Hypothetical Planning

Tamara Babaian
CIS Department, Bentley College,
Waltham, MA 02452
tbabaian@bentley.edu

Abstract

We present a novel method for interleaving planning with execution, called iterative deepening in hypotheticals. The method consists of performing an iterative deepening search in the space of partially ordered hypothetical plans. Hypothetical plans are partial plans in which the achievement of an otherwise unachievable goal may be conditioned on certain outcomes of sensing. This approach has been implemented within the PSIPLAN-S framework and used in a collaborative bibliography assistant, called Writer's Aid.

1 Introduction and Motivation

A planning agent operating in a real world must often deal with domains in which only incomplete information about the domain is available and furthermore, the complete information can never be acquired due to the large number of domain individuals. In such environments, using sensing actions judiciously and effectively to discover information that is relevant, but yet unknown, becomes critical.

Given correct, but incomplete description of the initial situation, a solution plan that *provably* achieves the goal may not exist. However, it may possible to construct the solution by interleaving the process of planning with execution of some information gathering steps. The method that we describe in this paper, called *hypothetical planning*, provides a mechanism for enacting such interleaved planning with execution. It is formulated using entailment and reasoning about knowledge and ignorance and guarantees non-redundancy of information gathering in that sensing actions are carried out only when the critical information is missing.

Hypothetical plans hypothesize on the value of an unknown subgoal; by verifying a hypothesis via execution of a sensing action, the planner eventually reduces the incompleteness of the knowledge so that a solution plan is found or the goal is proven to be unsatisfiable. For example, having no information on the location of a paper, the planner may adopt a hypothesis that the paper is available from a certain collection, and verify the information by querying the collection. Hypothetical plans in addition to causal links between a subgoal and an entailing it effect, contain *hypothetical links*, which link knowledge effects to domain subgoals. The idea is as follows: if neither p, nor $\neg p$ is known to be true prior to S, and there is a sensing action a_s whose effect entails knowing the truth value of p, then by executing a_s the planner may find out that p is true. A hypothetical plan leads to a solution plan, if after verifying the hypothesis, the plan can be successfully completed, which is not guaranteed even when the hypothesis is confirmed by an observation.

An alternative approach to planning with incomplete information is conditional planning, i.e. creating branching plans based on the possible outcomes of a sensing action. Applied to the above scenario, conditional planning would involve planning ahead for each of the two possible outcomes of checking if the paper is available from the searched collection. However, in the environments with a high degree of incompleteness, planning ahead for every contingency is computationally prohibitive, especially when a sensing action involves information on multiple atoms.

Furthermore, predicting all possible outcomes of sensing in a meaningful way becomes impossible when the sensing action may discover new objects. In such situations, the agent needs to proceed with execution and then complete the plan given the observation. For example, consider the goal of removing all fragile objects from a room. Given no prior information on the contents of the room, it is impossible to predict which objects are in it, if any of them are fragile, and therefore need to be removed. Thus, it does not make sense to create plans for removing any objects until the information on the contents of the room becomes available.

Suppose that the agent operating in the room can perform the sensing action of identifying all objects in the room, and another one, determining if the object is marked as fragile. Hypothetical planning would hypothesize that by using the first action the agent may discover that no objects are inside the room, thus yielding the goal of having no fragile objects satisfied. If however, upon executing the first action some objects are found inside the room, the agent now has a choice of either creating a plan to move all discovered objects out, or first identifying which are fragile and only removing those marked as fragile. The first solution can be obtained without any further information gathering, the second solution again requires hypothetical planning and execution.

An approach to interleaving planning with execution performed by XII [Golden *et al.*, 1994] and PUCCINI [Golden, 1998] planners (both based on the approach used in IPEM [Ambros-Ingerson and Steel, 1988]) is the other alternative to hypothetical planning. This method treats execution as one of the nondeterministic choices within the planning algorithm. In hypothetical planning execution is triggered by the need, thus it is more tightly constrained, and used only when necessary. The hypothetical planner's behavior is thus not dependent on the model of nondeterminism in the planner implementation and is better suited for an application in which the time of response is critical and sensing operations may take considerable time or are otherwise costly.

Hypothetical planning has been implemented in a partial order planning [Russell and Norvig, 1995] algorithm called PSIPOP-SE and used at the core of a collaborative bibliography assistant, called Writer's Aid [Babaian *et al.*, 2002]. PSIPOP-SE extends a sound and complete open world planner PSIPOP[Babaian and Schmolze, 2000] to planning with sensing, knowledge goals and interleaved execution. It uses PSIPLAN-S representation for reasoning and planning with incomplete information, sensing and knwoeldge goals.

When the set set of agents sensing actions is rich, the use of hypothetical plans may considerably expand the search space. To limit the search space, PSIPOP-SE explores the search space gradually increasing the maximum allowed plan number of hypotheses made in supporting a subgoal. This parameter is called the *hypothetical level of a plan.* Hypothetical level of a simple plan is 0. An example of a plan with hypothetical level two is a plan that hypothesizes that a paper is available from the author's homepage, and then, having no information about the author's homepage, hypothesizes that the url for the homepage can be found from a known index. Verification of each hypothesis reduces the uncertainty, therefore the size of subspace of hypothetical plans on each consecutive level is reduced, while the lower-level hypothetical subspace is explored. In our experiments Writer's Aid was unable to explore the entire space of plans of hypothetical level up to 2 at once due to the large size of this space, but was successful at exploring subspaces gradually, starting from maximum hypothetical level of 0. We call this approach **iterative deepening in hypotheticals**

The rest of the paper is organized as follows. An overview of the PSIPLAN representation is presented in the next section. The definition of a hypothetical link and the partial order planning algorithm interleaving planning with execution PSIPOP-SE are presented in Section 3.

2 Overview of PSIPLAN

PSIPLAN assumes infinite number of domain constants, and no other function symbols. PSIPLAN propositions include **ground domain atoms**, **domain ψ-forms** and **knowledge ψ-forms**. The general form of a ψ-form is

$$[Q(\vec{x}) \text{ except } \{\sigma_1, \ldots, \sigma_n\}],$$

and it represents a possibly infinite set of ground propositions that are obtained by instantiating the formula $Q(\vec{x})$ called the *main form*, with all possible ground assignments on the variables in $\vec{x}$, except for the instances specified by the substitutions σ_i called the *exceptions*. Each σ_i is a substitution on a subset of variables of $\vec{x}$. The main form $Q(\vec{x})$ of a *domain* ψ-form is a disjunction of negated literals. In *knowledge* ψ-forms $Q(\vec{x})$ has a form $KW(P(\vec{x}))$, where $P(\vec{x})$ is a disjunction of negated literals. All variables in $\vec{x}$ are implicitly universally quantified.

The combination of **domain atoms and ψ-forms** is necessary to describe situations as the following one, in which the agent knows that

> *The only bibliographies preferred by Ed are the digital library of the ACM, and maybe the ResearchIndex database.*

In PSIPLAN-S the example statement above is expressed by stating that

1. *ACM's digital library is a preferred bibliography*, which is represented by a ground atom:
$$a = PrefBib(ACM) \tag{1}$$
2. *Nothing is a preferred bibliography except for ACM and the ResearchIndex*, which is represented by the ψ-form:
$$\psi = [\neg PrefBib(b) \text{ except } \{\{b = ACM\}, \{b = RI\}\}] \tag{2}$$
Thus, ψ denotes all ground instances of the formula $\neg PrefBib(b)$ minus two exceptions: $\neg PrefBib(ACM)$ and $\neg PrefBib(RI)$ and is equivalent to the universally quantified predicate calculus formula $\forall b. \neg PrefBib(b) \vee (b = ACM) \vee (b = RI)$

Formally, we define the set of ground propositions represented by a ψ-form as follows

1. $\phi([Q(\vec{x})]) = \{Q(\vec{x})\sigma \mid Q(\vec{x})\sigma \text{ is ground }\}$
2. $\phi([Q(\vec{x}) \text{ except } \{\sigma_1, \ldots, \sigma_n\}]) =$ $\phi([Q(\vec{x})]) - \phi([Q(\vec{x})\sigma_1]) - \ldots - \phi([Q(\vec{x})\sigma_n])$

Note that assuming infinite number of individual domain objects, a finite set of PSIPLAN-S domain propositions can represent an infinite number of ground negated clauses without the knowledge of all domain objects by the virtue of implicit universal quantification in ψ-forms. However, it can represent only finite *"positive knowledge"*, i. e. finite number of atoms.

The algorithms for reasoning with ψ-formsare not presented in this paper (see [Babaian, 2000]), however, we note that these computations are carried out by manipulations on the main form and exceptions of the ψ-forms without expanding the ψ-form into the corresponding set of ground propositions.

Knowledge ψ-forms similarly to domain ψ-forms, represent a conjunction of all ground instances of the main form, however each ground instance in this case is a *knowledge proposition*. Knowledge propositions have form $KW(p)$, where p is a ground clause and represent *knowing p or knowing* not p, i.e. that the value of a domain clause p is known without committing to a particular value. For example, $KW(PrefBib(ACM))$ represents knowing-whether ACM is a preferred bibliography. Note that $KW(p)$ is semantically equivalent to $KW(\neg p)$. However, in the main form of a ψ-form the KW-fied formula is always a negated clause, as in the knowledge ψ-form below that represents knowing the set of all preferred bibliographies.

$$\tilde{\psi} = [KW(\neg PrefBib(b))] \qquad (3)$$

Knowledge propositions in PSIPLAN-S are used to reason about knowledge and ignorance, represent information goals and results of sensing actions. For example, posted as a goal, $\tilde{\psi}$ requires knowing the value of each ground instance of $PrefBib(b)$, or in other words, knowing the set of preferred bibliographies. The effect of checking if RI is a preferred bibliography, is a knowledge proposition $KW(PrefBib(RI))$. A negated kw-proposition $\neg KW(p)$ represents ignorance about p.

Semantics

A *world state* is a truth assignment on domain atoms. $w(q)$ denotes that q is *true* in the world state w. Let $\mathcal{W}$ denote the set of all world states.

To define a model we use *k-states* of Baral and Son [Baral and Son, 2001]. A k-state is a pair (w, W), where w denotes a world state from $\mathcal{W}$, and W denotes a a set of world states. A k-state represents a knowledge state of an agent who actually being in the world state w thinks it can be in any of the world states of W.

A set of *models* is denoted by α and defined below. We are assuming that the agent's knowledge is *correct*, hence we require that for any k-state (w, W) in a model $w \in W$. In what follows, c represents a ground negated domain clause and q represents a ground domain proposition, i.e. domain atom or a ground negated clause.

1. $\alpha(q) = \{(w, W) \,|\, w \in W \wedge \forall w' \in W.\, w'(q)\}$
2. $\alpha(KW(c)) = \{(w, W) \,|\, w \in W \wedge ([\forall w' \in W.\, w'(c)] \vee [\forall w' \in W.\, w'(\neg c)])\}$
3. $\alpha(\neg KW(c)) = \{(w, W) \,|\, w \in W \wedge [\exists w' \in W.\, w'(c)] \wedge [\exists w'' \in W.\, w''(\neg c)]\}$.
4. $\alpha(\{q_1, \ldots, q_k\}) = \cap_{i=1}^{k} \alpha(q_k)$.

A set of ground propositions $q_1, \ldots, q_k$ *k-entails* (or, for brevity, entails) another ground proposition q, denoted $q_1, \ldots, q_k \models_k q$ if $\alpha(\{q_1, \ldots, q_k\}) \subseteq \alpha(q)$.

Note that according to this semantics the k-entailment of ground domain propositions is equivalent to the ordinary entailment. Furthermore,

$$q \models_k KW(q), \text{ and,} q \models_k KW(\neg q).$$

A set of models of a PSIPLAN-S proposition is defined as the set of models of the set of ground propositions it represents.

Definition 1 For a PSIPLAN-S proposition p,$\alpha(p)$ is defined as the set of models $\alpha(\phi(p))$.

Definition 2 For a set of PSIPLAN-S propositions $p_1, \ldots p_m, p$

$$p_1, \ldots p_m \models_k p \text{ if and only if } \alpha(\{p_1, \ldots, p_m\}) \subseteq \alpha(p)$$

2.1 ψ-form Entailment

While we do not have the space to present the details of the algorithms for computing entailment in PSIPLAN, we state several key properties underlying those algorithms, and illustrate them with examples. The property critical for the efficiency of ψ-form reasoning is formulated in Theorem 1 below: given a set of ψ-forms $\Psi = \{\psi_1, \ldots, \psi_n\}$, $\Psi \models_k \psi$ only if there is a ψ-form $\psi_i \in \Psi$ that *nearly entails* ψ, i.e. main part of ψ_i entails the main part of ψ, or $[\mathcal{M}(\psi_i)] \models_k [\mathcal{M}(\psi)]$.

Theorem 1 Given a set of ψ-forms $\Psi = \{\psi_1, \ldots, \psi_n\}$ and a ψ-form ψ, $\Psi \models_k \psi$ only if there is a ψ-form ψ_i in Ψ such that $[\mathcal{M}(\psi_i)] \models_k [\mathcal{M}(\psi)]$.

E-Difference For any two sets of ground propositions A and B, *e-difference* is defined as follows.

$$B \dot{-} A = \{b \,|\, b \in B \;\wedge\; A \not\models_k b\}$$

As ψ-forms are compact representations of sets of ground propositions, we extend the e-difference operation to ψ-forms. The following example illustrates the e-difference operation.

Example 1 Let ψ denote $[Kn(\neg In(R, z))$ except $\{\{z = A\}, \{z = B\}\}]$, which represents that there are no items in room R except for possibly A and B. Further, let $\tilde{\psi}$ denote $[KW(\neg In(R, x) \vee \neg Fragile(x))]$, which can represent a goal of knowing for all objects (x) if they are inside room R and also fragile. ψ entails *most* of $\tilde{\psi}$, indeed, since $\neg In(R, z)$ is *true* for all values of z except possibly A and B, then so is the disjunction inside the $\tilde{\psi}$'s KW clause. Thus, the only parts of $\tilde{\psi}$ that are not entailed by ψ are

$$\tilde{\psi}_1 = [KW(\neg In(R, A) \vee \neg Fragile(A))]$$
$$\tilde{\psi}_2 = [KW(\neg In(R, B) \vee \neg Fragile(B))]$$

and therefore $\tilde{\psi} \dot{-} \psi = \{\tilde{\psi}_1, \tilde{\psi}_2\}$

The e-difference operator plays a key role in computing entailment. The next Theorem describes the necessary and sufficient conditions for entailment of a domain or a knowledge ψ-form by a set of domain atoms and ψ-forms.

We call a set s of domain propositions *saturated*, when there are no possible resolutions between a ground atom a and a ground negated clause $\neg a \vee \neg a_1 \vee \neg a_n$, represented by some ψ-form in s. A saturated equivalent of such a set can be computed in polynomial time in the number of propositions (ψ-forms and atoms) in s.

Theorem 2 Let $s = A \cup \Psi$ be a consistent saturated set of domain atoms (A) and ψ-forms (Ψ), and ψ is any ψ-form (either domain or knowledge). $s \models_k \psi$ if and only if

1. there exist $a_1, \ldots, a_n \in A$, such that $\psi = [KW(\neg a_1 \vee \ldots \vee \neg a_n)]$, or
2. there exists $\psi \in \Psi$, such that $[\mathcal{M}(\psi_k)] \models_k [\mathcal{M}(\psi)]$, and, furthermore, $s - \psi \models_k (\psi \dot{-} \psi_k)$

PSIPLAN-S SOK

SOK (State Of Knowledge) database is a consistent set of PSIPLAN-S domain atoms or psiforms. It represents the knowledge available to the system in the following way:

1. a domain proposition p is true in the world, if and only if $SOK \models_k p$,
2. furthermore, we make a Closed Know-Whether Assumption (**CKWA**) and assume that if $SOK \not\models_k KW(p)$ then the truth value of p is not known, i.e. $\neg KW(p)$

The set of possible worlds corresponding to this representation consists of all world states in which everything known to the agent is true, and only things known to the agent are guaranteed to be true. Such representation is sound and complete, due to soundness and completeness of reasoning about domain and knowledge propositions from a set of domain propositions in PSIPLAN. Importantly, the inference procedures also run in polynomial time and are fast, which bears directly on the speed of planning with PSIPLAN-S. PSIPLAN-S thus ensures precise and fast reasoning about knowledge and ignorance.

PSIPLAN-S Actions and SOK update

PSIPLAN-S distinguishes two types of actions: *domain actions* that change the world (e.g., an action of downloading a paper from a url), and *sensing actions* that do not change the world but only return information about it (e.g., querying a bibliography).

Each domain action has a list of *preconditions*, $\mathcal{P}$, and an encoding of the effects of the action as a set of literals, called the *assert list*, $\mathcal{A}$. The propositions in $\mathcal{P}$ can include literals and quantified ψ-forms, where the term *quantified* is used informally to denote a ψ-form that uses at least one variable, and thus represents infinite number of ground instances. We assume that an action is deterministic and can change the truth-value of only a *finite* number of atoms, thus assert list contains literals only, and no quantified ψ-forms.

To update SOK s after executing a domain action a_d all propositions whose truth value[1] could have been changed must be removed from s – these are all propositions entailed by the *negation* of some effect of a_d. The propositions entailed by effects of a_d are also removed, and then the effects of a are added to the new SOK. The agent's SOK after executing a domain action a_d in the SOK s is computed by function $update(s, a_d)$ below.

$$update(s, a_d) = ((s \dot{-} \mathcal{A}^-(a_d)) \dot{-} \mathcal{A}(a_d)) \cup \mathcal{A}(a_d), \quad (4)$$

where $\mathcal{A}^-(a_d)$ denotes the set of propositions obtained by negating each proposition in a_d's, assert list $\mathcal{A}(a_d)$.

Sensing actions also have preconditions. Effects of the sensing are given by its *knowledge list*, denoted $\mathcal{K}$. The propositions in $\mathcal{K}$ are kw-ψ-forms. After a sensing action is executed, it returns an *observation list* of kn-propositions corresponding to the information that was learned, denoted Δ.

Download(?p, ?s, ?u)
  $\mathcal{P}: HasPaper(?u, ?s, ?p)$
  $\mathcal{A}: Got(?p)$

QueryBib(?b, ?kwd)
  $\mathcal{P}: PrefBib(?b)$
  $\mathcal{K}$: $[KW(\neg Rel(p, ?kwd) \vee \neg InCollection(p, ?b))]$

Figure 1: Example of Writer's Aid's domain and sensing actions. The variable p is implicitly universally quantified. Other variables are action schema parameters

Figure 1 provides examples of two PSIPLAN-S actions. *Download(?p, ?s, ?u)* is an action of downloading paper $?p$ from url $?u$ of source $?s$. *QueryBib(?b, ?kwd)* is a sensing action that identifies all papers, which according to bibliography $?b$ are related to keyword $?kwd$. The effect of this action is encoded in the knowledge list that contains a quantified ψ-form, and states that as a result of this action the set of all papers in collection of bibliography $?b$ that are related to keyword $?kwd$ will be identified.

For example, suppose after executing sensing action $a = QueryBib(ACM, XII)$ with effect $[KW(\neg Rel(p, ?kwd) \vee \neg InCollection(p, ?b))]$ papers $Paper_1$ and $Paper_2$ were found as the only ones related to keyword XII, i.e. $\Delta(a)$ consists of the following propositions:

$$\begin{array}{l} [\neg Rel(p, XII) \vee \neg InCollection(p, ACM) \\ except\{p = Paper_1\}, \{p = Paper_2\}] \\ Rel(Paper_1, XII), InCollection(Paper_1, ACM) \\ Rel(Paper_2, XII), InCollection(Paper_2, ACM) \end{array} \quad (5)$$

After the execution of a sensing action a_s, the set of observed propositions, denoted below by $\Delta(a_s)$ is added to the SOK, i.e.

$$update(s, a_s) = s \cup \Delta(a_s) \quad (6)$$

After propositions from $\Delta(a)$ are added to the SOK, all possible resolutions from SOK propositions are computed and added to the new SOK – this is a necessary step that guarantees soundness and completeness of domain goal inference in PSIPOP-SE.

[1] true or false

3 Planning with Hypotheticals

We assume the reader's familiarity with Partial Order Planning (POP) [Russell and Norvig, 1995]. PSIPOP-SE is a partial order planner that builds on its predecessors: a sound and complete open world partial order planning algorithm PSIPOP [Babaian and Schmolze, 2000] and PSIPOP-S[Babaian, 2000], which is an extension of PSIPOP to planning with sensing and knowledge goals. All three algorithms are based on PSIPLAN-S representation and calculus. PSIPOP-SE extends PSIPOP-S to planning with execution.

A **hypothetical link** is a link between an effect of a sensing action and a domain subgoal, when the truth value of the subgoal proposition is unknown and it is possible that the result of sensing will reveal that the subgoal is true. To define hypothetical links formally, we first need to define the $kwfy()$ operation for PSIPLAN-S domain propositions. Intuitively, the purpose of $kwfy(p)$ is to reflect the existing knowledge regarding all ground propositions represented by p. $kwfy(p)$ defines the smallest PSIPLAN-S knowledge proposition implied by p.

Definition 3 $kwfy(p)$ **operator.**

- For a domain atom a, $kwfy(a) = [KW(\neg a)]$.
- For a domain ψ-form $[P(\vec{x})$ except $\{\sigma_1, \ldots, \sigma_n\}]$,

$$kwfy(\psi) = [KW(P(\vec{x})) \text{ except } \{\sigma_1, \ldots, \sigma_n\}].$$

A hypothetical link is created between an effect k of a sensing step S_s and a (domain) precondition p on step S_p if and only if

1. $k \models_k kwfy(p)$, i.e. the effect of sensing will result in knowing the truth value of every ground propositions denoted by p, and
2. $kwfy(p)$ does not hold immediately prior to step S_p, i.e. the values of at least some ground propositions denoted by p are not known prior to S_p.

Hypothetical links are similar in spirit to Golden's *observational* links [Golden, 1998], but observational links to p do not require agent's ignorance regarding p and are formulated using conditional effects rather than knowledge propositions.

In the example, illustrated in Figure 2, the planner attempts to find support to a precondition to the *Download* action. The precondition $HasPaper(P, ?s, ?u)$ requires that paper P be available for download from some source $?s$ at some url $?u$. Suppose, that neither the agent's current state of knowledge nor its domain actions can bring about the achievement of the goal, however there is a sensing action $QuerySourceForPaper(P, ?s)$ with effect $k = [KW(\neg HasPaper(P, ?s, u))]$. This effect entails $kwfy(HasPaper(P, ?s, ?u))$, which equals $[KW(\neg HasPaper(P, ?s, ?u))]$. Note that here, as everywhere else, variables $?u, ?s$ are implicitly existentially quantified and treated as Skolem constants, while u in the knowledge effect k is the ψ-form's universally quantified variable. To ensure that the sensing would not be redundant, the planner first tries to prove that given the current partial plan, the value of $HasPaper(P, ?s, ?u)$, is not already known, by calling procedure *VerifyIgnorance*().

Procedure *VerifyIgnorance*()is passed a partial plan and a domain subgoal p on step S_p, and tries to find support to the goal p without adding any new actions. When it fails to find support for $kwfy(p)$, by the CKWA we can assume value of p is not known, and the procedure returns true. Otherwise, it returns false.

VerifyIgnorance$(plan, p, S_p)$
  `if exist effects` $e_1, \ldots, e_n$ `of steps in` *plan*
  `possibly before` S_p`, such that` $e_1, \ldots, e_n \models_k kwfy(p)$
    `return false`
  `else return true`

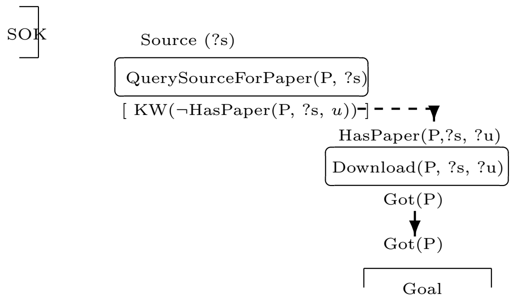

Figure 2: A depiction of a hypothetical plan. (Steps are represented by boxes containing action operator's name and parameters. – with dashed arrows.

The maximum number of consecutive hypotheses made in supporting any subgoal in a plan is called the **hypothetical level** of a plan Hypothetical level of a regular (also here called *simple*) partial order plan is 0. The space of hypothetical plans is explored gradually, by limiting the maximum allowed hypothetical level of a plan to avoid too much hypothesizing.

PSIPOP-SE **algorithm** is outlined in Figure 3. Note that this formulation is generalized and leaves out many details of PSIPLAN-S reasoning and associated goal satisfaction and threat resolution techniques, which can be found in [Babaian and Schmolze, 2000; Babaian, 2000], in order to focus on the details of planning with hypotheticals. PSIPOP-SE is a nondeterministic algorithm that is passed the initial plan encoding just the current SOK and goal state as its initial and goal steps, and an additional parameter `maxHL` denoting the maximum hypothetical level of explored plans. The following fields are added to the standard plan structure to support hypothetical planning: `hlevel` denotes the hypothetical level of the plan, `suspendedGoals` denotes a list of sets of goals, planning for which is suspended until the sensing step(s) are executed.

PSIPOP-SE starts by calling procedure POPH. POPH is searching for a way of supporting an open goal of a partially ordered plan that is passed in as a parameter, and simultaneously explores the hypothetical support for the goal. Hypothetical plans are created by procedure *FindHypPlans*, which nondeterministically chooses a sensing step - source of the hypothetical link to the goal in consideration, suspending the rest of the plan's open goals, and setting the set of plan's goals to the precondition of the added sensing step. The hypothetical plans returned by *FindHypPlans* are not expanded further unless the search for a simple solution plan results in failure.

If POPH returns with a failure, in other words, a simple plan that achieves a goal does not exist, PSIPOP-SE nondeterministically chooses a hypothetical plan from `HPlans`. The picked hypothetical plan has as its list of open goals the preconditions of the earliest source of the first in order hypothetical link, and the rest of the plan's open preconditions as its suspended goals. These subgoals were suspended because unless the target condition of the hypothetical link is found to be true, it does not make sense to continue planning to satisfy the rest of subgoals of the plan.

To enable execution of the first in order information gathering action, PSIPOP-SE calls procedure HPOP, which searches for a (partial) plan that makes the sensing step-source of the hypothetical link executable from the initial state. If such completion is found, HPOP executes the plan up until the source of the hypothetical link, otherwise, the next hypothetical plan is explored.

Upon execution of each action SOK is updated according to equations (4) and (6) in procedure *UpdateAfterExecution*. The executed plan is updated as well: the executed steps are removed, links originating in the executed steps are now drawn from the initial step denoting the SOK, previously suspended goals are restored and the planner continues to work towards completing the plan.

It is possible that due to the executed portion of the plan some sensing acts may have become redundant, as previously unknown propositions became known. To avoid redundant information gathering, procedure $HPOP$ verifies that the sensing is necessary by calling *VerifyIgnorance*, when picking the next hypothetical plan to expand. Note also that some causal links may be invalidated because the truth value of a proposition was reversed by the executed actions. This would not happen to the executed current plan, but it may affect other hypothetical plans in HPlans. Thus, $HPOP$ may discard some invalid links originating from the initial step (SOK) that are no longer valid, adding their target conditions to the plan's goals.

4 Conclusions and Future Work

We have presented a novel method for interleaving planning with execution, which enables information gathering to be used in support of planning goals. The method has been implemented within a partial order planner, however, its formulation is based on the general concepts of entailment, reasoning about knowledge and ignorance, which could make the method applicable to other planning and acting frameworks.

Future research needs to focus on fully exploring the properties of hypothetical planning on problems from a

PSIPOP-SE (init-plan, maxHL)

```
HPlans = ∅ // hypothetical plans
if POPH(init-plan, HPlans, maxHL) fails
      Choose a plan ph from HPlans
      remove ph from HPlans
      HPOP(ph, maxHL, HPlans)
```

POPH(plan, maxHL, HPlans)

```
if (plan.goals = ∅) return plan
else plan' = copy(plan)
   Choose a goal g from plan'.goals
   if FindSupport(plan',g) fails or
   ResolveThreats(plan',g) fails)
      result =∅
   HPlans=HPlans∪FindHypPlans(plan',maxHL, g)
   if result =∅ then fail
   else POPH(plan', maxHL, HPlans)
```

FindHypPlans(plan, maxHL, g)

```
// where g denotes a precondition p on step S_p
if (plan.hlevel<maxHL) and VerifyIgnorance(p, S_p) =
true
   Choose a sensing operator S_s with effect k
   such that k |=_k kwfy(p).  If found S_s:
      planh = copy(plan)
      add hypoth.  link S_s --> S_p to planh.links
      planh.hlevel = planh.hlevel + 1
      push (planh.goals) to planh.suspendedGoals
      planh.goals = P(S_s)
      return planh
```

HPOP(planh, maxHL, HPlans)

```
-- Complete and execute a hypothetical planh
Remove invalid causal links with source in the SOK
from planh.links,
add their goals to plan.goals
Find the earliest step S_s - source of hypothetical
link in planh.
Suppose it is linked to precondition p of S_p
if VerifyIgnorance(p, S_p) = true and ph.goals≠ ∅
      // find an executable completion of ph, phe
      phe = POPH(planh, maxHL, HPlans))
   else phe = ph;
   Execute phe up to and including S_s
   UpdateAfterExecution (phe)
   if (phe.hlevel > 0)
   // remaining plan still has hypothetical links
      HPOP(phe, maxHL, HPlans)
   else POPH(phe, maxHL, HPlans)
```

UpdateAfterExecution (ph)

```
For each executed step S in ph
   SOK = update(SOK,S) // equations (4,6)
   Replace S with SOK in all causal links
   originating from S to the rest of plan
ph.hlevel = ph.hlevel - 1
ph.goals = pop a list from ph.suspendedGoals
```

Figure 3: Nondeterministic algorithm PSIPOP-SE.

variety of domains, generalizing the hypothetical planning approach to planning in domains with irreversible actions, and examining formal issues related to soundness and completeness of the search for hypothetical plans in PSIPOP-SE.

References

[Ambros-Ingerson and Steel, 1988] Jose A. Ambros-Ingerson and Sam Steel. Integrating planning, execution and monitoring. In *Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI-88)*, pages 83–88, St. Paul, Minnesota, 21–26 August 1988. Morgan Kaufmann.

[Babaian and Schmolze, 2000] T. Babaian and J. Schmolze. Psiplan: open world planning with ψ-forms. In *Proceedings of AIPS'00*, pages 292–300, 2000.

[Babaian *et al.*, 2002] Tamara Babaian, Barbara J. Grosz, and Stuart M. Shieber. A writer's collaborative assistant. In *Proc. of IUI'02*, pages 7–14. ACM Press, January 2002.

[Babaian, 2000] Tamara Babaian. *Knowledge Representation and Open World Planning Using ψ-forms.* PhD thesis, Tufts University, 2000.

[Baral and Son, 2001] Chitta Baral and Tran Cao Son. Formalizing sensing actions – a transition function based approach. *Artificial Intelligence*, 125, 2001.

[Golden *et al.*, 1994] K. Golden, O. Etzioni, and D. Weld. Omnipotence without omniscience: Efficient sensor management for planning. In *Proceedings of AAAI-94*, 1994.

[Golden, 1998] Keith Golden. Leap before you look: Information gathering in the puccini planner. In *Proceedings of AIPS'98*. AAAI Press, June 1998.

[Russell and Norvig, 1995] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach.* Prentice Hall, Englewood Cliffs, NJ, 1995.