, 1]$, which are constructed by the steps (1)–(3):

(1) Initially, T is only the node (ε, S_ϕ).

(2) Consider the tree $T = (G, Pr)$ with $G = (V, E)$ built thus far. For every leaf (h, S) such that $\Pi.h = \alpha$ is executable in S, enlarge T as follows:

(2.1) If α is a sensing action, then add to T every arrow $(h, S) \to (h\alpha_\sigma, S')$ such that $S' = \Phi(S, \alpha_\sigma) \neq \emptyset$, and σ is a possible outcome of α.

(2.2) If α is deterministic (resp., nondeterministic), then add to T every arrow $(h, S) \to (h\alpha, S')$ such that $S' = \Phi(S, \alpha)$ (resp., $S' \in F_\alpha(S)$).

(2.3) If α is probabilistic, then add to T every arrow $e = (h, S) \to (h\alpha, S')$ such that $S' = \Phi_v(S, \alpha)$ for some $v \in \mathrm{dom}(V(\alpha))$ along with the probability $Pr(e) = \sum_{v \in \mathrm{dom}(V(\alpha)),\, S' = \Phi_v(S,\alpha)} Pr_\alpha(v)$.

(3) Repeat (2) until T is free of leaves (h, S) such that $\Pi.h = \alpha$ is executable in S.

Goodness

We define the goodness of a conditional plan Π for obtaining a goal ψ under an initial observation ϕ using the belief tree $T_{\phi\Pi} = ((V, E), Pr)$ as follows. The success (resp., failure) leaves of $T_{\phi\Pi}$ have the goodness 1 (resp., 0). We then propagate the goodness to every node of $T_{\phi\Pi}$, using the goodness of the children and the probabilities eventually associated with an arrow. The goodness of Π is the goodness of the root of $T_{\phi\Pi}$. Formally, the *goodness* of Π for obtaining ψ given ϕ is defined as $g_{\phi,\psi}(\Pi) = g(R)$, where R is the root of $T_{\phi\Pi} = ((V, E), Pr)$, and $g\colon V \to [0, 1]$ is defined by:

- $g(v) = 1$ (resp., $g(v) = 0$) for every leaf $v = (h, S) \in V$ (*success leaf* (resp., *failure leaf*)) such that $\Pi.h = \varepsilon$ and $\forall s \in S\colon s \models \psi$ (resp., $\Pi.h \neq \varepsilon$ or $\exists s \in S\colon s \not\models \psi$);
- $g(v) = \min_{v \to v' \in E} g(v')$ for every node $v = (h, S) \in V$ such that $\Pi.h$ is either a sensing action, or deterministic effect action, or nondeterministic effect action;
- $g(v) = \sum_{v \to v' \in E} Pr(v \to v') \cdot g(v')$ for all $v = (h, S) \in V$ such that $\Pi.h$ is a probabilistic effect action.

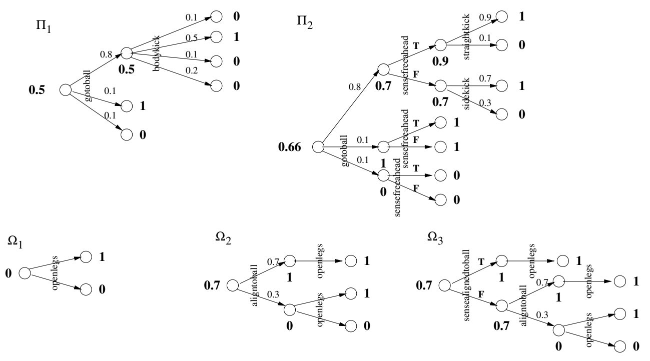

Figure 2: Computing the Goodness of a Conditional Plan

Example 3 The belief trees and the goodness of the conditional plans of Example 2 are shown in Fig. 2. □

Conditional Planning

The problem of conditional planning in our framework can be defined as follows. Given an extended action description $D = (KB, C)$, an initial state description ϕ_I, and a goal description ψ_G, compute a conditional plan Π that, when executed from a state in which ϕ_I is true, reaches a state in which the goal ψ_G is true (for any possible outcome of sensing) with maximum goodness.

We now present a planning method for solving this problem, which is divided into three steps: (1) computing the first-order extension (FOE) of the extended action description; (2) computing all valid conditional plans for the planning problem; (3) evaluating the goodness for each of these conditional plans and selecting the best one (that is, the conditional plan with maximum goodness).

Note that all proofs and further details can be found in the extended report (Iocchi *et al.* 2003).

FOE Generation

Given an epistemic knowledge base Σ consisting of an extended action description $D = (KB, C)$ and an initial state description ϕ_I, the *first-order extension* (*FOE*) of Σ, denoted $FOE(\Sigma)$, is the non-epistemic $\mathcal{ALC}$ knowledge base that consists of the domain constraints, the specification of the initial state, and the assertions which are consequences (up to renaming of individuals) of the epistemic sentences in Σ. The FOE of Σ provides a unique characterization of the knowledge that is shared by all the models of Σ and that is relevant with respect to the planning problem. The computation of the FOE is described in Appendix B.

The FOE constitutes the basis of a sound and complete planning method. More precisely, it is possible to reduce a planning problem in the epistemic knowledge base Σ to an entailment problem in $FOE(\Sigma)$ in the same way as described in (Iocchi, Nardi, & Rosati 2000).

Plan Extraction

The second step of the planning method is achieved by the Plan Generation Algorithm shown in Fig. 3 for extracting

```
Algorithm Plan Generation
Input: $FOE(\Sigma)$, initial state description $\phi_I$, goal description $\psi_G$.
Output: $SP = \{\Pi_i = \{(S_i, A, S_j)\}\}$: set of conditional plans
        with positive goodness.

  $I$ is an initial state in which $\phi_I$ holds;
  $SP = \mathit{findAllPaths}(FOE(\Sigma), I, \psi_G, \emptyset)$;
  while $\exists P \in SP\colon (S_i, A_i, S_j) \in P \ \wedge\ (S_i, \neg A_i, S_k) \notin P$ do
      $SP_{aux} = \mathit{findAllPaths}(FOE(\Sigma), S_k, \psi_G, \mathcal{F}_P(I, S_k))$;
      $SP = SP - \{P\}$;
      foreach $P_{aux} \in SP_{aux}$ do
          $P_{new} = P \cup \{(S_i, \neg A_i, S_k)\} \cup P_{aux}$ ;
          $SP = SP \cup \{P_{new}\}$
      end for
  end while;
  $SP = \mathit{unify}(SP)$;
  return $SP$ .
```

Figure 3: Plan Generation Algorithm

plans from the FOE of the knowledge base Σ. The output of the algorithm is the set of all valid conditional plans for the planning problem. Each plan is a direct acyclic graph represented as a set of tuples (S_i, A_i, S_j), whose meaning is that from the e-state S_i it is possible to execute the action A_i leading to the successor e-state S_j. The action A_i can be either an action without sensing effects or one of the two possible outcomes (true or false) of a sensing effect. In the latter case, also the other outcome (denoted by $\neg A_i$) must be present in the plan starting from the same e-state S_i. If, during the algorithm, this condition is not satisfied for a given (partially built) plan P (that is, $(S_i, A_i, S_j) \in P \wedge (S_i, \neg A_i, S_k) \notin P$), then the state S_i in P must be further expanded.

It uses the function $\mathit{findAllPaths}(FOE(\Sigma), S, \psi_G, \mathcal{F})$, which returns the set of all possible paths (without cycles) in the FOE from the e-state S to an e-state in which ψ_G holds, without considering any of the states specified in $\mathcal{F}$. Since a path is a sequence of actions, in this function, only one outcome of a sensing action or one context of an action with either nondeterministic or probabilistic effects is considered.

The returned linear path may have states that must be further expanded, since they may lack the other outcome of sensing. Moreover, since we are only interested in generating conditional plans (without cycles), it is necessary to limit the search of the paths in $\mathit{findAllPaths}$, by excluding the states that have already been considered in the path from the initial state to the current state: this is obtained by the computation of the set of states $\mathcal{F}_P(I, S_k)$, that includes all the states in the current plan P for which it exists a path to S_k. In this way, $\mathit{findAllPaths}$ never returns a path that can produce cycles when combined with the current plan P.

Observe that the first part of the Plan Generation Algorithm only finds those plans that are valid by considering a single context for each nondeterministic or probabilistic action. However, it is also necessary to derive plans that can be executed when considering at the same time multiple contexts for an action. To this end, it is possible to generate more general plans by combining pairs of previously computed ones. This is performed by a unification operation (implemented by the unify procedure) that unifies pairs of terms that suitably represent conditional plans.

Intuitively, the algorithm first builds all the possible subgraphs of FOE(Σ) that have the following features: (i) there are no states unexpanded (that is, in which only an outcome of sensing has been included), (ii) there are no cycles, and (iii) in each leaf node, the goal ψ_G holds. Then, it completes this set of plans by unifying plans in order to derive more general plans that consider combination of contexts.

Optimal Plan Selection

The third step of the planning method is a procedure that computes the goodness for all the valid conditional plans retrieved in the previous step and returns the best one.

Observe that, for efficiency reasons, the planning algorithm given above generally builds conditional plans in the form of DAGs, while the goodness of such plans is given in terms of conditional plans expressed in the form of trees. In fact, DAGs can be considered as a compact representation of trees and thus the computation of the goodness for such plans can be easily obtained by a transformation of the DAG into a tree (by recursively duplicating those subgraphs in the DAG whose root has more than one parent).

Example 4 The first planning problem of Example 2 admits several other conditional plans besides Π_1 and Π_2. Among them is $\Pi_3 = \mathsf{gotoball}; \mathsf{senseballclose}; \textbf{if } \mathsf{cb} \textbf{ then } \{\textbf{if } \mathsf{fa} \textbf{ then } \{\mathsf{straightkick}\} \textbf{ else } \{\mathsf{sidekick}\}\}$, which has goodness 0.66. The conditional plans Π_2 and Π_3 have the same goodness, which is also the maximally possible goodness, and thus they are both optimal plans. □

Related Work

In comparison with the previous approaches in the literature cited in the introduction, we provide a very rich framework for modeling dynamic systems: epistemic states of the agent, sensing actions, inertial properties, exogenous events, and static domain constraints. Furthermore, we have devised effective techniques for plan generation in this framework (Iocchi, Nardi, & Rosati 2000). As for the modeling of actions with probabilistic effects, the most closely related approach is Poole's independent choice logic (1997; 1998), which uses a similar way of adding probabilities to an approach based on acyclic logic programs. But the central conceptual difference is that Poole's independent choice logic does not allow for qualitative uncertainty in addition to probabilistic uncertainty. Poole circumvents the problem of dealing with qualitative uncertainty by imposing the strong condition of acyclicity on logic programs.

From a more general perspective, our approach is also related to planning under uncertainty in AI, since it can roughly be understood as a combination of conditional planning under nondeterministic uncertainty in AI (Geffner 2002) with conditional planning under probabilistic uncertainty in AI, both in partially observable environments.

Planning under probabilistic uncertainty in AI can be divided into (a) generalizations of classical planning in AI and (b) decision-theoretic planning in AI, which is based on work in operations research and decision science. The planning problems of (a) can be roughly described as follows (Kushmerick, Hanks, & Weld 1994): Given a probability distribution over initial states, probabilistic actions, a set of goal states, and a success threshold θ, compute a sequence of actions that reaches the goal with a probability of at least θ. There are extensions that also allow for observations, and their solutions may also contain conditionals and loops (Draper, Hanks, & Weld 1994). Some planning systems are, e.g., the partial-order planner C-BURIDAN (Draper, Hanks, & Weld 1994), which generalizes BURIDAN (Kushmerick, Hanks, & Weld 1994), the partial-order planner MAHINUR (Onder & Pollack 1999), C-MAXPLAN and ZANDER (Majercik & Littman 2003), and PTLPLAN (Karlsson 2001). Standard planning problems of (b) are either fully observable Markov decision processes (MDPs) (Puterman 1994) or the more general partially observable Markov decision processes (POMDPs) (Kaelbling, Littman, & Cassandra 1998). POMDPs also include costs and/or rewards associated with actions and/or states, and their solutions are mappings from situations to actions that have a high expected utility, rather than courses of actions that achieve a goal with an appropriately high probability. A recent overview on planning under probabilistic uncertainty in AI is given in (Boutilier, Dean, & Hanks 1999).

In summary, our approach can be seen as combining conditional planning under nondeterministic uncertainty and conditional planning under probabilistic uncertainty, where the latter is perhaps closest to generalizations of classical planning in AI. But instead of giving a threshold for the success probability of a plan, we aim at all plans with highest possible success probability. In contrast to the decision-theoretic framework, we do not assume costs and/or rewards associated with actions and/or states. Furthermore, sensing actions in our approach are more flexible than observations in POMDPs, since they allow for preconditions, and they can be performed at any time point when executable.

Summary and Outlook

We have presented the description logic PN-$\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$ for reasoning about actions with sensing under qualitative and probabilistic uncertainty, which is an extension of $\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$ by actions with nondeterministic and probabilistic effects. We have defined a formal semantics of PN-$\mathcal{ALCK}^{\alpha}_{\mathcal{NF}}$ in terms of deterministic, nondeterministic, and probabilistic transitions between epistemic states. We have then introduced the notions of a conditional plan and its goodness under qualitative and probabilistic uncertainty. We have formulated the problem of conditional planning and presented a sound and complete algorithm for solving it. We have also described an application in a robotic-soccer scenario.

As for semantic aspects, an interesting topic of future research is to also allow for sensing actions with noisy outcomes. Moreover, it would be interesting to extend conditional plans to cyclic execution structures. From the computational perspective, an interesting topic of further work is to exploit the notion of goodness to implement heuristics for efficient search in the space of epistemic states, and to provide a detailed complexity analysis of our algorithm and the general problem of conditional planning in our framework.

Acknowledgments. This work has been partially supported by the Austrian Science Fund under project Z29-N04 and by a Marie Curie Individual Fellowship of the European Union programme "Human Potential" under contract number HPMF-CT-2001-001286 (disclaimer: The authors are solely responsible for information communicated and the European Commission is not responsible for any views or results expressed). We are grateful to the reviewers for their constructive comments, which helped to improve our work.

Appendix A: $\mathcal{ALCK}_{\mathcal{NF}}$

The description logic $\mathcal{ALCK}_{\mathcal{NF}}$ models a domain of interest in terms of concepts and roles, which represent classes of individuals and binary relations between classes of individuals, respectively. Formally, we assume a nonempty finite set of *atomic concepts* $\mathcal{A}$ and a nonempty finite set of *atomic roles* $\mathcal{R}$. We use $\bot$ (resp., $\top$) to denote the *bottom* (resp., *top*) *concept*. The set of all *concepts* and *roles* is then inductively defined as follows. Every element of $\mathcal{A} \cup \{\bot, \top\}$ is a concept. If C and D are concepts, and R is a role, then $\neg C$, $C \sqcap D$, $C \sqcup D$, $\exists R.C$, $\forall R.C$, $\mathbf{K}C$, and $\mathbf{A}C$ are concepts. Every element of $\mathcal{R}$ is a role. If R is a role, then $\mathbf{K}R$ and $\mathbf{A}R$ are roles. The operators $\mathbf{K}$ and $\mathbf{A}$ are called the *minimal knowledge operator* and the *default assumption operator*, respectively.

A knowledge base encodes subset relationships between classes, the membership of individuals to classes, and the membership of pairs of individuals to binary relations between classes. Formally, we assume a set of individual names $\mathcal{N}$. An *inclusion axiom* is an expression of the form $C \sqsubseteq D$, where C and D are concepts. A *concept membership axiom* is of the form $C(a)$, where C is a concept, and a is an individual name. A *role membership axiom* has the form $R(a, b)$, where R is a role, and a and b are individual names. A *knowledge base* is a set of inclusion, concept membership, and role membership axioms.

The description logic $\mathcal{ALCK}_{\mathcal{NF}}$ is a special case of Lifschitz's logic MKNF (Lifschitz 1994). Since MKNF has a semantics in possible-world structures, also $\mathcal{ALCK}_{\mathcal{NF}}$ can be given a similar semantics, where each possible world corresponds to a standard description logic interpretation.

Informally, individual names $a \in \mathcal{N}$, the atomic concepts $A \in \mathcal{A}$, the concepts $\bot$ and $\top$, and the atomic roles $P \in \mathcal{R}$ are interpreted with respect to standard description logic interpretations, which consist of a domain Δ and a function associating with the above items elements of Δ, subsets of Δ, the empty set, the set Δ, and binary relations on Δ, respectively. The concepts $\neg C$, $C \sqcap D$, $C \sqcup D$, $\exists R.C$, and $\forall R.C$ are interpreted recursively, as usual. Finally, the operators $\mathbf{K}$ and $\mathbf{A}$ are interpreted with respect to two sets of possible worlds $\mathcal{M}$ and $\mathcal{N}$, respectively, where each possible world is a standard description logic interpretation. For example, $\mathbf{K}C(d)$ encodes that d is "known" to be an instance of C, which holds if d is an instance of C in every possible world of $\mathcal{M}$. Similarly, $\mathbf{A}C(d)$ encodes that d is "assumed to be" an instance of C, which holds if d is an instance of C in every possible world of $\mathcal{N}$.

Formally, a *classical interpretation* $\mathcal{I} = (\Delta, \cdot^{\mathcal{I}})$ consists of a nonempty denumerable *domain* Δ and a function $\cdot^{\mathcal{I}}$ that associates with each individual name from $\mathcal{N}$ an element of Δ (under the usual *unique name assumption*, that is, for any two different individual names $a, b \in \mathcal{N}$, it holds that $a^{\mathcal{I}} \neq b^{\mathcal{I}}$), with each atomic concept from $\mathcal{A}$ a subset of Δ, and with each atomic role from $\mathcal{R}$ a subset of $\Delta \times \Delta$. An *epistemic interpretation* $\mathcal{E} = (\mathcal{I}, \mathcal{M}, \mathcal{N})$ over the domain Δ consists of a classical interpretation $\mathcal{I}$ over the domain Δ and two sets of classical interpretations $\mathcal{M}$ and $\mathcal{N}$ over the domain Δ. The function $\cdot^{\mathcal{E}}$ then interprets individual names, concepts, and roles by induction as follows (where a is an individual name, A is an atomic concept, C and D are concepts, P are atomic roles, and R is a role):

$$\begin{array}{rcl} a^{\mathcal{E}} & = & a^{\mathcal{I}} \\ A^{\mathcal{E}} & = & A^{\mathcal{I}} \\ \bot^{\mathcal{E}} & = & \emptyset \\ \top^{\mathcal{E}} & = & \Delta \\ (\neg C)^{\mathcal{E}} & = & \Delta - C^{\mathcal{E}} \\ (C \sqcap D)^{\mathcal{E}} & = & C^{\mathcal{E}} \cap D^{\mathcal{E}} \\ (C \sqcup D)^{\mathcal{E}} & = & C^{\mathcal{E}} \cup D^{\mathcal{E}} \\ (\exists R.C)^{\mathcal{E}} & = & \{d \in \Delta \mid \exists d' \colon (d, d') \in R^{\mathcal{E}} \text{ and } d' \in C^{\mathcal{E}}\} \\ (\forall R.C)^{\mathcal{E}} & = & \{d \in \Delta \mid \forall d' \colon \text{ if } (d, d') \in R^{\mathcal{E}} \text{ then } d' \in C^{\mathcal{E}}\} \\ (\mathbf{K}C)^{\mathcal{E}} & = & \bigcap\{C^{(\mathcal{J}, \mathcal{M}, \mathcal{N})} \mid \mathcal{J} \in \mathcal{M}\} \\ (\mathbf{A}C)^{\mathcal{E}} & = & \bigcap\{C^{(\mathcal{J}, \mathcal{M}, \mathcal{N})} \mid \mathcal{J} \in \mathcal{N}\} \\ P^{\mathcal{E}} & = & P^{\mathcal{I}} \\ (\mathbf{K}R)^{\mathcal{E}} & = & \bigcap\{R^{(\mathcal{J}, \mathcal{M}, \mathcal{N})} \mid \mathcal{J} \in \mathcal{M}\} \\ (\mathbf{A}R)^{\mathcal{E}} & = & \bigcap\{R^{(\mathcal{J}, \mathcal{M}, \mathcal{N})} \mid \mathcal{J} \in \mathcal{N}\}\,. \end{array}$$

For example, it holds $d \in (\mathbf{K}C)^{(\mathcal{I}, \mathcal{M}, \mathcal{N})}$ iff $d \in C^{(\mathcal{J}, \mathcal{M}, \mathcal{N})}$ for all classical interpretations $\mathcal{J} \in \mathcal{M}$. Furthermore, $d \in (\mathbf{A}\neg C)^{(\mathcal{I}, \mathcal{M}, \mathcal{N})}$ iff $d \in \neg C^{(\mathcal{J}, \mathcal{M}, \mathcal{N})}$ for all classical interpretations $\mathcal{J} \in \mathcal{N}$. Similarly, $d \in (\exists \mathbf{K}R.\top)^{(\mathcal{I}, \mathcal{M}, \mathcal{N})}$ iff some $d' \in \Delta$ exists with $(d, d') \in R^{(\mathcal{J}, \mathcal{M}, \mathcal{N})}$ for all $\mathcal{J} \in \mathcal{M}$.

An epistemic interpretation $\mathcal{E} = (\mathcal{I}, \mathcal{M}, \mathcal{N})$ is a *model* of an inclusion axiom $C \sqsubseteq D$, or $\mathcal{E}$ *satisfies* $C \sqsubseteq D$, denoted $\mathcal{E} \models C \sqsubseteq D$, iff $C^{\mathcal{E}} \subseteq D^{\mathcal{E}}$. The epistemic interpretation $\mathcal{E}$ is a *model* of a concept (resp., role) membership axiom $C(a)$ (resp., $R(a, b)$), or $\mathcal{E}$ *satisfies* $C(a)$ (resp., $R(a, b)$), denoted $\mathcal{E} \models C(a)$ (resp., $\mathcal{E} \models R(a, b)$), iff $a^{\mathcal{E}} \in C^{\mathcal{E}}$ (resp., $(a^{\mathcal{E}}, b^{\mathcal{E}}) \in R^{\mathcal{E}}$). The epistemic interpretation $\mathcal{E}$ is a *model* of a knowledge base KB iff it is a model of every $F \in KB$.

We finally define satisfiability and logical consequence for knowledge bases KB in terms of preferred models of KB. A model $\mathcal{E} = (\mathcal{I}, \mathcal{M}, \mathcal{N})$ of KB is a *preferred* model of KB iff (i) $\mathcal{I} \in \mathcal{M}$, (ii) $\mathcal{M} = \mathcal{N}$, (iii) $(\mathcal{J}, \mathcal{M}, \mathcal{N}) \models KB$ for all $\mathcal{J} \in \mathcal{M}$, and (iv) $\mathcal{M}$ is maximal with (iii) (that is, there exists no $\mathcal{M}' \supset \mathcal{M}$ such that $(\mathcal{J}, \mathcal{M}', \mathcal{N}) \models KB$ for all $\mathcal{J} \in \mathcal{M}'$). A knowledge base KB is *satisfiable* (resp., *unsatisfiable*) iff KB has a (resp., no) preferred model. An axiom F is a *logical consequence* of KB, denoted $KB \models F$, iff every preferred model of KB is also a model of F.

Appendix B: Computing the FOE

Given an epistemic knowledge base $\Sigma = \Gamma_S \cup \Gamma_D \cup \Gamma_I$, consisting of the domain constraints Γ_S, the action descriptions Γ_D, and an initial state description Γ_I, the *first-order extension* of Σ, denoted $FOE(\Sigma)$, is the $\mathcal{ALC}$ knowledge base which consists of (i) the domain constraints in Γ_S, (ii) the specification of the initial state through the assertions in Γ_I, and (iii) the assertions that are consequences (up to renaming of individuals) of Σ. The FOE of Σ provides a unique characterization of the knowledge that is shared by all the models of Σ and that is relevant w.r.t. the planning problem.

To compute the FOE, we replace each sensing action R_S by two special actions R_S^+ and R_S^-. We denote by $\Gamma_D^{\pm}$ the set of axioms Γ_D in which those for the sensing actions R_S are replaced by the following axioms:

$$\mathbf{K}\phi \sqsubseteq \exists \mathbf{K}R_S^+.\top \qquad \top \sqsubseteq \forall R_S^+.S$$
$$\mathbf{K}\phi \sqsubseteq \exists \mathbf{K}R_S^-.\top \qquad \top \sqsubseteq \forall R_S^-.\neg S\,.$$

A nondeterministic or a probabilistic action R_N is replaced by n actions R_N^i, with the same preconditions and deterministic effects and different nondeterministic effects

$$\mathbf{K}\phi \sqsubseteq \exists \mathbf{K}R_N^i.\top \quad \mathbf{K}\phi \sqsubseteq \forall R_N^i.\psi \quad \mathbf{K}(\phi \sqcap V_{R_N}^{v_i}) \sqsubseteq \forall R_N^i.\psi$$

for every context v_i, $i \in \{1, \ldots, n\}$. We also use only a finite number of instances of the default frame axioms

$$\mathbf{K}L \sqsubseteq \forall \mathbf{K}R_S^{\pm}.\mathbf{A}\neg L \sqcup \mathbf{K}L \quad \mathbf{K}L \sqsubseteq \forall \mathbf{K}R_N^i.\mathbf{A}\neg L \sqcup \mathbf{K}L$$

```
Algorithm FOE Computation
Input: $\Sigma = \Gamma_S \cup \Gamma_D^{\pm} \cup \Gamma_I$
Output: $FOE(\Sigma)$

    Procedure $create\_new\_state(s, R)$
    $s'$ = new state name;
    $\mathcal{A}' = \mathcal{A} \cup \{R(s, s')\} \cup \{D(s') \mid$
            $D \in post(s, R, \Gamma_S \cup \mathcal{A}, \Gamma_D^{\pm})\}$;
    if $\exists s'' \in all\_states$ such that
    $concepts(\Gamma_S \cup \mathcal{A}, s'') = concepts(\Gamma_S \cup \mathcal{A}', s')$ then
      $\mathcal{A} = \mathcal{A} \cup R(s, s'')$
    else
      $\mathcal{A} = \mathcal{A}'$;
      $active\_states = active\_states \cup \{s'\}$;
      $all\_states = all\_states \cup \{s'\}$
    end if;

$active\_states = \{init\}$;
$all\_states = \{init\}$;
$\mathcal{A} = \Gamma_I$;
repeat
  $s = choose(active\_states)$;
  for each action $R$ do
    if $\exists \mathbf{K}C \sqsubseteq \exists \mathbf{K}R.\top \in \Gamma_D$ such that $\Gamma_S \cup \mathcal{A} \models C(s)$ then
      if $R$ has no sensing effect then
        $create\_new\_state(s, R)$
      else
        $create\_new\_state(s, R^+)$;
        $create\_new\_state(s, R^-)$
      end if
    end if
  end for;
  $active\_states = active\_states - \{s\}$;
until $active\_states = \emptyset$;
return $\Gamma_S \cup \mathcal{A}$.
```

Figure 4: FOE Computation Algorithm

obtained by instantiating the frame axiom schema for each atomic concept $L = A$ or its negation $L = \neg A$.

The FOE of Σ is computed by the algorithm shown in Fig. 4. Here, $concepts(\Gamma_S \cup \mathcal{A}, s) = \{C \mid \Gamma_S \cup \mathcal{A} \models C(s)\}$ denotes the set of concepts that are *valid* for the explicitly named individual s, occurring in the set of instance assertions $\mathcal{A}$, with respect to the $\mathcal{ALC}$ knowledge base $\Gamma_S \cup \mathcal{A}$. Moreover, $post(s, R, \Gamma_S \cup \mathcal{A}, \Gamma_D^{\pm}) = \{D \mid \mathbf{K}C \sqsubseteq \forall R.D \in \Gamma_D^{\pm}$ and $\Gamma_S \cup \mathcal{A} \models C(s)\}$ denotes the effect of the application of all the triggered rules belonging to the set $\Gamma_D^{\pm}$ involving the action R in the state s, namely the set of postconditions (concepts) of the rules that are triggered by s.

Informally, starting from the initial state $init$, the algorithm applies to each state the rules in the set $\Gamma_D^{\pm}$ which are triggered by such a state. A new state is thus generated, unless a state with the same properties has already been created. In this way, the effect of the rules is computed, obtaining a sort of "completion" of the knowledge base.

The FOE of Σ is unique, that is, every order of extraction of the states from $active\_states$ produces the same set of assertions, up to renaming of states. Furthermore, the algorithm terminates, that is, the condition $active\_states = \emptyset$ is eventually reached, since the number of states generated is bound by the number of axioms in $\Gamma_D^{\pm}$. More precisely, the number of generated states n_s is given by $n_s \leq 2^{n_r} + 1$, where n_r is the number of axioms in $\Gamma_D^{\pm}$.

Finally, $concepts(\Gamma_S \cup \mathcal{A}, s) = concepts(\Gamma_S \cup \mathcal{A}', s')$ can be checked by verifying whether, for each concept C such that either $C(init) \in \Gamma_I$ or $\mathbf{K}C$ is in the postcondition of some axiom in Γ_D: $\Gamma_S \cup \mathcal{A} \models C(s)$ iff $\Gamma_S \cup \mathcal{A}' \models C(s')$.

References

Bacchus, F.; Halpern, J. Y.; and Levesque, H. J. 1999. Reasoning about noisy sensors and effectors in the situation calculus. *Artif. Intell.* 111(1–2):171–208.

Baral, C.; Tran, N.; and Tuan, L.-C. 2002. Reasoning about actions in a probabilistic setting. In *Proceedings AAAI/IAAI-2002*, 507–512.

Boutilier, C.; Reiter, R.; Soutchanski, M.; and Thrun, S. 2000. Decision-theoretic, high-level agent programming in the situation calculus. In *Proceedings AAAI/IAAI-2000*, 355–362.

Boutilier, C.; Dean, T.; and Hanks, S. 1999. Decision-theoretic planning: Structural assumptions and computational leverage. *J. Artif. Intell. Res.* 11:1–94.

Boutilier, C.; Reiter, R.; and Price, B. 2001. Symbolic dynamic programming for first-order MDPs. In *Proceedings IJCAI-2001*, 690–700.

Donini, F. M.; Nardi, D.; and Rosati, R. 2002. Description logics of minimal knowledge and negation as failure. *ACM Transactions on Computational Logic (TOCL)* 3(2):1–49.

Draper, D.; Hanks, S.; and Weld, D. S. 1994. Probabilistic planning with information gathering and contingent execution. In *Proceedings AIPS-1994*, 31–36.

Eiter, T., and Lukasiewicz, T. 2003. Probabilistic reasoning about actions in nonmonotonic causal theories. In *Proceedings UAI-2003*, 192–199.

Eiter, T.; Faber, W.; Leone, N.; Pfeifer, G.; and Polleres, A. 2003. A logic programming approach to knowledge-state planning, II: The DLV$^{\mathcal{K}}$ system. *Artif. Intell.* 144(1–2):157–211.

Finzi, A., and Pirri, F. 2001. Combining probabilities, failures and safety in robot control. In *Proceedings IJCAI-2001*, 1331–1336.

Geffner, H. 2002. Perspectives on artificial intelligence planning. In *Proceedings AAAI/IAAI-2002*, 1013–1023.

Gelfond, M., and Lifschitz, V. 1993. Representing action and change by logic programs. *J. Logic Program.* 17:301–322.

Giunchiglia, E.; Lee, J.; Lifschitz, V.; McCain, N.; and Turner, H. 2004. Nonmonotonic causal theories. *Artif. Intell.* 153:49–104.

Großkreutz, H., and Lakemeyer, G. 2001. Belief update in the pGOLOG framework. In *Proceedings KI/ÖGAI-2001*, volume 2174 of *LNCS/LNAI*, 213–228.

Großkreutz, H. 2002. *Towards More Realistic Logic-Based Robot Controllers in the GOLOG Framework*. Ph.D. Dissertation, RWTH Aachen.

Halpern, J. Y., and Tuttle, M. R. 1993. Knowledge, probability, and adversaries. *J. ACM* 40(4):917–962.

Iocchi, L.; Lukasiewicz, T.; Nardi, D.; and Rosati, R. 2003. Reasoning about actions with sensing under qualitative and probabilistic uncertainty. Technical Report INFSYS RR-1843-03-05, Institut für Informationssysteme, TU Wien.

Iocchi, L.; Nardi, D.; and Rosati, R. 2000. Planning with sensing, concurrency, and exogenous events: Logical framework and implementation. In *Proceedings KR-2000*, 678–689.

Kaelbling, L. P.; Littman, M. L.; and Cassandra, A. R. 1998. Planning and acting in partially observable stochastic domains. *Artif. Intell.* 101(1–2):99–134.

Karlsson, L. 2001. Conditional progressive planning under uncertainty. In *Proceedings IJCAI-2001*, 431–438.

Kushmerick, N.; Hanks, S.; and Weld, D. S. 1994. An algorithm for probabilistic least-commitment planning. In *Proceedings AAAI-1994*, 1073–1078.

Levesque, H. J. 1996. What is planning in presence of sensing? In *Proceedings AAAI-1996*, 1139–1149.

Lifschitz, V. 1994. Minimal belief and negation as failure. *Artif. Intell.* 70(1–2):53–72.

Lobo, J.; Mendez, G.; and Taylor, S. R. 2001. Knowledge and the action description language $\mathcal{A}$. *Theory and Practice of Logic Programming (TPLP)* 1(2):129–184.

Majercik, S. M., and Littman, M. L. 2003. Contingent planning under uncertainty via stochastic satisfiability. *Artif. Intell.* 147(1–2):119–162.

Mateus, P.; Pacheco, A.; Pinto, J.; Sernadas, A.; and Sernadas, C. 2001. Probabilistic situation calculus. *Ann. Math. Artif. Intell.* 32(1–4):393–431.

Mateus, P.; Pacheco, A.; and Pinto, J. 2002. Observations and the probabilistic situation calculus. In *Proc. KR-2002*, 327–338.

Onder, N., and Pollack, M. E. 1999. Conditional, probabilistic planning: A unifying algorithm and effective search control mechanisms. In *Proceedings AAAI/IAAI-1999*, 577–584.

Poole, D. 1997. The independent choice logic for modelling multiple agents under uncertainty. *Artif. Intell.* 94(1–2):7–56.

Poole, D. 1998. Decision theory, the situation calculus and conditional plans. *Electronic Transactions on Artificial Intelligence* 2(1–2):105–158.

Puterman, M. L. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley.

Reiter, R. 2001. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. MIT Press.

Son, T. C., and Baral, C. 2001. Formalizing sensing actions — A transition function based approach. *Artif. Intell.* 125(1–2):19–91.