Reasoning about Actions with Sensing under Qualitative and Probabilistic Uncertainty

LUCA IOCCHI, THOMAS LUKASIEWICZ, DANIELE NARDI, and RICCARDO ROSATI
Sapienza Università di Roma

We focus on the aspect of sensing in reasoning about actions under qualitative and probabilistic uncertainty. We first define the action language $\mathcal{E}$ for reasoning about actions with sensing, which has a semantics based on the autoepistemic description logic $\mathcal{ALCK}_{\mathcal{NF}}$, and which is given a formal semantics via a system of deterministic transitions between epistemic states. As an important feature, the main computational tasks in $\mathcal{E}$ can be done in linear and quadratic time. We then introduce the action language $\mathcal{E}+$ for reasoning about actions with sensing under qualitative and probabilistic uncertainty, which is an extension of $\mathcal{E}$ by actions with nondeterministic and probabilistic effects, and which is given a formal semantics in a system of deterministic, nondeterministic, and probabilistic transitions between epistemic states. We also define the notion of a belief graph, which represents the belief state of an agent after a sequence of deterministic, nondeterministic, and probabilistic actions, and which compactly represents a set of unnormalized probability distributions. Using belief graphs, we then introduce the notion of a conditional plan and its goodness for reasoning about actions under qualitative and probabilistic uncertainty. We formulate the problems of optimal and threshold conditional planning under qualitative and probabilistic uncertainty, and show that they are both uncomputable in general. We then give two algorithms for conditional planning in our framework. The first one is always sound, and it is also complete for the special case in which the relevant transitions between epistemic states are cycle-free. The second algorithm is a sound and complete solution to the problem of finite-horizon conditional planning in our framework. Under suitable assumptions, it computes every optimal finite-horizon conditional plan in polynomial time. We also describe an application of our formalism in a robotic-soccer scenario, which underlines its usefulness in realistic applications.

Categories and Subject Descriptors: I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods—*Representation languages*

General Terms: Languages, Algorithms

Additional Key Words and Phrases: reasoning about actions, sensing, qualitative and probabilistic uncertainty, action languages, description logics, imprecise probabilities

Author's addresses: DIS, Sapienza Università di Roma, Via Ariosto 25, 00185 Rome, Italy; e-mail: {iocchi, lukasiewicz, nardi, rosati}@dis.uniroma1.it. Alternative address of T. Lukasiewicz: Institut für Informationssysteme, TU Wien, Favoritenstraße 9-11, 1040 Vienna, Austria; e-mail: lukasiewicz@kr.tuwien.ac.at.

This paper is a significantly extended and revised version of two papers that appeared in: *Proceedings of the 10th International Workshop on Non-Monotonic Reasoning (NMR-2004)*, pp. 240–248, 2004; *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI-2004)*, pp. 818–822, IOS Press, 2004.

This work has been partially supported by the German Research Foundation (DFG) under the Heisenberg Programme, by the Austrian Science Fund Projects P18146-N04 and Z29-N04, and by a Marie Curie Individual Fellowship of the EU 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 of this paper and its NMR-2004 and ECAI-2004 abstracts for their constructive comments, which helped to improve this work.

1. INTRODUCTION

Representation and reasoning about actions is a basic component for the design of cognitive robots. In reasoning about the actions of mobile robots operating 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], easily allow for expressing qualitative uncertainty about the initial situation of the world and the effects of actions through disjunctive knowledge. Similarly, recent formalisms for reasoning about actions that are inspired by the early action language $\mathcal{A}$ [Gelfond and 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.

The need for dealing with quantitative uncertainty has lead to a number of proposals for probabilistic reasoning about actions. They include probabilistic extensions of the situation calculus [Bacchus et al. 1999; Mateus et al. 2001], of logic programming formalisms [Poole 1997], and of the action language $\mathcal{A}$ [Baral et al. 2002].

Even though there is extensive work on reasoning about actions under qualitative and probabilistic uncertainty separately, there is relatively little work that orthogonally combines qualitative and probabilistic uncertainty in a uniform framework for reasoning about actions. One seminal such approach is due to Halpern and Tuttle [1993], which combines nondeterminism and probabilistic uncertainty in a game-theoretic framework. In particular, Halpern and Tuttle [1993] draw the following important conclusion:

> " This discussion leads us to conclude 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. The following example illustrates this strong need for modeling both qualitative and probabilistic uncertainty.

Example 1.1 *(Robotic Soccer)* In a robotic soccer domain, the action "align to ball" may succeed resp. fail with the probability 0.7 resp. 0.3, while the goalkeeper's action "open legs" may either save the goal or not save the goal. That is, the former action has probabilistic effects, while the latter action has nondeterministic effects. More precisely, in the latter case, it may not be possible to assign probabilities to the possible effects, which in fact depend on external factors (such as the speed and the kind of kick performed by an opponent robot) and thus cannot be given a priori. That is, we only know that the goalkeeper's action "open legs" may save the goal resp. not save the goal with the probability p resp. $1-p$, where the value $p \in [0,1]$ is unknown. Hence, rather than having exactly one probability distribution, we have the very different situation of a set of possible probability distributions for the effects of an action. Observe in particular that we cannot simply assume the uniform distribution, that is, that $p = 1 - p = 0.5$ holds.

The work [Eiter and Lukasiewicz 2003] is among the few papers that orthogonally combine qualitative and probabilistic uncertainty in a uniform framework for reasoning about actions. However, this approach does not deal with the crucial issue of *sensing* in reasoning about actions under qualitative and probabilistic uncertainty, which is needed to operate in dynamic environments in which it is not possible to acquire all the necessary information before executing a task (that is, in the initial state). In contrast to actions that change the state of the world (which are thus also called *physical actions*), *sensing actions* in reasoning about actions (see especially [Levesque 1996; Lobo et al. 1997; Son et al. 2004]) are actions that change the knowledge about the state of the world, that is, they allow an agent or a robot to obtain information about certain properties of the world. Sensing actions are strongly motivated by the overwhelming part of real-world applications where the initial state of the world is not fully specified or where exogenous actions may occur, and consequently an agent or a robot is forced to use sensors of some sort to determine the values of certain properties of the world. One important way to represent the sensing capabilities of the robotic agent is through an *epistemic* operator, which allows to distinguish what the agent *knows* from what is true in the world [Levesque 1996; Iocchi et al. 2000].

In this paper, we develop a formalism that allows for *sensing* in reasoning about actions under *qualitative* and *probabilistic uncertainty*, thus formulating and addressing the problem of conditional planning under qualitative and probabilistic uncertainty. The proposed formalism provides a complete integration of the notion of *qualitative belief*, with that of *probabilistic belief*. Furthermore, we show that, in this setting, under rather feasible hypotheses, the basic reasoning task can be solved in polynomial time.

More specifically, the contributions of this paper can be summarized as follows:

—We present the action language $\mathcal{E}$ for reasoning about actions with sensing. We define a formal semantics of action descriptions in $\mathcal{E}$ by systems of transitions between *epistemic states* (or *e-states*), which are sets of possible states of the world. We show that all basic computational tasks in $\mathcal{E}$ (among which there are especially the tasks of deciding whether an action is executable in an e-state, and of computing the successor e-state after executing an action in an e-state) can be done in linear resp. quadratic time.

—We define the action language $\mathcal{E}+$ for reasoning about actions with sensing under qualitative and probabilistic uncertainty, which is an extension of the action language $\mathcal{E}$ by actions with nondeterministic and probabilistic effects. Note that such an extension can also be defined for $\mathcal{C}+$ and related action languages as core action language instead of $\mathcal{E}$. We define a formal semantics of action descriptions in $\mathcal{E}+$ through systems of deterministic, nondeterministic, and probabilistic transitions between e-states.

—We introduce the concept of a belief graph, which represents the belief state of an agent after a sequence of deterministic, nondeterministic, and probabilistic actions. We also define the notions of lower and upper probabilities of fluent formulas in belief graphs, and we finally prove the important result that every belief graph is a compact representation of a set of unnormalized probability distributions, which intuitively shows that combining nondeterminism with precise probabilities leads to imprecise probabilities.

—We introduce the concept of a conditional plan in our framework for reasoning about actions under qualitative and probabilistic uncertainty. We define the notion of goodness of a conditional plan for achieving a goal from an initial observation, and the problems of optimal and threshold conditional planning under qualitative and probabilistic uncertainty. We then show that both problems are uncomputable in the general case.

—We present an algorithm for cycle-free conditional planning under qualitative and probabilistic uncertainty, which computes a set of conditional plans with goodness above a given threshold $\theta \geqslant 0$. The algorithm is always sound, and it is also complete when the relevant transition system between e-states is acyclic. That is, in the latter case, the algorithm returns the set of *all* conditional plans with goodness above θ.

—We also present an algorithm for finite-horizon conditional planning under qualitative and probabilistic uncertainty, which computes all optimal conditional plans of length below a given horizon $h \geqslant 0$. An important feature of this algorithm is that every optimal conditional plan can be computed in polynomial time, when the horizon is bounded by a constant, which is a reasonable assumption in many applications in practice.

The concepts and techniques presented in this paper are illustrated along a robotic-soccer scenario. The robotic application is implemented with a heterogeneous layered architecture [Iocchi 1999], where the formalism presented in this article is used to drive the high-level behavior of the system (that is, to select high-level actions to perform), while a numerical level is responsible for sensor processing and action execution. The heterogenous representation of the information within the system allows for appropriately integrating various techniques (such as image processing, probabilistic localization, fuzzy control, path planning procedures, and specialized control techniques). The layered architecture also allows for the effective implementation of complex behaviors, even though the used formalism is propositional. Such a scenario thus gives evidence of the usefulness of our formalism in realistic applications.

The rest of this paper is organized as follows. In Section 2, we define the action language $\mathcal{E}$. Section 3 extends $\mathcal{E}$ by actions with nondeterministic and probabilistic effects. In Section 4, we introduce the concept of a belief graph, and in Section 5, we formally define the conditional planning problem in our framework. Sections 6 and 7 provide algorithms for cycle-free and finite-horizon conditional planning in our framework, respectively. In Section 8, we discuss related work. Section 9 summarizes the main results and gives an outlook on future research. A notation table is given in Appendix A. To not distract from the flow of reading, some technical details have been moved to Appendix B.

2. THE ACTION LANGUAGE $\mathcal{E}$

In this section, we introduce the action language $\mathcal{E}$, which is syntactically similar to the action language $\mathcal{A}$ and its variants including the recent $\mathcal{C}+$, but which has a formal semantics in description logics. More precisely, it is equivalent to a fragment of the autoepistemic description logic $\mathcal{ALCK}_{\mathcal{NF}}$ [Donini et al. 2002] for modeling dynamic systems (see [Iocchi et al. 2006] for the proof that $\mathcal{E}$ is semantically founded on $\mathcal{ALCK}_{\mathcal{NF}}$), which has been successfully implemented and used for a robotic soccer team [Iocchi et al. 2000].

As a central feature, the action language $\mathcal{E}$ allows for sensing actions and for modeling the *epistemic state* of an agent, which is the set of all world states that the agent considers possible in a given situation. Intuitively, the epistemic state encodes what the agent knows about the world, in contrast to what is true in the world [Levesque 1996; Son and Baral 2001]. Reasoning about actions in the presence of sensing is then done by modeling the dynamics of the agent's epistemic state, rather than the dynamics of the world.

A dynamic system is specified in $\mathcal{E}$ through an initial state description and an action description, which express what an agent knows about the initial properties of the world and how this knowledge changes through the execution of actions, respectively. We now describe the syntax and the semantics of initial state and action descriptions.

2.1 Syntax

An action description in $\mathcal{E}$ consists of a set of formulas that encode dynamic knowledge about the preconditions and effects of actions as well as static background knowledge about the world. The states and properties of the world are described through fluent formulas, which are Boolean combinations of elementary propositions, called fluents. They may directly or indirectly change through the execution of actions.

We first define fluents, actions, and fluent formulas. We assume a nonempty finite set of *fluents* $\mathcal{F}$ and a nonempty finite set of *actions* $\mathcal{A}$, which are divided into *physical actions* and *sensing actions* (with binary sensing outcome). We use $\bot$ and $\top$ to denote the constants false and true, respectively. The set of *fluent formulas* is the closure of $\mathcal{F} \cup \{\bot, \top\}$ under the Boolean operators $\neg$ and $\wedge$ (that is, if ϕ and ψ are fluent formulas, then also $\neg\phi$ and $(\phi \wedge \psi)$). We use $(\phi \vee \psi)$ and $(\psi \Leftarrow \phi)$ to abbreviate $\neg(\neg\phi \wedge \neg\psi)$ and $\neg(\phi \wedge \neg\psi)$, respectively, and adopt the usual conventions to eliminate parentheses. A *fluent literal* ℓ is either a fluent f or the negation of a fluent $\neg f$. A *fluent conjunction* ϕ is either $\bot$, or $\top$, or a fluent formula of the form $\ell_1 \wedge \cdots \wedge \ell_n$, where $\ell_1, \ldots, \ell_n$ are fluent literals and $n \geqslant 1$.

We next introduce precondition, conditional effect, sensing effect, default frame, and domain constraint axioms in the action language $\mathcal{E}$. We use *precondition axioms* to encode the preconditions of actions. They are expressions of the form

$$\textbf{executable } \alpha \textbf{ if } \phi\,, \tag{1}$$

where ϕ is a fluent conjunction, and α is an action. Informally, the axiom (1) encodes that the action α is executable in every state that satisfies ϕ. In particular, if $\phi = \top$, then α is always executable. We use *conditional effect axioms* to represent the different conditional effects of physical actions. They are of the form

$$\textbf{caused } \psi \textbf{ after } \alpha \textbf{ when } \phi\,, \tag{2}$$

where ϕ and ψ are fluent conjunctions, and α is a physical action. Informally, the axiom (2) encodes that if the current state of the world satisfies ϕ, then the successor state after executing the action α satisfies ψ. If $\phi = \top$, then the axiom (2) is also called an *effect axiom* and abbreviated as **caused** ψ **after** α. *Sensing effect axioms* associate with sensing actions their possible two sensing outcomes. They have the form

$$\textbf{caused to know } \omega \textbf{ or } \neg\omega \textbf{ after } \alpha\,, \tag{3}$$

where ω is a fluent literal, and α is a sensing action. Informally, after executing α, the agent knows that ω is either true or false. Note that, for ease of presentation, we consider only sensing actions with two outcomes, but the formalism and all our results can be easily extended to sensing actions with more than two outcomes. *Default frame axioms* associate with actions properties of the world that they generally do not change. They are of the form

$$\textbf{inertial } \phi \textbf{ after } \alpha\,, \tag{4}$$

where ϕ is a fluent conjunction, and α is a physical action. Informally, if ϕ holds in the current state of the world, then ϕ also holds in the successor state after executing the action α, if this is consistent with the effects of α. Finally, *domain constraint axioms* describe background knowledge, and are of the form

$$\textbf{caused}\ \psi\ \textbf{if}\ \ell\,, \tag{5}$$

where ℓ is a fluent literal, and ψ is a fluent conjunction. Informally, every state of the world that satisfies ℓ should also satisfy ψ. Such an axiom (5) represents static background knowledge about the world, which is invariant relative to the execution of actions.

We are now ready to define the notions of an initial state description and of an action description as follows. An *initial state description* δ_I is a fluent conjunction. An *action description* AD is a finite set of precondition axioms, conditional effect axioms, sensing effect axioms, default frame axioms, and domain constraint axioms.

The following example shows how some actions of a goalkeeper in robotic soccer (RoboCup Four-Legged League) can be expressed in the action language $\mathcal{E}$.

Example 2.1 *(Robotic Soccer cont'd)* The fluents are ballclose (the goalkeeper is close to the ball), ballinarea (the ball is in the penalty area), freeahead (the space ahead the goalkeeper is free), inposition (the goalkeeper is in the correct position), ballmoving (the ball is moving towards the goal), alignedtoball (the goalkeeper is aligned with the direction of the ball), and goalsaved (the goal has been saved). We assume the physical actions gotoball (a movement towards the ball, which may touch the ball and move it outside the penalty area), bodykick, straightkick, and sidekick (three different kinds of kicks with different capabilities), openlegs (a position for intercepting a ball kicked towards the goal), and aligntoball (a movement for aligning to the direction of the ball moving towards the goalkeeper's own goal), as well as several sensing actions for some of the properties.

An action description is shown in Fig. 1. In particular, the action gotoball is executable only if the ball is in the penalty area and not moving towards the goal (1). The action openlegs has the effect that the goalkeeper is able to save the goal when it is aligned to the ball direction (8), which encodes a possible capability of saving the goal even when the alignment is unknown. After the sensing action senseballclose, the goalkeeper knows if the ball is close or not (9). All fluents are inertial (12), and thus they generally do not change through the execution of an action. Finally, the ball is in the penalty area, if the goalkeeper is close to the ball (13), since we assume that the goalkeeper is always in its own area.

2.2 Semantics

An initial state description δ_I represents an epistemic state, which is a set of possible states of the world, while an action description AD encodes a system of transitions between epistemic states (which forms a directed graph where the nodes represent epistemic states and the arrows encode transitions between epistemic states through actions).

We first define states and epistemic states, which are truth assignments to the fluents resp. sets of states that satisfy every domain constraint axiom in AD and that are representable by a fluent conjunction. Formally, a *state* s of an action description AD is a truth assignment to the fluents in $\mathcal{F}$. A set of states S *satisfies* a fluent formula ϕ, denoted $S \models \phi$, iff every $s \in S$ satisfies ϕ. It *satisfies* a domain constraint axiom **caused** ψ **if** ℓ iff either $S \not\models \ell$ or $S \models \psi$.[1] An *epistemic state* (or *e-state*) S of AD is a nonempty set of states s of AD

[1]Notice that this definition provides an epistemic interpretation of domain constraints, which is different from the usual interpretation. Based on such an interpretation, the constraint can be read as follows: if ℓ is *known* in the epistemic state S (that is, is true in every state s belonging to S), then ψ is known in S.

(i) precondition axioms:

(1) **executable** gotoball **if** ballinarea$\wedge\neg$ballmoving
(2) **executable** bodykick **if** ballclose
(3) **executable** straightkick **if** ballclose$\wedge$freeahead
(4) **executable** sidekick **if** ballclose$\wedge\neg$freeahead
(5) **executable** aligntoball **if** ballmoving
(6) **executable** openlegs **if** ballmoving
(7) **executable** sensealignedtoball **if** ballmoving

(ii) conditional effect axioms and effect axioms:

(8) **caused** goalsaved **after** openlegs **when** alignedtoball
(9) **caused** ballclose **after** gotoball
(10) **caused** $\neg$ballinarea **after** bodykick
(11) **caused** $\neg$ballinarea **after** straightkick
(12) **caused** $\neg$ballinarea **after** sidekick

(iii) sensing effect axioms:

(13) **caused to know** ballclose **or** $\neg$ballclose **after** senseballclose
(14) **caused to know** freeahead **or** $\neg$freeahead **after** sensefreeahead
(15) **caused to know** alignedtoball **or** $\neg$alignedtoball **after** sensealignedtoball

(iv) default frame axioms:

(16) **inertial** ℓ **after** α (for every fluent literal ℓ and every action α)

(v) domain constraint axioms:

(17) **caused** ballinarea **if** ballclose

Fig. 1. Robotic Soccer Example: Action description AD.

such that (i) S satisfies every domain constraint axiom in AD, and (ii) there exists a fluent conjunction ϕ such that S is the set of all states s of AD that satisfy ϕ.

We next define the executability of actions in e-states and the transitions between e-states through the execution of physical and sensing actions. An action α is *executable* in an e-state S of AD iff $S \models \phi$ for every precondition axiom **executable** α **if** ϕ in AD.

Given an e-state S of AD and a physical action α that is executable in S, let $direct(S, \alpha)$ denote the conjunction of all ψ such that **caused** ψ **after** α **when** ϕ is in AD and $S \models \phi$. We say that S' is a *successor e-state* of S under the physical action α iff S' is an e-state of AD such that (i) S' satisfies $direct(S, \alpha)$, (ii) S' satisfies every domain constraint axiom in AD, and (iii) S' satisfies a maximal subset of default frame axioms (that is, there exists no $S'' \neq \emptyset$ such that (1) $S'' \subset S'$, (2) S'' satisfies $direct(S, \alpha)$, (3) S'' satisfies every domain constraint axiom in AD, and (4) there exists a default frame axiom **inertial** ϕ **after** α in AD such that $S \models \phi$, $S' \not\models \phi$ and $S'' \models \phi$). Intuitively, a successor e-state of S under α encodes the direct effects of α (expressed through $direct(S, \alpha)$), the indirect effects due to the domain constraint axioms, and a maximal propagation of inertial properties that are consistent with these direct and indirect effects.

Analogously, S' is a *successor e-state* of S under a sensing action α with outcome $o \in \{\omega, \neg\omega\}$ iff S' is an e-state of AD such that (i) S' satisfies o, (ii) S' satisfies every domain constraint axiom in AD, and (iii) S' satisfies a maximal subset of default frame axioms (that is, no $S'' \neq \emptyset$ exists such that (1) $S'' \subset S'$, (2) S'' satisfies o, (3) S'' satisfies every domain constraint axiom in AD, and (4) there is a default frame axiom **inertial** ϕ **after** α in AD with $S \models \phi$, $S' \not\models \phi$ and $S'' \models \phi$). Intuitively, a successor e-state of S under a sensing action α encodes the sensing outcome of α, the indirect effects due to the domain constraint axioms, and the propagation of inertial properties consistent with them.

The following result shows an important uniqueness property for successor e-states, namely that there exists at most one successor e-state of an e-state S of AD under a physical action α (resp., a sensing action α with outcome o), denoted $\Phi(S,\alpha)$ (resp., $\Phi(S,\alpha_o)$). Notice that we here use the notation α_o to denote the pair consisting of a sensing action α and an outcome o. This notation allows for handling in a uniform way the two cases of a physical action α (without outcome) and a sensing action α with outcome o.

Proposition 2.2 *Let AD be an action description in $\mathcal{E}$, let S be an e-state of AD, and let α be a physical action (resp., sensing action with outcome $o \in \{\omega, \neg\omega\}$). If a successor e-state of S under α (resp., α with outcome o) exists, then it is unique.*

We are now ready to define the formal semantics of action and initial state descriptions as follows. An action description AD represents the directed graph $G_{AD} = (N, E)$, where N is the set of all e-states of AD, and $E \subseteq N \times N$ contains $S \rightarrow S'$ labeled with "α" (resp., "α_o") iff (i) α is a physical action (resp., sensing action with outcome $o \in \{\omega, \neg\omega\}$) that is executable in S, and (ii) $S' = \Phi(S,\alpha)$ (resp., $S' = \Phi(S,\alpha_o)$). An initial state description δ_I encodes the greatest e-state of AD that satisfies δ_I, denoted S_{δ_I}, if it exists (if there is an e-state that satisfies δ_I, then there is also a greatest such e-state). We denote by G_{AD,δ_I} the subgraph of G_{AD} consisting of all successors of S_{δ_I} along with their incident arrows.

Example 2.3 *(Robotic Soccer cont'd)* Consider the action description AD shown in Fig. 1 and the initial state description $\delta_I = \neg\mathsf{ballmoving} \wedge \mathsf{ballinarea}$, where the ball is in the penalty area and not moving. A portion of the directed graph G_{AD,δ_I} is shown in Fig. 2.

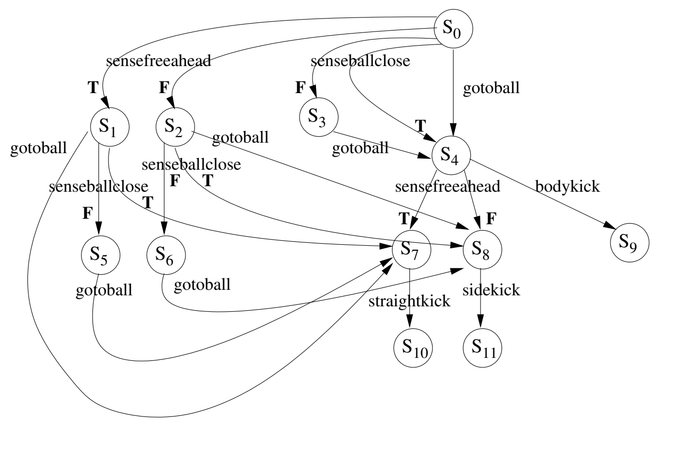
We finally define the notion of consistency for action and initial state descriptions. An action description is consistent iff it has at least one e-state and each action execution is defined. An initial state description is consistent if its e-state is defined. Formally, an action description AD is *consistent* iff (i) AD has at least one e-state S, (ii) $\Phi(S,\alpha)$ is defined for each e-state S of AD and each physical action α that is executable in S, and (iii) $\Phi(S,\alpha_o)$ is defined for each e-state S of AD and each sensing action α with outcome $o \in \{\omega, \neg\omega\}$ that is executable in S. An initial state description δ_I is *consistent* if S_{δ_I} is defined. In the sequel, we implicitly assume that all action and initial state descriptions are consistent.[2]

2.3 Computation

The main computational tasks related to action descriptions AD in $\mathcal{E}$ are (i) deciding whether an action α is executable in an e-state S, (ii) computing the e-state S_ϕ for a fluent conjunction ϕ (if it exists), (iii) deciding if an e-state S satisfies a fluent conjunction ϕ, and (iv) computing the successor e-state of an e-state S under an action α (if it exists). In this section, we provide upper bounds for the complexity of these tasks, which show that they all can be solved efficiently. In detail, (i)–(iii) can all be done in linear time in the size of AD, while (iv) can be done in quadratic time in the size of AD.

For fluent literals $\ell = f$ (resp., $\ell = \neg f$), we use $\neg.\ell$ to denote $\neg f$ (resp., f), and for sets of fluent literals L, we define $\neg.L = \{\neg.\ell \mid \ell \in L\}$. For fluent conjunctions ϕ, we denote by

[2]Our definition of consistency is thus stronger than simply requiring the existence of a model. This is analogous to other approaches in reasoning about actions, e.g., [Pirri and Reiter 1999; Zhang et al. 2002; Lang et al. 2003].



$S_0 = S_{\delta_I} \models \neg$ballmoving$\wedge$ballinarea
$S_1 \models \neg$ballmoving$\wedge$ballinarea$\wedge$freeahead
$S_2 \models \neg$ballmoving$\wedge$ballinarea$\wedge\neg$freeahead
$S_3 \models \neg$ballmoving$\wedge$ballinarea$\wedge\neg$ballclose
$S_4 \models \neg$ballmoving$\wedge$ballinarea$\wedge$ballclose
$S_5 \models \neg$ballmoving$\wedge$ballinarea$\wedge\neg$ballclose$\wedge$freeahead
$S_6 \models \neg$ballmoving$\wedge$ballinarea$\wedge\neg$ballclose$\wedge\neg$freeahead
$S_7 \models \neg$ballmoving$\wedge$ballinarea$\wedge$ballclose$\wedge$freeahead
$S_8 \models \neg$ballmoving$\wedge$ballinarea$\wedge$ballclose$\wedge\neg$freeahead
$S_9 \models \neg$ballmoving$\wedge\neg$ballinarea
$S_{10} \models \neg$ballmoving$\wedge\neg$ballinarea$\wedge$freeahead
$S_{11} \models \neg$ballmoving$\wedge\neg$ballinarea$\wedge\neg$freeahead

Fig. 2. A part of the directed graph G_{AD,δ_I} for $\delta_I = \neg$ballmoving$\wedge$ballinarea.

$Lit(\phi)$ the set of all fluent literals in ϕ, if ϕ is satisfiable, and the set of all fluent literals, otherwise. For e-states S, we denote by $Lit(S)$ the set of all fluent literals satisfied by S.

Given an action description AD, an e-state S of AD (represented by $Lit(S)$), and an action α, deciding whether α is executable in S can be done in linear time in the size of AD along the set of all precondition axioms in AD using standard data structures. Similarly, given AD and a fluent conjunction ϕ, computing the e-state S_ϕ (represented by $Lit(S_\phi)$) of AD and deciding whether a given e-state S (represented by $Lit(S)$) of AD satisfies ϕ can also both be done in linear time in the size of AD using standard data structures.

In the rest of this section, we provide a quadratic-time algorithm for computing the successor e-state of an e-state under a physical action (which can also easily be adapted to compute the successor e-state of an e-state under a sensing action). The algorithm, called **Compute-Successor**, is presented in Fig. 3. It takes as input an action description AD, an e-state S of AD (represented by $Lit(S)$), and a physical action α, and it returns as output the successor e-state S' of S under α (represented by $Lit(S')$). The set of fluent literals $L' = Lit(S')$ is constructed as follows. We start by initializing L' to an empty set, which is first augmented with all the fluent literals corresponding to the direct effects of the action α in S (steps 2–3 of the algorithm). Then, all the indirect effects due to the domain constraint axioms are added to L' (steps 4–8). Then, it is verified (step 9) whether the set of literals L' computed so far is *consistent*, that is, for each literal ℓ belonging to L', the literal $\neg.\ell$ does not belong to L'. Finally, the effects of the default frame axioms are computed and added to L' (steps 10–19). In particular, for each default frame axiom **inertial** ϕ **after** α such that ϕ holds in the initial e-state S (step 11), the set of literals L_{aux} initially contains the inertial literals propagated by the default frame axiom (that is, the ones

Algorithm Compute-Successor

Input: action description AD, e-state S of AD (represented by $Lit(S)$), and physical action α.

Output: successor e-state S' of S under α (represented by $Lit(S')$), if it exists,
and "there exists no successor e-state of S under α", otherwise.

```
1.  $L' = \emptyset$;
2.  for each conditional effect axiom "caused $\psi$ after $\alpha$ when $\phi$" in $AD$ do
3.    if $Lit(\phi) \subseteq Lit(S)$ then $L' = L' \cup Lit(\psi)$;
4.  repeat
5.    $L'' = L'$;
6.    for each domain constraint axiom "caused $\psi$ if $\ell$" in $AD$ do
7.      if $\ell \in L'$ then $L' = L' \cup Lit(\psi)$
8.  until $L'' = L'$;
9.  if $L'$ is not consistent then return "there exists no successor e-state of $S$ under $\alpha$";
10. for each default frame axiom "inertial $\phi$ after $\alpha$" in $AD$ do
11.   if $Lit(\phi) \subseteq Lit(S)$ then begin
12.     $L_{aux} = Lit(\phi)$;
13.     repeat
14.       $L'_{aux} = L_{aux}$;
15.       for each domain constraint axiom "caused $\psi$ if $\ell$" in $AD$ do
16.         if $\ell \in L_{aux}$ then $L_{aux} = L_{aux} \cup Lit(\psi)$
17.     until $L'_{aux} = L_{aux}$;
18.     if $L' \cup L_{aux}$ is consistent then $L' = L' \cup L_{aux}$
19.   end;
20. return $L'$.
```

Fig. 3. Algorithm Compute-Successor

occurring in ϕ); then (steps 13–17), L_{aux} is closed with respect to the domain constraint axioms (that is, it is augmented with the literals indirectly derived by the domain constraint axioms); finally, it is verified (step 18) whether the set of literals L_{aux} thus computed is consistent with L', that is, the set of literals $L' \cup L_{aux}$ is consistent: if this is the case, then the default frame axiom can be applied and the literals in ϕ (and all their indirect consequences) are propagated in the successor state L' by adding the literals in L_{aux} to the set L'. The following theorem shows that **Compute-Successor** is correct.

Proposition 2.4 *Given an action description AD in the action language $\mathcal{E}$, an e-state S of AD (represented by $Lit(S)$), and a physical action α,* **Compute-Successor** *returns the successor e-state S' of S under α (represented by $Lit(S')$), if it exists, and* **Compute-Successor** *returns "there exists no successor e-state of S under α", otherwise.*

Proof. First, it is easy to verify that there exists no successor e-state of S under α iff the set of fluent literals obtained by the union of the direct effects of α in S and the indirect effects given by the domain constraint axioms is unsatisfiable. Thus, the algorithm returns no set of fluent literals (step 9) iff there exists no successor e-state of S under α. Then, we prove that, for each AD, S, and α as stated in the theorem, the algorithm returns the set of fluent literals $L' = Lit(S')$, where S' is the successor e-state of S under α. First, notice that, when ϕ is a fluent conjunction, then $S \models \phi$ iff $Lit(\phi) \subseteq Lit(S)$ (steps 3 and 11 of the algorithm). Now, the first for–each cycle at step 2 guarantees that the above e-state represented by L' satisfies $direct(S, \alpha)$, while the two repeat–until loops guarantee that the e-state represented by L' satisfies all domain constraint axioms in AD. Finally, the last for–each cycle at step 9 guarantees that the e-state represented by L' satisfies a maximal subset of default frame axioms as requested by the definition of successor e-state. Hence, the returned L' is equal to $Lit(S')$, where S' is the successor e-state of S under α. $\Box$

Finally, as an immediate consequence of the previous result, we state an important upper bound for the complexity of computing successor e-states. The following theorem shows that computing successor e-states can be done in quadratic time. Here, we denote by $|AD|$ (resp., $\|AD\|$) the number of elements in AD (resp., the size of AD).

Proposition 2.5 *Let AD be an action description in the action language $\mathcal{E}$, let α be a physical action, and let S be an e-state of AD (represented by $Lit(S)$). The successor e-state $S' = \Phi(S, \alpha)$ (represented by $Lit(S')$) can be computed in time $O(|AD| \cdot \|AD\|)$. Moreover, if α is a sensing action, and o is an outcome of α, the successor e-state $S' = \Phi(S, \alpha_o)$ (represented by $Lit(S')$) can be computed in time $O(|AD| \cdot \|AD\|)$.*

Proof. For physical actions α, the proof is an immediate consequence of the algorithm **Compute-Successor** in Fig. 3. Indeed, it is easy to see that the algorithm runs in time $O(|AD| \cdot \|AD\|)$ using standard data structures (note that the size of $Lit(S)$ is linearly bounded by $\|AD\|$). The case when α is a sensing action can be proved analogously. $\Box$

3. THE ACTION LANGUAGE $\mathcal{E}+$

In this section, we introduce the action language $\mathcal{E}+$, which is an extension of the action language $\mathcal{E}$ by actions with nondeterministic and probabilistic effects. We define the syntax and semantics of extended action descriptions in $\mathcal{E}+$, which extend action descriptions in $\mathcal{E}$ by axioms to encode nondeterministic and probabilistic effects of actions.

3.1 Syntax

We divide the set of physical actions into *deterministic*, *nondeterministic*, and *probabilistic physical actions*. The nondeterministic and probabilistic conditional effects of the latter two types of actions are encoded in nondeterministic and probabilistic conditional effect axioms, respectively. A *nondeterministic conditional effect axiom* has the form

$$\textbf{caused}\ \psi_1, \ldots, \psi_n\ \textbf{after}\ \alpha\ \textbf{when}\ \phi\,, \tag{6}$$

where $\psi_1, \ldots, \psi_n$ and ϕ are fluent conjunctions, α is a nondeterministic physical action, and $n \geqslant 2$. Informally, if the current state of the world satisfies ϕ, then the successor state after executing α satisfies ψ_i for some $i \in \{1, \ldots, n\}$. A *probabilistic conditional effect axiom* is an expression of the form

$$\textbf{caused}\ \psi_1 : p_1, \ldots, \psi_n : p_n\ \textbf{after}\ \alpha\ \textbf{when}\ \phi\,, \tag{7}$$

where $\psi_1, \ldots, \psi_n$ and ϕ are fluent conjunctions, α is a probabilistic physical action, $p_1, \ldots, p_n > 0$, $p_1 + \cdots + p_n = 1$, and $n \geqslant 2$. Informally, if the current state of the world satisfies ϕ, then the successor state after executing α satisfies ψ_i with the probability p_i, for all $i \in \{1, \ldots, n\}$. Note that similar specifications of probabilistic knowledge can also be found in probabilistic reasoning about actions (see Section 8) and in probabilistic agent systems (see, e.g., [Dix et al. 2006]). If $\phi = \top$, then (6) (resp., (7)) is also called a *nondeterministic* (resp., *probabilistic*) *effect axiom*, and we omit "**when** ϕ" in (6) (resp., (7)).

(vi) nondeterministic conditional effect axioms:

(18) **caused** goalsaved, ¬goalsaved **after** openlegs

(vii) probabilistic conditional effect axioms:

(19) **caused** ballclose: 0.8, ¬ballinarea: 0.1, ¬ballclose: 0.1 **after** gotoball
(20) **caused** ¬ballinarea∧¬inposition: 0.1, ¬ballinarea∧inposition: 0.5, ¬inposition: 0.1, ⊤: 0.3 **after** bodykick
(21) **caused** ¬ballinarea: 0.9, ⊤: 0.1 **after** straightkick
(22) **caused** ¬ballinarea: 0.7, ⊤: 0.3 **after** sidekick
(23) **caused** alignedtoball: 0.7, ¬alignedtoball: 0.3 **after** aligntoball

Fig. 4. Robotic Soccer Example: Nondeterministic and probabilistic conditional effect axioms.

We define extended action descriptions as follows. An *extended action description* EAD is a finite set of precondition, conditional effect, sensing effect, default frame, domain constraint, nondeterministic conditional effect, and probabilistic conditional effect axioms.

Example 3.1 *(Robotic Soccer cont'd)* The physical actions gotoball, bodykick, straightkick, sidekick, and aligntoball of the robotic soccer scenario in Example 2.1 have either nondeterministic or probabilistic effects, and thus cannot be encoded in action descriptions in $\mathcal{E}$. However, using nondeterministic and probabilistic conditional effect axioms, they can be easily be expressed in extended action descriptions in $\mathcal{E}+$. More precisely, the extended action description EAD is given by the precondition, conditional effect, sensing effect, default frame, and domain constraint axioms in Fig. 1 and the nondeterministic and probabilistic conditional effect axioms in Fig. 4. In particular, after executing the nondeterministic physical action openlegs, the goal is saved or not (14). After executing the probabilistic physical action gotoball, the ball is close with probability 0.8, or the ball is not in the penalty area with probability 0.1, or the ball is not close with probability 0.1 (15).

3.2 Semantics

We define the semantics of an extended action description EAD by a system of deterministic, nondeterministic, and probabilistic transitions between e-states. To this end, we extend the transition system of an action description AD by nondeterministic and probabilistic transitions between e-states through nondeterministic and probabilistic physical actions, respectively. These transitions are defined by associating with each pair (S, α) of a current e-state S and a nondeterministic (resp., probabilistic) physical action α executable in S, a set (resp., probability distribution on a set) of successor e-states after executing α in S.

Note that the above probabilistic transitions are similar to the probabilistic transitions in fully observable Markov decision processes (MDPs) [Puterman 1994] and partially observable Markov decision processes (POMDPs) [Kaelbling et al. 1998]. However, they are between e-states and thus involve *sets of states* rather than *single states*.

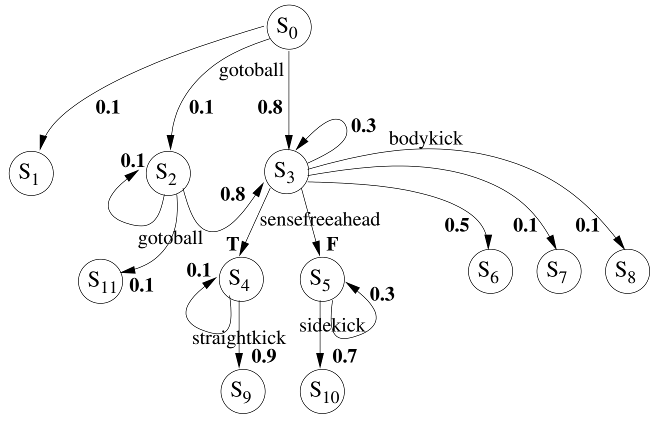
In the sequel, let EAD be an extended action description. We define states, e-states, the executability of actions in e-states, and the transitions between e-states through the execution of deterministic physical actions and sensing actions in the same way as in Section 2.2, but relative to EAD instead of AD. Hence, it now only remains to define the nondeterministic and probabilistic transitions between e-states through the execution of nondeterministic and probabilistic physical actions, respectively.

Let S be an e-state of EAD, and α be a nondeterministic (resp., probabilistic) physical action executable in S. We now define the set (resp., probability distribution on a set) of successor e-states after executing α in S. We first collect the set of all axioms (6) (resp., (7)) in EAD that are *relevant to S and α*, that is, for which $S \models \phi$ holds. Let $\{$**caused** $\psi_{j,1}, \ldots, \psi_{j,n_j}$ **after** α **when** $\phi_j \mid j \in J\}$ (resp., $\{$**caused** $\psi_{j,1} \colon p_{j,1}, \ldots, \psi_{j,n_j} \colon p_{j,n_j}$ **after** α **when** $\phi_j \mid j \in J\}$) denote this set. For every combination $c = (\psi_j)_{j \in J} = (\psi_{j,i_j})_{j \in J}$ (called *context*) from $C_{S,\alpha} = \{(\psi_j)_{j \in J} \mid \forall j \in J \colon \psi_j \in \{\psi_{j,1}, \ldots, \psi_{j,n_j}\}\}$, we then compute one successor e-state (which is associated with the probability $Pr_{S,\alpha}(c) = \Pi_{j \in J}\, p_{j,i_j}$, if α is probabilistic). We thus assume that any two nondeterministic (resp., probabilistic) conditional effect axioms relevant to S and α are logically (resp., probabilistically) independent. Formally, the *successor e-state* of S after executing α in the context $c = (\psi_j)_{j \in J} \in C_{S,\alpha}$, denoted $\Phi_c(S, \alpha)$, is the e-state $\Phi(S, \alpha)$ under the action description obtained from EAD by removing all axioms (6) and (7) and adding **caused** $\bigwedge_{j \in J} \psi_j$ **after** α. We finally define the overall nondeterministic (resp., probabilistic) transition as follows. If α is nondeterministic, then the *set of successor e-states* of S under α is defined as $F_\alpha(S) = \{\Phi_c(S, \alpha) \mid c \in C_{S,\alpha}\}$. If α is probabilistic, then the *probability distribution on the successor e-states* of S under α, denoted $Pr_\alpha(\,\cdot\,|S)$, is defined by $Pr_\alpha(S'|S) = \sum_{c \in C_{S,\alpha},\, S' = \Phi_c(S,\alpha)} Pr_{S,\alpha}(c)$ for all e-states S' of EAD. Intuitively, executing a nondeterministic action α in an e-state S nondeterministically leads to some $S' \in F_\alpha(S)$, while executing a probabilistic action α in S leads to S' with the probability $Pr_\alpha(S'|S)$.

We are now ready to define the semantics of an extended action description EAD in terms of a system of deterministic, nondeterministic, and probabilistic transitions between its e-states as follows. The extended action description EAD represents the directed graph $G_{EAD} = (N, E)$, where N is the set of all e-states of EAD, and $E \subseteq N \times N$ contains (i) an arrow $S \to S'$ labeled with "α" for every e-state $S \in N$ and deterministic physical action α that is executable in S, where $S' = \Phi(S, \alpha)$, (ii) an arrow $S \to S'$ labeled with "α_o" for every e-state $S \in N$ and sensing action α with outcome $o \in \{\omega, \neg\omega\}$ that is executable in S, where $S' = \Phi(S, \alpha_o)$, (iii) an arrow $S \to S'$ labeled with "α_c" for every e-state $S \in N$, nondeterministic physical action α that is executable in S, and context $c \in C_{S,\alpha}$, where $S' = \Phi_c(S, \alpha)$, and (iv) an arrow $S \to S'$ labeled with "α_c, pr" for every e-state $S \in N$, probabilistic physical action α that is executable in S, and context $c \in C_{S,\alpha}$, where $pr = Pr_{S,\alpha}(c)$ and $S' = \Phi_c(S, \alpha)$. We denote by G_{EAD,δ_I} the subgraph of G_{EAD} that consists of all successors of S_{δ_I} along with their incident arrows.

We finally define the consistency of extended action descriptions. We say EAD is *consistent* iff (i) EAD has at least one e-state S, (ii) $\Phi(S, \alpha)$ is defined for every e-state S of EAD and every deterministic physical action α that is executable in S, (iii) $\Phi(S, \alpha_o)$ is defined for every e-state S of EAD and every sensing action α with outcome $o \in \{\omega, \neg\omega\}$ that is executable in S, and (iv) $\Phi_c(S, \alpha)$ is defined for every e-state S of EAD, nondeterministic or probabilistic physical action α that is executable in S, and context $c \in C_{S,\alpha}$. In the sequel, we implicitly assume that all extended action descriptions are consistent.

Example 3.2 *(Robotic Soccer cont'd)* Let the extended action description EAD be given by the axioms in Figs. 1 and 4 excluding the axioms (9) to (12). Furthermore, let the initial state description be given by $\delta_I = \neg\mathsf{ballmoving} \wedge \mathsf{ballinarea} \wedge \mathsf{inposition}$, where the goalkeeper is in the correct position, and the ball is in the penalty area and not moving. Then, a portion of the directed graph G_{EAD,δ_I} is shown in Fig. 5.



$S_0 = S_{\delta_I} \models$ ¬ballmoving∧ballinarea∧inposition
$S_1 \models$ ¬ballmoving∧¬ballinarea∧inposition
$S_2 \models$ ¬ballmoving∧ballinarea∧inposition∧¬ballclose
$S_3 \models$ ¬ballmoving∧ballinarea∧inposition∧ballclose
$S_4 \models$ ¬ballmoving∧ballinarea∧inposition∧ballclose∧freeahead
$S_5 \models$ ¬ballmoving∧ballinarea∧inposition∧ballclose∧¬freeahead
$S_6 \models$ ¬ballmoving∧¬ballinarea∧inposition
$S_7 \models$ ¬ballmoving∧¬ballinarea∧¬inposition
$S_8 \models$ ¬ballmoving∧ballinarea∧¬inposition∧ballclose
$S_9 \models$ ¬ballmoving∧¬ballinarea∧inposition∧freeahead
$S_{10} \models$ ¬ballmoving∧¬ballinarea∧inposition∧¬freeahead
$S_{11} \models$ ¬ballmoving∧¬ballinarea∧inposition∧¬ballclose

Fig. 5. A part of the directed graph G_{EAD,δ_I} for $\delta_I =$ ¬ballmoving∧ballinarea∧inposition.

3.3 Computation

The computational results of Section 2.3 about action descriptions AD in $\mathcal{E}$ all carry over to extended action descriptions EAD in $\mathcal{E}+$. That is, (i) deciding if an action α is executable in an e-state S, (ii) computing the e-state S_ϕ for a fluent conjunction ϕ (if it exists), and (iii) deciding if an e-state S satisfies a fluent conjunction ϕ can all be done in linear time in the size of EAD, while (iv) computing the successor e-state (if it exists) of an e-state S under a (deterministic, nondeterministic, or probabilistic) physical action α and a context c, if α is nondeterministic or probabilistic, along with its probability, if α is probabilistic, or under a sensing action α with outcome o can be done in quadratic time in the size of EAD.

4. BELIEF GRAPHS

In this section, we define the notion of a belief graph and the concepts of lower and upper probabilities of fluent formulas in belief graphs. We then show that every belief graph is a compact representation of a finite set of unnormalized probability distributions over the set of all e-states. In the sequel, let EAD be an extended action description.

4.1 Belief Graphs

Intuitively, a belief graph encodes the overall epistemic state of an agent after starting from a single initial e-state and then performing a finite sequence of actions. A belief graph consists of a directed acyclic graph (which is a directed graph that does not contain any directed path forming a cycle) in which every node represents an e-state and every arrow represents a transition between two e-states. Given an initial e-state and a sequence of actions $\alpha_1, \ldots, \alpha_n$, their belief graph is built by using the initial e-state as a root and then adding for every action α_i, $i \in \{1, \ldots, n\}$, a new layer of descendent nodes, namely, the set of all possible successor e-states after executing α_i in the e-states added before.

Formally, every belief graph $B = (V, E, \ell, Pr)$ consists of a directed acyclic graph $G = (V, E)$, a labeling function ℓ that associates with every node $v \in V$ an e-state $\ell(v) = S$ of EAD, and a partial mapping Pr that associates with some arrows $e \in E$ a real number $Pr(e) \in [0, 1]$. Every belief graph $B = (V, E, \ell, Pr)$ has exactly one node $r \in V$ without parents, called the *root* of B, and some nodes without children, called the *leaves* of B. A *deepest leaf* of B is a leaf of B that has the maximum distance from the root of B. An action α is *executable* in a belief graph B iff α is executable in the label S of some deepest leaf v of B. More precisely, *belief graphs* are inductively defined as follows. Any node v labeled with an e-state S of EAD is a belief graph. In particular, for fluent conjunctions ϕ such that S_ϕ is defined, we denote by B_ϕ the belief graph that consists of a single node v labeled with S_ϕ. If B is a belief graph and α is a deterministic (resp., nondeterministic) physical action executable in B, then $B \circ \alpha$ is also a belief graph, which is obtained from B by (i) adding a new node v' labeled with S' for every $S' = \Phi(S, \alpha)$ (resp., $S' \in F_\alpha(S)$) such that S is the label of a deepest leaf v of B in which α is executable, and (ii) connecting the nodes v and v' of such S and S', respectively, by a new arrow $v \rightarrow v'$. If B is a belief graph and α is a probabilistic physical action executable in B, then $B \circ \alpha$ is also a belief graph, which is obtained from B by (i) adding a new node v' labeled with S' for every $S' = \Phi_c(S, \alpha)$ such that (i.1) $c \in C_{S,\alpha}$ and (i.2) S is the label of a deepest leaf v of B in which α is executable, and (ii) connecting the nodes v and v' of such S and S', respectively, by a new arrow $e = v \rightarrow v'$ with the probability $Pr(e) = Pr_\alpha(S'|S)$. If B is a belief graph and α is a sensing action with outcome $o \in \{\omega, \neg\omega\}$ executable in B, then $B \circ \alpha_o$ is also a belief graph, which is obtained from B by (i) adding a new node v' labeled with S' for every $S' = \Phi(S, \alpha_o)$ such that S is the label of a deepest leaf v of B in which α is executable, and (ii) connecting the nodes v and v' of such S and S', respectively, by a new arrow $e = v \rightarrow v'$. Informally, $B \circ \alpha$ (resp., $B \circ \alpha_o$) is the successor belief graph after executing the action α (resp., α with outcome o) in B.

Example 4.1 *(Robotic Soccer cont'd)* Consider the fluent conjunction $\delta_I = \mathsf{ballinarea} \wedge \mathsf{inposition} \wedge \neg\mathsf{ballmoving}$. Fig. 6, left side, shows the belief graphs after executing the following sequences of actions in B_{δ_I} (that is, the belief graph associated with δ_I): (1.a) gotoball and bodykick; (1.b) gotoball, sensefreeahead with outcome T, and straightkick; and (1.c) gotoball, sensefreeahead with outcome F, and sidekick.

Consider next the fluent conjunction $\delta_I = \mathsf{ballmoving}$. Fig. 6, right side, shows the belief graphs after executing the following sequences of actions in the belief graph B_{δ_I}: (2.a) openlegs; (2.b) aligntoball and openlegs; (2.c) sensealignedtoball with outcome T and openlegs; and (2.d) sensealignedtoball with outcome F, aligntoball, and openlegs.

Observe that the number of nodes n_B of a belief graph B depends on the length l of its sequence of actions and the width of its nondeterministic and probabilistic branchings. Hence, n_B may be large. However, n_B is polynomial in the size of EAD under suitable assumptions, that is, in the special case where l is bounded by a constant and the maximal

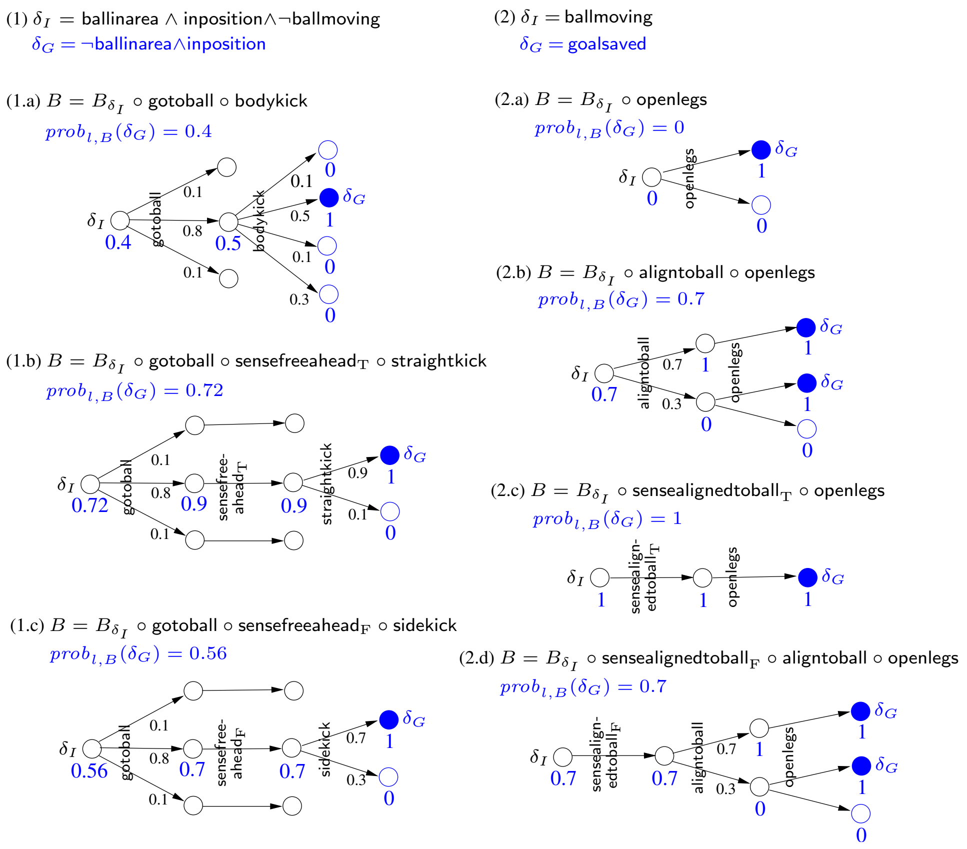

(1) δ_I = ballinarea ∧ inposition∧¬ballmoving
δ_G = ¬ballinarea∧inposition
(1.a) $B = B_{\delta_I} \circ$ gotoball ∘ bodykick
$prob_{l,B}(\delta_G) = 0.4$
(1.b) $B = B_{\delta_I} \circ$ gotoball ∘ sensefreeahead$_T$ ∘ straightkick
$prob_{l,B}(\delta_G) = 0.72$
(1.c) $B = B_{\delta_I} \circ$ gotoball ∘ sensefreeahead$_F$ ∘ sidekick
$prob_{l,B}(\delta_G) = 0.56$
(2) δ_I = ballmoving
δ_G = goalsaved
(2.a) $B = B_{\delta_I} \circ$ openlegs
$prob_{l,B}(\delta_G) = 0$
(2.b) $B = B_{\delta_I} \circ$ aligntoball ∘ openlegs
$prob_{l,B}(\delta_G) = 0.7$
(2.c) $B = B_{\delta_I} \circ$ sensealignedtoball$_T$ ∘ openlegs
$prob_{l,B}(\delta_G) = 1$
(2.d) $B = B_{\delta_I} \circ$ sensealignedtoball$_F$ ∘ aligntoball ∘ openlegs
$prob_{l,B}(\delta_G) = 0.7$

Fig. 6. Belief graphs and lower probabilities of fluent formulas.

number of nondeterministic and probabilistic conditional effect axioms (6) resp. (7) that are relevant to some S and α is also bounded by a constant (see also Section 7).

4.2 Lower and Upper Probabilities of Fluent Formulas

We next evaluate the truth of fluent formulas in belief graphs. Since a belief graph as an overall epistemic state of an agent contains qualitative and probabilistic uncertainty, it specifies a set of probability values for the truth of a fluent formula, rather than an exact binary truth value. We especially deal with the smallest and the largest probability value of a fluent formula ϕ in a belief graph B, called the *lower* and the *upper probability* of ϕ in B, respectively. Intuitively, given the qualitative and probabilistic knowledge of B, the fluent formula ϕ holds with at least (resp., most) its lower (resp., upper) probability in B.

Formally, let $B = (V, E, \ell, Pr)$ be a belief graph with the root $r \in V$, and let ϕ be a fluent formula. Let $G_d = (V_d, E_d)$ denote the subgraph of $G = (V, E)$ where (i) V_d is the set of all nodes $v \in V$ on a path from r to a deepest leaf in G, and (ii) E_d is the restriction of E to the nodes in V_d. Then, the *lower probability* of ϕ in B, denoted $prob_{l,B}(\phi)$, is the value $prob_{l,r}(\phi)$, where the function $prob_{l,\cdot}(\phi)\colon V_d \to [0,1]$ is defined as follows:

—$prob_{l,v}(\phi)$ is 1 for every leaf $v \in V_d$ with $\ell(v) \models \phi$, and 0 for all other leaves $v \in V_d$;
—$prob_{l,v}(\phi) = \min_{e=v\to v'\in E_d} prob_{l,v'}(\phi)$ for every $v \in V_d$ where $Pr(e)$ is undefined;
—$prob_{l,v}(\phi) = \sum_{e=v\to v'\in E_d} Pr(e) \cdot prob_{l,v'}(\phi)$ for every $v \in V_d$ where $Pr(e)$ is defined.

Informally, the deepest leaves v of B whose e-state $\ell(v)$ satisfies (resp., does not satisfy) ϕ associate with ϕ the lower probability 1 (resp., 0). We then propagate the lower probability to every node of G_d, using the lower probabilities of the children and the probabilities that are associated with some arrows. The lower probability of ϕ in B is then the lower probability that the root r associates with ϕ. Similarly, the *upper probability* of ϕ in B, denoted $prob_{u,B}(\phi)$, is the value $prob_{u,r}(\phi)$, where $prob_{u,\cdot}(\phi)\colon V_d \to [0,1]$ is defined by:

—$prob_{u,v}(\phi)$ is 1 for every leaf $v \in V_d$ with $\ell(v) \not\models \neg\phi$, and 0 for all other leaves $v \in V_d$;
—$prob_{u,v}(\phi) = \max_{e=v\to v'\in E_d} prob_{u,v'}(\phi)$ for every $v \in V_d$ where $Pr(e)$ is undefined;
—$prob_{u,v}(\phi) = \sum_{e=v\to v'\in E_d} Pr(e)\cdot prob_{u,v'}(\phi)$ for every $v \in V_d$ where $Pr(e)$ is defined.

Finally, the *executability probability* of a belief graph B is defined as $prob_{l,B}(\top)$. Intuitively, this is the probability with which the sequence of actions behind B is executable.

Example 4.2 *(Robotic Soccer cont'd)* The lower probabilities of $\delta_G = \neg\mathsf{ballinarea} \wedge \mathsf{inposition}$ in the belief graphs of Fig. 6 (1.a), (1.b), and (1.c) are given by 0.4, 0.72, and 0.56, respectively, while the lower probabilities of $\delta_G = \mathsf{goalsaved}$ in the belief graphs of Fig. 6 (2.a), (2.b), (2.c), and (2.d) are given by 0, 0.7, 1, and 0.7, respectively. The executability probabilities of the belief graphs of Fig. 6 (1.a) to (1.c) are all 0.8, while the executability probabilities of the belief graphs of Fig. 6 (2.a) to (2.d) are all 1.

The following lemma shows that the lower probability of a fluent formula ϕ in a belief graph B is always below the upper probability of ϕ in B. This result can be easily proved along the recursive definition of the lower and the upper probability of ϕ in B.

Lemma 4.3 *If B is a belief graph and ϕ is a fluent formula, then $prob_{l,B}(\phi) \leqslant prob_{u,B}(\phi)$.*

4.3 Representation Results

We finally show that every belief graph is a compact representation of a set of unnormalized probability distributions over the set $\mathcal{S}$ of all e-states of EAD. That is, every belief graph can be associated with a set of unnormalized probability distributions such that (i) deciding the executability of an action, (ii) executing an action, and (iii) evaluating the lower and the upper probability of a fluent formula in a belief graph B can be defined in an isomorphic way on the set of unnormalized probability distributions of B.

Let $B = (V, E, \ell, Pr)$ be a belief graph with the root $r \in V$, and let $G_d = (V_d, E_d)$ be the subgraph of $G = (V, E)$ defined in Section 4.2. Then, the set of unnormalized probability distributions associated with B, denoted $\boldsymbol{\mu}_B$, is defined as $\boldsymbol{\mu}_r$, where the function $\boldsymbol{\mu}_\cdot$ associates with every node $v \in V_d$ a set of unnormalized probability distributions by:

—$\boldsymbol{\mu}_v = \{\mu_v\}$ for every leaf $v \in V_d$, where $\mu_v(\ell(v)) = 1$ and $\mu_v(S) = 0$ for all other $S \in \mathcal{S}$;
—$\boldsymbol{\mu}_v = \bigcup \{\boldsymbol{\mu}_{v'} \mid e = v \to v' \in E_d\}$ for every node $v \in V_d$ such that $Pr(e)$ is undefined;
—$\boldsymbol{\mu}_v = \bigcup \{\sum_{e=v\to v'\in E_d} Pr(e) \cdot \mu_{v'} \mid \forall e = v \to v' \in E_d\colon \mu_{v'} \in \boldsymbol{\mu}_{v'}\}$ for every node $v \in V_d$ such that $Pr(e)$ is defined, where $(\sum_{e=v\to v'\in E_d} Pr(e) \cdot \mu_{v'})(S) = \sum_{e=v\to v'\in E_d} Pr(e) \cdot \mu_{v'}(S)$ for all e-states $S \in \mathcal{S}$.

Example 4.4 *(Robotic Soccer cont'd)* The belief graph in Fig. 6 (1.a) has one unnormalized probability distribution, which maps the e-states of the deepest leaves to the probabilities 0.08, 0.4, 0.08, and 0.24, while the belief graph in Fig. 6 (2.a) has two probability distributions, one that maps the first leaf to 1, and one that maps the second leaf to 1.

The following theorem shows that the executability of an action α in a belief graph B can be expressed in terms of $\boldsymbol{\mu}_B$, that is, B's set of unnormalized probability distributions over the set $\mathcal{S}$ of all e-states of EAD. It also shows that there exists an operation $\circ'$ such that $\boldsymbol{\mu}_B \circ' \alpha = \boldsymbol{\mu}_{B \circ \alpha}$ for all belief graphs B and all actions α that are executable in B.

Theorem 4.5 *Let EAD be an extended action description, let B be a belief graph, and let α be an action, which is executable in B for (b) to (e). Let $\mathcal{S}$ be the set of all e-states of EAD, and let $\boldsymbol{\mu}_B$ be B's set of unnormalized probability distributions over $\mathcal{S}$. Then:*

(a) The action α is executable in B iff it is executable in some e-state $S \in \mathcal{S}$ such that $\mu(S) > 0$ for some $\mu \in \boldsymbol{\mu}_B$.

(b) If α is a deterministic physical action, then $\boldsymbol{\mu}_{B \circ \alpha} = \{\mu \circ \alpha \mid \mu \in \boldsymbol{\mu}_B\}$, where $(\mu \circ \alpha)(S') = \sum_{S \in \mathcal{S}:\, S' = \Phi(S,\alpha)} \mu(S)$ for all $S' \in \mathcal{S}$.

(c) If α is a sensing action with outcome $o \in \{\omega, \neg\omega\}$, then $\boldsymbol{\mu}_{B \circ \alpha_o} = \{\mu \circ \alpha_o \mid \mu \in \boldsymbol{\mu}_B\}$, where $(\mu \circ \alpha_o)(S') = \sum_{S \in \mathcal{S}:\, S' = \Phi(S,\alpha_o)} \mu(S)$ for all $S' \in \mathcal{S}$.

(d) If α is a nondeterministic physical action, then $\boldsymbol{\mu}_{B \circ \alpha} = \{\mu \circ \widetilde{\alpha} \mid \mu \in \boldsymbol{\mu}_B, \widetilde{\alpha} \in inst(\alpha)\}$, where $(\mu \circ \widetilde{\alpha})(S') = \sum_{S \in \mathcal{S}:\, S' = \Phi(S,\widetilde{\alpha})} \mu(S)$ for all $S' \in \mathcal{S}$, and $inst(\alpha)$ denotes the set of all actions $\widetilde{\alpha}$ such that $\Phi(S, \widetilde{\alpha}) \in F_\alpha(S)$ for all $S \in \mathcal{S}$. Intuitively, $inst(\alpha)$ is the set of all possible "deterministic instances" of α.

(e) If α is a probabilistic physical action, then $\boldsymbol{\mu}_{B \circ \alpha} = \{\mu \circ \alpha \mid \mu \in \boldsymbol{\mu}_B\}$, where $(\mu \circ \alpha)(S') = \sum_{S \in \mathcal{S}:\, \exists c \in C_{S,\alpha}:\, S' = \Phi_c(S,\alpha)} Pr_\alpha(S'|S) \cdot \mu(S)$ for all $S' \in \mathcal{S}$.

The next theorem shows that (i) lower and upper probabilities of fluent formulas in a belief graph B and (ii) the executability probability of a belief graph B can also be expressed in terms of B's set of unnormalized probability distributions.

Theorem 4.6 *Let EAD be an extended action description, let B be a belief graph, and let ϕ be a fluent formula. Let $\mathcal{S}$ be the set of all e-states of EAD, and let $\boldsymbol{\mu}_B$ be the set of unnormalized probability distributions over $\mathcal{S}$ associated with B. Then, (a) $prob_{l,B}(\phi)$ (resp., $prob_{u,B}(\phi)$) is given by $\min_{\mu \in \boldsymbol{\mu}_B} \sum_{S \in \mathcal{S},\, S \models \phi} \mu(S)$ (resp., $\max_{\mu \in \boldsymbol{\mu}_B} \sum_{S \in \mathcal{S},\, S \not\models \neg\phi} \mu(S)$), and (b) the executability probability of B is given by $\min_{\mu \in \boldsymbol{\mu}_B} \sum_{S \in \mathcal{S}} \mu(S)$.*

5. CONDITIONAL PLANNING

The conditional planning problem in our framework can be described as follows. Given an extended action description EAD, an initial state description δ_I, and a *goal description* δ_G, which is a fluent conjunction, compute the best conditional plan to achieve δ_G from δ_I. We first define conditional plans and their goodness for achieving δ_G from δ_I. We then formally state the conditional planning problems and provide some uncomputability results.

$$
\begin{aligned}
CP_1 &= \mathsf{gotoball};\, \mathsf{bodykick}\\
CP_2 &= \mathsf{gotoball};\, \mathsf{sensefreeahead};\, \textbf{if } \mathsf{freeahead} \textbf{ then } \{\mathsf{straightkick}\}\\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad \textbf{else } \{\mathsf{sidekick}\}\\
CP_3 &= \mathsf{gotoball};\, \mathsf{senseballclose};\, \textbf{if } \mathsf{ballclose} \textbf{ then } \{\mathsf{sensefreeahead};\, \textbf{if } \mathsf{freeahead} \textbf{ then } \{\mathsf{straightkick}\}\\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \textbf{else } \{\mathsf{sidekick}\}\}\\
CP_4 &= \mathsf{openlegs}\\
CP_5 &= \mathsf{aligntoball};\, \mathsf{openlegs}\\
CP_6 &= \mathsf{sensealignedtoball};\, \textbf{if } \mathsf{alignedtoball} \textbf{ then } \{\mathsf{openlegs}\}\\
&\qquad\qquad\qquad\qquad\qquad\qquad \textbf{else } \{\mathsf{aligntoball};\, \mathsf{openlegs}\}
\end{aligned}
$$

Fig. 7. Conditional plans.

5.1 Conditional Plans

Intuitively, a conditional plan (see especially [Levesque 1996; Lobo et al. 1997; Son et al. 2004]) is a binary directed tree where every arrow represents an action, and every branching expresses the two outcomes of a sensing action, which can thus be used to select the proper actions. We recall that a *directed tree* is a directed acyclic graph in which every node has exactly one parent, except for the *root*, which has no parents; nodes without children are called *leaves*. Formally, a *conditional plan* CP is either (i) the *empty conditional plan*, denoted λ, or (ii) of the form $\alpha\,; CP'$, or (iii) of the form $\beta\,;$ **if** ω **then** $\{CP_\omega\}$ **else** $\{CP_{\neg\omega}\}$, where α is a physical action, β is a sensing action with outcomes ω and $\neg\omega$, and CP', CP_ω, and $CP_{\neg\omega}$ are conditional plans. We call α and β in (i) and (ii), respectively, the *root action* of CP, and we often abbreviate "$\pi\,;\lambda$" in (i) by "π". The *length* of a conditional plan CP, denoted $\mathit{length}(CP)$, is inductively defined by (i) $\mathit{length}(\lambda) = 0$, (ii) $\mathit{length}(\alpha\,; CP') = 1 + \mathit{length}(CP')$, and (iii) $\mathit{length}(\beta\,;$ **if** ω **then** $\{CP_\omega\}$ **else** $\{CP_{\neg\omega}\}) = 1 + \max(\mathit{length}(CP_\omega), \mathit{length}(CP_{\neg\omega}))$.

Example 5.1 *(Robotic Soccer cont'd)* Consider first the following initial state description $\delta_I = \mathsf{ballinarea} \wedge \mathsf{inposition} \wedge \neg\mathsf{ballmoving}$, which encodes the initial state where the robot is in its standard position and the ball is in the robot's own area and not moving, and the goal description $\delta_G = \neg\mathsf{ballinarea} \wedge \mathsf{inposition}$, which encodes the goal state where the robot should kick away the ball and remain in its position. Some potential conditional plans CP_1, CP_2 and CP_3 for achieving δ_G from δ_I are shown in Fig. 7. Consider next an initial state description $\delta_I = \mathsf{ballmoving}$, where the ball is moving, and a goal description $\delta_G = \mathsf{goalsaved}$, where the goal has been saved. Some potential conditional plans CP_4, CP_5, and CP_6 for achieving δ_G from δ_I are also shown in Fig. 7.

5.2 Goodness of Conditional Plans

We next define the notion of goodness for conditional plans. Intuitively, the best conditional plans are those that reach a goal state from an initial state with highest probability.

We first define the goodness of a conditional plan for achieving a goal state from a belief graph. Given a belief graph B and a conditional plan CP, we say that CP is *executable* in B iff either (i) $CP = \lambda$, or (ii) $CP = \alpha$; CP' and α and CP' are executable in B and $B \circ \alpha$, respectively, or (iii) $CP = \beta$; **if** ω **then** $\{CP_\omega\}$ **else** $\{CP_{\neg\omega}\}$ and β, CP_ω, and $CP_{\neg\omega}$ are executable in B, $B \circ \beta_\omega$, and $B \circ \beta_{\neg\omega}$, respectively. Given a belief graph B, a conditional plan CP that is executable in B, and a goal description δ_G, the *goodness* of CP for achieving δ_G from B, denoted $goodness(CP, B, \delta_G)$, is defined as follows:

$$\begin{cases} prob_{l,B}(\delta_G) & \text{if } CP = \lambda \\ goodness(CP', B \circ \alpha, \delta_G) & \text{if } CP = \alpha;\, CP' \\ \min(goodness(CP_\omega, B \circ \beta_\omega, \delta_G), & \\ \qquad goodness(CP_{\neg\omega}, B \circ \beta_{\neg\omega}, \delta_G)) & \text{if } CP = \beta; \textbf{if } \omega \textbf{ then } \{CP_\omega\} \textbf{ else } \{CP_{\neg\omega}\}. \end{cases}$$

Informally, if CP is empty, then its goodness for achieving δ_G from B is the lower probability of δ_G in B. Otherwise, if CP consists of a physical action α and a conditional plan CP', then its goodness for achieving δ_G from B is the goodness of CP' for achieving δ_G from the successor belief graph of B after executing α. Finally, if CP consists of a sensing action β and one conditional plan CP_o for each outcome $o \in \{\omega, \neg\omega\}$, then its goodness is the minimum of the goodness values of CP_ω and $CP_{\neg\omega}$ for achieving δ_G from the successor belief graphs of B after executing β and observing ω and $\neg\omega$, respectively. We next extend the notion of goodness for conditional plans from belief graphs to initial state descriptions as follows. Given an initial state description δ_I, a conditional plan CP that is executable in the belief graph B_{δ_I} (that is, the belief graph that consists only of the e-state S_{δ_I}, which is the greatest e-state S_{δ_I} of EAD that satisfies δ_I), and a goal description δ_G, the *goodness* of CP for achieving δ_G from δ_I, denoted $goodness(CP, \delta_I, \delta_G)$, is defined as the goodness of CP for achieving δ_G from B_{δ_I}.

Example 5.2 *(Robotic Soccer cont'd)* The goodness values of the conditional plans CP_1 and CP_2 in Fig. 7 for achieving $\delta_G = \neg\mathsf{ballinarea} \wedge \mathsf{inposition}$ from $\delta_I = \mathsf{ballinarea} \wedge \mathsf{inposition} \wedge \neg\mathsf{ballmoving}$ are given by 0.4 and $\min(0.72, 0.56) = 0.56$, respectively, where 0.4 and 0.72 and 0.56 are the lower probabilities of δ_G in the belief graphs in Fig. 6 (1.a), (1.b), and (1.c), respectively. The conditional plan CP_3 has the goodness 0.56 for achieving δ_G from δ_I. The goodness values of the conditional plans CP_4, CP_5, and CP_6 in Fig. 7 for achieving $\delta_G = \mathsf{goalsaved}$ from $\delta_I = \mathsf{ballmoving}$ are given by 0, 0.7, and $\min(1, 0.7) = 0.7$, respectively, where 0, 0.7, 1, and 0.7 are the lower probabilities of δ_G in the belief graphs in Fig. 6 (2.a), (2.b), (2.c), and (2.d), respectively.

The following result shows that the goodness of a conditional plan CP is the minimum of the goodness values of all linearizations of CP, which are roughly all possible sequences of actions from the root to a leaf of CP. Formally, *linearizations* of a conditional plan CP are defined as follows. The only linearization of the empty conditional plan $CP = \lambda$ is λ itself. A linearization of $CP = \alpha;\, CP'$ has the form $\alpha; l$, where l is a linearization of CP'. A linearization of $CP = \beta; \textbf{if } \omega \textbf{ then } \{CP_\omega\} \textbf{ else } \{CP_{\neg\omega}\}$ has the form $\beta_o; l_o$ where $o \in \{\omega, \neg\omega\}$ and l_o is a linearization of CP_o. The executability in belief graphs and the goodness for achieving a goal description from a belief graph or an initial state description are then naturally extended from conditional plans to their linearizations.

Proposition 5.3 *Let EAD be an extended action description, let δ_I be an initial state description, let δ_G be a goal description, and let CP be a conditional plan that is executable in B_{δ_I}. Then, the goodness of CP for achieving δ_G from δ_I is the minimum of the goodness values of all the linearizations of CP for achieving δ_G from δ_I.*

5.3 Problem Statements

The conditional planning problem in our framework of extended action descriptions in $\mathcal{E}+$ can now be formalized as the problem of finding a conditional plan with maximum possible goodness for achieving a goal state from an initial state and as the problem of finding a conditional plan with a goodness of at least a given threshold as follows:

OPTIMAL CONDITIONAL PLANNING: Given an extended action description EAD, an initial state description δ_I, and a goal description δ_G, compute a conditional plan CP that has the maximal goodness among all conditional plans for achieving δ_G from δ_I.

THRESHOLD CONDITIONAL PLANNING: Given an extended action description EAD, an initial state description δ_I, a goal description δ_G, and a threshold $\theta > 0$, compute a conditional plan CP that has a goodness $g \geqslant \theta$ for achieving δ_G from δ_I (if one exists).

Example 5.4 *(Robotic Soccer cont'd)* Some conditional plans of goodness $g \geqslant \theta = 0.4$ for achieving $\delta_G = \neg\mathsf{ballinarea} \wedge \mathsf{inposition}$ from $\delta_I = \mathsf{ballinarea} \wedge \mathsf{inposition} \wedge \neg\mathsf{ballmoving}$ are given by CP_1, CP_2, and CP_3. In fact, the latter two conditional plans have the maximum possible goodness, and thus they are both optimal.

Observe that THRESHOLD CONDITIONAL PLANNING can be easily reduced to OPTIMAL CONDITIONAL PLANNING by first computing a conditional plan of maximal goodness g and then checking whether $g \geqslant \theta$. The following theorem shows that the above two problems are both uncomputable. Its proof is similar to the undecidability proof of the plan existence problem in sequential (unconditional) probabilistic planning given in [Madani et al. 2003]. Note that the variant of THRESHOLD CONDITIONAL PLANNING where the condition $g \geqslant \theta$ (> 0) is replaced by $g > \theta$ ($\geqslant 0$) is also uncomputable.

Theorem 5.5 *The two problems* OPTIMAL CONDITIONAL PLANNING *and* THRESHOLD CONDITIONAL PLANNING *are both uncomputable.*

6. CYCLE-FREE CONDITIONAL PLANNING

In this section, we show that OPTIMAL and THRESHOLD CONDITIONAL PLANNING are both computable in the special case in which G_{EAD,δ_I} is acyclic. More precisely, we present an algorithm for solving THRESHOLD CONDITIONAL PLANNING. For every given problem instance, the algorithm terminates and returns *some* conditional plans of goodness $g \geqslant \theta$ for achieving δ_G from δ_I. In the special case in which G_{EAD,δ_I} is acyclic, the algorithm returns *all* conditional plans of goodness $g \geqslant \theta$ for achieving δ_G from δ_I.

The algorithm is shown in Fig. 8. It uses the function *find_all_cycle_free_paths*, which takes as input the directed graph G_{EAD,δ_I}, an e-state S_0, and a fluent formula ϕ, and which returns as output the set of all paths without cycles from S_0 to an e-state S_n that satisfies ϕ. Every such path $P = S_0 \rightarrow_{\alpha_1} S_1 \rightarrow_{\alpha_2} S_2 \cdots S_{n-1} \rightarrow_{\alpha_n} S_n$ is encoded as the sequence $\alpha_1; \alpha_2; \ldots; \alpha_n$ of labels of the arrows of P. Recall that every α_i is either (a) a deterministic physical action or a sensing action along with one of its outcomes, or (b) a nondeterministic (resp., probabilistic) physical action along with one of its contexts (resp., one of its contexts and a probability value). We then write $P^\star$ to denote the sequence of actions $\alpha'_1; \alpha'_2; \ldots; \alpha'_n$, where (a) $\alpha'_i = \alpha_i$ if α_i is a deterministic physical action or a sensing action along with one of its outcomes, and (b) α'_i is obtained from α_i by removing the

```
Algorithm Cycle-Free Conditional Planning
Input: extended action description $EAD$, initial state description $\delta_I$, goal description $\delta_G$,
       and threshold $\theta > 0$.
Output: set of conditional plans $CP$ such that $goodness(CP, \delta_I, \delta_G) \geqslant \theta$.
 1. $S_L = find\_all\_cycle\_free\_paths(G_{EAD,\delta_I}, S_{\delta_I}, \delta_G)$;
 2. $S_L = \{P^\star \mid P \in S_L,\ goodness(P^\star, \delta_I, \delta_G) \geqslant \theta\}$;
 3. $S_{CP} = S_L$;
 4. while $\exists CP \in S_{CP}$ such that $r\alpha_o \ltimes CP$ but not $r\alpha_{\neg o} \ltimes CP$ do begin
 5.   $L_{aux} = \{L \in S_L \mid r\alpha_{\neg o} \ltimes L\}$;
 6.   $S_{CP} = S_{CP} - \{CP\}$;
 7.   for each $L \in L_{aux}$ do begin
 8.     $CP_{new} = unify(CP, L)$;
 9.     $S_{CP} = S_{CP} \cup \{CP_{new}\}$
10.   end
11. end;
12. return $S_{CP}$.
```

Fig. 8. Algorithm Cycle-Free Conditional Planning

context (resp., the context and the probability value) if α_i belongs to a nondeterministic (resp., probabilistic) physical action. For sensing actions α with outcome $o \in \{\omega, \neg\omega\}$, we write $\neg\neg\omega$ to denote ω. For fragments of conditional plans CP, we denote by $p \ltimes CP$ that p is a prefix of a linearization of CP. We define $unify(CP, L)$ by $unify(\alpha; CP', \alpha; L') = \alpha; unify(CP', L')$ and $unify(\alpha_o; CP', \alpha_{\neg o}; L') = \alpha;$ **if** o **then** $\{CP'\}$ **else** $\{L'\}$.

The algorithm in Fig. 8 works as follows. Step 1 computes the set of all paths P without cycles in G_{EAD,δ_I} from S_{δ_I} to an e-state S that satisfies δ_G. By Proposition 6.2 below, their sequences of actions $P^\star$ are candidates for linearizations of the desired conditional plans. In step 2, using Proposition 5.3, we keep only those linearizations with a goodness of at least θ for achieving δ_G from δ_I. In steps 3–11, we then combine them to conditional plans, and in step 12, we finally return these conditional plans.

Example 6.1 *(Robotic Soccer cont'd)* Consider the initial state description $\delta_I = \mathsf{ballinarea} \wedge \mathsf{inposition} \wedge \neg\mathsf{ballmoving}$, where the ball is in the penalty area and not moving, and the goalkeeper is in the correct position, and the goal description $\delta_G = \neg\mathsf{ballinarea} \wedge \mathsf{inposition}$, where the ball is outside the penalty area, and the goalkeeper is in the correct position. By applying the algorithm in Fig. 8, supposing the threshold $\theta = 0.5$, we compute the set of all cycle-free paths in G_{EAD,δ_I} from S_{δ_I} to some e-state S satisfying δ_G. Consider the two paths $P_1^\star, P_2^\star \in S_L$ in step 2 given by $P_1^\star = \mathsf{gotoball}; \mathsf{sensefreeahead}_{\mathrm{T}}; \mathsf{straightkick}$ and $P_2^\star = \mathsf{gotoball}; \mathsf{sensefreeahead}_{\mathrm{F}}; \mathsf{sidekick}$ (with goodness 0.72 resp. 0.56 as shown in Fig. 6). The path $CP = P_1^\star$ satisfies the condition in step 4 of the algorithm, thus entering the loop. In the next steps, L_{aux} contains $P_2^\star$ and these two paths are unified through the unify function in step 8. The resulting CP_{new}, which is included in the output, is the conditional plan CP_2 shown in Fig. 7 with goodness 0.56.

The following result shows that linearizations from conditional plans of positive goodness for achieving δ_G from δ_I correspond to paths in G_{EAD,δ_I} from S_{δ_I} to an e-state S that satisfies δ_G, which essentially states the correctness of step 1 of the algorithm.

Proposition 6.2 *Let EAD be an extended action description, let δ_I be an initial state description, and let δ_G be a goal description. Let CP be a conditional plan of positive goodness for achieving δ_G from δ_I. Then, for every linearization $L = \alpha_1; \alpha_2; \ldots; \alpha_n$ of CP, there exists a deepest leaf node in $B_{\delta_I} \circ \alpha_1 \circ \alpha_2 \circ \cdots \circ \alpha_n$ whose e-state satisfies δ_G.*

The next result shows that the algorithm always terminates with *some* conditional plans of goodness $g \geqslant \theta$ for achieving δ_G from δ_I in its output. Moreover, if G_{EAD,δ_I} is acyclic, then *all* conditional plans of goodness $g \geqslant \theta$ for achieving δ_G from δ_I are returned.

Theorem 6.3 *Let EAD be an extended action description, let δ_I be an initial state description, let δ_G be a goal description, and let $\theta > 0$ be a threshold. Then, (a) Cycle-Free Conditional Planning terminates, and (b) the algorithm returns a set of conditional plans of goodness $g \geqslant \theta$ for achieving δ_G from δ_I; if G_{EAD,δ_I} is acyclic, then it returns the set of all conditional plans of goodness $g \geqslant \theta$ for achieving δ_G from δ_I.*

As a corollary, we obtain that THRESHOLD CONDITIONAL PLANNING is computable in the case in which G_{EAD,δ_I} is acyclic. Observe that a variant of Cycle-Free Conditional Planning where "$\geqslant \theta$" is replaced by "$> \theta$" can be used for computing a set of conditional plans of goodness $g > \theta \geqslant 0$, and thus in particular for computing the set of all conditional plans of positive goodness in the acyclic case. Since we can then compute the goodness of every such conditional plan and select the ones of maximal goodness, also OPTIMAL CONDITIONAL PLANNING is computable in the case in which G_{EAD,δ_I} is acyclic.

Corollary 6.4 OPTIMAL CONDITIONAL PLANNING *and* THRESHOLD CONDITIONAL PLANNING *are both computable for the class of all instances in which G_{EAD,δ_I} is acyclic.*

7. FINITE-HORIZON CONDITIONAL PLANNING

In this section, we define the problem of finite-horizon conditional planning, which is roughly the problem of finding a conditional plan of bounded length with maximal goodness for achieving a goal description from an initial state description. We then show how some (and even all) optimal conditional plans of bounded length can be computed, which thus proves that this problem is computable. We also show that finite-horizon conditional planning can be used to perform cycle-free conditional planning. Formally, the optimization problem of finite-horizon conditional planning is defined as follows:

FINITE-HORIZON CONDITIONAL PLANNING: Given an extended action description *EAD*, an initial state description δ_I, a goal description δ_G, and a horizon $h \geqslant 0$, compute a conditional plan CP of length $l \leqslant h$ with maximal goodness for achieving δ_G from δ_I.

We now show how to compute a solution to this problem. In the sequel, let EAD be an extended action description, and let δ_G be a goal description. Let $\mathcal{A}' = \mathcal{A} \cup \{nop\}$, where nop is a new deterministic physical action that is executable in every e-state S of EAD and that satisfies $\Phi(S, nop) = S$ for every such S. Informally, nop is the empty action, which is always executable and does not change the e-state. It subsequently allows us to consider only conditional plans that have a length l of exactly the horizon h and whose linearizations all have a length l of exactly the horizon h, even if the optimal conditional plans or some of their linearizations have a length $l < h$, since we can always enlarge such shorter conditional plans and linearizations by filling in nop. We first define the function V^n, $n \geqslant 0$, which associates with every belief graph B and goal description δ_G the maximal goodness of a conditional plan of length $l \leqslant n$ to achieve δ_G from B:

$$V^n(B, \delta_G) = \begin{cases} prob_{l,B}(\delta_G) & \text{if } n = 0 \\ \max\,\{Q^n(B, \alpha, \delta_G) \mid \alpha \in \mathcal{A}',\ \alpha \text{ is executable in } B\} & \text{if } n > 0, \end{cases}$$

where $Q^n(B, \alpha, \delta_G)$ denotes the maximal goodness of a conditional plan that starts with the action α and has the length $l \leqslant n$ to achieve δ_G from B:

$$Q^n(B, \alpha, \delta_G) = \begin{cases} V^{n-1}(B \circ \alpha, \delta_G) & \text{if } \alpha \text{ is a physical action} \\ \min\,\{V^{n-1}(B \circ \alpha_o, \delta_G) \mid o \in \{\omega, \neg\omega\}\} & \text{otherwise.} \end{cases}$$

Informally, $V^0(B, \delta_G)$ is the lower probability of δ_G in B, while $V^n(B, \delta_G)$, $n > 0$, is the maximum of $Q^n(B, \alpha, \delta_G)$ subject to all actions $\alpha \in \mathcal{A}'$ that are executable in B. If α is a physical action, then $Q^n(B, \alpha, \delta_G)$ is the maximal goodness of a conditional plan of length $l \leqslant n{-}1$ to achieve δ_G from $B \circ \alpha$. If α is a sensing action with outcomes ω and $\neg\omega$, then $Q^n(B, \alpha, \delta_G)$ is the minimum of the maximal goodness of a conditional plan of length $l \leqslant n{-}1$ to achieve δ_G from $B \circ \alpha_o$ subject to $o \in \{\omega, \neg\omega\}$.

The following result shows that $V^n(B, \delta_G)$ is indeed the maximal goodness of a conditional plan of length $l \leqslant n$ to achieve the goal description δ_G from the belief graph B.

Theorem 7.1 *Let EAD be an extended action description, and let δ_G be a goal description. Let B be a belief graph, and let $\alpha \in \mathcal{A}'$ be an action that is executable in B. Then, $V^n(B, \delta_G)$ (resp., $Q^n(B, \alpha, \delta_G)$) is the maximal goodness of a conditional plan (resp., a conditional plan that starts with the action α) of length $l \leqslant n$ for achieving δ_G from B.*

We next specify a solution to FINITE-HORIZON CONDITIONAL PLANNING in terms of the function CP^n, $n \geqslant 0$, which assigns to every belief graph B and goal description δ_G a conditional plan of length $l = n$ with maximal goodness for achieving δ_G from B:

$$CP^n(B, \delta_G) = \begin{cases} \lambda & \text{if } n = 0 \\ Aux^n(B, \alpha, \delta_G), \text{ where } \alpha \in \mathcal{A}' \text{ such that (i) } \alpha \text{ is} & \\ \quad \text{executable in } B \text{ and (ii) } V^n(B, \delta_G) = Q^n(B, \alpha, \delta_G) & \text{if } n > 0, \end{cases}$$

where $Aux^n(B, \alpha, \delta_G)$ is the conditional plan that (i) starts with an optimal action α, (ii) has the length $l = n$, and (iii) has maximal goodness for achieving δ_G from B:

$$Aux^n(B, \alpha, \delta_G) = \begin{cases} \alpha;\, CP^{n-1}(B \circ \alpha, \delta_G) & \text{if } \alpha \text{ is a physical action} \\ \alpha; \textbf{if } \omega \textbf{ then } \{CP^{n-1}(B \circ \alpha_\omega, \delta_G)\} & \\ \qquad \textbf{else } \{CP^{n-1}(B \circ \alpha_{\neg\omega}, \delta_G)\} & \text{otherwise.} \end{cases}$$

Informally, $CP^0(B, \delta_G)$ is the empty conditional plan, while $CP^n(B, \delta_G)$, $n > 0$, is the conditional plan $Aux^n(B, \alpha, \delta_G)$. If α is a physical action, then $Aux^n(B, \alpha, \delta_G)$ is built from α and one conditional plan of length $l = n{-}1$. Otherwise, $Aux^n(B, \alpha, \delta_G)$ is constructed from α and two conditional plans of length $l = n{-}1$, one for each outcome of α.

The following theorem shows that $CP^n(B_{\delta_I}, \delta_G)$ provides indeed a conditional plan of length $l \leqslant h$ with maximal goodness for achieving δ_G from δ_I, and thus the problem of FINITE-HORIZON CONDITIONAL PLANNING can be solved by computing $CP^n(B_{\delta_I}, \delta_G)$.

Theorem 7.2 *Let EAD be an extended action description, let δ_I be an initial state description, let δ_G be a goal description, and let $h \geqslant 0$ be a horizon. Then, the conditional plan obtained from $CP^h(B_{\delta_I}, \delta_G)$ by removing all the occurrences of the action nop is a conditional plan of length $l \leqslant h$ with maximal goodness for achieving δ_G from δ_I.*

As an immediate corollary of the previous theorem, we thus obtain that the problem of FINITE-HORIZON CONDITIONAL PLANNING is computable.

Corollary 7.3 FINITE-HORIZON CONDITIONAL PLANNING *is computable.*

The next result provides an upper bound for the complexity of solving FINITE-HORIZON CONDITIONAL PLANNING by using the function CP^n (as described in Theorem 7.2) in terms of basic operations on belief graphs. In particular, it implies that for horizons bounded by a constant, a polynomial number of such basic operations is sufficient.

Theorem 7.4 *Let EAD be an extended action description, let δ_I be an initial state description, let δ_G be a goal description, and let $h \geqslant 0$ be a horizon. Then, the conditional plan $CP^h(B_{\delta_I}, \delta_G)$ can be computed by (i) $O(a \cdot b^{h+1})$ checks whether an action $\alpha \in \mathcal{A}$ is executable in a belief graph, (ii) $O(b^{h+2})$ executions of an action $\alpha \in \mathcal{A}'$ in a belief graph, and (iii) $O(b^{h+1})$ evaluations of δ_G on a belief graph, where $a = |\mathcal{A}|$, $b = |\mathcal{A}_e| + 2 \cdot |\mathcal{A}_s| + 1$, and $\mathcal{A}_e$ and $\mathcal{A}_s$ denote the set of all physical and sensing actions in $\mathcal{A}$, respectively.*

As a corollary, we also obtain an upper bound for the complexity of using the function CP^n in terms of basic operations on e-states, which implies that for horizons bounded by a constant, a polynomial number of basic operations on e-states is sufficient.

Corollary 7.5 *Let EAD be an extended action description, let δ_I be an initial state description, let δ_G be a goal description, and let $h \geqslant 0$ be a horizon. Then, $CP^h(B_{\delta_I}, \delta_G)$ can be computed by (i) $O(a \cdot b^{h+1} \cdot o^h)$ checks whether an action $\alpha \in \mathcal{A}$ is executable in an e-state, (ii) $O(b^{h+2} \cdot o^h)$ executions of an action $\alpha \in \mathcal{A}'$ in an e-state, and (iii) $O(b^{h+1} \cdot o^h)$ evaluations of δ_G on an e-state, where a and b are as in Theorem 7.4, and o is the maximal number of alternatives of nondeterministic and probabilistic actions.*

Since every basic operation on e-states can be done in linear or quadratic time in the size of EAD (see Section 3.3), it thus follows that using the function CP^n can be done in polynomial time when the horizon is bounded by a constant. Furthermore, using CP^n can be done in polynomial time in the size of EAD, when the horizon is bounded by a constant and the maximal number of nondeterministic and probabilistic conditional effect axioms (6) resp. (7) that are relevant to some S and α is also bounded by a constant.

We next show how to compute all conditional plans of length $l \leqslant h$ with maximal goodness for achieving δ_G from δ_I. To this end, we generalize the function CP^n to the following function $\boldsymbol{CP}^n$, which assigns to every belief graph B and goal description δ_G the set of all conditional plans of length $l \leqslant n$ with maximal goodness for achieving δ_G from B:

$$\boldsymbol{CP}^n(B, \delta_G) = \begin{cases} \lambda & \text{if } n = 0 \\ \bigcup\{\boldsymbol{Aux}^n(B, \alpha, \delta_G) \mid \alpha \in \mathcal{A}', \alpha \text{ is executable in } B, & \\ \quad \text{and } V^n(B, \delta_G) = Q^n(B, \alpha, \delta_G)\} & \text{if } n > 0, \end{cases}$$

```
Algorithm Finite-Horizon Conditional Planning
Input: extended action description $EAD$, initial state description $\delta_I$, goal description $\delta_G$,
       and horizon $h \geqslant 0$.
Output: set of all conditional plans $CP$ of length $l \leqslant h$ such that $goodness(CP, \delta_I, \delta_G)$ is maximal.
1. $S_{CP} := \boldsymbol{CP}^h(B_{\delta_I}, \delta_G)$;
2. $S_{CP} := \{CP' \mid CP \in S_{CP},\ CP'$ is obtained from $CP$ by removing all occurrences of $nop\}$;
3. return $S_{CP}$.
```

Fig. 9. Algorithm Finite-Horizon Conditional Planning

where the sets of conditional plans $\boldsymbol{Aux}^n(B, \alpha, \delta_G)$ are defined as follows:

$$\boldsymbol{Aux}^n(B, \alpha, \delta_G) = \begin{cases} \{\alpha; CP \mid CP \in \boldsymbol{CP}^{n-1}(B \circ \alpha, \delta_G)\} & \text{if } \alpha \text{ is a physical action} \\ \{\alpha; \textbf{if } \omega \textbf{ then } \{CP_\omega\} \textbf{ else } \{CP_{\neg\omega}\} \mid & \\ \qquad CP_\omega \in \boldsymbol{CP}^{n-1}(B \circ \alpha_\omega, \delta_G) & \\ \qquad CP_{\neg\omega} \in \boldsymbol{CP}^{n-1}(B \circ \alpha_{\neg\omega}, \delta_G)\} & \text{otherwise.} \end{cases}$$

The following result shows that $\boldsymbol{CP}^h(B_{\delta_I}, \delta_G)$ provides indeed the set of all conditional plans of length $l \leqslant h$ with maximal goodness for achieving δ_G from δ_I.

Theorem 7.6 *Let EAD be an extended action description, let δ_I be an initial state description, let δ_G be a goal description, and let $h \geqslant 0$ be a horizon. Then, the set of conditional plans obtained from $\boldsymbol{CP}^h(B_{\delta_I}, \delta_G)$ by removing all the occurrences of nop is the set of all conditional plans of length $l \leqslant h$ with maximal goodness for achieving δ_G from δ_I.*

An algorithm for computing the set of all optimal conditional plans of length $l \leqslant h$ for achieving δ_G from δ_I using the function $\boldsymbol{CP}^h$ is shown in Fig. 9. The following example illustrates the underlying computation via the functions V^h and Q^h.

Example 7.7 *(Robotic Soccer cont'd)* Consider again the initial state description $\delta_I = \mathsf{ballinarea} \wedge \mathsf{inposition} \wedge \neg\mathsf{ballmoving}$ and the goal description $\delta_G = \neg\mathsf{ballinarea} \wedge \mathsf{inposition}$. For the horizon $h = 2$, the algorithm in Fig. 9 computes the set of all conditional plans of length $l \leqslant 2$ with maximal goodness for achieving δ_G from δ_I. In particular, the returned set of conditional plans contains $CP_1 = \mathsf{gotoball}; \mathsf{bodykick}$, shown in Fig. 7, which is computed via the functions V^2, Q^2, V^1, Q^1, and V^0 as follows:

$$\begin{aligned} V^2(B_{\delta_I}, \delta_G) &= \max\{Q^2(B_{\delta_I}, \alpha, \delta_G) \mid \alpha \in \{\mathsf{gotoball}, \mathsf{sensefreeahead}, \mathsf{senseballclose}, nop\}\} \\ &= Q^2(B_{\delta_I}, \mathsf{gotoball}, \delta_G) \\ &= V^1(B_{\delta_I} \circ \mathsf{gotoball}, \delta_G) \\ &= \max\{Q^1(B_{\delta_I} \circ \mathsf{gotoball}, \alpha, \delta_G) \mid \alpha \in \{\mathsf{bodykick}, \mathsf{gotoball}, \mathsf{sensefreeahead}, \\ &\qquad\qquad \mathsf{senseballclose}, nop\}\} \\ &= Q^1(B_{\delta_I} \circ \mathsf{gotoball}, \mathsf{bodykick}, \delta_G) \\ &= V^0(B_{\delta_I} \circ \mathsf{gotoball} \circ \mathsf{bodykick}, \delta_G) \\ &= prob_{l,\, B_{\delta_I} \circ \mathsf{gotoball} \circ \mathsf{bodykick}}(\delta_G) \\ &= 0.4 \quad \text{(see Fig. 6).} \end{aligned}$$

Note that a slightly modified version of the function CP^h (resp., $\boldsymbol{CP}^h$), where the condition "$V^n(B, \delta_G) = Q^n(B, \alpha, \delta_G)$" is replaced by the condition "$Q^n(B, \alpha, \delta_G) \geqslant \theta$"

can be used for computing a conditional plan (resp., the set of all conditional plans) of length $l \leqslant h$ with goodness $g \geqslant \theta > 0$ for achieving δ_G from δ_I.

The next result shows that, if G_{EAD,δ_I} is acyclic, then for sufficiently large horizons $h \geqslant 0$, the set of all solutions of an instance of FINITE-HORIZON CONDITIONAL PLANNING coincides with the set of all solutions of the corresponding instance of OPTIMAL CONDITIONAL PLANNING, which in turn is a subset of the set of all solutions of a corresponding instance of THRESHOLD CONDITIONAL PLANNING (if it is solvable). Hence, if G_{EAD,δ_I} is acyclic, then the problems of OPTIMAL and THRESHOLD CONDITIONAL PLANNING can both be reduced to FINITE-HORIZON CONDITIONAL PLANNING.

Theorem 7.8 *Let EAD be an extended action description, let δ_I be an initial state description, let δ_G be a goal description. Suppose that G_{EAD,δ_I} is acyclic. Then, there exists a horizon $h \geqslant 0$ such that the set of all conditional plans of maximal goodness for achieving δ_G from δ_I is given by the set of conditional plans obtained from $\boldsymbol{CP}^h(B_{\delta_I}, \delta_G)$ by removing all the occurrences of the action nop.*

8. RELATED WORK

The literature contains several probabilistic extensions of formalisms for reasoning about actions. In particular, Bacchus et al. [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; 2000] 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] 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 and Pirri [2001] add probabilities to the situation calculus to quantify and compare the safety of different sequences of actions. Boutilier et al. [2001] introduce and explore an approach to 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 by Boutilier et al. [2000] presents a generalization of Golog, called DTGolog, that combines robot programming in Golog with decision-theoretic planning in MDPs. Other probabilistic extensions of the situation calculus and Golog are given in [Mateus et al. 2001; Grosskreutz and Lakemeyer 2001]. A probabilistic extension of the action language $\mathcal{A}$ is given by Baral et al. [2002], which aims especially at an elaboration-tolerant representation of MDPs and at formulating observation assimilation and counterfactual reasoning.

Among the above approaches, the most closely related is perhaps Poole's independent choice logic (ICL) [1997], which uses a similar way of adding probabilities to an approach based on acyclic logic programs. But, as a central conceptual difference, like all the other above approaches, Poole's ICL does not allow for qualitative uncertainty in addition to probabilistic uncertainty. Poole circumvents the problem of dealing with qualitative uncertainty by imposing the strong acyclicity condition on logic programs. Moreover, Poole's formalism is inspired more by the situation calculus and less by description logics.

Another closely related work is [Eiter and Lukasiewicz 2003], which proposes the action language $\mathrm{P}\mathcal{C}+$ for probabilistic reasoning about actions, and which is among the few works in the literature that deal with both qualitative and probabilistic uncertainty in reasoning about actions. More precisely, $\mathrm{P}\mathcal{C}+$ allows for expressing nondeterministic and probabilistic effects of actions as well as qualitative and probabilistic 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, and it is then shown how the problems of prediction, postdiction, and unconditional planning under qualitative and probabilistic uncertainty can be formulated in $\mathrm{P}\mathcal{C}+$. However, this work especially does not address sensing.

A further group of important related works is represented by the probabilistic agent programs in [Subrahmanian and Ward 1996; Dix et al. 2000; Dix et al. 2006], which also deal with reasoning about actions in the context of multiple alternative possible world states. More concretely, Subrahmanian and Ward [1996] present an approach to STRIPS-style probabilistic planning and show that it can be equivalently expressed in terms of probabilistic logic programs. Furthermore, Dix et al. [2000] present an approach to probabilistic agent programs, which is based on the ordinary agent programs introduced by Eiter et al. [1999], and which is similar in spirit to Poole's ICL [1997]. Finally, [Dix et al. 2006] is a generalization of [Dix et al. 2000] by temporal probabilistic knowledge. Similarly to our work here, Dix et al. [2000; 2006] allow for dealing with probabilistic uncertainty about the world state. However, differently from here, they do not additionally allow for dealing with qualitative uncertainty about the world state, and they do not allow for directly expressing nondeterministic and probabilistic effects of actions. Notice also that their multiple alternative possible world states are due to probabilistic initial states, while ours are due to actions with nondeterministic and probabilistic effects. Differently from our work, they also do not define a semantics based on autoepistemic description logics, they do not consider belief trees, and they do not focus on solving the conditional planning problem.

From a more general perspective, our approach is also related to planning under uncertainty in AI, since it can be roughly understood as a combination of (i) conditional planning under nondeterministic uncertainty [Geffner 2002] with (ii) conditional planning under probabilistic uncertainty, both in partially observable environments. Previous work on planning under probabilistic uncertainty can be roughly divided into (a) generalizations of classical planning and (b) decision-theoretic planning. The former (see for example [Draper et al. 1994; Onder and Pollack 1999; Karlsson 2001]) typically considers the problem of determining a sequence of actions given a success threshold, with some extensions that consider also sensing and conditional plans. Decision-theoretic planning, on the other hand, deals with fully observable Markov decision processes (MDPs) [Puterman 1994] or the more general partially observable Markov decision processes (POMDPs) [Kaelbling et al. 1998], which also include costs and/or rewards associated with actions and/or states, and their solutions are mappings from situations to actions of high expected utility, rather than courses of actions achieving a goal with high probability. Summarizing, our approach can perhaps best be seen as combining conditional planning under nondeterministic and under probabilistic uncertainty, where the latter is perhaps closest to generalizations of classical planning in AI. 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.

9. CONCLUSION

In this paper, we have presented the language $\mathcal{E}+$ for reasoning about actions with sensing under qualitative and probabilistic uncertainty.

The proposed framework has several interesting features of reasoning about actions, such as sensing, persistence, and static constraints, and it combines them with nondeterministic and probabilistic effects of actions. The proposed formalism also provides a complete integration of the epistemic and probabilistic beliefs of an agent.

We have formulated the problem of conditional planning under qualitative and probabilistic uncertainty, and we have presented two algorithms for conditional planning in our framework. The first one is always sound, and it is also complete for the special case where the relevant transitions between epistemic states are cycle-free. The second algorithm is a sound and complete solution to the problem of finite-horizon conditional planning. Under the assumption that the horizon is bounded by a constant, it computes every optimal finite-horizon conditional plan in polynomial time.

Finally, several examples have illustrated our formalism. They describe a robotic soccer scenario in which we model at the same time the sensing abilities of a robot, as well as nondeterministic and probabilistic uncertainty in the execution of its actions. More precisely, the examples show how this scenario can be modeled in our formalism, and they illustrate the concepts of belief graph and conditional plan, the evaluation of different possible conditional plans, and their computation using the presented algorithms. They show not only the need for an integrated formalism in realistic applications, but also that the choices in modeling uncertainty in the actions affect the behavior of the robot.

While from the representation standpoint our formalism provides a rather rich framework, a number of issues still deserve further investigation. Specifically, we are currently addressing extensions of the proposed framework that generalize it by introducing noise in sensing actions (for example, along the lines of [Bacchus et al. 1995; Shapiro 2005]), as well as actions with costs and/or rewards (for example, such as in POMDPs [Kaelbling et al. 1998]). Moreover, we are improving the implementation of the prototype planner to make it suitable for quantitative experiments and performance evaluation.

Another interesting topic of future research would be to elaborate an extension of the presented formalism to multi-agent systems. Furthermore, it would also be very interesting to investigate whether $\mathcal{E}+$ can be applied in web services: The semantic foundation of $\mathcal{E}+$ on description logics is in spirit of a recent trend towards combining action languages with description logics [Baader et al. 2005] for modeling web services in the *Semantic Web* [Berners-Lee 1999; Fensel et al. 2002]. Here, description logics in general play a crucial role as a formal foundation for the *OWL Web Ontology Language* [W3C 2004; Horrocks et al. 2003] and autoepistemic description logics in particular as a mechanism for combining rules and ontologies [Motik et al. 2006; Motik and Rosati 2007].

A. APPENDIX: NOTATION TABLE

Symbol	**Description**	**Section**
$\mathcal{F}$	set of fluents	2.1
$\mathcal{A}$	set of actions	2.1
ϕ, ψ	fluent formulas	2.1
ℓ_i, ω	fluent literals	2.1
α	action in $\mathcal{A}$	2.1
δ_I	initial state description	2.1
AD	action description	2.1
s	state of AD	2.2
S	epistemic state (or e-state) of AD	2.2
$\Phi(S, \alpha)$	successor e-state of an e-state S of AD under action α	2.2
G_{AD}	directed graph represented by AD	2.2
S_{δ_I}	e-state encoded by δ_I	2.2
G_{AD,δ_I}	subgraph of G_{AD} with all successors of S_{δ_I}	2.2
L	set of fluent literals	2.3
$Lit(\phi)$	set of all fluent literals in ϕ	2.3
$Lit(S)$	set of all fluent literals satisfied by S	2.3
p_i	probability value	3.1
EAD	extended action description	3.1
$C_{S,\alpha}$	set of contexts when executing α in S	3.1
$\Phi_c(S, \alpha)$	the successor e-state of S after executing α in the context c	3.1
$F_\alpha(S)$	set of successor e-states of S under α	3.1
$Pr_\alpha(\cdot \mid S)$	probability distribution on the successor e-state of S under α	3.1
G_{EAD}	directed graph represented by EAD	3.1
G_{EAD,δ_I}	subgraph of G_{EAD} with all successors of S_{δ_I}	3.1
$B = (V, E, \ell, Pr)$	belief graph	4.1
$prob_{l,B}(\phi)$	lower probability of ϕ in B	4.2
$prob_{u,B}(\phi)$	upper probability of ϕ in B	4.2
$\boldsymbol{\mu}_B$	set of unnormalized probability distributions associated with B	4.3
δ_G	goal description	5
CP	conditional plan	5.1
λ	empty conditional plan	5.1
$goodness(CP, B, \delta_G)$	goodness of CP for achieving δ_G from B	5.2
$V^n(B, \delta_G)$	maximal goodness of a conditional plan of length $l \leqslant n$ to achieve δ_G from B	7
$Q^n(B, \alpha, \delta_G)$	maximal goodness of a conditional plan that starts with action α and has the length $l \leqslant n$ to achieve δ_G from B	7
$CP^n(B, \delta_G)$	conditional plan of length $l = n$ with maximal goodness for achieving δ_G from B	7
$Aux^n(B, \alpha, \delta_G)$	conditional plan that starts with an optimal action α, has the length $l = n$, and has maximal goodness for achieving δ_G from B	7

Fig. 10. Notation

B. APPENDIX: PROOFS

Proof of Proposition 2.2. It is sufficient to show that the default frame axioms in AD do not produce any nondeterministic choice. Let I be the set of all default frame axioms **inertial** ϕ **after** α in AD that are *applicable* in S, that is, such that the set of all fluent literals in ϕ, denoted $Lit(\phi)$, is contained in the set of all fluent literals representing S, denoted $Lit(S)$. It then follows that also $Lit^\star(\phi)$ is contained in $Lit(S)$, for every **inertial** ϕ **after** α in I, where $Lit^\star(\phi)$ is the closure of $Lit(\phi)$ under all domain constraint axioms in AD. This shows in particular that the union of all $Lit^\star(\phi)$ such that **inertial** ϕ **after** α is in I is consistent. It thus follows that for any consistent set of fluent literals L, the set of fluent literals $L \cup Lit^\star(\phi_1) \cup \cdots \cup Lit^\star(\phi_n)$, where **inertial** ϕ_i **after** α belongs to I for every $i \in \{1, \ldots, n\}$, is consistent iff every $L \cup Lit^\star(\phi_i)$ with $i \in \{1, \ldots, n\}$ is consistent. Hence, any consistent set of fluent literals $L \cup Lit^\star(\phi_1) \cup \cdots \cup Lit^\star(\phi_n)$, where (i) **inertial** ϕ_i **after** α belongs to I for every $i \in \{1, \ldots, n\}$, and (ii) $\{$**inertial** ϕ_i **after** $\alpha \mid i \in \{1, \ldots, n\}\}$ is maximal, is unique and exactly given through all **inertial** ϕ_i **after** α in I such that $L \cup Lit^\star(\phi_i)$ is consistent. $\Box$

Proof of Theorem 4.5. (a) Recall first that an action α is executable in B iff it is executable in the e-state $\ell(v) = S$ of some deepest leaf v of B. Observe then that the e-states of the deepest leaves of B are exactly the e-states $S \in \mathcal{S}$ such that $\mu(S) > 0$ for some $\mu \in \boldsymbol{\mu}_B$.

(b) (resp., (c)) The set of unnormalized probability distributions over $\mathcal{S}$ associated with the belief graph $B \circ \alpha$ (resp., $B \circ \alpha_o$) coincides with the set of unnormalized probability distributions over $\mathcal{S}$ associated with the belief graph obtained from B by replacing the e-state $\ell(v) = S$ of every deepest leaf v such that α is executable in S by the e-state $S' = \Phi(S, \alpha)$ (resp., $S' = \Phi(S, \alpha_o)$). The latter is given by the set of all $\mu \circ \alpha$ (resp., $\mu \circ \alpha_o$) with $\mu \in \boldsymbol{\mu}_B$.

(d) The set of unnormalized probability distributions over $\mathcal{S}$ associated with $B \circ \alpha$ coincides with the union of all $\boldsymbol{\mu}_{\widetilde{\alpha}}$ such that $\widetilde{\alpha} \in inst(\alpha)$, where every $\boldsymbol{\mu}_{\widetilde{\alpha}}$ is the set of unnormalized probability distributions over $\mathcal{S}$ associated with the belief graph obtained from B by replacing the e-state $\ell(v) = S$ of every deepest leaf v such that α is executable in S by the e-state $S' = \Phi(S, \widetilde{\alpha})$. Every such $\boldsymbol{\mu}_{\widetilde{\alpha}}$ is given by the set of all $\mu \circ \widetilde{\alpha}$ with $\mu \in \boldsymbol{\mu}_B$.

(e) Recall that for every deepest leaf v of B, the set of unnormalized probability distributions $\boldsymbol{\mu}_v$ associated with v in B is given by the probability distribution μ_v that maps the e-state $\ell(v) = S$ to 1 and all other e-states $S \in \mathcal{S}$ to 0. Observe then that for every deepest leaf v of B such that α is executable in the e-state $\ell(v) = S$, the set of unnormalized probability distributions $\boldsymbol{\mu}_v$ associated with v in $B \circ \alpha$ is given by the unnormalized probability distribution μ that maps every $S' \in \mathcal{S}$ for which some $c \in C_{S,\alpha}$ exists with $S' = \Phi_c(S, \alpha)$ to $Pr_\alpha(S'|S)$ and all other e-states $S' \in \mathcal{S}$ to 0. Hence, the set of unnormalized probability distributions over $\mathcal{S}$ associated with $B \circ \alpha$ is given by the set of all $\mu \circ \alpha$ with $\mu \in \boldsymbol{\mu}_B$. $\Box$

Proof of Theorem 4.6. (a) Let $B = (V, E, \ell, Pr)$, and let $r \in V$ be the root of B. Let the subgraph $G_d = (V_d, E_d)$ of $G = (V, E)$ be defined as in Section 4.2. By induction on the recursive structure of G_d, we show that $prob_{l,v}(\phi) = \min_{\mu \in \boldsymbol{\mu}_v} \sum_{S \in \mathcal{S},\, S \models \phi} \mu(S)$ for all $v \in V_d$. Analogously, it can be shown that $prob_{u,v}(\phi) = \max_{\mu \in \boldsymbol{\mu}_v} \sum_{S \in \mathcal{S},\, S \not\models \neg\phi} \mu(S)$ for all $v \in V_d$. Since the above holds in particular for the root r of B, this then proves (a).

Basis: Let $v \in V_d$ be a leaf. Then, $prob_{l,v}(\phi)$ is 1 if $\ell(v) \models \phi$, and 0 otherwise. Furthermore, $\boldsymbol{\mu}_v$ is given by $\{\mu_v\}$, where $\mu_v(\ell(v)) = 1$ and $\mu_v(S) = 0$ for every other e-state

$S \in \mathcal{S}$. Hence, $\min_{\mu \in \boldsymbol{\mu}_v} \sum_{S \in \mathcal{S},\, S \models \phi} \mu(S) = \sum_{S \in \mathcal{S},\, S \models \phi} \mu_v(S)$ is 1 if $\ell(v) \models \phi$, and 0 otherwise. This shows that $prob_{l,v}(\phi) = \min_{\mu \in \boldsymbol{\mu}_v} \sum_{S \in \mathcal{S},\, S \models \phi} \mu(S)$.

Induction: Let $v \in V_d$ be a non-leaf node. Suppose first that $Pr(e)$ is undefined for all outgoing arrows e of v. Then, $prob_{l,v}(\phi) = \min_{v \to v' \in E_d} prob_{l,v'}(\phi)$. By the induction hypothesis, the latter coincides with $\min_{v \to v' \in E_d} \min_{\mu \in \boldsymbol{\mu}_{v'}} \sum_{S \in \mathcal{S},\, S \models \phi} \mu(S) = \min_{\mu \in \boldsymbol{\mu}_v} \sum_{S \in \mathcal{S},\, S \models \phi} \mu(S)$. Suppose next that $Pr(e)$ is defined for all outgoing arrows e of v. Then, $prob_{l,v}(\phi) = \sum_{e = v \to v' \in E_d} Pr(e) \cdot prob_{l,v'}(\phi)$. By the induction hypothesis, the latter is equal to $\sum_{e = v \to v' \in E_d} Pr(e) \cdot \min_{\mu \in \boldsymbol{\mu}_{v'}} \sum_{S \in \mathcal{S},\, S \models \phi} \mu(S) = \min_{\mu \in \boldsymbol{\mu}_v} \sum_{S \in \mathcal{S},\, S \models \phi} \mu(S)$. In summary, this shows that $prob_{l,v}(\phi) = \min_{\mu \in \boldsymbol{\mu}_v} \sum_{S \in \mathcal{S},\, S \models \phi} \mu(S)$.

(b) Immediate by (a) and the definition of the executability probability of B. □

Proof of Proposition 5.3. By induction on the structure of conditional plans CP, we show that for every belief graph B in which CP is executable, $goodness(CP, B, \delta_G)$ is the minimum of $goodness(l, B, \delta_G)$ subject to all linearizations l of CP.

Basis: Let $CP = \lambda$. Since λ is the only linearization of CP, $goodness(CP, B, \delta_G)$ is the minimum of $goodness(l, B, \delta_G)$ subject to all linearizations l of CP.

Induction: Let $CP = \alpha; CP'$. Then, $goodness(CP, B, \delta_G) = goodness(CP', B \circ \alpha, \delta_G)$. By the induction hypothesis, the latter is the minimum of $goodness(l', B \circ \alpha, \delta_G)$ subject to all linearizations l' of CP', which coincides with the minimum of $goodness(l, B, \delta_G)$ subject to all linearizations l of CP. Finally, let $CP = \beta$; **if** ω **then** $\{CP_\omega\}$ **else** $\{CP_{\neg\omega}\}$. Then, $goodness(CP, B, \delta_G)$ is the minimum of $goodness(CP_o, B \circ \beta_o, \delta_G)$ subject to $o \in \{\omega, \neg\omega\}$. By the induction hypothesis, each of the latter is given by the minimum of $goodness(l_o, B \circ \beta_o, \delta_G)$ subject to all linearizations l_o of CP_o, which coincides with the minimum of $goodness(l, B, \delta_G)$ subject to all linearizations l of CP. □

Proof of Theorem 5.5. Let THRESHOLD CONDITIONAL PLAN EXISTENCE denote the following decision problem: Given an extended action description EAD, an initial state description δ_I, a goal description δ_G, and a threshold $\theta > 0$, decide whether there exists a conditional plan CP that has a goodness of at least θ for achieving δ_G from δ_I. Observe then that THRESHOLD CONDITIONAL PLAN EXISTENCE can be easily reduced to THRESHOLD CONDITIONAL PLANNING, which in turn can be easily reduced to OPTIMAL CONDITIONAL PLANNING. It is thus sufficient to show that THRESHOLD CONDITIONAL PLAN EXISTENCE is undecidable. We show this by a reduction from the language emptiness problem for probabilistic finite automata (PFA), which is undecidable by [Paz 1971] and [Condon and Lipton 1989]. More precisely, a *probabilistic finite automaton (PFA)* is a tuple (S, Σ, T, s_0, s_a), where S is a nonempty finite set of states, Σ is a finite input alphabet, $T = \{T_a \mid a \in \Sigma\}$ where every T_a is a transition function that associates with every state $s \in S$ a probability distribution $T_a(\cdot \mid s)$ over the set of states S, $s_0 \in S$ is an initial state, and $s_a \in S$ is an accepting state. The language emptiness problem is the problem of deciding, given a PFA (S, Σ, T, s_0, s_a) and a threshold $\theta > 0$, whether there exists an input string $w \in \Sigma^\star$ that the PFA accepts with a probability of at least θ (that is, the probabilities of all possible transitions from s_0 to s_a on w sum up to a value of at least θ).

We reduce the language emptiness problem for PFAs to THRESHOLD CONDITIONAL PLAN EXISTENCE as follows. Let (S, Σ, T, s_0, s_a) be a PFA, where $S = \{s_0, \ldots, s_n = s_a\}$ and $n \geqslant 0$, and let $\theta > 0$ be a threshold. We then define the set of actions $\mathcal{A}$ as the input alphabet Σ, where each $a \in \mathcal{A}$ is a probabilistic physical action, and the set of fluents $\mathcal{F}$ as the set of states S. The extended action description EAD contains one conditional probabilistic effect axiom of the form **caused** $\phi_0 \colon p_0, \ldots, \phi_n \colon p_n$ **after** a **when** ϕ_j for every action $a \in \mathcal{A}$ and $j \in \{0, \ldots, n\}$, where $\phi_i = s_i \wedge \bigwedge_{k \in \{0,\ldots,n\}-\{i\}} \neg s_k$ and $p_i = T_a(s_i | s_j)$ for all $i \in \{0, \ldots, n\}$. Let $\delta_I = s_0 \wedge \bigwedge_{k \in \{1,\ldots,n\}} \neg s_k$ and $\delta_G = s_n \wedge \bigwedge_{k \in \{0,\ldots,n-1\}} \neg s_k$. Then, there is an input string $w \in \Sigma^\star$ that the PFA accepts with a probability of at least θ iff there is a conditional plan CP with a goodness of at least θ for achieving δ_G from δ_I. $\Box$

Proof of Proposition 6.2. Towards a contradiction, suppose that there exists a linearization $L = \alpha_1; \alpha_2; \ldots; \alpha_n$ of CP such that the e-state $\ell(v) = S$ of every deepest leaf node v in $B_{\delta_I} \circ \alpha_1 \circ \alpha_2 \circ \cdots \circ \alpha_n$ does not satisfy δ_G. Hence, the goodness of L for achieving δ_G from δ_I is given by 0. Thus, by Proposition 5.3, the goodness of CP for achieving δ_G from δ_I is also given by 0. But this contradicts CP having a positive goodness for achieving δ_G from δ_I. This shows that for every linearization $L = \alpha_1; \alpha_2; \ldots; \alpha_n$ of CP, there exists a deepest leaf node in $B_{\delta_I} \circ \alpha_1 \circ \alpha_2 \circ \cdots \circ \alpha_n$ whose e-state satisfies δ_G. $\Box$

Proof of Theorem 6.3. (a) Immediate by the observation that (i) there are only finitely many acyclic paths in G_{EAD,δ_I} from S_{δ_I} to some e-state S that satisfies δ_G, and thus (ii) both the while-loop and the for-loop terminate after a finite number of iterations.

(b) We now prove that for all conditional plans CP, it holds that CP is returned by the algorithm iff CP has a goodness of at least θ for achieving δ_G from δ_I, where the "$\Leftarrow$"-part of the statement holds only in the special case in which G_{EAD,δ_I} is acyclic.

($\Rightarrow$) Suppose CP is a conditional plan returned by the algorithm. By step 2, CP consists only of linearizations of goodness of at least θ for achieving δ_G from δ_I. Hence, by Proposition 5.3, CP has also a goodness of at least θ for achieving δ_G from δ_I.

($\Leftarrow$) Suppose CP is a conditional plan of goodness of at least θ for achieving δ_G from δ_I. By Proposition 6.2, for every linearization $L = \alpha_1; \alpha_2; \ldots; \alpha_n$ of CP, there exists a deepest leaf node in $B_{\delta_I} \circ \alpha_1 \circ \alpha_2 \circ \cdots \circ \alpha_n$ whose e-state satisfies δ_G. Thus, every such linearization L of CP has a corresponding path P in G_{EAD,δ_I} from S_{δ_I} to some e-state S that satisfies δ_G. Since G_{EAD,δ_I} is acyclic, also P is acyclic, and thus P is included in S_L in step 1 of the algorithm. By Proposition 5.3, L has a goodness of at least θ for achieving δ_G from δ_I, and thus L is included in S_L in step 2 of the algorithm. It thus follows that S_{CP} and S_L in step 3 contain all linearizations of CP, and thus CP is constructed in steps 4–11 and included in the set of conditional plans returned in step 12. $\Box$

Proof of Theorem 7.1. We prove by induction on $n \geqslant 0$ that, for every belief graph B and goal description δ_G, it holds that $V^n(B, \delta_G)$ (resp., $Q^n(B, \alpha, \delta_G)$) is the maximal goodness of a conditional plan (resp., a conditional plan that starts with the action α) over $\mathcal{A}' = \mathcal{A} \cup \{nop\}$ of length $l = n$ to achieve δ_G from B. This then proves that, for every belief graph B and goal description δ_G, it holds that $V^n(B, \delta_G)$ (resp., $Q^n(B, \alpha, \delta_G)$) is the maximal goodness of a conditional plan (resp., a conditional plan that starts with the action α) over $\mathcal{A}$ of length $l \leqslant n$ to achieve δ_G from B.

Basis: For $n = 0$, only the empty conditional plan λ has the length $l = 0$. Since λ has the goodness $prob_{l,B}(\delta_G)$ for achieving δ_G from B, it follows that $V^0(B, \delta_G) = prob_{l,B}(\delta_G)$ is the maximal goodness of a conditional plan of length $l = 0$ to achieve δ_G from B.

Induction: Let $n > 0$. By the induction hypothesis, $V^{n-1}(B', \delta_G)$ is the maximal goodness of a conditional plan of length $l = n-1$ to achieve δ_G from the belief graph B'. This shows that $Q^n(B, \alpha, \delta_G)$ is the maximal goodness of a conditional plan that starts with α of length $l = n$ to achieve δ_G from B. It thus follows that $V^n(B, \delta_G)$ (which is the maximum of $Q^n(B, \alpha, \delta_G)$ subject to all actions $\alpha \in \mathcal{A}'$ that are executable in B) is the maximal goodness of a conditional plan of length $l = n$ to achieve δ_G from B. $\Box$

Proof of Theorem 7.2. We prove by induction on $h \geqslant 0$ that, for every belief graph B and goal description δ_G, it holds that $CP^h(B, \delta_G)$ is a conditional plan of length $l \leqslant h$ with maximal goodness for achieving δ_G from B. This then shows that $CP^h(B_{\delta_I}, \delta_G)$ is a conditional plan of length $l \leqslant h$ with maximal goodness for achieving δ_G from δ_I.

Basis: For $h = 0$, only the empty conditional plan λ is of length 0. Thus, $CP^0(B, \delta_G) = \lambda$ is a conditional plan of length $l \leqslant 0$ with maximal goodness for achieving δ_G from B.

Induction: Let $h > 0$. By the induction hypothesis, for every belief graph B', it holds that $CP^{h-1}(B', \delta_G)$ is a conditional plan of length $l \leqslant h-1$ with maximal goodness for achieving δ_G from B'. By Theorem 7.1, $V^h(B, \delta_G)$ (resp., $Q^h(B, \alpha, \delta_G)$) is the maximal goodness of a conditional plan (resp., a conditional plan that starts with the action α) of length $l \leqslant h$ to achieve δ_G from B. It thus follows that $CP^h(B, \delta_G)$ is a conditional plan of length $l \leqslant h$ with maximal goodness for achieving δ_G from B. $\Box$

Proof of Theorem 7.4. The value $V^n(B, \delta_G)$ and all values $Q^n(B, \alpha, \delta_G)$ such that (a) $\alpha \in \mathcal{A}'$ and (b) α is executable in B can be computed by (i) at most $a \cdot b^n$ checks whether an action $\alpha \in \mathcal{A}$ is executable in a belief graph, (ii) at most b^{n+1} executions of an action $\alpha \in \mathcal{A}'$ in a belief graph, and (iii) at most b^n evaluations of δ_G on a belief graph. Hence, if $\mathcal{A}$ is nonempty, then $CP^h(B, \delta_G)$ can be computed by (i) at most $a \cdot b^{h+1}$ checks whether an action $\alpha \in \mathcal{A}$ is executable in a belief graph, (ii) at most $b^{h+2} + 2^h$ executions of an action $\alpha \in \mathcal{A}'$ in a belief graph, and (iii) at most b^{h+1} evaluations of δ_G on a belief graph. $\Box$

Proof of Theorem 7.6. Immediate by the proof of Theorem 7.2. $\Box$

Proof of Theorem 7.8. Since the subgraph of G_{EAD} that consists of all successors of S_{δ_I} is finite and has no cycles, the set of all conditional plans is finite. Thus, some $h \geqslant 0$ exists such that every conditional plan has a length $l \leqslant h$. By Theorem 7.6, the set of conditional plans obtained from $\boldsymbol{CP}^h(B_{\delta_I}, \delta_G)$ by removing all the occurrences of the action nop is the set of all conditional plans with maximal goodness for achieving δ_G from δ_I. $\Box$

REFERENCES

BAADER, F., LUTZ, C., MILICIC, M., SATTLER, U., AND WOLTER, F. 2005. Integrating description logics and action formalisms: First results. In *Proceedings AAAI-2005*. AAAI Press / MIT Press, 572–577.

BACCHUS, F., HALPERN, J. Y., AND LEVESQUE, H. J. 1995. Reasoning about noisy sensors and effectors in the situation calculus. In *Proceedings IJCAI-1995*. Morgan Kaufmann, 1933–1940.

BACCHUS, F., HALPERN, J. Y., AND LEVESQUE, H. J. 1999. Reasoning about noisy sensors and effectors in the situation calculus. *Artif. Intell. 111*, 171–208.

BARAL, C., TRAN, N., AND TUAN, L.-C. 2002. Reasoning about actions in a probabilistic setting. In *Proceedings AAAI-2002*. AAAI Press, 507–512.

BERNERS-LEE, T. 1999. *Weaving the Web*. Harper, San Francisco, CA.

BOUTILIER, C., REITER, R., AND PRICE, B. 2001. Symbolic dynamic programming for first-order MDPs. In *Proceedings IJCAI-2001*. Morgan Kaufmann, 690–700.

BOUTILIER, C., REITER, R., SOUTCHANSKI, M., AND THRUN, S. 2000. Decision-theoretic, high-level agent programming in the situation calculus. In *Proceedings AAAI-2000*. AAAI Press / MIT Press, 355–362.

CONDON, A. AND LIPTON, R. 1989. On the complexity of space bounded interactive proofs. In *Proceedings FOCS-1989*. IEEE Computer Society, 462–467.

DIX, J., KRAUS, S., AND SUBRAHMANIAN, V. S. 2006. Heterogeneous temporal probabilistic agents. *ACM Trans. Comput. Log. 7,* 1, 151–198.

DIX, J., NANNI, M., AND SUBRAHMANIAN, V. S. 2000. Probabilistic agent programs. *ACM Trans. Comput. Log. 1,* 2, 208–246.

DONINI, F. M., NARDI, D., AND ROSATI, R. 2002. Description logics of minimal knowledge and negation as failure. *ACM Trans. Comput. Log. 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*. AAAI Press, 31–36.

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.

EITER, T. AND LUKASIEWICZ, T. 2003. Probabilistic reasoning about actions in nonmonotonic causal theories. In *Proceedings UAI-2003*. Morgan Kaufmann, 192–199.

EITER, T., SUBRAHMANIAN, V. S., AND PICK, G. 1999. Heterogeneous active agents, I: Semantics. *Artif. Intell. 108,* 1–2, 179–255.

FENSEL, D., WAHLSTER, W., LIEBERMAN, H., AND HENDLER, J., Eds. 2002. *Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential*. MIT Press.

FINZI, A. AND PIRRI, F. 2001. Combining probabilities, failures and safety in robot control. In *Proceedings IJCAI-2001*. Morgan Kaufmann, 1331–1336.

GEFFNER, H. 2002. Perspectives on artificial intelligence planning. In *Proceedings AAAI-2002*. AAAI Press, 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,* 1–2, 49–104.

GROSSKREUTZ, H. AND LAKEMEYER, G. 2001. Belief update in the pGOLOG framework. In *Proceedings KI/ÖGAI-2001*. LNCS, vol. 2174. Springer, 213–228.

HALPERN, J. Y. AND TUTTLE, M. R. 1993. Knowledge, probability, and adversaries. *J. ACM 40,* 4, 917–962.

HORROCKS, I., PATEL-SCHNEIDER, P. F., AND VAN HARMELEN, F. 2003. From $\mathcal{SHIQ}$ and RDF to OWL: The making of a web ontology language. *J. Web Semantics 1,* 1, 7–26.

IOCCHI, L. 1999. *Design and Development of Cognitive Robots.* Ph.D. thesis, University of Rome "La Sapienza", Rome, Italy. Available at `www.dis.uniroma1.it/~iocchi/publications.html`.

IOCCHI, L., LUKASIEWICZ, T., NARDI, D., AND ROSATI, R. 2006. Reasoning about actions with sensing under qualitative and probabilistic uncertainty. Tech. Rep. INFSYS RR-1843-03-05, Institut für Informationssysteme, Technische Universität Wien. March.

IOCCHI, L., NARDI, D., AND ROSATI, R. 2000. Planning with sensing, concurrency, and exogenous events: Logical framework and implementation. In *Proceedings KR-2000*. Morgan Kaufmann, 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*. Morgan Kaufmann, 431–438.

LANG, J., LIN, F., AND MARQUIS, P. 2003. Causal theories of action: A computational core. In *Proceedings IJCAI-2003*. Morgan Kaufmann, 1073–1078.

LEVESQUE, H. J. 1996. What is planning in presence of sensing? In *Proceedings AAAI-1996*. AAAI Press / MIT Press, 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. 1997. Adding knowledge to the action description language A. In *Proceedings AAAI-1997*. AAAI Press / MIT Press, 454–459.

MADANI, O., HANKS, S., AND CONDON, A. 2003. On the undecidability of probabilistic planning and related stochastic optimization problems. *Artif. Intell. 147,* 1–2, 5–34.

MATEUS, P., PACHECO, A., PINTO, J., SERNADAS, A., AND SERNADAS, C. 2001. Probabilistic situation calculus. *Ann. Math. Artif. Intell. 32*, 393–431.

MOTIK, B., HORROCKS, I., ROSATI, R., AND SATTLER, U. 2006. Can OWL and logic programming live together happily ever after? In *Proceedings ISWC-2006*. LNCS, vol. 4273. Springer, 501–514.

MOTIK, B. AND ROSATI, R. 2007. A faithful integration of description logics with logic programming. In *Proceedings IJCAI-2007*. 477–482.

ONDER, N. AND POLLACK, M. E. 1999. Conditional, probabilistic planning: A unifying algorithm and effective search control mechanisms. In *Proceedings AAAI-1999*. AAAI Press / MIT Press, 577–584.

PAZ, A. 1971. *Introduction to Probabilistic Automata*. Academic Press, New York.

PIRRI, F. AND REITER, R. 1999. Some contributions to the metatheory of the situation calculus. *J. ACM 46,* 3, 325–361.

POOLE, D. 1997. The independent choice logic for modelling multiple agents under uncertainty. *Artif. Intell. 94,* 1-2, 7–56.

POOLE, D. 2000. Logic, knowledge representation, and Bayesian decision theory. In *Proceedings CL-2000*. LNCS, vol. 1861. Springer, 70–86.

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.

SHAPIRO, S. 2005. Belief change with noisy sensing and introspection. In *Proceedings NRAC-2005*.

SON, T. C. AND BARAL, C. 2001. Formalizing sensing actions: A transition function based approach. *Artif. Intell. 125,* 1–2, 19–91.

SON, T. C., TU, P. H., AND BARAL, C. 2004. Planning with sensing actions and incomplete information using logic programming. In *Proceedings LPNMR-2004*. LNCS/LNAI, vol. 2923. Springer, 261–274.

SUBRAHMANIAN, V. S. AND WARD, C. 1996. A deductive database approach to planning in uncertain environments. In *Proceedings LID-1996*. LNCS, vol. 1154. Springer, 83–98.

W3C. 2004. OWL web ontology language overview. W3C Recommendation (10 February 2004). Available at `www.w3.org/TR/2004/REC-owl-features-20040210/`.

ZHANG, D., CHOPRA, S., AND FOO, N. Y. 2002. Consistency of action descriptions. In *Proceedings PRICAI-2002*. LNCS/LNAI, vol. 2417. Springer, 70–79.

Received March 2006; revised March 2007; accepted July 2007