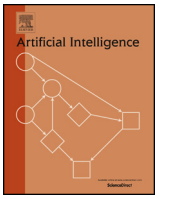
Artificial Intelligence 307 (2022) 103696

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint



Analyzing generalized planning under nondeterminism ☆,☆☆

Vaishak Belle [a,b,*]

[a] School of Informatics, University of Edinburgh, Edinburgh, UK
[b] Alan Turing Institute, London, UK

ARTICLE INFO

Article history:
Received 25 November 2020
Received in revised form 11 October 2021
Accepted 7 March 2022
Available online 16 March 2022

Keywords:
Knowledge representation
Reasoning about action
Cognitive robotics
Plans with loops
Iterative planning

ABSTRACT

In automated planning, there has been a recent interest in solving a class of problems, where a single solution applies for multiple, possibly infinitely many, instances. This necessitates a generalized notion of plans, such as plans with loops. However, the correctness of such plans is non-trivial to define, making it difficult to provide a clear specification of what we should be looking for. In an influential paper, Levesque proposed a formal specification for analyzing the correctness of such plans. He motivated a logical characterization within the situation calculus that included binary sensing actions. This characterization argued that from each state considered possible initially, the plan should terminate while satisfying the goal. Increasingly, classical plan structures are being applied to stochastic environments such as robotics applications. This raises the question as to what the specification for correctness should look like, since Levesque's account makes the assumption that actions are deterministic.

In this work, we aim to generalize Levesque's account to handle actions with nondeterministic outcomes, which may also be accorded probabilities. By appealing to an extension of the situation calculus to handle probabilistic nondeterminism, we will show that Levesque's definition, as well as a notion of goal achievability proposed by Lin and Levesque, have limited appeal under stochastic nondeterminism. In essence, they correspond to one correct execution, which is unlikely to be adequate. Rather, we propose to delineate between goal satisfaction and termination leading to a range of correctness criteria. To better study these criteria, and to position the results in a broader context while still allowing for the generality of the situation calculus, we consider an abstract framework to study the correctness of plans with loops, in domains that are possibly unbounded, and/or stochastic, and/or continuous. Within that framework, we then prove numerous relationships between the criteria, including some impossibility results for categorically satisfying goals. Finally, we show that these notions provide a more granular view than those discussed in the literature, such as strong planning and strong cyclic planning.

☆ This article belongs to Special Issue: Epistemic Planning.

☆☆ Preliminary versions of this work appeared in [4,3]. The author is grateful to Hector J. Levesque for discussions, and also thanks the referees for numerous suggestions and helpful remarks. This work is partly supported by a Royal Society University Research Fellowship.

* Correspondence to: School of Informatics, University of Edinburgh, Edinburgh, UK.
E-mail address: vaishak@ed.ac.uk.

https://doi.org/10.1016/j.artint.2022.103696

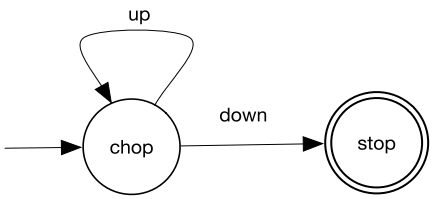


Fig. 1. Controller for the tree chop problem.

1. Introduction

Automated planning is a central concern in high-level symbolic AI research, with applications in logistics, robotics and service composition [32,31]. In the simple case of an agent operating in a known world, the output of a planner is a sequence of actions to be performed. In an incompletely known world, an agent can appeal to its many sensors, and so what is expected is a *conditional plan* that branches on sensing outcomes, which is a tree-like structure. There has been a recent interest in solving a class of problems, possibly with sensing actions, where graph-like plan structures, such as plans with loops, are desirable.

1.1. Motivation

Consider the problem of chopping a tree [66,51]. If we are told that a *chop* action reduces the thickness of the tree by a unit, and that the agent can sense whether the tree is still standing, the plan structure in Fig. 1 is a simple and compact controller that achieves the objective, reacting only to the current observation. Intuitively, the controller is reasonable for a tree of any thickness, because (a) the controller will execute as many actions as needed to bring down the tree, and (b) the controller will stop thereafter. Interestingly, we would now observe that had the task been that of clearing a table, a similar plan structure works too [41]. That is, suppose we are considering the task of picking up an object on a table, putting that away, and repeating that until no more objects remain on the table. Let us also suppose we do not know the number of objects in advance. Then, Fig. 1 but with *chop* replaced by a composite action of *pick* and *put away*, and where we sense *clear* instead of *down* can be seen to solve all instances of the problem.

The natural question to ask is this: in which sense are such plans *correct*? That is, how precisely can we argue that Fig. 1 represents a correct controller, and why is to preferable to, say, a sequential plan with three chop actions? The correctness of the latter plan clearly depends on the problem instance: if we are dealing with a tree that is three units thick, the plan is indeed sufficient, but it is immediately seen to be incorrect if the tree's thickness were, say, four units. So, the correctness of Fig. 1 can be positioned in a setting where: (a) we do not know the thickness of the tree, and (b) our only sensing action tells us whether the tree is still standing. The broader point here is that in the absence of a precise specification, it is difficult to identify what we should be looking for.

In an influential paper, Levesque [52] proposed a formal specification for analyzing the correctness of iterative plans. To analyze such plans, Levesque motivated an epistemic characterization within the logical language of the situation calculus that included binary sensing actions. In that account, the planning task is to find a structure such that for every situation (that is, world state) considered initially possible, it leads to a final situation where the goal holds. In the tree chop problem, for example, mirroring our observation earlier, the goal state is reached when the controller executes as many actions as needed to bring down the tree (*correctness*), after which the controller stops (*termination*). Since the same controller solves a set of possibly infinitely many instances, the area has been recently referred to as *generalized planning* [37].

This concern is related to a number of efforts in AI and computer science. (See the penultimate section for a comprehensive discussion.) For example, as discussed above, the concern is related to automated planning [31], although the classical notion concerns itself with a specified initial state, e.g., either a tree of unit thickness, or one with a thickness of three units, and so on. In the presence of multiple initial states resulting from the incomplete knowledge, we are then interested in a conditional plan where we may branch on sensing outcomes. Generalized planning is motivated in a similar setting, where a solution is sought for multiple classical planning instances which might differ, for example, only in the initial states, corresponding to the incomplete knowledge [52,41]. Put differently, a generalized plan is a conditional plan, but possibly with iteration to account for possibly infinitely many initial states. The concern also relates to notions of computability and program correctness [26,35,65]. However, those notions are not ideal for stipulating the incomplete knowledge of a robot, as the available actions are internal to the machine. That is, we might have actions that read and write a Turing machine tape, or change the values of registers and variables. In contrast, a robot can affect changes to the environment that it operates in. Finally, the concern is most closely related to the notion of *achievability*. Lin and Levesque [53] ask the following question: given a set of actions $\mathcal{A}$, what can we say about what the robot is capable of achieving? They argue that given a set of initial states and a goal formula, if a function (e.g., a program, possibly with loops) can be obtained from $\mathcal{A}$ that on execution makes the goal true, then the goal can be said to be *achievable*. (Achievability is also a precursor to the notion of *knowing how*: if we are to stipulate that an agent has provably sufficient knowledge to execute a plan, then the plan must in the very least be achievable; see Lin and Levesque [53] for discussions.)

While Levesque's characterization does not immediately yield a practical algorithm, the specification serves as a general skeleton to explore the synthesis of program-like plans [25,51,17,70,38,41]. Moreover, recent advances in epistemic planning [47,59] are encouraging, and a formal characterization like the one by Levesque can help in relating algorithmic machinery from the two communities.

1.2. The case for nondeterminism

Overall, the attractiveness, then, of iterative/loopy plan structures like the one in Fig. 1 is as follows: (a) they allow action repetition, as needed when there is uncertainty about the initial state, and (b) they behave like conditional plans for a large (possibly infinite) number of problem instances. Moreover, owing to their compact, reactive nature, they are also ideal for systems with limited resources (*e.g.* mobile robots [57]). With mobile robots, there is indeed uncertainty about the initial state, but owing to noisy effectors, there may also be uncertainty about how noise affects the execution of actions. As a consequence, this new flavor of uncertainty would affect our understanding about both correctness (since an intended action can fail) and termination (since an action can fail infinitely often). This raises the question as to what the specification for loopy plans should look like in general, since Levesque's account makes the assumption that actions are deterministic, or noise-free: the intended action always happens in the real world.[1]

As a side remark, it is worth noting that when an action has a sensing outcome (as admitted by Levesque's account), one could view the action being executed as enabling a nondeterministic transition at the belief level [67]. For example, assume an action opens the door to reveal whether there is treasure behind the door. Before the action, the agent does not know what is behind the door. On executing the action, that belief state will either change to one that knows that there is treasure, or change to one that knows that there is no treasure. One way or another, the next (belief) state is one where there is complete knowledge of what is behind the door, and with regards to the action, which outcome took place. Such a scenario is distinct from noisy actions in general, where the intended action can fail, and in fact, fail repeatedly. Moreover, there may not be a way of knowing in the next state whether the action actually succeeded or failed, unless there is an appropriate sensing action that reveals that information. For these reasons, a proper treatment of nondeterministic actions that affect the physical world (and not just beliefs) is valuable. We return to discuss this matter in more detail towards the end of the article.

To the best of our knowledge, no attempt has been made to address noise at the level of generality of the original paper, and this is precisely our aim here. In particular, in this work, we aim to generalize Levesque's account to handle actions with nondeterministic outcomes, which may also be accorded probabilities. By appealing to an extension of the epistemic situation calculus to handle probabilistic nondeterminism, we will show that Levesque's definition, as well as the notion of goal achievability proposed by Lin and Levesque, have limited appeal under stochastic nondeterminism. In essence, they correspond to one correct execution, which is unlikely to be adequate. Rather, we propose to delineate between goal satisfaction and termination leading to a range of correctness criteria. To better study these criteria, and to position the results in a broader context while still allowing for the generality of the situation calculus, we consider an abstract framework to study the correctness of plans with loops, in domains that are possibly unbounded, and/or stochastic, and/or continuous. Within that framework, we then prove numerous relationships between the criteria, including some impossibility results for categorically satisfying goals.

There are a number of benefits to consider such a generalization. Although the limitation of Levesque's notion for noisy actions is hinted at in many prior accounts, there is yet to emerge an analysis of the sort we discuss. For example, Cimatti et al. [25] argued that there are conceptual difficulties in understanding the correctness of plans when actions have nondeterministic outcomes. They defined the notions of *weak, strong* and *strong cyclic* solutions formulated in terms of action histories that reach the goal state. An informal probabilistic interpretation for these notions was also suggested: weak plans, for example, reach the goal with a non-zero probability. However, while nondeterminism is addressed in their work, Cimatti et al. [25] assume a finite state system. These notions are almost universally adopted in the planning community today [40,15,71,20], but there has been no study on the sensitivity of such correctness notions to the structural assumptions of the underlying plan framework.

More concretely, strong cyclic planning has become the "de-facto solution concept" for so-called fully observable nondeterministic (FOND) problems, as they admit "re-try until success" [28]. FOND problems are extensions to classical planning problems in allowing nondeterministic effects for actions, but once executed, the agent knows of the outcome. But implicit in the definition of strong cyclic planning is the notion of *fairness*: when an action is executed infinitely many times, all of its possible effects occur infinitely often too. In other words, if an intended action can fail with some probability, in an infinite execution, it must succeed infinitely often. So how should we modify that definition if we are dealing with a domain where the effect of noisy actions are not observable in the next state? What if fairness cannot be assumed? Otherwise, what if fairness is assumed but there are uncountably many possible effects for some actions (e.g., an intended action can fail in uncountably many ways), in which case fairness may not matter?

Consider, for example, that nondeterministic outcomes are most prevalent in robotic applications, where the noise of effectors is often characterized by continuous probability distributions [73], that is, there are uncountably many outcomes.

[1] In this work, we refer to actions with nondeterministic outcomes equivalently as nondeterministic actions, and sometimes also as noisy actions.

Consider that in many real-world applications, fluents typically take values from infinite or uncountable sets (*e.g.* resources, time). In these scenarios, how are we to define categorical notions of termination and goal satisfaction? Under which conditions (if any) are categorical claims unachievable, thereby warranting a compromise in terms of probabilistic notions? Our investigations will shed some light on these issues. To the best of our knowledge, the relationships between various kinds of termination and goal-based correctness criteria in unbounded domains, and their probabilistic and continuous versions, have not been discussed in a formal and general way elsewhere. In sum, we make the following contributions:

1. We provide a generalization to Levesque's notion of correctness, so as to formalize an integrated view of goal satisfaction and termination with nondeterministic actions.
2. We argue that this motivates a delineation of goal satisfaction and termination. Termination can be further restricted to account for plans of bounded length. To study the properties and relationships of these notions, we consider proving results in an abstract framework for broader applicability.
3. We then consider probabilistic variants of these notions, including in the presence of continuous noise, and identify that categorical goals are not realizable in some stochastic domains.
4. We then relate our results to the notion of goal achievability [53], as well as weak, strong and strong cyclic planning [25], and show that our correctness notions are more granular.

At the outset, we remark that we are only concerned with formal characterizations here: at no point will we concern ourselves with algorithmic ideas or plan heuristics. Moreover, as will become clear, we will also not recommend any single correctness criteria, but instead discuss the relationships between various criteria, along with sample plans that meet such specifications in a manner that is independent of plan generation. Thus, the thrust is in showing: (a) what the generalization of Levesque's correctness notion for stochastic actions looks like, and (b) study the resulting idea in more detail in a general manner via an abstract framework. In the penultimate section, we report on recent progress on leveraging our results towards the synthesis of plans with loops with respect to the identified correctness criteria.

We are structured as follows. We begin with Levesque's definition first, while also introducing the epistemic situation calculus with noisy actions. After reporting on the limitations of Levesque's definition in the noisy setting, we consider other correctness criteria. This then motivates the abstract framework, where properties of the different criteria are studied in further detail. We then report on the connection between our notions and Lin and Levesque's proposal for goal achievability, as well as the notions of weak, strong and strong cyclic solutions. We then discuss related work and conclude. Throughout this paper, we extend Levesque's account to handle noisy actions, but assume the sensing to be exact. As we show, this is still fertile ground to explore correctness notions. Noisy sensing and how it affects beliefs is clearly a very important direction for our line of inquiry, and an analysis on how it augments our results is left for the future.

2. Preliminaries

Levesque's proposal [52] was formulated in the rich, first-order language of the situation calculus [64]. One could consider such a language for the axiomatization of domains, from which an agent can reason about actions and plans, based on what it knows. We will motivate Levesque's characterization of the correctness of plans with loops. However, while Levesque's proposal allows for knowledge, acting and sensing [68], it assumes actions are deterministic. Thus, for our objectives in this work, we will also motivate an extension of the epistemic situation calculus for noisy acting and reasoning about degrees of belief [1,10]. This extension has been recently shown to capture involved robotics applications [73] such as localization [9]. (The extension also allows for noisy sensing, but as discussed above, that direction is left for the future.)

Roughly, the idea behind noisy acting is this: if the agent intends to move towards a wall, a noisy action (say) by 2 units would allow the agent to believe that it is now closer to the wall, but not necessarily be certain about the exact distance moved: it can be by 2.1 units or even 1.83 units, for example. A distribution would capture the likelihood of possible values. The agent may then consider planning for the most likely future states, based on the noisy outcomes in the current state as well as successor states.

We note that a fragment of the planning literature assumes an angelic interpretation to the outcomes of noisy actions [56]. That is, the agent explicitly or an environment implicitly favors the goal getting achieved, and so encourages the intended action to succeed. This can be contrasted with a demonic interpretation, where the worst outcome for goal satisfiability can be assumed to happen. Our framework does not resort to either interpretation: we assume only that the probability distribution on the outcomes are known, and so when actions are taken, the likelihood of an outcome is determined by that distribution [13].

2.1. A theory of knowledge and action

We will not go over the language $\mathcal{L}$ of the situation calculus in detail [58,64], but simply note that it is a many-sorted dialect of predicate calculus, with sorts for *actions*, *situations*, denoting a (possibly) empty sequence of actions, and *objects*, for everything else. A special constant S_0 denotes the real world initially, and the term $do(a, s)$ denotes the situation obtained on doing a in s. Initial situations are defined as those without a predecessor:

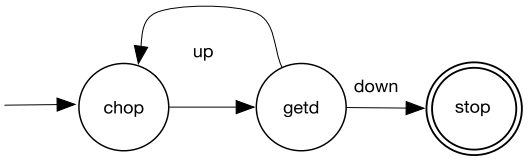


Fig. 2. Controller for chopping a tree of unknown non-zero thickness d, by means of a binary sensing action *getd*.

$$Init(s) \doteq \neg\exists a, s'.\ s = do(a, s').$$

Fluents, whose last argument is always a situation, can be used to capture changing properties. Following Reiter [64], application domains are axiomatized as *basic action theories*, which stipulate the conditions under which actions are executable, and their affects on fluents, while embodying a monotonic solution to the frame problem. For example, for the tree chop problem, using the functional fluent d to mean the thickness of the tree, we may have[2]:

$$Poss(chop(x), s) \equiv d(s) \geq x.$$

$$\begin{aligned} d(do(a, s)) = u \ \equiv \\ (a = chop(x) \wedge d(s) = u + x) \vee \\ (a \neq chop(x) \wedge d(s) = u). \end{aligned}$$

We let *chop* be an abbreviation for $chop(1)$. These axioms say that *chop* is only possible when the tree's thickness is non-zero, and that the current value of d is u if and only if its previous value was also u and no *chop* action occurred, or its previous value was $u + 1$ and a single *chop* action was executed.

A special binary fluent $K(s', s)$ denotes that s' is a possible world when the agent is at s. As usual, knowledge is defined as truth at accessible worlds:

$$Know(\phi, s) \doteq \forall s'.\ K(s', s) \supset \phi[s'].$$

Given the space of situations, we first insist that only the initial situations are accessible from S_0:

$$K(s, S_0) \supset Init(s).$$

We would also need to establish how the accessibility relation changes as actions occur: that is, if $K(s, S_0)$ should it not be the case that $K(do(a, s), do(a, S_0))$ too? Such matters, including the role of sensing actions, are handled using a successor state axiom for K, discussed below.

A modeler also axiomatizes the initial beliefs of the agent[3]:

$$K(s, S_0) \supset (1 \leq d(s) \leq 10). \tag{1}$$

That is, all initial situations where the d-value is between 1 and 10 are accessible from the real world. In other words, as far as the agent is concerned, initially d could be any value between 1 and 10. This is equivalently written using *Know* and a special term *now* to denote the current situation as:

$$Know(d(now) = 1 \vee \ldots \vee d(now) = 10, S_0).$$

The observations obtained by the agent are described using a special function *SF*. While Fig. 1 enabled sensing as part of the chop action, for the situation calculus, it is helpful to consider a separate sensing action for the tree chop problem. So let Fig. 2 now denote the controller for the tree chop problem, for which we may have:

$$SF(a, s) = \begin{cases} down & a = getd \wedge d(s) = 0 \\ up & a = getd \wedge d(s) \neq 0 \\ 1 & \text{otherwise} \end{cases} \tag{2}$$

which says that on doing the sensing action *getd*, if the tree is still standing, the sensor returns *up*, else it would return *down*. Physical actions such as *chop* do not provide observations, and so they will return trivial values for *SF*, say, 1.

[2] Free variables are assumed to be implicitly quantified from the outside.

[3] Like in modal logic [29], constraints on K correspond to appropriate properties for *Know* in the truth theory [68]. We assume, as is usual, that *Know* has the full power of introspection and closed under logical reasoning – so-called **S5** – by consequence of a stipulation that K is an equivalence relation. Also note that, unlike standard modal logic, where "worlds" are static states of affairs that provide truth values to propositions, the model theory of the situation calculus instantiates *trees*: each world describes the values of fluents initially, but also after any sequence of actions.

A domain-independent successor state axiom for K then determines how the agent's knowledge changes over actions:

$$K(s', do(a, s)) \equiv \exists s''[K(s'', s) \wedge s' = do(a, s'') \wedge Poss(a, s'') \\ \wedge (SF(a, s'') = SF(a, s)))]$$

which has the effect of eliminating worlds that disagree with the sensing outcome at the real world, leading to *knowledge expansion.*

Given a basic action theory $\mathcal{D}$, which is assumed to include the foundational axioms and the above domain-independent axiom for K, to reason whether a property ϕ is true after doing an action a, we would check if $\mathcal{D} \models \phi(do(a, S_0))$. To reason whether the agent *knows* that this property is true, we could check if $\mathcal{D} \models Know(\phi, do(a, S_0))$. As explained by the abbreviation above, this is equivalent to checking $\mathcal{D} \models \forall s.\ K(s, S_0) \supset \phi(do(a, s))$.

2.2. Plan correctness

To reason about plans, we will need to consider a plan representation of suitable generality. Broadly, we would like to allow computable functions, such as programs or plans with loops. In Levesque's original formulation [52], he argued for a simple type of program syntax without any internal state but with while loops, referred to as *robot programs*; Lin and Levesque [53] later showed that robot programs are sufficient for considering effective goal achievability. (We will return to this notion towards the end of the article, and define it formally.) In Hu and Levesque [39], it is further shown that loopy plans can be considered instead that are strictly more general than robot programs: while every robot program has an equivalent loopy plan, there are loopy plans that cannot be expressed as robot programs. Thus, we follow the semantics from Hu and Levesque [39,40] focusing on such loopy plans.

Consider that since such plans represent only finitely many parameterless actions and observations, it is helpful to imagine a possibly infinite set of actions $\mathcal{A} = \{\alpha_1, \alpha_2, \ldots\}$ that are parameterless, and a special symbol *stop* not in $\mathcal{A}$ to denote termination. We also imagine a possibly infinite set of observations $\mathcal{O} = \{\beta_1, \beta_2, \ldots\}$ that are parameterless. We then define:

Definition 1. Suppose $acts \subseteq \mathcal{A}$ is a finite set of actions, and $obs \subseteq \mathcal{O}$ is a finite set of observations. A *finite reactive plan* $\mathcal{X}$ is a tuple $\langle \mathcal{Q}, q_0, q_F, \gamma, \delta \rangle$:

- $\mathcal{Q}$ is a finite set of control states;
- $q_0 \in \mathcal{Q}$ is the initial control state, and $q_F \in \mathcal{Q}$ is the final control state;
- $\gamma \in [\mathcal{Q} \rightarrow (acts \cup \{stop\})]$ is a labeling function, such that for any $q \in \mathcal{Q}$, $\gamma(q) = stop$ iff $q = q_F$; and
- $\delta \in [(\mathcal{Q} - \{q_F\}) \times obs \rightarrow \mathcal{Q}]$ is a transition function.

Such controllers are prominent in the literature on generalized planning [17,40,38], and are essentially Moore machines [36].

To reason about these plans in an environment enabled in the situation calculus, we need to encode the plan structure as an $\mathcal{L}$-sentence, and axiomatize the execution semantics over situations. Thus, the control states, along with the labeling and transition functions will be distinguished constants and functions respectively in $\mathcal{L}$. Following Hu and Levesque [40], we define:

Definition 2. Let Σ be the union of the following axioms for:

1. domain closure for the control states: $(\forall q)\ \{q = q_0 \vee q = Q_1 \vee \ldots \vee q = Q_n \vee q = q_F\}$;
2. unique names axiom for the control states: $Q_i \neq Q_j$ for $i \neq j$;
3. action association: $\forall Q \in \mathcal{Q}$ of the form $\gamma(Q) = a$; and
4. transitions: $\forall Q \in (\mathcal{Q} - \{q_F\})$ of the form $\delta(Q, o) = Q'$.

Example 3. For Fig. 2, we might have $\mathcal{Q} = \{q_0, Q, q_F\}$. Then Σ would be the union of: $\gamma(q_0) = chop$, $\gamma(q_F) = stop$, $\gamma(Q) = getd$, $\delta(Q, up) = q_0$, $\delta(Q, down) = q_F$, $\delta(q_0, 1) = Q$, $q_0 \neq Q$, $Q \neq q_F$, $q_0 \neq q_F$, and $\forall q\ (q = q_0 \vee q = Q \vee q = q_F)$.

Definition 4. We use $T^*(q, s, q', s')$ as abbreviation for $\forall T[\ldots \supset T(q, s, q', s')]$, where the ellipsis is the conjunction of the universal closure of:

- $T(q, s, q, s)$;
- $T(q, s, q'', s'') \wedge T(q'', s'', q', s') \supset T(q, s, q', s')$; and
- $\gamma(q) = a \wedge Poss(a, s) \wedge SF(a, s) = o \wedge \delta(q, o) = q' \supset T(q, s, q', do(a, s))$.

In English: T^* is the reflexive transitive closure of the one-step transitions in the plan.[4] We are now prepared to reason about plan correctness as formulated by Levesque [52], which we denote by **DET** to say actions are deterministic, i.e., noise-free.

Definition 5. For any goal formula $\phi \in \mathcal{L}$, basic action theory $\mathcal{D}$, plan $\mathcal{X}$ and its encoding Σ, we say $\mathcal{X}$ is correct for ϕ iff

$$\mathcal{D} \cup \Sigma \models \forall s.\, K(s, S_0) \supset \exists s'[T^*(q_0, s, q_F, s') \wedge \phi(s')]. \qquad (\textbf{DET})$$

Basically, this definition says that for every situation considered accessible initially, executing $\mathcal{X}$ at the initial situation leads to a final situation where the final control state is reached (and so, termination is achieved) but also that this is a situation where the goal formula ϕ is true.

Example 6. Let $\mathcal{D}_{dyn}$ denote the tree chop axioms, and then it is easy to see that $\mathcal{D}_{dyn} \cup \{(1), (2)\} \cup \Sigma$, where Σ represents the encoding of Fig. 2, is correct for the goal $d = 0$.

2.3. A theory of probabilistic beliefs

Our objective now is to be able to generalize the above notion to a stochastic setting. The account of knowledge, deterministic acting and exact sensing was extended by Bacchus, Halpern, and Levesque [1] – BHL henceforth – to deal with degrees of belief in formulas, and in particular, with how degrees of belief should evolve in the presence of noisy sensing and acting, in accordance with Bayesian conditioning. (As discussed before, sensing is assumed to be exact in what follows.) The main advantage of a logical account like BHL is that it allows a specification of belief that can be partial or incomplete, in keeping with whatever information is available about the application domain. The account is based on 3 distinguished fluents: p, l and alt. (The presentation in the sequel is based on the account by Belle and Levesque [10].) The p fluent here is a numeric analogue to K in that $p(s', s)$ denotes the weight (or density) accorded to s' when the agent is at s. Initial constraints about what is known can be provided as usual. For starters, as we would for K, we need to insist that only initial situations are accessible from S_0:

$$p(s, S_0) > 0 \supset Init(s).$$

With that, we can define what initial knowledge looks like; for example:

$$p(s, S_0) = \begin{cases} .1 & \text{if } (1 \leq d(s) \leq 10) \\ 0 & \text{otherwise} \end{cases} \qquad (3)$$

is saying that the tree's thickness d takes a value that is uniformly drawn from $\{1, \ldots, 10\}$. We provide a definition for a belief modality Bel shortly, but the above constraint can be equivalently written as:

$$Bel(d(now) = 1, S_0) = .1$$

and so on for the other values. As argued earlier, the logical account also allows for uncertainty about the prior distribution [43], by means of expressions such as:

$$Bel(d(now) = 1, S_0) \geq .05$$

which says that any distribution where d takes a value of 1 with a probability greater than or equal to .05 is a permissible one.

The l and alt fluents are used for modeling noise. In particular, to handle noisy actions, the idea is that if $chop(x)$ represents an action that decrements the tree's thickness by x units, assume a new action type $chop(x, y)$ in that x is the intended argument and y is taken to be what actually happens. The latter argument is chosen by nature, and as such, out of the agent's control. These action types are retrofitted in successor state axioms:

$$\begin{aligned} d(do(a, s)) = u \equiv (a = chop(x, y) \wedge u = d(s) - y) \vee \\ (a \neq chop(x, y) \wedge u = d(s)) \end{aligned} \qquad (4)$$

says that d is actually affected by the second argument.

[4] We note that this is a second-order logical formulation. The situation calculus allows some second-order logical features [64], including the second-order induction on situations as part of the domain-independent axioms.

Since the agent is assumed to not control y, we use *alt* to model the possible *alternatives* to an intended action[5]:

$$alt(chop(x, y), a', z) \equiv a' = chop(x, z) \tag{5}$$

says that, for example, $chop(1, 2)$ is indistinguishable from $chop(1, 3)$. To suggest that some alternatives are more likely than others, an axiom like

$$l(chop(x, y), s) = \mathcal{N}(y; x, .25) \tag{6}$$

then says that the actual value is normally distributed around the intended value, with a variance of .25. In robotics terminology [73], this is the equivalent to an action having additive Gaussian noise. (For noise-free actions, these specifications are trivial, with every action being an alternative only to itself, and the likelihoods being 1.)

Note that because the l-axiom is situation dependent, it can be *context-dependent* in the sense of involving situation-dependent properties as conditions for the probability distribution. For example, depending on the humidity, we can specify that variance be adjusted, or even that the type of distribution changes. This is a powerful feature of the language; further examples can be found in [1,8,7].

Likelihoods and *alt*-axioms determine the probability of successors, enabled by the following successor state axiom:

$$\begin{aligned} & p(s', do(a, s)) = u \equiv \\ & \quad \exists a', z, s'' \, [alt(a, a', z) \wedge s' = do(a', s'') \wedge Poss(a', s'') \wedge SF(a, s) = SF(a', s'') \wedge \\ & \qquad\qquad u = p(s'', s) \times l(a', s'')] \\ & \vee \neg \exists a', z, s'' \\ & \quad [alt(a, a', z) \wedge s' = do(a', s'') \wedge Poss(a', s'') \wedge SF(a, s) = SF(a', s'') \wedge u = 0] \end{aligned}$$

which essentially says that if two situations s and s' are considered epistemically possible, on doing a at s, $do(a, s)$ and $do(b, s')$ will also be considered epistemically possible, where b is any *alt*-related action to a. Moreover, the p-value of $do(b, s')$ is that of s' multiplied by the likelihood of the outcome b. Note that for the sake of handling exact sensing, we re-introduce the condition that the worlds have to agree on sensing outcomes, which is not present in the original formulation [10].

Putting all this together, the degree of belief in ϕ at s is defined as the weight of worlds where ϕ is true:

$$Bel(\phi, s) \doteq \sum_{\{s' \,:\, \phi(s')\}} p(s', s) \Big/ \sum_{s'} p(s', s)$$

We write $K(s', s)$ to mean $p(s', s) > 0$, and $Know(\phi, s) \doteq Bel(\phi, s) = 1$. We finally note that although this expression involves a summation and so is limited to discrete distributions, a corresponding abbreviation can also be specified for *Bel* in continuous and mixed discrete-continuous domains; see Belle and Levesque [10] for details.

3. Correctness with noisy acting

Our objective now will be to motivate a definition for analyzing the correctness of plan structures when actions are noisy (that is, they are nondeterministic, and the actual outcome is not directly observable). We assume that sensing is exact.[6] This then also handles the case where actions are nondeterministic but observable immediately after, by way of a sensing action to inform the planner about the outcome. Like in Levesque's setting, moreover, there is still uncertainty about what is true initially, and a solution is sought that is correct for all initial situations. We note that although the limitation of Levesque's notion for noisy actions is hinted at in many prior accounts [25], our contributions are different in the following regards. We wish to formally generalize the T^* semantics, and from there, we will tease out termination and goal satisfaction properties as separate entities, and study them.

[5] A more involved version would introduce *alt* as a fluent, allowing possible outcomes to be determined by a situation.

[6] Recall that sensor readings are either binary (in indicating, say, whether the light is on) or real-valued (e.g., the temperature for this hour). Exact sensing means that the value returned for the corresponding fluent (light-status or temperature) is precisely the value for that fluent in the situation where the action was performed. This was perhaps best illustrated in Equation (2): if *getd* is performed in the situation s, depending on whether the d-value is zero or not, we observe *down* or *up* respectively. The idea is that when the action is performed in the real world, the observation obtained by the agent reflects reality, and consequently, those worlds that disagree with that observation are discarded. This is captured in the successor state axiom for K.

With noisy sensing, even if the d-value at s was 0, it is possible that the reading on the sensor might say *up*. By repeatedly testing the sensor, we would have a probabilistic error model. Say, with a probability of 0.8, the d-value being 0 means *down* is read, but with a probability of 0.2, *up* is obtained. Likewise, if the d-value is not 0, say, there is a 0.9 probability of reading *up*, but with a probability of 0.1, *down* is obtained. Consequently, when the action is performed in the real world and if *up* were obtained, because of the noise, those worlds that disagree with that observation are *not discarded* but their weights *appropriately adjusted* using the specified probabilistic error model [1,5]. In sum, noisy sensing becomes relevant with an explicit model of belief, as it is beliefs that capture the updated weights on the worlds.

As discussed in Section 1, noisy acting but exact sensing is still fertile ground to explore correctness notions, as in the case of [25], among others. Modeling beliefs explicitly is clearly a very important direction for our line of inquiry, especially in autonomous physical robotics applications where noisy sensing is commonplace [73,8,9], and an analysis on how it augments our results is left for the future (cf. also discussions in the penultimate section).

As far as the syntax of the plan structure goes, we will not want it to be any different than Definition 1; however, we will need to revisit Definition 4 to internalize the noisy aspects of acting by using *alt*.

Definition 7. We use $U^*(q, s, q', s')$ as abbreviation for $\forall U[\ldots \supset U(q, s, q', s')]$, where the ellipsis is the conjunction of the universal closure of:

- $U(q, s, q, s)$;
- $U(q, s, q'', s'') \wedge U(q'', s'', q', s') \supset U(q, s, q', s')$; and
- $\gamma(q) = a \wedge \exists b, z\ (alt(a, b, z) \wedge Poss(b, s) \wedge SF(b, s) = o \wedge \delta(q, o) = q') \supset U(q, s, q', do(b, s))$.

The main new ingredient here over T^*, of course, is how control states transition from a situation to a successor. The reflexive transitive closure of U basically says that if the controller advises a and b is any action that is *alt*-related to a, we consider the transition wrt the executability and the sensing outcome of the action b. The idea, then, is allow U^* to capture the least set that accounts for all the successors of a situation where a noisy action is performed. We follow the previous notion of correctness to define:

Definition 8. For any goal formula $\phi \in \mathcal{L}$, basic action theory $\mathcal{D}$, plan $\mathcal{X}$ and its encoding Σ, we say $\mathcal{X}$ is correct for ϕ iff

$$\mathcal{D} \cup \Sigma \models \forall s.\ K(s, S_0) \supset \exists s'[U^*(q_0, s, q_F, s') \wedge \phi(s')]. \qquad (\textbf{ONE})$$

Like Levesque's definition, this says that for every initial world, by executing the plan we reach a successor situation that terminates (i.e., control state q_F reached) and where the goal ϕ is true. (Here, **ONE** refers to the existence of at least one such successor situation, elaborated below.)

With this definition in hand, the first natural question to ask ourselves is this: how do we know we have generalized T^* reasonably? It is fairly easy to show a simple downward compatibility result, in that the new semantics coincides with Levesque's account when the action theory is *noise-free*: that is, *alt*-axioms are trivial $\forall z(alt(a, a', z) \equiv a = a')$ and l assigns 1 to every action. Then:

Theorem 9. *Suppose $\mathcal{D}$ is a noise-free action theory, $\mathcal{X}$ and Σ are as above, and ϕ is any situation-suppressed formula. Then, $\mathcal{X}$ is correct for ϕ in the sense of* **DET** *iff $\mathcal{X}$ is correct in the sense of* **ONE**.

Proof. U^* differs from T^* is only one aspect, that of *alt*-related actions governing the transition to a successor situation. By assumption, $\forall z(alt(a, a', z) \equiv a = a')$; so Definition 5's constraint on T^* coincides with Definition 8's constraint on U^*. □

Beyond this sanity check, however, we immediately notice an interesting difference between what these definitions offer in terms of termination and goal satisfiability when applied in their respective domains. We state the following observations informally below, and without proof. They will be proved formally in a more general manner in the subsequent sections. For example:

- Suppose $\mathcal{D}$ is noise-free. Then starting from S_0, if there is a situation s such that $T^*(q_0, S_0, q_F, s)$ holds and $\phi(s)$, then there cannot be $s' \neq s$ such that $T^*(q_0, S_0, q_F, s')$.
  That is, an action sequence that is terminating (situations where the control state q_F is reached) is unique, and if that sequence also enables the goal, then the plan is deemed correct.
- Suppose $\mathcal{D}$ is not noise-free. Starting from S_0, suppose there is a situation s such where $U^*(q_0, S_0, q_F, s)$ holds and $\phi(s)$. Then it does not follow that this is only terminating action sequence: there may be $s' \neq s$ such that $U^*(q_0, S_0, q_F, s')$ and perhaps even $\neg\phi(s')$.
  That is, the existence of one terminating sequence where the goal is true does not negate the existence of other terminating action sequences, which may or may not enable the goal.

This is perhaps not immediately surprising: nondeterministic actions by their very nature mean that an action other than what was intended can occur, and these may nonetheless lead to successor situations where plans terminate but goals do not hold. By strictly following Levesque's definition, we are only insisting that there is one terminating sequence enabling the goal. (A similar observation can be made about the goal achievability notion in Lin and Levesque [53], which we will introduce and relate to in a subsequent section.) The problem persists even when we have complete knowledge about the initial state, that is, when $K(s, S_0)$ only for $s = S_0$.

The problem here is not U^* per se but rather the condition imposed on the successor situations obtained as result of executing the plan. Clearly, in the noise-free case, **DET** gives an integrated view of termination and goal satisfiability because terminating sequences are unique. But for the noisy setting, it will be useful to delineate these notions, discussed below. Thus, we refer to the notion of correctness from Definition 8 by **ONE** to say that *there is at least one terminating sequence that enables the goal.* (We will later show that **ONE** coincides with the well-known notion of *weak planning* [25].)

Let us consider the condition that every plan execution leads to a terminating one, referred to as **TER**.

Definition 10. Suppose $\mathcal{D}, \mathcal{X}$ and Σ are as in Definition 8. Then we refer to the following the property holding as *termination*:

$$\mathcal{D} \cup \Sigma \models \forall s.\ K(s, S_0) \supset \forall s'\ [U^*(q_0, s, q, s') \supset \exists s''\ (U^*(q, s', q_F, s''))]. \tag{\textbf{TER}}$$

Analogously, we could now consider a condition on goal satisfiability: that is, every terminating sequence is one that enables the goal, referred to as **PC** for partial correctness.

Definition 11. Suppose $\mathcal{D}, \mathcal{X}, \Sigma$ and ϕ are as in Definition 8. Then we refer to the following the property holding as *partial correctness*:

$$\mathcal{D} \cup \Sigma \models \forall s.\ K(s, S_0) \supset \forall s'\ [U^*(q_0, s, q_F, s') \supset \phi(s')]. \tag{\textbf{PC}}$$

This says that starting from every initial situation, every terminating plan is also a goal satisfying one.

Example 12. Imagine a tree chop problem with noise-free sensing (i.e., let *SF* work as in (2)), but with noisy actions. Let $chop \in \mathcal{A}$ correspond to the $\mathcal{L}$-action $chop(1, 1)$, with

$$l(chop(x, y), s) = \begin{cases} .9 & \text{if } x = y \\ .1 & \text{if } y = 0 \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

That is, the chop action does nothing with a small probability. Letting the initial theory be a p-based axiom for (1), we see that the plan from Fig. 2 is correct in the sense of **ONE**. We notice that **TER** holds too, because regardless of how many times the action fails, there is clearly one action sequence that terminates, where the appropriate number of chops succeeds and then stops. However, because this sequence has chop actions as long as *up* is obtained, and stops as soon as *down* is obtained, it is also the one enabling the goal of bringing down the tree. Therefore, **PC** holds too, and consequently, **ONE**, **TER** and **PC** hold together. Interestingly, the action can fail an unbounded number of times, so terminating plans are not bounded in length. (This will then motivate a correctness criteria on its own, as we shall see in the next section.)

Of course, it will be easy to consider problems where **ONE**, **TER** and **PC** do not hold together. Indeed, our earlier observations can be restated as follows:

- In noise-free problems, **ONE** is equivalent to {**PC**, **TER**}.
- In noisy problems, **ONE** neither implies **PC**, nor **TER**. We could also argue that even if both **ONE** and **TER** hold, **PC** need not. This is because the existence of one terminating sequence that is goal satisfying need not mean that the other terminating sequences also enable the goal.

So, Levesque's proposal for an integrated view of an execution that terminates and satisfies the goal is not appealing in noisy domains. Moreover, there seem to be numerous properties that need to be further explored. For example, what do **ONE** and **TER** imply in terms of whether terminating sequences are of finite length? Should we consider probabilistic notions of **TER** and **PC**? How are those notions further generalized to continuous domains?

Clearly, it is useful to study and enumerate such properties in more detail, but these results have applicability far beyond the situation calculus. To better study them, and to position the results in a broader context while still allowing for the generality of the situation calculus, we consider an abstract framework to study the correctness of plans with loops, in domains that are possibly unbounded, and/or stochastic, and/or continuous. Within that framework, we then prove numerous relationships between the conditions, including some impossibility results for categorically satisfying goals.

Before concluding this section, let us briefly consider a nuanced view on correctness wrt the worlds considered accessible initially. As in Levesque's proposal, we have formalized correctness wrt all situations that are K-related to S_0. In the context of the p fluent, in particular, we viewed K as capturing the "support" of p: $K(s', s)$ stood for $p(s', s) > 0$, thereby capturing all initial worlds that have non-zero weights relative to S_0. But weaker specifications are possible still.

Definition 13. Suppose $\mathcal{D}, \mathcal{X}, \Sigma$ and ϕ are as in Definition 8. Then we define the properties $\mathbf{GEQ}_\kappa$ and $\mathbf{BEL}_\kappa$ for some $\kappa \in [0, 1]$ as follows:

$$\mathcal{D} \cup \Sigma \models \forall s.\ p(s, S_0) \geq \kappa \supset \exists s'[U^*(q_0, s, q_F, s') \wedge \phi(s')] \tag{$\mathbf{GEQ}_\kappa$}$$

$$\mathcal{D} \cup \Sigma \models Bel(\exists s'[U^*(q_0, now, q_F, s') \wedge \phi(s')], S_0) \geq \kappa \tag{$\mathbf{BEL}_\kappa$}$$

Here, $\mathbf{GEQ}_\kappa$ looks at worlds with weights greater than or equal to κ, and $\mathbf{BEL}_\kappa$ says that the sum (or integral) of initial worlds where there is a terminating and goal enabling plan is greater than or equal to κ. (Here, we are assuming that the p-values of situations is within [0,1], otherwise it would have to be normalized for the above definition.) Since $K(s', s)$ is an abbreviation for $p(s', s) > 0$, and $Know(\phi, s) \doteq Bel(\phi, s) = 1$, we would immediately obtain:

Proposition 14. ONE *is equivalent to* $\mathbf{GEQ}_\kappa$ *for* $\kappa = 0$*, and it is equivalent to* $\mathbf{BEL}_\kappa$ *for* $\kappa = 1$.

Example 15. Imagine a tree chop problem with two types of trees, wooden ones and metal ones; the chop action has no effect on a metal tree [67]. Suppose we have 3 worlds: the first with a wooden tree of thickness 1 and weight .4, the second with a wooden tree of thickness 2 and weight .4, and the third with a metal tree of arbitrary non-zero thickness with weight .2. If the goal is $d = 0$, and the plan is the one from Fig. 2, then $\mathbf{GEQ}_\kappa$ holds for (say) $\kappa = .3$ and $\mathbf{BEL}_\kappa$ holds for (say) $\kappa = .7$.

We will not attempt to consider such variants in much detail. Our controller framework can be extended to handle such variants too, as we shall later see. More broadly, we do not think that there is one preferred definition for correctness. While **ONE** attempts correctness in the sense of Levesque [52], planning for likely states, as in $\mathbf{GEQ}_\kappa$, is very common in robotics when exploring large biased search spaces [69,45].

4. A controller framework

We are interested in the correctness of program-like plans that work in multiple, possibly infinitely many, domain instances. We believe these notions of correctness do not depend on the details of how the planning problem is formalized. Building on and extending the formulation from Lin and Levesque [53] (also see Hu and De Giacomo [37]) to now handle probabilistic nondeterminism, we introduce an abstract framework for our purposes. (Here, by *abstract*, we simply mean that the framework is defined in terms of a set of atomic states, actions and observations [12]. That is, we will not worry about how precisely the domain is specified, as we would, for example, in the situation calculus via basic action theories, objects, and relations.)

The overall idea is to develop a *specification* of a (robot) controller operating in a dynamic environment. The controller encapsulates the actions that some putative agent performs, and the environment is the world in which these actions occur. We will not want to assume, however, that the controller has access to one form of information or another. So from the controller's perspective, the only feedback from the environment are observations received after every action.

Formally, imagine a possibly infinite set of actions $\mathcal{A} = \{\alpha_1, \alpha_2, \ldots\}$ that are parameterless, and a special symbol *stop* not in $\mathcal{A}$ to denote termination. We also imagine a possibly infinite set of observations $\mathcal{O} = \{\beta_1, \beta_2, \ldots\}$ that are parameterless. The controller, then, simply responds to accumulated sensor readings by advising an action[7]:

Definition 16. A controller $\mathcal{C}$ is any function from $\mathcal{O}^*$ to $\mathcal{A} \cup \{stop\}$.

After an action, the environment changes one state to another, and responds with an observation:

Definition 17. An *environment* $\mathcal{E}$ is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \Delta, \Omega \rangle$:

- $\mathcal{S}$ is a possibly infinite set of states;
- $\mathcal{A}$ is a possibly infinite set of actions (as above);
- $\mathcal{O}$ is a possibly infinite set of observations (as above);
- $\Delta \in [\mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}^{\geq 0}]$ is a probabilistic transition function: that is, on doing α at s, $\mathcal{E}$ changes to s' with a likelihood of $\Delta(s, \alpha, s')$; and
- $\Omega \in [\mathcal{S} \to \mathcal{O}]$ is a sensor model.

Put together, a *system* encapsulates a controller operating in an environment:

[7] Our framework is very general in not making any assumptions about the state, action or observation spaces. Regarding our definition for a controller, we will also not place any constraints on its specification, allowing it to be any function from observations to actions. (In the interest of computability, we will later only require it to be computable, and choose one such representation for the sake of concreteness.)

However, there is a need for nondeterministic (and in fact, stochastic) controllers that map to a set of (or distribution over, respectively) actions. Specifically, in partially observable Markov decision processes [44], the use of certain optimization techniques for the synthesis of controllers becomes possible when they are stochastic. Since we are only interested in whether action sequences are goal enabling, and how to account for those sequences in the presence of noise, there is nothing to be gained from nondeterministic controllers. If we allowed for nondeterministic controllers purely for the sake of generality, it would needlessly complicate the exposition to follow.

In the penultimate section, we will briefly discuss Markov decision processes in the context of our results. It is conceivable that to draw deeper connections, a stochastic controller might be worth considering to position its notion of correctness against the other sorts discussed here.

Definition 18. A *system* is a pair $(\mathcal{C}, \mathcal{E})$, where $\mathcal{C}$ is a controller, and $\mathcal{E}$ is an environment.

The dynamics of a system is characterized by its *runs*:

Definition 19. A *history* σ of the system $(\mathcal{C}, \mathcal{E})$ is an element of the set $\mathcal{S}^*$. The *final state* of $\sigma = s_0 \cdots s_n$, written $end(\sigma)$, is s_n, and the *sensing outcomes* for σ, written $sensed(\sigma)$, is $\Omega(s_0) \cdots \Omega(s_n)$. For the empty history ϵ, $end(\epsilon) = sensed(\epsilon) = \epsilon$.

A non-empty history σ is a *run* at a state s of a system $(\mathcal{C}, \mathcal{E})$ if, inductively, either σ is s, or $\sigma = \sigma' \cdot s'$ such that σ' is a run of the system at s, $\mathcal{C}(sensed(\sigma')) = \alpha$ and $\Delta(end(\sigma'), \alpha, s') > 0$. The run σ is said to be *terminating* if $\mathcal{C}(sensed(\sigma)) = stop$. The run σ is said to be *extendable* if there is a non-empty history σ' such that $\sigma \cdot \sigma'$ is also a run at s of $(\mathcal{C}, \mathcal{E})$. A run σ is said to be *undefined* if $\mathcal{C}(sensed(\sigma)) \neq stop$ and σ is not extendable. A *proper* system is one without undefined runs.

The picture is this: starting with state s, on reading $\beta_1 = \Omega(s)$, the robot controller chooses to perform $\alpha_1 = \mathcal{C}(\beta_1)$. The environment probabilistically changes to some s' provided $\Delta(s, \alpha_1, s') > 0$ and returns $\beta_2 = \Omega(s')$ to the controller. The controller now advises $\alpha_2 = \mathcal{C}(\beta_1 \cdot \beta_2)$, and so on, until the controller says *stop*. In this work, we implicitly assume that systems are proper: that is, given a system $(\mathcal{C}, \mathcal{E})$ and any state s of $\mathcal{E}$, there are no undefined runs at s.[8] Put differently, a history does not depend on controller, but a run does. That is, a run is a history produced by the controller.

We will leverage the notion of runs for defining correctness wrt planning problems starting in some initial state. Since runs are defined for individual states, it is useful to distinguish an individual planning problem with a distinguished initial state, which we refer to as *basic,* from problems involving a set of initial states. Following Hu and De Giacomo [37], among others [41], we refer to the latter as *generalized* planning problems.

Definition 20. For any $\mathcal{E} = \langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \Delta, \Omega \rangle$, a *basic planning problem* $\mathcal{P}$ is a tuple $\langle s_0, \mathcal{E}, \mathcal{G} \rangle$ where $s_0 \in \mathcal{S}$ is the initial state and $\mathcal{G} \subseteq \mathcal{S}$ are goal states.

Definition 21. A *generalized planning problem* $\overline{\mathcal{P}}$ is a (possibly infinite) set of basic problems $\{\mathcal{P}_1, \mathcal{P}_2, \ldots\}$, where $\mathcal{P}_i$ agree on everything except the initial state.

Intuitively, a basic planning problem captures an instance of a planning problem starting from a distinguished initial state. A generalized planning problem, then, captures a class of basic (planning) problems where the instance differ in terms of initial situations. The idea is that we will look for a plan that solves all of the contained problems. It is not hard to see this is very much in the spirit of Levesque's notion [52], where the basic planning problem instances correspond to the situations that are considered possible initially. One might also view generalized planning as a "meta-level" framework corresponding to the "object-level" specification of incomplete knowledge in Leveque's formalization.

Note that the requirement in that definition that $\mathcal{P}_i$ agree on everything except the initial state essentially amounts to all situations having the same view of the dynamics of the world. Dropping that requirement would lead to a more general problem instance than currently captured in the situation calculus formulations. Recent papers have argued for such flexibility [18,37,41], a point we discuss further in the penultimate section. In that regard, nothing in our technical results hinge on whether we take the more conservative view of generalized planning where the instances only differ in the initial situation. Formally:

Definition 22. An *unconstrained generalized planning problem* $\overline{\mathcal{P}}$ is a (possibly infinite) set of basic problems $\{\mathcal{P}_1, \mathcal{P}_2, \ldots\}$.

In the sequel, every definition and result formulated for generalized planning problems applies to the unconstrained version as well. Of course, when synthesizing plans, even the more flexible definitions of generalized planning in the literature [18,37,41] argue for some common structure, *e.g.* the set of actions and observations being the same across all instances [37], but as far as the formal study in this article is concerned, no such restrictions are needed.

4.1. Types of environments

It will be useful to distinguish between types of environments that vary in the nature of the state space or the transition function, for example. We follow this terminology:

Definition 23. Suppose $\mathcal{E} = \langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \Delta, \Omega \rangle$. Then:

- $\mathcal{E}$ is *fully observable* when $\mathcal{O} = \mathcal{S}$ and Ω is such that for every $s \in \mathcal{S}$, $\Omega(s) = s$.
- $\mathcal{E}$ is *finite* if $\mathcal{S} \times \mathcal{A} \times \mathcal{O}$ is finite; otherwise $\mathcal{E}$ is *infinite.*

[8] Systems with undefined runs will need to abort before the end of computation, a complication we ignore.

- $\mathcal{E}$ is *noise-free* (or *deterministic*) if for every s and α, there is a unique s' such that $\Delta(s, \alpha, s') = 1$ and for $s'' \neq s'$, $\Delta(s, \alpha, s'') = 0$; otherwise $\mathcal{E}$ is *noisy*.
- $\mathcal{E}$ is *discrete* if for every $(s, \alpha) \in \mathcal{S} \times \mathcal{A}$, $\Delta(s, \alpha, \cdot)$ is a probability mass function. If $\Delta(s, \alpha, \cdot)$ is a probability density function, then $\mathcal{E}$ is *continuous*.[9]

For simplicity, we will not treat environments with mixed discrete-continuous transition models.[10]

The terminology is extended for a basic planning problem $\mathcal{P} = \langle s_0, \mathcal{E}, \mathcal{G} \rangle$ in terms of its environment $\mathcal{E}$. For example, if $\mathcal{E}$ is noise-free, then we say that $\mathcal{P}$ is a noise-free basic planning problem, or simply noise-free problem for short. Analogously, we say a generalized planning problem is noise-free iff all of the contained basic problems are noise-free.

It is easy to see that standard planning formulations are encapsulated in such a setup [37,32]. For example,

- A *classical planning problem* is essentially a finite, fully observable basic planning problem. Equivalently, it is a finite, fully observable, generalized planning problem that is a singleton, that is, its of size 1.
- A *conditional planning problem* is essentially a finite generalized planning problem (where the contained basic planning problems differ only in the initial state).
- A *conformant planning problem* is essentially a finite generalized planning problem where we further insist that $\mathcal{O} = \{1\}$, with 1 being a trivial sensing result, and $\Omega(s) = 1$ for all states. That is, the agent is not able to make (non-trivial) observations.
- A so-called *fully observable non-deterministic* (FOND) planning problem [28] is (in our terms) a fully observable, finite, noisy, discrete basic planning problem.

See Hu and De Giacomo [38] for discussions on how this notion of generalized planning also captures a range of other formulations from the planning literature.

4.2. Effective controllers

Our controller formulation above is general, and the notions of correctness investigated in the sequel are applicable at that level of generality. In practise, however, when algorithms are designed [53], we would want these controllers to be computable, which can be achieved by ensuring that $\mathcal{C}$ is recursive.

Definition 24. A controller $\mathcal{C}$ is *effective* if the function $\mathcal{C}$ is recursive.

As discussed previously, one simple candidate for an effective controller are finite plans with loops (Definition 1), such as the one seen in Fig. 1. Our discussions will make use of such reactive plans owing to their simplicity and for illustrating sample plans. (The formal claims made below do not hinge on this particular representation for controllers.)

For any environment $\mathcal{E}$, a finite reactive plan $\mathcal{X}$ can be viewed as a controller $\mathcal{C}$ that behaves as follows for a run $\sigma = s_0 \cdots s_n$ of the system $(\mathcal{C}, \mathcal{E})$:

$$\mathcal{C}(\Omega(s_0) \cdots \Omega(s_n)) = \gamma(q(\Omega(s_0) \cdots \Omega(s_n))))$$

where, $q(\Omega(s_0) \cdots \Omega(s_n)))$ is defined inductively:

$$= \begin{cases} q_0 & \text{if } n = 0 \\ \delta(q(\Omega(s_0) \cdots \Omega(s_{n-1})), \Omega(s_n)) & \text{otherwise} \end{cases}$$

5. From correctness to categorical adequacy

To formally define the reasonableness of controllers for a problem specification, we will need to consider a correctness condition. But as argued previously, unlike in Levesque's proposal, we do not have a single notion of correctness, and precisely because we decoupled goal satisfaction and termination, we would need to consider combinations of conditions. To avoid overloading the notion of correctness, we will instead consider whether a controller is *adequate* wrt criteria such as **PC** and **TER**. That is, if the runs of a system $(\mathcal{C}, \mathcal{E})$ satisfy a property θ (*e.g.* there is a terminating run), we say that $\mathcal{C}$ is a θ-adequate controller.

[9] Recall that a mass function gives the probability that a discrete random variable is exactly equal to some value. A density function is for continuous random variables. The probability of any single point is always zero with the latter, and we would need to integrate over an interval to yield a probability. Thus, note that $\mathcal{S}$ would be uncountable when $\mathcal{E}$ is continuous.

[10] Such an extension would mainly affect the treatment of probabilistic correctness discussed in subsequent sections. Currently, they appeal to sums for the discrete environments, and integrals for continuous environments. If we are to deal with mixed discrete-continuous transition models, we would need to sum when a discrete action is performed, and integrate when a continuous action is performed; see [10], for example.

In this section, we follow the criteria considered in Section 3 along with two extra ones that are all *categorical*: they pertain to whether a run or the set of all runs satisfy a property. And so, the probabilities of the individual outcomes (provided they are non-zero), and by extension, the probabilities of runs do not matter. In the next section, we will take these probabilities into account to specify probabilistic variants of **PC** and **TER**.

Henceforth, for any $\mathcal{C}$, in the context of a basic problem $\mathcal{P} = \langle s_0, \mathcal{E}, \mathcal{G} \rangle$, we will often simply speak of *runs* of the system $(\mathcal{C}, \mathcal{E})$ with the understanding that these runs are at s_0. Also, the length of a history σ is defined inductively: the length of ϵ is 0, and the length of $\sigma' \cdot s$ is 1 plus the length of σ'. Then adequacy is defined in the following natural way based on the runs:

Definition 25. Suppose $\mathcal{P} = \langle s_0, \mathcal{E}, \mathcal{G} \rangle$ is a basic planning problem, and $\mathcal{C}$ is any controller. We say $\mathcal{C}$ is θ-adequate for $\mathcal{P}$ if the runs (at s_0) of $(\mathcal{C}, \mathcal{E})$ satisfies θ.

We can extend this definition to state how generalized planning problems are solved for some notion of adequacy:

Definition 26. We say a controller $\mathcal{C}$ is θ-adequate for a generalized planning problem $\overline{\mathcal{P}}$ iff $\mathcal{C}$ is θ-adequate for every basic problem in $\overline{\mathcal{P}}$.

Recall that Levesque's notion stipulates that a controller is correct for a basic action theory when the controller is correct for each situation considered accessible initially. As discussed before, Definition 26 captures precisely that idea, when the planning problems in $\overline{\mathcal{P}}$ differ only in the initial states.

Thus, controllers for generalized problems are adequate (in whichever sense) for all of the contained basic problems. Put differently, we can articulate conditions for basic problems, and to extend that condition for generalized problems we only need to stipulate that the controller is adequate to every contained basic problem. We are now prepared to define our notions of adequacy. We already identified **ONE**, **PC** and **TER** previously. But then in the interest of finite computation, we might also be interested in whether runs are *bounded* in length: that is, is there a number n such that all runs are lesser than or equal to n? We refer to this notion by **BND**. Finally, when considering finitely many states [25], bounded runs are invariably linked to the absence of *cycles,* where states previously visited are stipulated to not be visited at a subsequent point. We refer to such as notion by **ACYC**. Formally:

Definition 27. Given a basic planning problem with s_0 as its initial state and $\mathcal{G}$ as the goal states, for the set of runs (at s_0), we define:

ONE There is a terminating run σ such that $end(\sigma) \in \mathcal{G}$.
PC For every terminating run σ, we have $end(\sigma) \in \mathcal{G}$.
TER For every run σ, there is a history σ' such that $\sigma \cdot \sigma'$ is a terminating run.
BND Every run is *bounded*, that is, there is a number $n \in \mathbb{N}$ such that there is no run of length greater than n.
ACYC If $s \cdots s'$ is a run of length greater than 1, then $s \neq s'$.

To summarize the intuitions: **ONE** says that there is a run to the goal; **PC** denotes *partial correctness,* that is, every terminating run stops at a goal state; **TER** denotes *termination*, that is, it ensures that every run can be extended to a terminating one; **BND** denotes *boundedness*, that is, it disallows runs of arbitrary length[11]; finally, **ACYC** denotes *acyclicity*, that is, it disallows visiting the same state more than once.

For the main results of this section, we study some key properties of these notions for the various types of problems. We first begin with what we informally observed in Section 3. The reason Levesque's notion is appropriate in noise-free domains is because there is a sense in which runs are unique, as a result of which, **ONE** captures both goal satisfaction as well as termination.

Proposition 28. *In noise-free problems, if there is a run of length k then it is unique.*

Proof. By induction on k. Suppose $\mathcal{P} = \langle s_0, \mathcal{E}, \mathcal{G} \rangle$ is a noise-free basic planning problem, and $\mathcal{C}$ any controller. Clearly there is only one run of length 1, namely s_0. Suppose σ is a run of length k; by hypothesis it is unique. Suppose $\mathcal{C}(sensed(\sigma)) = stop$, then we are done. Suppose $\mathcal{C}(sensed(\sigma)) = \alpha \in \mathcal{A}$. In noise-free environments, there is a unique s' such that $\Delta(end(\sigma), \alpha, s') \neq 0$ and so, $\sigma \cdot s'$ is a run of length $k+1$ and it is unique. □

[11] In the context of BHL, to formalize a property such as **BND**, not surprisingly, requires second-order notions and can be fairly involved. For example, for the initial situation s_0, we would say: $\exists n \forall P(\kappa \wedge \forall q', s' \ [U^*(q_0, s_0, q', s') \supset \exists m \leq n \ P(m, q', s'))]$ where κ is the universal closure of: (a) $P(0, q_0, s)$; and (b) $P(n, q, s) \wedge U(q, s, q', s') \supset P(n+1, q', s')$.

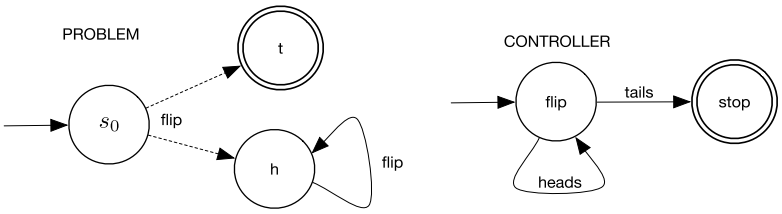


Fig. 3. Coin flip problem with bad flips and its controller (double stroked circles indicate goal states in the planning problem and the final control state in the controller).

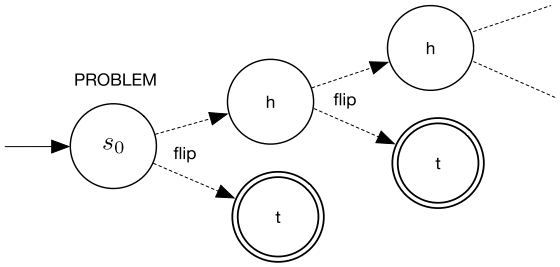


Fig. 4. Coin flip problem where after infinitely many heads, a tails can still be observed.

Theorem 29. *In noise-free problems,* **ONE** *is equivalent to* {**PC**, **TER**}. *This is not the case in noisy problems.*

Proof. For noise-free problems, suppose **ONE** holds: there is a terminating run σ, say of length k, and $end(\sigma) \in \mathcal{G}$. By Proposition 28, σ is the only run of length k, *i.e.* every other run $\sigma' \neq \sigma$ can be extended to σ, and by assumption, it is terminating and so **TER** holds. By that account, σ is also the only terminating run and so **PC** holds. Conversely, **TER** ensures that there is a terminating run and by Proposition 28, **PC** ensures that this reaches the goal, and so **ONE** is true.

For noisy problems, we provide a counterexample in Fig. 3. Consider an environment $\mathcal{E}$ with states $\{s_0, h, t\}$ and a basic problem $\langle s_0, \mathcal{E}, \{t\}\rangle$. A coin flip nondeterministically changes the environment from s_0 to either h where heads is observed, or to t where tails is observed. No actions are possible at t. Coin flips are possible at h, but the environment changes to h itself. In other words, if tails is not observed for the first coin flip, then it will not be observed for all coin flips. Using the controller, we see that **ONE** holds by getting a tails on the first flip. But **TER** does not hold because there is no history that makes the run $s_0 \cdot h$ a terminating one. □

Thus, we needed to have considered notions other than **ONE** in noisy domains. Let us also point out a consequence of these differences in terms of runs being bounded in length. In noise-free domains, when **TER** holds, so does **BND**. This can be intuitively argued for as follows: because **TER** holds, there is one terminating run and since it is also unique, its length gives the upper bound. Such a property is not true in noisy domains.

Theorem 30. *In noise-free problems, if* **TER** *holds then* **BND** *holds. This is not the case in noisy ones.*

Proof. Since **TER** holds, for any run σ, by Proposition 28, there is a unique (possibly empty) history σ' such that $\sigma \cdot \sigma'$ is terminating. Clearly, **BND** holds, as there cannot be a run of length greater than that of $\sigma \cdot \sigma'$.

For noisy domains, we provide a counterexample in Fig. 4. Imagine an environment where a coin flip action nondeterministically changes the current state s_i to s_{i+1} where heads is observed, or to s^*_{i+1} where tails is observed. Let us suppose all states $s^*_1, s^*_2, \ldots$ where tails are observed as terminating. Consider a basic planning problem with s_0 as the start state and the controller from Fig. 3. For this problem, given any run $\sigma = s_0 \cdot s_1 \cdots s_n$ that is not terminating (so tails has not been observed thus far), clearly there is a history, namely one where we observe tails on flipping $\sigma' = s^*_{n+1}$, such that $\sigma \cdot \sigma'$ is a terminating run. So **TER** indeed holds, but **BND** does not since the length of σ can be arbitrary. □

From the viewpoint of finite computation, we might therefore want to insist that **BND** holds. But what does that say about **TER**?

Theorem 31. *In any (basic planning) problem, if* **BND** *holds then* **TER** *holds.*

Proof. By assumption, all runs have length $\leq n$, for some $n \in \mathbb{N}$. In other words, there cannot be a run that is extendable to one of arbitrary length. Let Γ be the set of all runs that are not extendable. (For all runs $\sigma \notin \Gamma$, there must be a history

σ' such that $\sigma \cdot \sigma' \in \Gamma$ because σ cannot be extended arbitrarily by assumption.) Since none of the runs $\sigma \in \Gamma$ can be undefined (recall that systems are implicitly assumed to be proper), they must be terminating. □

Let us reflect on this. In nondeterministic domains, we first discover that **ONE** is too weak, and so by delineating goal satisfaction and termination, we could settle on controllers that are {**PC**, **TER**}-adequate. By means of the above theorem, we now show that in the interest of finite computation, we should insist on {**PC**, **BND**}-adequate controllers instead. That is, because **BND** implies **TER**, but not the other way around, and so {**PC**, **BND**}-adequacy is much more attractive.

It is also worth noting that we could place structural constraints on the domain by means of which **BND** holds. For example, in finite state systems, it should be easy to see that **ACYC** implies bounded runs.

Proposition 32. *In finite problems, if* **ACYC** *holds then* **BND** *holds. This is not the case in infinite ones.*

Proof. In finite problems, $\mathcal{S}$ is finite. Suppose **ACYC** holds but **BND** does not. Then there is a run of length $> |\mathcal{S}|$, which means that by the pigeon hole principle, the run must mention some state twice. Contradiction.

If $\mathcal{S}$ is infinite, a system can be easily constructed for which a run visits every state exactly once, and so **ACYC** would hold but **BND** would not. The coin flip problem in Fig. 4 is one such example: runs of the form $s_0 \cdot s_1 \cdots s_n$ never visit the same state twice but are unbounded. □

Readers may be curious how these properties relate to notions such as *weak, strong* and *strong cyclic* planning [25]. We show that our delineation offers a more granular view in a subsequent section: for example, the above properties allow us to illustrate subtleties when applying notions such as strong planning for finite problems. Moreover, precisely because a solution to a generalized problem reduces to the same solution applying to each of the contained basic problems, it is easy to see that the properties proven are useful even in the context of a noisy and infinite basic planning problem.

Before concluding this section, let us note that thus far, we have focused on categorical adequacy, which pertain to whether a run or the set of all runs satisfy a property. This was achieved by only looking at outcomes with non-zero probabilities; the actual probabilities did not matter. What is then the value of considering probabilities? One main reason is that striving for categorical adequacy may be too challenging: it could be very difficult (or even impossible if the set of runs is uncountable) to account for every run. In the presence of non-uniform probability distributions, what is especially appealing is to focus on those outcomes that have a higher likelihood to occur. Essentially, a biased search. To explore an account of adequacy that can leverage such intuitions, we now consider probabilistic variants of **PC** and **TER**.

6. Probabilistic adequacy

As argued above, the motivation for exploring probabilistic variants of correctness is to allow search to be biased, by exploring the most likely outcomes, or more generally, the most likely runs. There are two fundamental questions that we aim to clarify with such variants:

(a) Can we show that the categorical and the probabilistic notions are equivalent when considering the case where probabilities add up (or integrate) to 1?
(b) Are there domains where it is impossible to aspire for categorical adequacy, in the sense that they only admit (strictly less than 1) probabilistic guarantees?

Here, it is tempting to imagine that (a) should be true. But as we show, when considering uncountable sets, owing to the fact that the probability of any single point would be 0, probabilistic adequacy does not imply categorical adequacy. This then impacts what we can say about (b).

We first start with discrete problems, where it turns out (a) can indeed be answered in the affirmative.

6.1. Discrete problems

We begin by applying transition probabilities to runs of arbitrary length. We define the *likelihood* of a run σ at s in the system $(\mathcal{C}, \mathcal{E})$, denoted $l(\sigma)$, inductively:

- if $\sigma = s$, then $l(s) = 1$;
- if $\sigma = \sigma' \cdot s'$ and $\mathcal{C}(sensed(\sigma')) = \alpha$, then $l(\sigma) = l(\sigma') \times \Delta(end(\sigma'), \alpha, s')$.

Definition 33. Suppose $\mathcal{P} = \langle s_0, \mathcal{E}, \mathcal{G} \rangle$ is a basic planning problem. Then with respect to the set of runs at s_0, we define the following quantities to capture the likelihood of termination and goal satisfaction respectively:

$$\textbf{LTER} = \sum_{\{\sigma \mid \sigma \text{ is a terminating run}\}} l(\sigma)$$

$$\textbf{LPC} = \frac{\sum_{\{\sigma \mid \sigma \text{ is a terminating run and } end(\sigma) \in \mathcal{G}\}} l(\sigma)}{\textbf{LTER}}$$

Here, **LTER** is the probabilistic analogue to **TER** in the sense that it accounts for the likelihoods of terminating runs. Abusing notation, when we want this expression to equal or exceed a number $\lambda \in \mathbb{R}$, we write $\textbf{LTER} = \lambda$ or $\textbf{LTER} > \lambda$ respectively. Its counterpart for **PC** is **LPC**, which accounts for the likelihoods of terminating runs that also reach goal states. The normalization factor $\textbf{LTER} \neq 0$ is naturally defined in terms of terminating runs regardless of whether they reach the goal states.

For our main results, we relate the probabilistic and categorical accounts, as motivated earlier. First, let us prove a lemma about a set that lumps together runs of length $\leq k$.

Lemma 34. *Suppose $\mathcal{P} = \langle s_0, \mathcal{E}, \mathcal{G} \rangle$ is any discrete problem, and $\mathcal{C}$ any controller. For any $k \in \mathbb{N}$, let*

$$\Gamma_k = \{\sigma \mid \sigma \textit{ is a run of length } k\} \cup \{\sigma \mid \sigma \textit{ is a terminating run of length } \leq k\}.$$

(Here, as with the other adequacy notions, we are considering the set of runs at s_0 of $(\mathcal{C}, \mathcal{E})$.) Then:

$$\sum_{\sigma \in \Gamma_k} l(\sigma) = 1.$$

Proof. By induction on k. For $k = 1$, the proof is trivial since $l(s_0) = 1$ by definition. Assume claim is true for Γ_k. Clearly $\Gamma_k \cap \Gamma_{k+1}$ is the set of all terminating runs of length $\leq k$. Consider $\Gamma^* = \Gamma_{k+1} - \Gamma_k$. Observe that $\sigma \in \Gamma^*$ only if there is a $\sigma' \in \Gamma_k$ and a history of length 1, say s, such that $\sigma = \sigma' \cdot s$, that is, $\mathcal{C}(sensed(\sigma')) = \alpha$ and $\Delta(end(\sigma'), \alpha, s) \neq 0$. Let $exts(\sigma') = \{\sigma \in \Gamma^* \mid \sigma \text{ is an extension of } \sigma'\}$. Since $\Delta(end(\sigma'), \alpha, \cdot)$ is a probability mass function, $l(\sigma') = \sum_{\sigma \in exts(\sigma')} l(\sigma)$. It then follows that:

$$\sum_{\sigma \in \Gamma^*} l(\sigma) = \sum_{\sigma \in (\Gamma_k - \Gamma_{k+1})} l(\sigma).$$

Using this equivalence in the below equality:

$$\sum_{\sigma \in \Gamma_{k+1}} l(\sigma) = \sum_{\sigma \in (\Gamma_k \cap \Gamma_{k+1})} l(\sigma) + \sum_{\sigma \in \Gamma^*} l(\sigma)$$

we obtain that $\sum_{\sigma \in \Gamma_{k+1}} l(\sigma) = \sum_{\sigma \in \Gamma_k} l(\sigma)$ which is 1 by the hypothesis. □

Basically, the lemma clarifies that by considering Γ_k for every k, the sum of the probabilities of the runs in those must be 1. With this lemma, we can show that stipulating probabilistic certainty implies categorical adequacy, both for termination and goal satisfaction.

Theorem 35. *In discrete problems, if* **LTER**$=1$*, then* **TER** *holds.*

Proof. Suppose **LTER** is 1 but **TER** does not hold: there is a run σ^* such that for every σ' we have that $\sigma^* \cdot \sigma'$ is not a terminating run. By definition, $l(\sigma^*) > 0$. (Recall that runs are admitted only by transitions with non-zero probability.) Suppose the length of σ^* is k and consider the set Γ_k as in Lemma 34. By Lemma 34, $\sum_{\sigma \in \Gamma_k} l(\sigma) = 1$ and so $\sum_{\sigma \in \Gamma^*} l(\sigma) < 1$, where $\Gamma^* = \Gamma_k - \{\sigma^*\}$. Thus, for every $n > k$, by assumption, every extension of σ^* of length n that is a run is not terminating, and so $\sum_{\sigma \in \Gamma^{**}} l(\sigma) < 1$, where $\Gamma^{**} = \Gamma_n - exts(\sigma^*)$, Γ_n is as in Lemma 34, and $exts(\sigma^*)$ is the set of all extensions of σ^* that are runs of length n. Contradiction. □

Theorem 36. *In discrete problems, if* **LPC**$=1$*, then* **PC** *holds.*

Proof. Suppose **LPC**$=1$, that is, **LPC=LTER**, but **PC** does not hold: there is a terminating run σ^* that does not reach a goal state. Suppose Γ is the set of terminating runs. By definition, $l(\sigma^*) > 0$, and so clearly:

$$\sum_{\sigma \in \Gamma} l(\sigma) > \sum_{\sigma \in (\Gamma - \{\sigma^*\})} l(\sigma).$$

So **LPC**, which is defined over the runs $\subseteq (\Gamma - \{\sigma^*\})$ reaching a goal state, cannot be equal to **LTER**. Contradiction. □

Of course, since categorical adequacy accounts for all runs (of non-zero probability), it immediately implies probabilistic adequacy:

Proposition 37. *In discrete problems,* **TER** *implies* **LTER**$_{=1}$*, and* **PC** *implies* **LPC**$_{=1}$*.*

Putting this together, we get the following equivalence between categorical adequacy and adequacy with probability 1:

Corollary 38. *In discrete problems,* **LTER**$_{=1}$ *iff* **TER***, and* **LPC**$_{=1}$ *iff* **PC***.*

Thus, we can view **LTER** as a *weakening* of **TER**, in the sense that if we were to insist that $\mathbf{LTER} = 1$, we would also have **TER**-adequate controllers. Likewise for **PC**.

We reiterate that had this result not been true, then we would see **LTER** as a *strict weakening* of **TER** in the sense that even by insisting that $\mathbf{LTER} = 1$, we would not recover **TER**. This is precisely what we show when considering continuous problems in the sequel.

6.2. Continuous problems

Here, we want to study the consequences of a continuous transition function on probabilistic adequacy. We will attempt to establish results in the same fashion as for discrete problems, culminating in something like Corollary 38, or rather the lack thereof.

Let us begin by noting that in continuous problems, the uncountable nature of the state space precludes sums. But it is not a simple matter of replacing sums in the previous notions with an integral. Consider Fig. 3, for example. There is one run of length 1, and another of unbounded length. For the discrete versions of **LTER** and **LPC**, it sufficed to simply consider the set of terminating runs, say, and count their likelihoods. For the continuous case, however, we need a suitable space to define the density function and the resulting integral. It is unclear how to integrate over the set of terminating runs that vary arbitrarily in length. So what we attempt first is a reformulation of **LTER** and **LPC** for discrete problems where the sums are defined over the set of states rather than runs. This reformulation is more involved than the way **LTER** and **LPC** were defined above, and so we also show that the simpler definition is equivalent to this reformulation as a sanity check. The scheme is then generalized to the continuous case.

Let us define $L(\sigma, n)$ as the sum of the likelihoods of all histories σ' of maximal length n such that $\sigma \cdot \sigma'$ is a terminating run. More precisely:

Definition 39. Suppose $\mathcal{P} = \langle s_0, \mathcal{E}, \mathcal{G} \rangle$ is any discrete problem, $\mathcal{C}$ any controller, and σ any history of $(\mathcal{C}, \mathcal{E})$. Then, define $L(\sigma, n)$ inductively:

- $L(\sigma, 1) = \begin{cases} 1 & \text{if } \sigma \text{ is terminating} \\ 0 & \text{otherwise} \end{cases}$
- $L(\sigma, n+1) = \begin{cases} L(\sigma, n) & \text{if } \sigma \text{ is terminating} \\ succ & \text{otherwise} \end{cases}$

  where,

$$succ \doteq \sum_{s} L(\sigma \cdot s, n) \times \Delta(end(\sigma), \mathcal{C}(sensed(\sigma)), s)$$

Let us define $L^{\bullet}(\sigma, n)$ inductively, which is exactly like $L(\sigma, n)$ except for the base case:

$$L^{\bullet}(\sigma, 1) = \begin{cases} 1 & \text{if } \sigma \text{ is terminating and } end(\sigma) \in \mathcal{G} \\ 0 & \text{otherwise.} \end{cases}$$

We will be equating $L(\sigma, n)$ in the limit with **LTER** and analogously, $L^{\bullet}(\sigma, n)$ in the limit will be equated to **LPC**.

Theorem 40. *In discrete problems,* **LTER** *yields the same value as:*

$$\lim_{n \to \infty} L(s_0, n) \tag{†}$$

Proof. Let $\mathcal{P} = \langle s_0, \mathcal{E}, \mathcal{G} \rangle$ be any discrete problem and $\mathcal{C}$ any controller. Let Γ be the set of terminating runs at s_0 of $(\mathcal{C}, \mathcal{E})$. Let $N = \max\{\text{length of } \sigma \mid \sigma \in \Gamma\}$.

For $n \geq 1$, let

$$\mathbf{LTER}_n = \sum_{\{\sigma \mid \sigma \text{ is a terminating run of length } \leq n\}} l(\sigma)$$

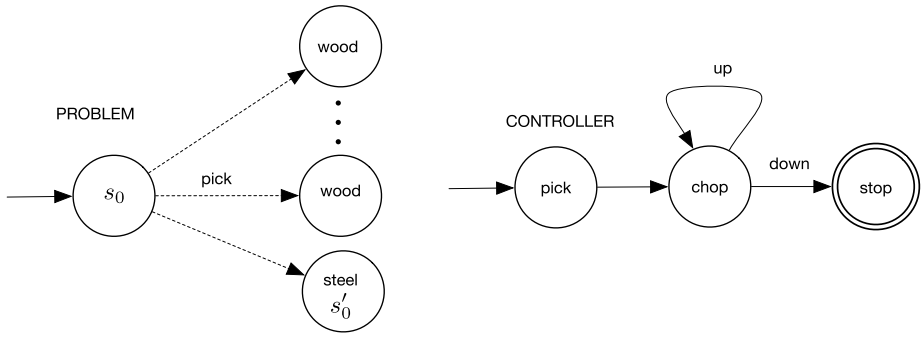


Fig. 5. Wood and steel trees.

An induction argument will show that $\textbf{LTER}_n = L(s_0, n)$. So clearly $\textbf{LTER} = \textbf{LTER}_N = L(s_0, N)$. By definition, we have $L(s_0, N) = L(s_0, N + k)$ for $k > 0$. Thus the sequence $L(s_0, 1), \ldots, L(s_0, N), \ldots$ converges and is $L(s_0, N)$. Therefore, **LTER**=(†). □

Theorem 41. *In discrete problems,* **LPC** *yields the same value as:*

$$\frac{\lim_{n\to\infty} L^{\bullet}(s_0, n)}{\textbf{LTER}} \tag{‡}$$

Proof. We need a simple variation of the argument for Theorem 40. We begin by defining Γ to be the set of all terminating runs that also reach a goal state. The rest proceeds analogously. □

These are now amenable to the use of integrals:

Definition 42. In continuous problems, **LTER** is given by (†) and **LPC** by (‡), where $L(\sigma, n)$ and $L^{\bullet}(\sigma, n)$ are as in Definition 39 except that

$$succ \doteq \int_s L(\sigma \cdot s, n) \times \Delta(end(\sigma), \mathcal{C}(sensed(\sigma)), s).$$

In a continuous setting, while a density is accorded to all points of the state space, the probability of any one point (or a finite set of points) is zero. This leads to surprising deviations from Corollary 38.

Theorem 43. *In continuous problems,* **LTER**=1 *does not imply* **TER**.

Proof. Proof by counterexample. The idea is to allow one run to never terminate but which nonetheless does not affect the outcome of integration. Imagine a pick action that nondeterministically changes an environment from s_0 to a state in $\mathcal{S}' = \{s'_x \mid x \in [0, 1]\}$, corresponding to a sample drawn from the interval $[0, 1]$. All states contain a tree and respond with an observation that the tree is up. States in $\mathcal{S}' - \{s'_0\}$ contain a wooden tree that can be brought down by doing some number of chop actions. However, the state s'_0 has a steel tree that can never be brought down; see Fig. 5. Here, integrating over $\mathcal{S}' - \{s'_0\}$ yields **LTER**=1, but **TER** does not hold because no extension of $s_0 \cdot s'_0$ will terminate. □

Theorem 44. *In continuous problems,* **LPC**=1 *does not imply* **PC**.

Proof. Proof by counterexample. The idea is to allow one run to not reach the goal but which will not affect the outcome of integration. Imagine a robot operating in a 1-dimensional world, at location 0. Its goal is to be any location but at 0 using a single move action. Assuming this action is characterized by a continuous noise model, such as a Gaussian, all outcomes, including the one where the robot moves by 0 units, are given a non-zero density. Assume all states vacuously respond with the observation *true*. A very simple controller can be specified for this problem; see Fig. 6. (Note that the argument to the move action will not affect the nature of this example.) So, all runs terminate, but the one where $loc = 0$ is not a goal state. Integrating over the rest still leads to **LPC**=1, but **PC** does not hold. □

On the other hand, categorical adequacy clearly implies probabilistic adequacy. In particular, Proposition 37 is easily shown to hold also in the continuous case:

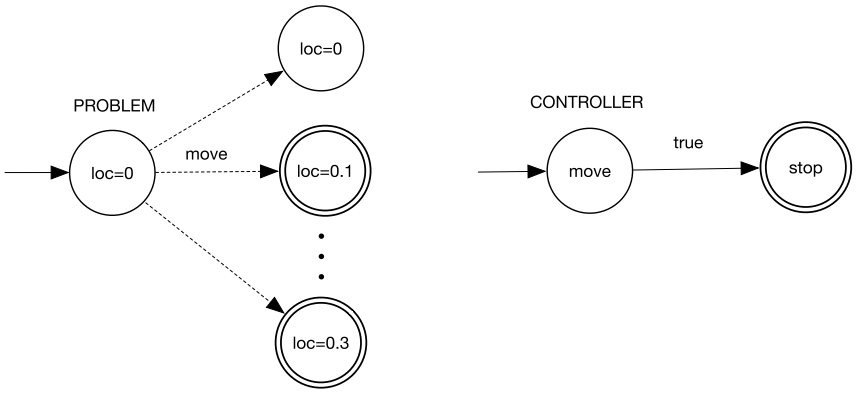


Fig. 6. Continuous problem with the goal $loc \neq 0$.

Proposition 45. *In continuous problems,* **TER** *implies* **LTER**$_{=1}$*, and* **PC** *implies* **LPC**$_{=1}$*.*

As a consequence, this means that equivalences from Corollary 38 do not hold for continuous problems:

Corollary 46. *In continuous problems,* **TER** *and* **LTER**$_{=1}$ *are not equivalent, and* **PC** *and* **LPC**$_{=1}$ *are not equivalent.*

With these results established, we will now consider whether planning problems can be specified in a way that precludes certain types of correctness.

6.3. Barriers to adequacy

Having investigated relations between termination and correctness criteria across environments, we are now in a position to ask: *are there planning problems for which only a particular sort of adequacy can be provided?* In this section, we answer positively to that question. We consider a set of planning problems using the inventory of examples dealt with so far, and discuss sample plans. Like with troublesome domains such as Fig. 5, it will be easy to argue that any variation of these sample plans can, for example, only provide some flavor of probabilistic adequacy, but never categorical ones. In particular, this not only suggest that the generalization of Levesque's proposal lead new insights on how categorical adequacy should be defined in the presence of nondeterminism, but that with stochastic nondeterminism, we have weaken our idea of correctness to probabilistic variants.

We will respond to the above question by thinking in terms of two dimensions: variants of termination in one dimension, and variants of goal satisfaction on another. In particular, consider the following three criteria for termination: **BND**, **LTER**$_{=1}$ and **LTER** < 1. (Rather than considering **TER**, we consider **BND**-adequacy as it is stronger than **TER** in the sense of ensuring finite termination, as established previously.) Consider the following three criteria for correctness: **PC**, **LPC**$_{=1}$ and **LPC** < 1. Below, we enumerate problems that are distinctive for every pair of termination and correctness criteria:

	PC	**LPC**$_{=1}$	**LPC** < 1
BND	(1)	(2)	(3)
LTER$_{=1}$	(4)	(5)	(6)
LTER < 1	(7)	(8)	(9)

1. *Standard tree chop (Fig. 1)*: the agent chops down a tree of thickness $n > 0$ by executing n chop actions. So there is a unique terminating run of length $n + 1$ reaching the goal. Therefore, both **BND** and **PC** hold.
2. *Point goal (Fig. 6)*: all runs $s_0 \cdot s'_x$ for $x \in [0, 1]$ are terminating, and so **BND**. As the goal is $loc \neq 0$, every run other than $s_0 \cdot s'_0$ reach goal states and so **PC** does not hold. Owing to the continuity, **LPC**$_{=1}$ holds.
3. *Interval goal*: consider the controller and environment from item 2 but assume that the planning problem has as its goal $loc > .5$. As before, all runs $s_0 \cdot s'_x$ for $x \in [0, 1]$ are terminating, and so **BND**. However, every run other than $s_0 \cdot s'_y$ for $y \in [0, .5]$ are goal states; so **PC** does not hold. Moreover, although the probability of any single s'_y is 0 (as in item 2), the probability of this set of points, which comprise an interval, is non-zero. So we get **LPC** < 1. (As with item 2, the argument to the move action will not affect the nature of this example.)
4. *Flip and chop (Fig. 7)*: a tails on a coin flip permits the agent to chop down a tree. All terminating runs reach the goal, and so **PC**, but there is no bound because of the coin flip. So **BND** does not hold, but **TER** and **LTER**$_{=1}$ hold.

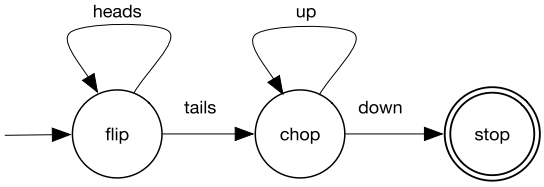


Fig. 7. Controller for flip and chop.

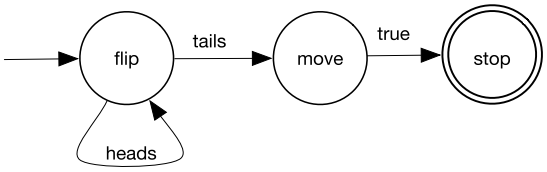


Fig. 8. Controller for flip and move.

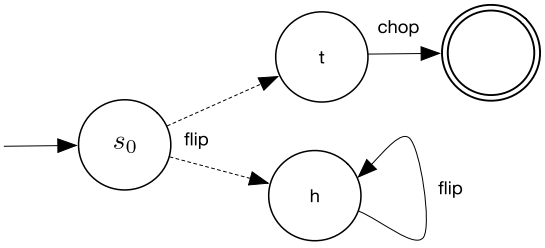


Fig. 9. Bad flip and chop problem.

5. *Flip and point goal (Fig. 8)*: a tails on a coin flip permits the agent to attempt $loc \neq 0$ using a single move action. As in item 4, owing to the coin flip, there is no bound on the runs, but **TER** and **LTER**=1 hold. Moreover, as established for item 2, **PC** does not hold but **LPC**=1.
6. *Flip and interval goal (Fig. 8)*: a tails on a coin flip permits the agent to attempt $loc > .5$ using a single move action. As seen in item 5, **LTER**=1, but **BND** does not hold. Moreover, as established for item 3, **LPC** < 1.
7. *Bad flip and chop*: consider the environment in Fig. 9 where a coin flip at s_0 nondeterministically changes to either h, where heads is observed, or to t where tails is observed. The state t also has a tree of unit thickness. The goal is to bring down this tree. Only coin flips are possible at h, but the environment changes to h itself. Only chop actions are possible at t. Using the controller from item 4, we observe that no extension of the run $s_0 \cdot h$ will terminate, and so **TER** does not hold. (Moreover, coin flips means **BND** does not hold.) In other words, **LTER** is determined from the transition probability for t and so is < 1. Nonetheless, all extensions of $s_0 \cdot t$ that are runs are both terminating and reach goal states, and so **PC** holds.
8. *Bad flip and point goal*: consider an environment like the one in Fig. 9, except that at t, we imagine a robot at location 0, that is, $loc = 0$. A move action nondeterministically changes the environment to $\{t'_x \mid x \in [0, 1]\}$ in that t'_x is a state where $loc = x$. The goal is to be at a state where $loc \neq 0$. Using the controller from item 5, we observe that, as in item 5, **PC** does not hold but **LPC**=1. Owing to the bad coin flip, from item 7, we obtain **LTER** < 1.
9. *Bad flip and interval goal*: consider the environment from item 8, except that the goal is $loc > .5$. As in item 6, owing to the interval goal, **LPC** < 1. Owing to the bad coin flip, from item 7, we obtain **LTER** < 1.

These distinct instances could be seen to a sort of impossibility result. For example, in a planning problem like in (8) seen in Fig. 9, is it possible to be able to design a **TER**-adequate controller? The answer clearly is in the negative. If the coin flip yields a heads, we are stuck in a state where the only action available is another flip, and regardless of the outcome of that flip, we remain in the state. One could perhaps argue that we could simply disallow runs that have such properties. For example, enforcing **ACYC** could prevent us from considering the branch where the first flip yielded a heads, but our view is that this is unrealistic: suppose a robot is on the planet Mars, and an action causes the robot to fall into a ditch. It could very well be that it would be stuck in that ditch forever, and the best we can hope for are probabilistic guarantees, unless there is some deterministic course of action that avoids such situations altogether. Overall, based on the above examples, we can establish the following result:

Theorem 47. *Given the 9 distinct combinations of correctness and termination criteria identified, there are planning problems that belong to one and none other.*

This result could be distilled into the following corollary when emphasizing those cases where **LTER** < 1 and/or **LPC** < 1:

Corollary 48. *There are planning problems where: (a) categorical adequacy is impossible, and (b) probabilistic termination (and/or goal satisfaction) with probability 1 is impossible.*

Thus, starting from Levesque's notion of correctness, we delineated goal satisfaction and termination to arrive at {**TER**, **PC**}-adequate controllers for nondeterministic actions. Since **TER** did not guarantee finite computation, we suggested {**BND**, **PC**}-adequate controllers. We then proposed probabilistic correctness notions and have now discovered that for certain planning problems, only strictly weak probabilistic guarantees are possible. This results in a nuanced view of correctness.

Before turning to related work, we will consider three additional results on the generality of our notion of adequacy in the context of some other accounts.

7. Adequacy and generality

In this section, we will study additional results on the generality of our notion of adequacy. We will first reflect on how our controller framework relates to the generality of BHL. Next, we will revisit the notion of goal achievability from Lin and Levesque [53] and formally establish that it is equivalent to Levesque's notion of correctness, and so our results also affect that notion. Finally, we will then consider the notions of weak, strong and strong cyclic from Cimatti et al. [25], and show that our results from categorical adequacy provides a more granular view on those notions.

7.1. Adequacy and BHL

Let us briefly reflect on how our model relates to BHL. In essence, we would like to provide clarity on three aspects:

- how does (categorical and) probabilistic adequacy as discussed above relate to the notion of correctness argued for BHL (Section 3)?
- what role do the probabilities on the initial states play?
- how might we handle uncertainty about the distributions (on the initial states and/or noise models), as allowed by BHL?

First, by means of Definition 26, it is easy see how categorical adequacy defined for a basic planning problem applies to a generalized planning problem: essentially, a controller is considered to be, say, **TER**-adequate for a generalized planning problem $\overline{\mathcal{P}}$ iff it is **TER**-adequate for all the basic problems contained in $\overline{\mathcal{P}}$. This then also informs us about how to provide an account for probabilistic adequacy. We would say a controller is considered to be, for example, (**LTER** > 0)-adequate for a generalized planning problem $\overline{\mathcal{P}}$ iff it is (**LTER** > 0)-adequate for all the basic problems contained in $\overline{\mathcal{P}}$.

Second, BHL stipulates a probability distribution on the set of initial states. We ignored that here, mainly because the notion of correctness from Section 3 stipulates that for each world considered accessible initially, a certain property holds. While Section 3 focused on categorical notions, the same view can also be applied to probabilistic adequacy. Roughly, this means that **LTER** $= 1$ should be interpreted as saying the probabilities of the set of all runs starting from some initial state sum up (or integrate) to 1. If we wished to capture a condition such as $\mathbf{GEQ}_\kappa$, we could simply let $\overline{\mathcal{P}}$ to only include those basic planning problems whose initial worlds have a probability that exceeds κ. Then, stipulating that the controller is **ONE**-adequate for such a constrained $\overline{\mathcal{P}}$ is capturing $\mathbf{GEQ}_\kappa$.

However, we may want to have a more global view of the likelihood of terminating runs, as needed for $\mathbf{BEL}_\kappa$. In this case, we might as well consider the total likelihood. This can be achieved as follows: let **LTER** for a basic planning problem $\mathcal{P}$ be defined as before. Let us denote this by $\mathbf{LTER}_\mathcal{P}$. Suppose we have a generalized planning problem $\overline{\mathcal{P}} = \{\mathcal{P}_1, \ldots, \mathcal{P}_k\}$, where $\mathcal{P}_i = \langle s_i, \mathcal{E}, \mathcal{G}\rangle$. (We are assuming discrete problems and that $\overline{\mathcal{P}}$ contains finitely many basic problems for simplicity.) Then let $\mathbf{LTER}_{\overline{\mathcal{P}}}$ be defined as

$$\Pr(s_1) \times \mathbf{LTER}_{\mathcal{P}_1} + \ldots + \Pr(s_k) \times \mathbf{LTER}_{\mathcal{P}_k},$$

where Pr is the distribution on the set of initial worlds $\{s_1, \ldots, s_k\}$. Then $\mathbf{LTER}_{\overline{\mathcal{P}}} = 1$ is only possible when $\mathbf{LTER}_{\mathcal{P}_i} = 1$ for each $\mathcal{P}_i$, for example. In fact it is easy to see:

Proposition 49. *Suppose* $\overline{\mathcal{P}} = \{\mathcal{P}_1, \ldots, \mathcal{P}_k\}$ *is a discrete generalized planning problem. Suppose the corresponding initial worlds are* $\{s_1, \ldots, s_k\}$*, and let* Pr *be the distribution on these initial worlds. Suppose further that* $\Pr(s_i) > 0$ *for every i. Then a controller is* **TER***-adequate iff* $\mathbf{LTER}_{\overline{\mathcal{P}}} = 1$.

Proof. For a discrete generalized planning problem $\overline{\mathcal{P}}$, a controller is **TER**-adequate iff it is **TER**-adequate for all the basic problems contained in $\overline{\mathcal{P}}$. But by Corollary 38, a controller is **TER**-adequate for a basic problem iff $\mathbf{LTER} = 1$. Thus, regardless of the distribution on the basic problems in $\overline{\mathcal{P}}$, $\mathbf{LTER}_{\overline{\mathcal{P}}}$ must be 1.

Note that if we did not insist that $\Pr(s_i) > 0$ then we could have the following situation. Suppose there is a controller that is **TER**-adequate for every basic problem other than $\mathcal{P}_i$ for some i. Suppose $\Pr(s_i) = 0$. Nonetheless, it follows that $\mathbf{LTER}_{\overline{\mathcal{P}}} = 1$. However, because the controller is not **TER**-adequate for $\mathcal{P}_i$, it is also not **TER**-adequate for $\overline{\mathcal{P}}$. And, so the claim does not hold. □

Thus, by considering $\mathbf{LTER}_{\overline{\mathcal{P}}} \geq \kappa$, we essentially have $\mathbf{BEL}_\kappa$ as defined in Section 3.

Finally, let us consider the case that there is uncertainty about the distribution Pr on the initial worlds, and perhaps also on the transition probabilities Δ.[12] In particular, suppose the agent is not sure which distribution among $\{\Pr^1, \ldots, \Pr^m\}$ applies to the initial worlds, and the agent is also not sure which probabilistic transition function among $\{\Delta^1, \ldots, \Delta^n\}$ applies to dynamics. Given a pair $(\Pr^i, \Delta^j)$, let $\mathbf{LTER}_{\overline{\mathcal{P}}}^{(i,j)}$ be defined exactly as above wrt the distribution $\Pr^i$ on the initial states and the transition probabilities provided by Δ^j. Then let us say that a controller is $(\mathbf{LTER}_{\overline{\mathcal{P}}}^{(*,*)} > 0)$-adequate for the generalized planning problem $\overline{\mathcal{P}}$ iff the controller is $(\mathbf{LTER}_{\overline{\mathcal{P}}}^{(i,j)} > 0)$-adequate for every $i \in \{1, \ldots, m\}, j \in \{1, \ldots, n\}$. Like in Proposition 49, under the assumption that none of the initial worlds in $\overline{\mathcal{P}}$ are accorded a zero probability by any of the distributions, it is not hard to show that a controller is **TER**-adequate for $\overline{\mathcal{P}}$ iff it is $(\mathbf{LTER}_{\overline{\mathcal{P}}}^{(*,*)} = 1)$-adequate.

All of this shows that we can recover the generality of BHL. Moreover, if we instead considered unconstrained generalized planning problems, our framework is easily seen to be more flexible than BHL. Indeed, stipulating correctness for multiple problem instances each with their own distinct action theories will be very awkward to formulate in the situation calculus, and may possibly require meta-level statements.

7.2. Goal achievability

Lin and Levesque [53] (LL henceforth) considered the problem of *goal achievability*, which is perhaps best explained using an example. Imagine an agent facing two doors, the first of which has a tiger that can kill the agent and the second has gold. The agent, however, does not know which door has what. Although the goal of getting the gold is achievable by opening the second door, we would not say the agent *knows how* to get to the gold. To that end, LL propose a notion of *goal achievability*. The rough idea is that given a controller, a goal is said to be *achievable* iff there is a controller such that for every initial situation considered epistemically possible, there is a terminating run that makes the goal true. Thus, it is essentially in the spirit of Levesque's proposal [52]. In this section, we will formalize the relationship between goal achievability as considered by LL and our work.

Formally, LL define a controller framework to define achievability. Their work assumes binary sensing outcomes, and does not address nondeterministic outcomes, and so slightly simpler to ours.

Definition 50. [53] A history is an element of the set $\mathcal{R} = (\mathcal{A} \times \{0, 1\})^*$, where $\{0, 1\}$ is the set of observations, a controller $\mathcal{C}$ is any function mapping histories to $\mathcal{A} \cup \{stop\}$, and an environment $\mathcal{E}$ is any function mapping $\mathcal{R} \times \mathcal{A}$ to $\{0, 1\}$. A controller is effective if $\mathcal{C}$ is recursive. A history σ is a run of the system $(\mathcal{C}, \mathcal{E})$ if inductively $\sigma = \epsilon$ is the empty sequence, or $\sigma = \sigma' \cdot (\alpha, \beta)$ where σ' is a run, $\mathcal{C}(\sigma') = \alpha \in \mathcal{A}$, and $\mathcal{E}(\sigma', \alpha) = \beta \in \{0, 1\}$.

Basically, LL's notion of an environment can be seen a simplified account of our model focusing purely on noise-free environments. Their environment implicitly changes after an action is performed by the controller. While histories in LL are different from ours in also mentioning the actions, this difference is not crucial: in any run of a system, actions mentioned in histories are precisely those advised by the controller. Other minor differences can be resolved analogously.

Although controllers and environments are defined in an abstract fashion by LL, the notion of goal achievability is formalized in the logical language of the situation calculus. In our work, we chose to develop our results entirely in the abstract framework for ease of presentation and clarity. Continuing in that spirit, we reformulate their intuitions in an abstract setting:

[12] While unusual, as mentioned previously, owing to the situation being an argument in the l-fluent specification, the BHL approach also admits uncertainty about the likelihood models. For example, we would be allowed to provide a l-axiom for the $chop(x, y)$ action from Section 2.3 instead as: $l(chop(x, y), s) = \mathcal{N}(y; x, .25) \vee l(chop(x, y), s) = \mathcal{N}(y; x, 1)$. (Basically, such a sentence would be included as part of the basic action theory and the foundational axioms, and we would be interested in the entailments of such a logical theory.) That is, the actual value is normally distributed around the intended value with a variance of 0.25 or the actual value is normally distributed around the intended value with a variance of 1. A more involved specification could be $\exists z(z > .25 \wedge l(chop(x, y), s) = \mathcal{N}(y; x, z))$, which says that we have a Gaussian error profile for the chop action whose variance is some value greater than 0.25. This would admit infinitely many distributions for the likelihood model. (For our controller framework, we will continue entertaining only finitely many possibilities for simplicity.)

Definition 51. Suppose $\overline{\mathcal{P}} = \{\langle s_0, \mathcal{E}, \mathcal{G}\rangle \mid s_0 \in \mathcal{I}\}$ is a noise-free generalized planning problem over goal states $\mathcal{G}$. We say that $\mathcal{G}$ is (effectively) achievable iff there is a (effective) controller $\mathcal{C}$ such that for every $\langle s_0, \mathcal{E}, \mathcal{G}\rangle \in \overline{\mathcal{P}}$, there is a terminating run σ at s_0 of $(\mathcal{C}, \mathcal{E})$ and $end(\sigma) \in \mathcal{G}$.

As a realization of effective controllers, LL consider *robot programs* [52], which we discussed previously. These are built from atomic actions $\mathcal{A}$, sequences, branches and loops. A central result in their work is that a goal is effectively achievable iff there is a robot program that achieves the goal. However, as discussed earlier, based on Hu and Levesque [39], it follows that loopy plans are also simple candidates but more expressive than robot programs. Putting it all together, we obtain the following result:

Proposition 52. *Suppose $\mathcal{O} = \{0, 1\}$, and $\overline{\mathcal{P}}$ is a noise-free generalized planning problem over goal states $\mathcal{G}$. Suppose $\mathcal{R}$ is a robot program, and $\mathcal{X}$ is the corresponding reactive plan. Then, $\mathcal{R}$ effectively achieves $\mathcal{G}$ iff $\mathcal{X}$ is* **ONE***-adequate for $\overline{\mathcal{P}}$.*

The proof can be seen to be straightforward, requiring us to only match the differences between LL's framework and a simplified version of ours. Indeed, since the environment is noise-free, it is not surprising that **ONE**-adequacy is deemed sufficient; for example, as seen in Theorem 29, {**PC**, **TER**}-adequacy is implied, and by Theorem 30, so is {**PC**, **BND**}-adequacy.

Overall, LL's notion of goal achievability has little appeal in noisy domains, as it follows the thrust of Levesque's notion, thereby suffering from the same limitations. Indeed, the results in our work show that in noisy domains, **ONE** does not imply **TER**. Moreover, insisting on **TER** does not imply that **BND** holds. So, in noisy domains, we should instead be insisting on {**BND**, **PC**}-adequate controllers.

7.3. Weak, strong and strong cyclic planning

We will now relate our framework and its results to the influential notions of *weak*, *strong* and *strong cyclic* planning in noisy but finite systems. These notions were first considered in [23,24,27] and later refined in Cimatti et al. [25] (CPRT henceforth). We will base our discussion on CPRT: in what follows, we first introduce their framework, and then discuss what these solutions mean in the context of adequacy.

Definition 53. [25] A *planning domain* $\mathcal{D}$ is a tuple $\langle \mathcal{Y}, \mathcal{S}, \mathcal{A}, \mathcal{R}\rangle$, where $\mathcal{Y}$ is a finite set of propositions, $\mathcal{S} \subseteq 2^{\mathcal{Y}}$ is the set of states, $\mathcal{A}$ is a finite set of actions, and $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{A} \times \mathcal{S}$ is the transition relation. A *planning problem* $\mathcal{B}$ over $\mathcal{D}$ is a tuple $\langle \mathcal{D}, \mathcal{I}, \mathcal{G}\rangle$ where $\mathcal{I}, \mathcal{G} \subseteq \mathcal{S}$ are the initial and goal states respectively.

A *state-action table* π for $\mathcal{D}$ is a set of pairs $\langle s, \alpha\rangle$, where $s \in \mathcal{S}$ and $\alpha \in \mathcal{A}$ provided there is a s' such that $\mathcal{R}(s, \alpha, s')$.

The *execution structure* induced by π from $\mathcal{I}$ is a tuple $\mathcal{K} = \langle \mathcal{Q}, \mathcal{T}\rangle$, where $\mathcal{Q} \subseteq \mathcal{S}$ and $\mathcal{T} \subseteq \mathcal{S} \times \mathcal{S}$, which is the smallest set satisfying:

- if $s \in \mathcal{I}$ then $s \in \mathcal{Q}$
- if $s \in \mathcal{Q}$ and there exists a state-action pair $\langle s, \alpha\rangle \in \pi$ such that $\mathcal{R}(s, \alpha, s')$ then $s' \in \mathcal{Q}$ and $(s, s') \in \mathcal{T}$.

A state $s \in \mathcal{Q}$ is a *terminal state* of $\mathcal{K}$ if there is no $s' \in \mathcal{Q}$ such that $(s, s') \in \mathcal{T}$. An *execution path* of $\mathcal{K}$ at $s_0 \in \mathcal{I}$ is a possibly infinite sequence $s_0 \cdot s_1 \cdot s_2 \cdots$ such that: either s_i is the last state of the sequence (that is, a terminal state) or $(s_i, s_{i+1}) \in \mathcal{T}$. Define that a state s' is *reachable* from a state s if there is a path from s to s'.

It is easy to see that a planning problem $\mathcal{B}$ defined as above is essentially a generalized planning problem $\overline{\mathcal{P}}$ of the following sort: a fully observable and finite environment $\mathcal{E}$ is constructed from $\mathcal{S}, \mathcal{A}$ and $\mathcal{R}$ in an obvious way, and then we let $\overline{\mathcal{P}} = \{\langle s_0, \mathcal{E}, \mathcal{G}\rangle \mid s_0 \in \mathcal{I}\}$.

CPRT consider controller specifications by appealing to the state-action table π. In particular, suppose for every s mentioned in π, there is a unique α such that $\langle s, \alpha\rangle \in \pi$. Such a π is called *deterministic*, and it can be viewed as a controller in that it specifies the action to be performed at any given state.[13] Then, π is said to be a solution of a certain type (to $\mathcal{B}$) if the execution structure $\mathcal{K}$ satisfies appropriate properties. In particular, π is a weak solution if for any $s_0 \in \mathcal{I}$, some terminal state of $\mathcal{K}$ is reachable that is also in $\mathcal{G}$. Formally:

Definition 54. Suppose $\mathcal{D}, \mathcal{B}, \mathcal{K}, \mathcal{Q}, \mathcal{I}, \mathcal{G}$ and the rest are as in Definition 53. Then, π is a *weak solution* if for any $s_0 \in \mathcal{I}$, some terminal state of $\mathcal{K}$ is reachable that is also in $\mathcal{G}$. Analogously, π is a *strong cyclic solution* to $\mathcal{B}$ if from any state in $\mathcal{Q}$ some terminal state of $\mathcal{K}$ is reachable and all terminal states of $\mathcal{K}$ are in $\mathcal{G}$. Finally, π is a *strong solution* if $\mathcal{K}$ is acyclic and all terminal states of $\mathcal{K}$ are in $\mathcal{G}$.

[13] Note that this does not rule out nondeterministic transitions. The case of state-action tables that are nondeterministic is addressed in terms of all possible deterministic instances; we omit that exposition here. The motivation behind this flexibility is that an agent is allowed to freely choose any of the available actions to execute at a state.

Note that here, $\mathcal{K}$ is defined to be acyclic if all its execution paths are finite. But because the state space is finite, this is implicitly understood to mean that the same state cannot appear more than once in an execution path. Indeed, CPRT [25, footnote 1] write: "since the execution paths of strong plans cannot pass twice through the same state."

With this complication put aside, it is fairly straightforward to see how these notions relate in terms of adequacy[14]:

Proposition 55. *Suppose $\mathcal{B}$ is as above, with a deterministic state-action table π, and suppose $\overline{\mathcal{P}}$ is the corresponding generalized planning problem. Then:*

- *weak solutions (for $\mathcal{B}$) are equivalent to* **ONE***-adequacy (in $\overline{\mathcal{P}}$);*
- *strong cyclic solutions are equivalent to* {**PC**, **TER**}*-adequacy;*
- *strong solutions are equivalent to* {**ACYC**, **PC**}*-adequacy.*

Proof. As discussed, $\mathcal{B}$ corresponds to $\overline{\mathcal{P}}$, in the sense that for any $s_0^i \in \mathcal{I}$ in $\mathcal{B}$, we have a corresponding basic planning problem $\mathcal{P}^i \in \overline{\mathcal{P}}$. For π to be a weak solution for s_0^i, there is an execution path $s_0^i \cdots s_n$ such that s_n is a terminal state but also $s_n \in \mathcal{G}$. Since π advises an action at every state, it is easily shown to be a controller in our framework; for a terminal state, let the controller advise *stop*. It is now the case that the corresponding run σ in our framework is clearly a terminating one and moreover $end(\sigma) \in \mathcal{G}$. This is precisely the condition postulated by **ONE**.

Analogously, for π to be a strong cyclic solution, $\mathcal{Q}$ consists of states reachable from s_0^i, and from every such state, a terminal state is reachable. In our framework, every run can be extended to a terminating one, which is equivalent to **TER**. Moreover, every terminal state is in $\mathcal{G}$, which is equivalent to **PC** that requires every terminating run to be goal satisfying.

Analogously, strong solutions require execution paths to not mention the same state twice, which equivalent to runs not mentioning the same state twice, *i.e.* **ACYC**. Moreover, all terminal states need to be in $\mathcal{G}$, which is equivalent to saying all terminating runs are goal satisfying. □

Thus, with our analysis, we have a much more granular view of correctness that enabled by CPRT's notions. We can now leverage results from Section 5 to interpret the applicability of these solution types. For example, in Theorem 32, we established that **BND** is implied by **ACYC**, and so:

Corollary 56. *Suppose $\mathcal{B}$ and $\overline{\mathcal{P}}$ are as in Proposition 55. Then, strong solutions imply* {**BND**, **PC**}*-adequacy.*

This is precisely what is informally hinted by CPRT. They write that strong solutions enable the goal with a bounded number of steps, whereas strong cyclic solutions can loop forever and be of unbounded length [25]. Indeed, we know from Section 5 that **TER** does not imply **BND** with nondeterministic actions, and so if we simply insist on {**PC**, **TER**}-adequacy, which are essentially strong cyclic solutions, we do not obtain bounded runs in general. We can either enable structural properties such as **ACYC** and insist on {**ACYC**, **PC**}-adequate controllers, as would strong solutions, or simply insist on {**BND**, **PC**}-adequate controllers. Admittedly, in finite domains, there is no appeal in allowing cycles, so insisting on **ACYC** is entirely reasonable. But if the domain is infinite, one way to ensure finite termination is to insist on {**BND**, **PC**}-adequate controllers where possible. This is the reason why not all of CPRT's notions are appealing in infinite domains:

Corollary 57. *Suppose $\mathcal{B}$ and $\overline{\mathcal{P}}$ are as in Proposition 55, except that the set of states $\mathcal{S}$ of $\mathcal{B}$ is infinite. Then, strong solutions do not imply* **BND***-adequacy.*

In sum, as argued above, to capture the spirit of strong solutions in the sense of only admitting finite execution paths (or in our lingo: bounded runs), {**PC**, **BND**}-adequacy is better suited across finite and infinite planning domains. An interesting

[14] There is an interesting technical subtlety in the declarative account of [27] (DTV henceforth) that is worth a remark. (We thank an anonymous referee for bringing this to our attention.) As mentioned above, CPRT's work is a refinement of DTV's formulation. Like CPRT, controllers in DTV are in the form of state-action tables. However, DTV do not introduce terminal states, which, as discussed above, are states "where the execution stops" [25]. Indeed, in DTV, execution paths are "infinite sequences of states" (as opposed to "possibly infinite" in CPRT), and moreover, "when the sequence reaches a goal state, the execution is extended with an infinite sequence of the same goal state." Thus, when DTV characterize planning solutions, they are simply expressed in terms of execution paths and goal satisfaction. For example, *weak* planning is the notion that there is at least one history that enables the goal. (In temporal-logic parlance [27], there exists at least one path where the goal is eventually true.)

However, notice that the termination in CPRT and our framework refers to the controller deciding to stop, and not that the environment "halts." Indeed, if we think of an open-ended general-purpose agent, after one controller has enabled the goal for a particular planning problem, a second planning problem might become relevant and an appropriate controller would be needed to achieve the goals of this latter problem. Put differently, transitions to states can continue to happen even after the first controller has stopped as long as the agent continues to act. Thus, the mention of termination in notions like **PC** and **TER** pertain only to the current controller deciding that it is done.

Given this, the natural question to ponder is whether there is any value is further decoupling terminal states from our framework. For example, **PC** could be defined as (roughly) saying: every run σ can be extended to a run σ' that is goal enabling, and moreover, every extension to σ' continues to be goal enabling. However, we think this is needlessly cumbersome, and moreover, it is unclear what we gain by not stopping the controller once the goal is achieved. For these reasons, we base our results on CPRT's reformulation as their terminal states map directly to our controller advising stop.

Note that there may be applications where we wish to allow goals to be fulfilled partially, or to suggest that the goal be true at some point and every point after that. This clearly demands a framework for temporally-extended goals [2], which is an orthogonal concern.

question for further investigation is whether simple structural stipulations for infinite domains exist that immediately imply **BND**-adequacy (such as **ACYC** in finite domains).

Overall, our analysis has provided a more detailed perspective on such correctness notions in the literature. We now turn to related work.

8. Related work and discussions

To properly disambiguate our contributions from related efforts in automated planning and decision-theoretic modeling, let briefly recap the main contributions of our work:

1. We provided a generalization to Levesque's T^* semantics, so that we could formalize an integrated view of goal satisfaction and termination with nondeterministic actions via U^*. We referred to this by **ONE**.
2. This motivated a delineation to **PC** and **TER**, and further to **BND** and **ACYC** as ways of ensuring finite termination. These latter notions and those discussed below were then formulated in an abstract framework for broader applicability.
3. We then considered probabilistic analogues of **PC** and **TER**, and in the presence of continuous noise, identified that categorical goals are not satisfiable in some domains.
4. We were able to recover the generality of BHL, and go beyond when considering unconstrained generalized planning problems with distribution uncertainty.
5. We related our results to the notion of goal achievability, as well as weak, strong and strong cyclic planning, and found that our categorical adequacy notions are more granular. Furthermore, our results were also explored for unbounded spaces.

Let us position these results against related efforts by grouping the latter into two broad camps: *generalized planning,* and *knowledge-level planning,* with the latter including *decision-theoretic planning.* At the outset, it will be clear that our contributions are closest in spirit to the efforts in generalized planning: we follow the underlying formulation in that area in maintaining a distinct set of goal states, and a solution either satisfies the goal or does not. Even when considering the probabilistic analogue, we do expect individual runs to satisfy goals and so the expression is simply capturing the ratio of the runs that satisfy goals versus the ones that do not. As argued by Bonet and Geffner [15], such formulations are both "simpler and crisper:" the formulation in decision-theoretic planning is based on rewards rather than goals, and the general thrust is in terms of either preserving optimality or approximating reward.

8.1. Generalized planning

Generalizing plans have been of interest since the early days of planning [30]. Algorithmic proposals to synthesize plans that generalize varied widely in methodology, ranging from interactive theorem proving [72] to learning from examples [75].[15] Our interest here is mainly about the accounts of correctness that apply to this area. Nonetheless, we provide a short summary of some of the underlying algorithmic ideas subsequently, as they have inspired recent work on computing plans against probabilistic adequacy.

In the presence of sensing, perhaps the first formal account on the correctness of recursive plans is that of Levesque [52]. In later work, Lin and Levesque [53] adapted this account to define goal achievability. The general thrust in this and many other works in generalized planning is essentially the same [70,17,38,41]: define the generalized planning problem as a set of basic planning problems differing only in their initial state, and a correct controller is one such that for every initial state, it leads to goal satisfaction and termination. We showed in this article that these notions in noise-free domains simply corresponds to obtaining **ONE**-adequate controllers for a generalized planning problem. In terms of a logical formalization, the T^*-semantics from Levesque [52] was extended to U^* in this work.

To address nondeterministic finite systems, [27] proposed the notions of *weak, strong* and *strong cyclic* solutions, a reformulation of which was the focus of CPRT [25].[16] These notions are widely used in the planning community [15]; see, for example, Bertoli et al. [11] for an account of strong planning with a sensor model. Recently, Srivastava et al. [71] consider the synthesis of loopy plans in domains with nondeterministic quantitative effects, for which strong cyclic solutions are considered as well. As shown in our work, these notions can be studied in a more granular manner with **BND**, **ACYC**, **TER**, **PC** and **ONE**, which becomes relevant in infinite domains, for example, where {**ACYC**, **PC**}-controllers do not imply **BND**, thereby showing an issue with the definition of strong planning.

At a conceptual level, based on the early work by CPRT, it is perhaps obvious that Levesque's notion (essentially **ONE**) is unappealing with nondeterministic actions, as it immediately corresponds to *weak* planning. (The name insinuates that it is not correct or strong.) However, where our contributions even in that regard are different pertain to the decoupling of goal satisfaction and termination: **PC** and **TER** served as primitives for all further analysis. On the one hand, we showed that

[15] For a review of sequential and conditional planning, we refer readers to treatments such as Geffner and Bonet [31].

[16] Variant additional stipulations in the literature include notions such as *fairness*, where every outcome of a nondeterministic action must occur infinitely often [15,28].

what CPRT suggested to be more appropriate was a combination of **PC** with **TER** or **BND**. On the other hand, by showing the equivalency of {**PC**, **TER**} to **ONE** in noise-free domains (and indeed, the equivalency of {**PC**, **BND**} to **ONE** in noise-free domains), we established a type of downward compatibility. Indeed, weak and strong cyclic are essentially equivalent to Levesque's notion in noise-free domains. Such insights, to the best of our knowledge, were not studied previously in a formal manner in the generalized planning literature. A full probabilistic analysis of those concepts was also lacking in that literature.

By revisiting Levesque's notion in an abstract framework, what was also gained is a more abstract treatment of the underlying principles without needing the use of a logical language. Nonetheless, we showed that handling of BHL's generality was still feasible. In particular, handling probabilities on initial states to stipulate correctness notions such as $\mathbf{GEQ}_\kappa$ and $\mathbf{BEL}_\kappa$ could be formulated, as could the handling of multiple distributions.

An additional layer of generality was in terms of how generalized planning was formulated: we remained consistent with the BHL account to suggest that the basic planning problems contained in a generalized planning problem differed only in the initial states. However, as discussed with the notion of an unconstrained generalized planning problem, nothing hinges on this restriction: adequacy in a generalized problem was simply reduced to adequacy for all the problems contained. Interestingly, some recent work [37,16,41] argue that expecting the contained basic planning problems to differ only in terms of initial states precludes relational planning domains where actions and objects change across instances. The notion of an unconstrained generalized planning problem thus admits such flexibility, allowing the contained problems to differ arbitrarily.

We reiterate that our results are closest in spirit to the above family of efforts on generalized planning. Although we discuss related threads of research below from knowledge-level planning and decision-theoretic modeling, the connection between those accounts and our framework could be equally positioned in terms of its connection to the above efforts on generalized planning.

8.2. Algorithms for generalized planning

While our interest in this work was purely related to formalization and analysis, we briefly review some of the underlying algorithmic ideas below. Early approaches to loopy plans can be seen as deductive methodologies, often influenced by program synthesis and correctness [35]. Manna and Waldinger [55] obtained recursive plans by matching induction rules, and Stephan and Biundo [72] refine generic plan specifications, but required input from humans. See Magnusson and Doherty [54] for a recent approach using induction. Most recent proposals differ considerably from this early work using deduction. For example:

- Levesque [51] expects two parameters with the planning problem; the approach plans for the first parameter, winds it to form loops and tests it for the second.
- Winner and Veloso [75] synthesize a plan sequence with partial orderings, and exploit repeated occurrences of sub-plans to obtain loops.
- Srivastava [70] considers an abstract state representation that groups objects into equivalences classes, the idea being that any concrete plan can be abstracted wrt these classes and repeated occurrences of sub-plans can be leveraged to generate compact loopy plans.
- Bonet et al. [17], Bonet and Geffner [15] integrate the dynamics of a reactive plan with a planning problem, and convert that to a conformant planning problem; the solution to this latter problem is shown to generalize to multiple instances of the original problem.
- Hu and De Giacomo [38] propose a bounded AND/OR search procedure that is able to synthesize loopy plans.

From the perspective of an algorithmic schema, the generic technique of Hu and De Giacomo [38] (HD henceforth) is perhaps the simplest to analyze. Here, an environment virtually identical to ours is assumed, and the transition relation is also nondeterministic (but non-probabilistic). Initially, the algorithm starts with the empty controller, at the initial controller state and the initial states of a generalized planning problem. The AND-step enumerates the outcomes of an action from a given combined state and history, and calls OR-step to synthesize a controller that is correct for every outcome. The OR-step function enumerates the extensions of a controller for the current controller state and observation. It thus selects a next action for the current observation, and then calls AND-step to test for correctness recursively on the outcomes of the chosen action. So, the procedure is essentially a blind search in controller space, reverting to the last non-deterministic choice point when a branch fails. The algorithm is shown to be both sound and complete.

In recent work [74], we show that this algorithm, along with its soundness and completeness proofs, can be lifted to **LTER** and **LPC** in discrete, finite problems. It should be clear that by ignoring probabilities, we would immediately obtain (**LTER** $= 1$)-adequate (or **LPC** $= 1$ analogously) controllers by reducing the case to HD. What is particularly tricky is adapting the algorithm to yield (**LTER** $\geq \lambda$)-adequate controllers for $\lambda \neq 1$. Our results show that this is possible indeed. To emphasize why $\lambda \neq 1$ are needed at all, we discuss the so-called *bridge walk* domain. It imagines an agent that stands on the handrail of a bridge. On one side is a river, and if the agent stays on the handrail then taking n steps brings the agent to the goal. However, with every step taken on the handrail, the agent has a 0.1 probability of falling into the river (which is an absorbing state, i.e., it only has state transitions to itself), and 0.9 probability of moving forward by one step. A simple

instance of this problem involving coin tosses has been discussed earlier. For $\mathbf{LPC} = .9^n$, clearly a controller that loops on the noisy action satisfies the goal. This is exactly what the algorithm synthesizes. Without any means to recover from falling in the river, it is clear that obtaining a ($\mathbf{LPC} > .9^n$)-adequate controller is not possible.

8.3. Knowledge-level planning: goal satisfaction

As discussed previously, generalized planning has an epistemic flavor to it, in that we are finding a plan structure that applies to all initial states considered epistemically possible by the agent. Thus, the area of knowledge-level planning, which treats the set of initial states as representing the beliefs of the agent, and monitors how these beliefs change as a result of physical and sensing actions is closely related [14].

We will first consider those accounts with a distinguished goal state, as opposed to those involving rewards that need to be maximized, which are considered subsequently.

Broadly, with knowledge-level planning, one imagines that by sensing actions, the number of states considered epistemically possible reduces (i.e., we have knowledge expansion), and the controller attempts to reach a goal belief state. When solutions are computed offline, its worth noting that sensing with deterministic actions in the real world then becomes a nondeterministic action at the level of beliefs: because based on what was observed, we get two distinct belief states (with binary sensing) [67]. As we have suggested earlier, plans with loops can behave like conditional plans for a large (possibly infinite) number of problem instances. See, for example, Sardiña et al. [67] for an account of knowledge-level (iterative) planning. See also Jiménez et al. [41] for discussions on the areas of planning with belief states and generalized planning, and observations about some cross-overs in terms of algorithmic strategies between the areas. What we have contributed here is an analysis of generalized planning correctness in terms of termination, goal satisfaction and their probabilistic analogues, which is related but different in thrust to the work in knowledge-level planning. Indeed, if we took the view that the states an environment $\mathcal{E}$ are atomic belief states, then our results would relate termination and goal satisfaction at the level of beliefs.

It is worth noting that the notion of goal achievability, which we have related to in terms of adequacy, can be seen as a precursor for reasoning about agent abilities [50]: to be able to realize a goal, the agent needs to know that there is a plan that is adequate for the goal; see Sardiña et al. [67] and Lin and Levesque [53] for discussions.

When dealing with beliefs, plan structures can themselves be more involved. For example, knowledge-based programs [63,50] and their stochastic extensions [6,49] allow belief formulas to be represented as test actions (as in, whether an agent knows a formula), based on which the program can branch or iterate [6]. On the one hand, the results by Lang and Zanuttini [49] indicate that such knowledge-based programs are likely exponentially more succinct than reactive plans. They may also be more understandable when communicating strategies to other agents. But on the other hand, such programs are more cumbersome to execute as they need to query the agent's beliefs. (In fact, we need to consider epistemic feasibility [50].) Thus, from a knowledge representation viewpoint, the question of which representation is more appropriate for an application context is an interesting one.

Putting aside the expressiveness of plan representations, the main benefit of knowledge-level accounts is that we can also address correctness with noisy sensing. Indeed, as sensors only affect the belief state, monitoring how that changes is relevant for stipulating that the goal belief state has been reached. A preliminary investigation on how T^* could be generalized to belief states has been considered recently [3], but a deeper investigation on whether such an account also allows a delineation of correctness in terms of goal satisfaction versus termination, or even probabilistic adequacy still needs to be considered. Such inquiries might bring the areas of generalized planning and knowledge-level planning closer. Interestingly, Chatterjee and Chmelík [21] discuss a strategy for reducing noisy observations to deterministic observations by extending the state space with observations. While such a move is not possible in continuous domains in general, exploring such a reduction to further position our results against an account of goal satisfaction and termination at the level of beliefs would be interesting as well.

8.4. Knowledge-level planning: rewards

Markov decision processes [62,46] are widely used in AI. To connect the threads in this area to our work, we further segment this section as follows. We first state some general background to that area. Then we draw connections to our notions of correctness. Finally, we motivate what a reward-based notion might look like in our framework.

8.4.1. Background

There is extensive work on synthesizing plans in stochastic domains modeled as Markov Decision Processes (MDPs) and partially observable MDPs (POMDPs) [61,33,60,48,22,42]. MDPs assume the states are observable (so the observation function can be ignored), whereas POMDPs assume that the actual state is not observable directly and a subjective probability distribution captures the agent's uncertainty about where it is. Both these models, classically, assume a reward function that rewards the agent with a real-valued quantity for doing a in s. Conversely, the agent could be paying a cost for doing an action. A solution is a controller that maps states to actions such that the cumulative reward is maximized. Since this reward has to handle stochastic transitions between states, the solution is attempting to maximize the expected reward. Such a controller can also involve loops [61,21].

8.4.2. Relation to generalized planning

Our controller framework and the ones most closely related to it [37] can be related to such models in the following way [15]: the framework can be seen as a POMDP specification but where sensing in exact, a distinguished set of goal states have to be reached, and where the initial uncertainty is expressed as a set of states rather than a distribution over those states.[17] A *proper policy* is one where the goal state is reached with certainty regardless of the initial state, and only in this setting, the distinction in terms of the initial uncertainty can be ignored. It is easy to see that given a generalized planning problem consisting of basic problems differing only in their initial states, a proper policy would correspond to (**LPC** $= 1$)-adequate controller provided the controller is *Markovian*. That is, the controller should advise actions only based on the current state, or more precisely, for a history σ, $\mathcal{C}(sensed(\sigma)) = \mathcal{C}(sensed(end(\sigma)))$.

Fundamentally, the notion of correctness boils down to maximizing the expected reward. (Moreover, a majority of algorithms either solve an approximation of the problem, or they come without correctness guarantees; see Treszkai and Belle [74] for discussions.) A representative example of work on correctness is perhaps that of Chatterjee and Chmelík [21], and we briefly discuss that work to show how our contributions differ. They consider a planning problem that essentially corresponds to a discrete, finite, noisy basic problem in our framework, but with rewards and a distinguished initial state. It is then shown that despite considering a distinguished initial state, they can capture a distribution on initial states by means of a dummy initial state that transitions to successor state according to the initial distribution. They look at runs (analogous to our notion) from the initial state, and consider the average reward over the set of all (possibly infinite) runs. (The reward function is assumed to be $\mathbb{R}^{[0,1]}$ for simplicity.) In particular, they consider the setting where the average reward is precisely 1 versus greater than or equal to some quantity $\lambda \in (0, 1)$, and discuss synthesis algorithms for such setting.

What we offer in this work is first separating termination from goal satisfaction, their relationships and probabilistic variants. (Indeed, there is no notion of termination per se in that work [21], but just a quantity to denote the accumulated reward over possibly unbounded runs. The work by Hansen [34], for example, discusses the implications of such considerations in decision-theoretic models.) We also stipulate the correctness of the controller for a generalized planning problem as being a correct solution for all basic instances contained within the problem. Thus, the thrust of our work and the insights offered are fairly orthogonal.

A different way to position that result from ours is to think of insisting on, say, {**LTER** $\geq \lambda_1$, **AVG** $\geq \lambda_2$}-adequate controllers for a given problem, where by **AVG** $\geq \lambda_2$ we mean to say the controller achieves an average reward $\geq \lambda_2$. (Here, let $\lambda_i \in [0, 1]$.) If we wanted to additionally consider a set of goal states, we might then insist on {**LTER** $\geq \lambda_1$, **AVG** $\geq \lambda_2$, **LPC** $\geq \lambda_3$}-adequate controllers.

We finally note that controllers in the decision-theoretic literature have also be studied to be stochastic. These are primarily useful for optimizing rewards when viewing a POMDP as a continuous space MDP [19]. Since our work purely focuses on the goal satisfaction and termination of iterative plans, allowing controllers to also be stochastic would undoubtedly make the treatment more involved. Thus, there is little to be gained by moving beyond the current definition for our purposes.

8.4.3. Rewards and adequacy

As hinted above, one way to incorporate results from the decision-theoretic literature is to consider reward-based adequacy notions that in the first instance can be stipulated alongside termination (and/or goal satisfaction). We present an example formulation here, and leave more detailed analysis and other variants for the future.

There are a variety of models [12], and we develop the formulation for a discrete, finite, noisy basic problem, as related to the work of Chatterjee and Chmelík [21] above. Since we want rewards to co-exist with goals, we will allow a distinguished set of goal states as usual, but in addition, we will consider a cost function $\mathcal{V} \in [\mathcal{S} \times \mathcal{A} \to \mathbb{R}_{[0,\infty)}]$, that is, for taking action $\alpha \in \mathcal{A}$ at state s, the agent pays $\mathcal{V}(s, \alpha)$. Thus, the basic problem is one of the sort $\langle s_0, \mathcal{E}, \mathcal{G}\rangle$, where the environment is discrete, finite, and noisy: $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \Delta, \Omega, \mathcal{V}\rangle$ and is now additionally augmented with the cost function. (Correspondingly, a generalized planning problem is a set of such basic problems. Also, the cost function being part of the environment means that this is a standard cost determined by the environment for every basic problem, but alternative formulations are also possible.)

With this setup, all of the previous adequacy notions are immediately definable. But what is of particular interest are the costs of plans, leading to a cost-sensitive adequacy criteria. So, for any run σ, let $cost(\sigma) = 0$ if σ has length ≤ 1. For $\sigma = \sigma' \cdot s$, where σ' has non-zero length, we define:

$$cost(\sigma) = cost(\sigma') + \mathcal{V}(end(\sigma'), \mathcal{C}(sensed(\sigma'))).$$

That is, the cost of σ is the cost of σ' plus $\mathcal{V}(\alpha, s')$ with the understanding that the action $\alpha = \mathcal{C}(sensed(\sigma'))$ was performed at $s' = end(\sigma')$ leading the agent to s, possibly nondeterministically. It is not hard to see that if actions have unit costs, then $cost(\sigma)$ is simply 1 less than the length of σ. That is, $cost(s_0) = 0$, $cost(s_0 \cdot s) = 1$, and so on.

[17] It is possible to capture the full definition of POMDPs too, of course. Notions like $\mathbf{BEL}_\kappa$ also allow distributions on initial states, and noisy sensing can perhaps be addressed via clever encodings [21].

Given a basic problem with s_0 as its initial state, the expected cost would be defined as:

$$\sum_{\{\sigma \mid \sigma \text{ is a terminating run}\}} cost(\sigma) \times l(\sigma) \qquad (\textbf{LCOST})$$

(We consider discrete problems for simplicity.) When we consider terminating runs that also reach goal states, we can define the expected cost for goal satisfying runs:

$$\sum_{\{\sigma \mid \sigma \text{ is a terminating run and } end(\sigma) \in \mathcal{G}\}} cost(\sigma) \times l(\sigma) \qquad (\textbf{LPCOST})$$

That is, **LCOST** is the cost based on the likelihood of terminating runs, and **LPCOST** also takes goal satisfaction (in other words, partial correctness) into account.

It is not hard to see that these quantities do not imply **TER** since it is defined purely for terminating runs.

Proposition 58. *Suppose* (**LCOST** $\leq \lambda$) *holds for some* $\lambda \in \mathbb{N}$*, then this does not imply that* **TER** *holds. Analogously, suppose* (**LPCOST** $\leq \lambda$)*, then this also does not imply that* **TER** *holds.*

Indeed, consider any system where Γ is the set of runs and $\Gamma' \subsetneq \Gamma$ is set of terminating runs (that is, $\Gamma - \Gamma'$ is not empty). Suppose every run in Γ' also reaches a goal state. Then the quantity **LPCOST** would be obtained from Γ' and supposing it is $\leq \lambda$. However, since $\Gamma - \Gamma'$ is not empty, there are runs that do not terminate. So **TER** does not hold.

What this means for us is that when looking for controllers, along with cost-optimal ones, we may also want to insist on adequacy wrt termination (or even bounded termination). That is, we could stipulate that we require $\{\textbf{TER}, \textbf{LCOST} \leq \lambda\}$-adequate controllers, which are controllers where **TER** holds but also the expected cost is $\leq \lambda$. In other words, we get provably terminating controllers that maximize the reward or minimize the cost. By extension, a $\{\textbf{TER}, \textbf{LCOST} \leq \lambda\}$-adequate controller for a generalized planning problem would mean that the controller is $\{\textbf{TER}, \textbf{LCOST} \leq \lambda\}$-adequate for each of the contained basic problems. Analogously, a $\{\textbf{BND}, \textbf{LCOST} \leq \lambda\}$-adequate controller is like a $\{\textbf{TER}, \textbf{LCOST} \leq \lambda\}$-adequate controller except that the terminating runs are additionally bounded in length.

Incidentally, there is extensive research on synthesizing plans for **LCOST** and **LPCOST**-specific objectives [76]. How these algorithms can be extended to reason about **TER** and **PC**, or conversely, how sound and complete algorithms for $\{\textbf{PC}, \textbf{TER}\}$-adequate controllers for generalized planning problems [38,74] may be further extended to optimize rewards remains to be seen.

9. Conclusions

Beginning with Levesque's notion for the correctness of expressive plans in the presence of incomplete knowledge in the situation calculus, we uncovered a rich and exciting landscape for stochastic nondeterminism. We delineated termination and goal satisfaction, which led to a number of insights on their relationships and their probabilistic variants. Overall, to the best of our knowledge, the relationships between various kinds of termination and goal-based correctness criteria in nondeterministic (possibly unbounded) domains, and their probabilistic and continuous versions, have not been discussed in a formal and general way elsewhere.

Interestingly, like in Levesque's original proposal [52], one can motivate a generic planning procedure as follows. Given a generalized planning problem $\overline{\mathcal{P}}$ and an adequacy criteria θ, we can provide the following procedure for effective controllers:

```
PLANNING PROCEDURE(P̄, θ) {
    REPEAT with C ∈ CONTROLLERS {
        IF C is θ-adequate for P̄
    THEN return C } }
```

Naturally, this is not an algorithm to be used in practice. Existing algorithms for generalized planning are the most promising to adapt to stochastic settings, and as mentioned previously, in recent work [74], we explore an extension to the bounded AND/OR search from Hu and De Giacomo [38]. For discrete finite problems, the algorithm can be shown to be sound and complete. Extending that approach to infinite and unbounded domains, possibly by leveraging abstraction techniques [71,18,40], is an worthwhile direction for the future. In addition to that, there are many other challenging directions. The most fruitful is perhaps seeing how we can relate adequacy, decision-theoretic planning and knowledge-level planning on the algorithmic front towards the synthesis of rigorous solutions for generalized planning problems.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] F. Bacchus, J.Y. Halpern, H.J. Levesque, Reasoning about noisy sensors and effectors in the situation calculus, Artif. Intell. 111 (1–2) (1999) 171–208.
[2] J.A. Baier, S.A. McIlraith, Planning with first-order temporally extended goals using heuristic search, in: AAAI, 2006.
[3] V. Belle, On plans with loops and noise, in: Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems, International Foundation for Autonomous Agents and Multiagent Systems, 2018, pp. 1310–1317.
[4] V. Belle, H. Levesque, Foundations for generalized planning in unbounded stochastic domains, in: KR, 2016.
[5] V. Belle, H.J. Levesque, Reasoning about continuous uncertainty in the situation calculus, in: Proc. IJCAI, 2013.
[6] V. Belle, H.J. Levesque, Allegro: belief-based programming in stochastic dynamical domains, in: IJCAI, 2015.
[7] V. Belle, H.J. Levesque, A logical theory of localization, in: Studia Logica, 2015.
[8] V. Belle, H.J. Levesque, Robot location estimation in the situation calculus, J. Appl. Log. 13 (4) (2015) 397–413.
[9] V. Belle, H.J. Levesque, A logical theory of localization, Stud. Log. 104 (4) (2016) 741–772.
[10] V. Belle, H.J. Levesque, Reasoning about discrete and continuous noisy sensors and effectors in dynamical systems, Artif. Intell. 262 (2018) 189–221.
[11] P. Bertoli, A. Cimatti, M. Roveri, P. Traverso, Strong planning under partial observability, Artif. Intell. (ISSN 0004-3702) 170 (4–5) (2006) 337–384, https://doi.org/10.1016/j.artint.2006.01.004, http://www.sciencedirect.com/science/article/pii/S0004370206000075.
[12] D.P. Bertsekas, Dynamic Programming and Optimal Control, vol. I and vol. II, second ed., Athena Scientific, Belmont, MA, 2001.
[13] P. Billingsley, Probability and Measure, third edition, Wiley-Interscience, Apr. 1995.
[14] B. Bonet, H. Geffner, Planning with incomplete information as heuristic search in belief space, in: AIPS, 2000, pp. 52–61.
[15] B. Bonet, H. Geffner, Policies that generalize: solving many planning problems with the same policy, in: IJCAI, 2015, http://ijcai.org/papers15/Abstracts/IJCAI15-396.html.
[16] B. Bonet, H. Geffner, Features, projections, and representation change for generalized planning, in: Proceedings of the 27th International Joint Conference on Artificial Intelligence, 2018, pp. 4667–4673.
[17] B. Bonet, H. Palacios, H. Geffner, Automatic derivation of memoryless policies and finite-state controllers using classical planners, in: ICAPS, 2009.
[18] B. Bonet, G. De Giacomo, H. Geffner, S. Rubin, Generalized planning: non-deterministic abstractions and trajectory constraints, in: IJCAI, 2017, pp. 873–879.
[19] D. Braziunas, POMDP solution methods: a survey, Technical report, Department of Computer Science, University of Toronto, 2003.
[20] A. Camacho, C. Muise, A. Ganeshen, S.A. McIlraith, From fond to probabilistic planning: guiding search for quality policies, in: Workshop on Heuristic Search and Domain Independent Planning, ICAPS, 2015, http://www.haz.ca/papers/camacho-hsdip15-probprp.pdf.
[21] K. Chatterjee, M. Chmelík, Pomdps under probabilistic semantics, Artif. Intell. 221 (2015) 46–72.
[22] K. Chatterjee, M. Chmelík, J. Davies, A symbolic sat-based algorithm for almost-sure reachability with small strategies in pomdps, in: Thirtieth AAAI Conference on Artificial Intelligence, 2016.
[23] A. Cimatti, M. Roveri, P. Traverso, Automatic obdd-based generation of universal plans in non-deterministic domains, in: AAAI/IAAI, 1998, pp. 875–881.
[24] A. Cimatti, M. Roveri, P. Traverso, Strong planning in non-deterministic domains via model checking, in: AIPS, vol. 98, 1998, pp. 36–43.
[25] A. Cimatti, M. Pistore, M. Roveri, P. Traverso, Weak, strong, and strong cyclic planning via symbolic model checking, Artif. Intell. 147 (1–2) (2003) 35–84.
[26] B. Cook, Automatically proving program termination, in: CAV, 2007, p. 1.
[27] M. Daniele, P. Traverso, M.Y. Vardi, Strong cyclic planning revisited, in: European Conference on Planning, Springer, 1999, pp. 35–48.
[28] N. D'Ippolito, N. Rodrıguez, S. Sardina, Fully observable non-deterministic planning as assumption-based reactive synthesis, J. Artif. Intell. Res. 61 (2018) 593–621.
[29] R. Fagin, J.Y. Halpern, Y. Moses, M.Y. Vardi, Reasoning About Knowledge, MIT Press, ISBN 0262061627, 1995.
[30] R. Fikes, P. Hart, N. Nilsson, Learning and executing generalized robot plans, Artif. Intell. 3 (1972) 251–288.
[31] H. Geffner, B. Bonet, A Concise Introduction to Models and Methods for Automated Planning, Morgan and Claypool Publishers, 2013.
[32] M. Ghallab, D. Nau, P. Traverso, Automated Planning: Theory and Practice, Morgan Kaufmann Publishers, 2004.
[33] C. Guestrin, D. Koller, C. Gearhart, N. Kanodia, Generalizing plans to new environments in relational MDPs, in: IJCAI, 2003.
[34] E.A. Hansen, Indefinite-horizon pomdps with action-based termination, in: AAAI, 2007, pp. 1237–1242.
[35] C.A.R. Hoare, An axiomatic basis for computer programming, Commun. ACM 12 (10) (1969) 576–580, https://doi.org/10.1145/363235.363259, http://doi.acm.org/10.1145/363235.363259.
[36] J.E. Hopcroft, J.D. Ullman, Introduction to Automata Theory, Languages, and Computation, Addison-Wesley Publishing Company, 1979.
[37] Y. Hu, G. De Giacomo, Generalized planning: synthesizing plans that work for multiple environments, in: Twenty-Second International Joint Conference on Artificial Intelligence, 2011.
[38] Y. Hu, G. De Giacomo, A generic technique for synthesizing bounded finite-state controllers, in: ICAPS, 2013.
[39] Y. Hu, H. Levesque, Planning with loops: some new results, in: ICAPS Workshop on Generalized Planning, 2009, p. 37.
[40] Y. Hu, H.J. Levesque, A correctness result for reasoning about one-dimensional planning problems, in: IJCAI, 2011, pp. 2638–2643.
[41] S. Jiménez, J. Segovia-Aguas, A. Jonsson, A review of generalized planning, Knowl. Eng. Rev. 34 (2019).
[42] S. Junges, N. Jansen, R. Wimmer, T. Quatmann, L. Winterer, J. Katoen, B. Becker, Finite-state controllers of pomdps via parameter synthesis, in: Proceedings of the UAI, 2018.
[43] L.P. Kaelbling, T. Lozano-Pérez, Integrated task and motion planning in belief space. I, Int. J. Robot. Res. 32 (9–10) (2013) 1194–1227.
[44] L.P. Kaelbling, M.L. Littman, A.R. Cassandra, Planning and acting in partially observable stochastic domains, Artif. Intell. (ISSN 0004-3702) 101 (1–2) (1998) 99–134, https://doi.org/10.1016/S0004-3702(98)00023-X, http://www.sciencedirect.com/science/article/pii/S000437029800023X.
[45] R.A. Knepper, M.T. Mason, Realtime Informed Path Sampling for Motion Planning Search, Springer International Publishing, Cham, ISBN 978-3-319-29363-9, 2017, pp. 401–417.
[46] A. Kolobov, Mausam, Planning with Markov decision processes: an AI perspective, Synth. Lect. Artif. Intell. Mach. Learn. 6 (1) (2012) 1–210.
[47] F. Kominis, H. Geffner, Beliefs in multiagent planning: from one agent to many, in: ICAPS, 2015, pp. 147–155, http://www.aaai.org/ocs/index.php/ICAPS/ICAPS15/paper/view/10617.
[48] A. Kumar, S. Zilberstein, History-based controller design and optimization for partially observable mdps, in: ICAPS, 2015.
[49] J. Lang, B. Zanuttini, Probabilistic knowledge-based programs, in: Twenty-Fourth International Joint Conference on Artificial Intelligence, 2015.
[50] Y. Lespérance, H.J. Levesque, F. Lin, R.B. Scherl, Ability and knowing how in the situation calculus, Stud. Log. 66 (1) (2000) 165–186.
[51] H. Levesque, Planning with loops, in: Proc. IJCAI, 2005, pp. 509–515.
[52] H.J. Levesque, What is planning in the presence of sensing?, in: Proc. AAAI/IAAI, 1996, pp. 1139–1146.
[53] F. Lin, H.J. Levesque, What robots can do: robot programs and effective achievability, Artif. Intell. 101 (1–2) (1998) 201–226.
[54] M. Magnusson, P. Doherty, Deductive planning with inductive loops, in: KR, 2008, pp. 528–534, http://www.aaai.org/Library/KR/2008/kr08-051.php.
[55] Z. Manna, R.J. Waldinger, A deductive approach to program synthesis, ACM Trans. Program. Lang. Syst. 2 (1) (1980) 90–121, https://doi.org/10.1145/357084.357090, http://doi.acm.org/10.1145/357084.357090.
[56] B. Marthi, S.J. Russell, J.A. Wolfe, Angelic hierarchical planning: optimal and online algorithms, in: ICAPS, 2008, pp. 222–231.

[57] M.J. Matarić, The Robotics Primer, Mit Press, 2007.
[58] J. McCarthy, P.J. Hayes, Some philosophical problems from the standpoint of artificial intelligence, in: Machine Intelligence, 1969, pp. 463–502.
[59] C. Muise, V. Belle, P. Felli, S. McIlraith, T. Miller, A. Pearce, L. Sonenberg, Planning over multi-agent epistemic states: a classical planning approach, in: Proc. AAAI, 2015.
[60] J.K. Pajarinen, J. Peltonen, Periodic finite state controllers for efficient pomdp and dec-pomdp planning, in: NIPS, 2011, pp. 2636–2644.
[61] P. Poupart, C. Boutilier, Bounded finite state controllers, in: NIPS, 2004, pp. 823–830.
[62] M.L. Puterman, Markov Decision Processes: Discrete Stochastic Dynamic Programming, 1st edition, John Wiley & Sons, Inc., New York, NY, USA, 1994.
[63] R. Reiter, On knowledge-based programming with sensing in the situation calculus, ACM Trans. Comput. Log. 2 (4) (2001) 433–457.
[64] R. Reiter, Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems, MIT Press, 2001.
[65] H. Rogers Jr., Theory of Recursive Functions and Effective Computability, The MIT Press, ISBN 0262680521, 1987.
[66] S. Sardina, G. De Giacomo, Y. Lespérance, H.J. Levesque, On the semantics of deliberation in indigolog—from theory to implementation, Ann. Math. Artif. Intell. 41 (2–4) (2004) 259–299.
[67] S. Sardiña, G. De Giacomo, Y. Lespérance, H.J. Levesque, On the limits of planning over belief states under strict uncertainty, in: KR, 2006, pp. 463–471.
[68] R.B. Scherl, H.J. Levesque, Knowledge, action, and the frame problem, Artif. Intell. (ISSN 0004-3702) 144 (1–2) (2003) 1–39.
[69] T. Siméon, J. Laumond, J. Cortés, A. Sahbani, Manipulation planning with probabilistic roadmaps, Int. J. Robot. Res. 23 (7–8) (2004) 729–746, https://doi.org/10.1177/0278364904045471.
[70] S. Srivastava, Foundations and Applications of Generalized Planning, PhD thesis, Department of Computer Science, University of Massachusetts Amherst, 2010.
[71] S. Srivastava, S. Zilberstein, A. Gupta, P. Abbeel, S. Russell, Tractability of planning with loops, in: AAAI, 2015.
[72] W. Stephan, S. Biundo, Deduction-based refinement planning, in: AIPS, 1996, pp. 213–220, http://www.aaai.org/Library/AIPS/1996/aips96-027.php.
[73] S. Thrun, W. Burgard, D. Fox, Probabilistic Robotics, MIT Press, 2005.
[74] L. Treszkai, V. Belle, A correctness result for synthesizing plans with loops in stochastic domains, Int. J. Approx. Reason. 119 (2020) 92–107.
[75] E. Winner, M.M. Veloso, LoopDISTILL: learning domain-specific planners from example plans, in: Workshop on AI Planning and Learning, ICAPS, 2007.
[76] H. Xu, S. Mannor, Probabilistic goal Markov decision processes, in: Twenty-Second International Joint Conference on Artificial Intelligence, 2011.