Formalizing a Spectrum of Plan Generalizations Based on Modal Truth Criteria

Subbarao Kambhampati*
Department of Computer Science and Engineering
Arizona State University
Tempe, AZ 85287
email: rao@asuvax.asu.edu

Abstract

Plan generalizations are of considerable importance in improving planning performance through reuse. In this paper, we provide a unified framework for analytic (non-inductive) plan generalization based on explanation of plan correctness with respect to modal truth criteria. Within this framework, we explore a large spectrum of generalizations based on the type of constraints on the plan that are being relaxed (ordering, binding, initial state specification), the strength and type of truth criteria used for explaining correctness and the number of explanations used as a basis for generalization. Apart from the straightforward precondition and order generalizations, the spectrum also includes such novel ones as possible correctness generalizations and disjunctive generalizations. In each case, we characterize the cost of producing the generalization, and the storage and usage advantages provided by it during plan reuse. Although there has been previous work on plan generalizations, this is the first time that a complete spectrum of generalizations are characterized within a single unified framework, facilitating a comparative analysis of their costs and benefits.

Submitted to AIPS-94 (2nd Intl. Planning Conference)
Content Areas: Learning, Explanation-based Generalization
Submitted also to Canadian AI Conference, 1994

*This research is supported in part by National Science Foundation under grant IRI-9210997, and ARPA/Rome Laboratory planning initiative under grant F30602-93-C-0039.

1 Introduction

Creating and using generalized plans is a central problem in machine learning and planning. Broadly speaking, a plan $\mathcal{P}$ is said to be a generalization of another plan $\mathcal{P}'$, if $\mathcal{P}$ provides more ways of achieving the goals of $\mathcal{P}'$. This may include allowing goal achievement from a larger class of initial states, or providing more ways of executing the actions in the plan. Generalized plans are of particular utility in plan reuse as storing and reusing generalized plans as against specific plans allows for storage compactions, and retrieval efficiency. An important approach for plan generalization is the so-called explanation-based generalization (EBG) [14]. Much of the previous work on EBG of plans concentrated on inflexible plan representations, and was limited to straight forward precondition and order generalization [14, 18, 10, 2, 17, 19, 3]

This paper aims to unify and extend the previous work on plan-generalizations. We start with the flexible partial order plan representation, and formalize explanation based plan generalization in terms of proofs of correctness with respect to modal truth criteria (which have been originally proposed for formalizing plan generation [1]). Specifically, the correctness of a given plan is first explained with respect to a truth criterion. This proof, called validation structure of the plan, is used as a basis for generalization. Broadly speaking, generalization involves retracting any constraints of the plan that are not ''justified'' with respect to the validation structure. The retractable constraints include the initial state specification, the orderings and the bindings of the plan.

We show that this unified framework results in a large spectrum of plan generalizations, based on the constraints being relaxed, the strength and type of truth criteria, and the number of validation structures being used as a basis for generalization. In addition to casting previous work on plan generalization in one common framework, our systematic exploration brings out additional novel plan generalizations such as possible correctness generalizations and disjunctive generalizations. We show that the different generalizations all form a generalization lattice, that foregrounds a variety of interesting generalization cost vs. instantiation cost tradeoffs among them. We believe that the systematic comparative analysis of plan generalizations facilitated by this work, is of critical importance in designing plan reuse frameworks with favorable computational tradeoffs [7].

The paper is organized as follows: Section 1.1 reviews the plan representation and notions of plan correctness. Section 2 discusses truth criteria and how they are used to build explanations of correctness of plans. In Section 3 and 4, these explanations are used as a basis for developing a variety of plan generalizations. Section 5 puts the entire gamut of generalizations discussed in this paper in a generalization lattice,

and analyzes the tradoffs offered by them. Section 6 summarizes the conclusions and discusses related work.

1.1 Preliminaries

We will begin with some standard terminology for partially ordered and partially instantiated plans (POPI plans): Given a planning problem $[\mathcal{I}, \mathcal{G}]$ where $\mathcal{I}$ is a conjunction of literals specifying the initial state and $\mathcal{G}$ is a conjunction of literals specifying the desired goal state, a *partially ordered partially instantiated plan* (POPI plan) $\mathcal{P}$ is a 3-tuple $\mathcal{P} : \langle T, O, \mathcal{B} \rangle$, where T is the set of actions in the plan, O is a partial ordering relation over T, and $\mathcal{B}$ is a set of codesignation (binding) and non-codesignation constraints (prohibited bindings) on the variables in $\mathcal{P}$. The notation $x \approx y$ is used to denote that the variable x codesignates (binds) with the variable or constant y. T contains two distinguished nodes t_I and t_G, where the effects of t_I and the preconditions of t_G correspond to the initial and final states of the plan, respectively. The actions are represented by instantiated STRIPS-type operators with *Add*, *Delete* and *Precondition* lists, all of which are conjunctions of first order literals. The subset of this representation where all three formulas can be represented as conjunctions of function-less first order literals, and all the variables have infinite domains, is called the TWEAK representation (c.f. [1, 12]).

As an illustration of this plan representation, consider the blocks world problem of stacking four blocks A, B, C and D, which are initially all on table and clear, into two stacks $On(A, B)$ and $On(C, D)$. A POPI plan for this problem will be $\langle \{t_I, t_1 : puton(A, B), t_2 : puton(C, D), t_G\}, \{t_I \prec t_1 \prec t_G, t_I \prec t_2 \prec t_G\}, \emptyset \} \rangle$, where the effects of t_I contain the initial state assertions, and the preconditions of t_G contain the goals $On(A, B) \wedge On(C, D)$ (note that t_2 and t_1 are not ordered with repsect to each other). We call this plan the 4BS plan, and will use it as an example throughout the paper.

A POPI plan $\mathcal{P} : \langle T, O, \mathcal{B} \rangle$ corresponds to a set of totally ordered totally instantiated plans called *completions*. Each completion of $\mathcal{P}$ corresponds to a topological sort of the operators of $\mathcal{P}$, with all variables assigned (constant) values that are consistent with $\mathcal{B}$. The totally ordered plan $\text{4BS}_c : \langle \{t_I, t_1 : puton(A, B), t_2 : puton(C, D), t_G\}, \{t_I \prec t_1 \prec t_G, t_1 \prec t_2, t_I \prec t_2 \prec t_G\}, \emptyset \} \rangle$ is a completion of the 4BS plan discussed above (it is same as 4BS except for the additional ordering constraint $t_1 \prec t_2$). The modal operators ''$\Box$'' and ''$\Diamond$'' are used to denote the *necessary* and *possible* truth of a statement over all the completions of a plan (see [1]).

The correctness of a POPI plan is defined in terms of its completions. A completion is said to be *correct* if it can be executed in a state matching the effects of t_I to produce a state matching the preconditions of t_G. Since a POPI plan corresponds to many completions, there are at least two different ways of defining POPI plan correctness: (i) A POPI plan $\mathcal{P} : \langle T, O, \mathcal{B} \rangle$ is said to be **necessarily correct** if and only if every

completion of $\mathcal{P}$ is correct. (ii) A POPI plan is said to be **possibly correct** if and only if at least one completion of $\mathcal{P}$ is correct.

2 Truth criteria and Explanations of Correctness

An important method for analytical generalization of plans is the so-called explanation based generalization. In this method, one develops an explanation as to why the example satisfies the concept under consideration. This explanation is then used as a guide to generalize the example. In the case of plan generalization, the concept under consideration is ''plan correctness'', and the example is a supposedly correct POPI plan. In this section, we will develop a generalized way of representing explanations of correctness. For now, we shall concentrate on necessary correctness as the target concept; we will come back to the possible correctness in Section 4.3.

The obvious way to verify and explain the (necessary) correctness of a POPI plan will involve generating all its completions and showing that they are all individually correct. Modal truth criteria provide a more efficient way of explaining correctness. A truth criterion specifies a set of constraints on the orderings, bindings and steps of the plan which, when satisfied, will ensure that a given precondition of a given step is necessarily true (in that it will be satisfied in all the completions of the plan). It is easy to see that when all the preconditions of all the steps are necessarily true, then the plan itself is necessarily correct. The following is an example truth criterion:

Definition 1 (MTC1) *A precondition C of a step s in plan $\mathcal{P} : \langle T, O, \mathcal{B} \rangle$ is said to be necessarily true according to MTC1 if and only if there is some step $s' \in T$ with an effect E such that $\Box(E \approx C)$, and for all $s'' \in T$ such that $\Diamond(s' \prec s'' \prec s)$, if D is a delete list literal of s'', then $\Box(D \not\approx C)$.*

Modal truth criteria can vary in their generality. Chapman provides the most general truth criterion (i.e., a truth criterion which is both necessary and sufficient) for POPI plans in TWEAK representation. For simplicity of exposition, we will use MTC1 to guide the development of the rest of the paper. We will however discuss the ramifications of using more general truth criteria such as MTC2 in Section 4.1.

Explanation Structures: Once we settle on a truth criterion, the explanation of correctness of a plan with respect to that truth criterion can be be characterized as a ''proof'' that all the preconditions of the plan are necessarily true according to this criterion. For POPI plans, any such proof can be represented (summarized) in the form of a set $\mathcal{V}$ of dependency links, collectively called a validation structure. Each individual dependency link, called validation, is a 4-tuple $\langle e, t', C, t \rangle$ where t' has an effect e which is used to support the precondition C of step t, such that C is necessarily true according to the truth criterion being used. The exact semantics of validations thus depend on the truth criterion being used. For the case of MTC1, we have the following formal characterization of validation structure.

Algorithm EXP-MTC ($\mathcal{P} : \langle T, O, \mathcal{B} \rangle$)
  $\mathcal{V} \leftarrow \emptyset$
  foreach $t \in T$ **do**
   foreach $\langle C, t \rangle$ (where $C \in precond(t)$) **do**
    Traverse $\mathcal{P}$ in the reverse topological sorted order and find the first operator t' s.t.
        $t' \prec t \wedge \exists e \in effects(t') \wedge \Box(e \approx C)$ and
        $\forall t''$s.t. $\Diamond(t' \prec t'' \prec t), \forall d \in delete(t'')\ \Box(d \not\approx C)$
    if such a t' is found **then** $\mathcal{V} \leftarrow \mathcal{V} \cup \{\langle e, t', C, t \rangle\}$
        else *return failure*

Figure 1: Algorithm for constructing explanation of correctness w.r.t MTC1

Definition 2 (Validation Structure) *$\mathcal{V}$ is a validation structure of $\mathcal{P} : \langle T, O, \mathcal{B} \rangle$, if (i) for every precondition C of a step t of the $\mathcal{P}$, there exists exactly one validation $v : \langle e, t', C, t \rangle$ in $\mathcal{V}$ supporting that precondition. (ii) for each validation $v : \langle e, t', C, t \rangle \in \mathcal{V}$, $\Box(e \approx C) \wedge \Box(t' \prec t)$ and $\forall t'' \in T$ s.t. $\Diamond(t' \prec t'' \prec t)$, $\forall d \in delete(t'')$, $\Box(d \not\approx C)$*

A validation for the 4BS plan is $\langle On(C, D), t_2, On(C, D), t_G \rangle$. Given a POPI plan, it is straightforward to compute and represent its correctness in terms of validations. Figure 1 provides an algorithm for doing this with respect to MTC1. This algorithm takes $O(n^3)$ running time (where n is the length of the plan).

3 Explanation-based Necessary correctness generalizations

Once we have a validation structure, $\mathcal{V}$ for a plan $\mathcal{P}$, the generalization step involves relaxing the constraints in the plan while still maintaining $\mathcal{V}$ as its validation structure. The relaxable constraints of the plan are the orderings (O), bindings ($\mathcal{B}$) and the specifications of the initial state (the effects of t_I). In general, we can remove a constraint that is not ''justified'' with respect to the validation structure, in that removing it does not cause any of the validations of the plan to fail (i.e., fail to satisfy the conditions specified in Definition 2). An initial state assertion (effect) is justified (required) only if it is used to support a validation (i.e., there is some validation of the form $\langle e, t_I, C, t \rangle$ for some step t and its precondition C belonging to the plan). Similarly, individual ordering and binding constraints are justified if and only if removing them will lead to the failure of some validation of the plan. Since a POPI plan contains multiple relaxable constraints, there are a larger variety of ways of generalizing the plan, while keeping a given validation structure.

In particular, given two correct POPI plans $\mathcal{P} : \langle T, O, \mathcal{B} \rangle$ and $\mathcal{P}' : \langle T', O', \mathcal{B}' \rangle$:

(i) $\mathcal{P}$ is considered a **precondition generalization** of $\mathcal{P}'$ if and only if and $T = T'$, $O = O'$, every codesignation and non-codesignation constraint that is consistent with $\mathcal{B}'$ is also consistent with $\mathcal{B}$, and finally the initial state of $\mathcal{P}$ are subsumed by that of $\mathcal{P}'$ (i.e., *effects*$(t_I) \vdash$ *effects*(t'_I)).
(ii) $\mathcal{P}$ is considered an **order generalization** of $\mathcal{P}'$ if and only if $T = T'$, $\mathcal{B} = \mathcal{B}'$ and $\mathsf{TC}(O') \supset \mathsf{TC}(O)$ (where TC is the transitive closure operation). In other words, every ordering constraint that is consistent with O' must also be consistent with O.
(iii) $\mathcal{P}$ is considered an **order and precondition generalization** (or **hybrid** generalization) of $\mathcal{P}'$ if and only if $T = T'$, $\mathsf{TC}(\mathcal{B}') \supset \mathsf{TC}(\mathcal{B})$ and $\mathsf{TC}(O') \supset \mathsf{TC}(O)$, and *effects*$(t_I) \vdash$ *effects*(t'_I) (i.e., all the orderings, bindings and initial state conditions of $\mathcal{P}$ are subsumed by those of $\mathcal{P}'$.

3.1 Explanation-Based Precondition (and Binding) Generalization

Precondition generalization involves relaxing bindings and the initial state specification (also called the applicability preconditions of the entire plan). To do this, we first schematize[1] (variablize) the plan $\mathcal{P}$ to produce $\mathcal{P}^s$. This process involves substituting the plan steps in $\mathcal{P}$ with the corresponding operator templates (with fresh variables), and variablizing all the constants in the initial and goal states) to produce. Next, we compute the weakest constraints (codesignation and non-codesignation) on the variable bindings to ensure that $\mathcal{V}^s$ will be a validation structure of $\mathcal{P}^s$. From the semantics of the validations provided in equation 3, these conditions can be stated as the conjunction of codesignation and non-codesignation constraints shown in expression 1 below:[2]

$$\bigwedge_{\forall v^s:\langle e^s,t'^s,C^s,t''^s\rangle \in \mathcal{V}^s} \left[\Box(e^s \approx C^s) \bigwedge \forall t^s \in T^s \, s.t. \begin{array}{l} \Diamond(t'^s \prec t^s \prec t''^s), \\ \forall d^s \in delete(t^s) \; \Box(d^s \not\approx C^s) \end{array} \right]$$

Evaluating these constraints gives rise to a generalized initial state, with a set of required codesignation and non-codesignation constraints on the variables in the initial state. As long as the initial state of a given planning problem satisfies these constraints, the generalized plan can be used in that situation to successfully achieve the generalized goals [11].[3] For the case of 4BS plan, the precondition generalization algorithm will produce a plan $\text{4BS}^p : \langle \{t_I, t_1 : puton(p,q), t_2 : puton(r,s), t_G\}, \{t_I \prec t_1 \prec t_G, t_I \prec t_2 \prec t_G\}, \mathcal{B} : \{q \not\approx r, q \not\approx s, p \not\approx r \vee s \not\approx table, p \not\approx r \vee q \not\approx table, p \not\approx s, p \not\approx r\}\rangle$ where the preconditions of t_G are $on(p,q), on(r,s)$ and the effects of t_I are $clear(p), clear(q), on(p,table), clear(r), clear(s)$ and $on(r,table)$.

[1] We shall use the superscript ''s'' to distinguish entities corresponding to the schematized plan.

[2] [10] contains a more elaborate description of this process.

[3] Note that the construction of initial explanation itself causes an important form of relaxation of the initial state specification by removing from consideration any part of the initial state that are not explicitly used to support any validation.

Since there are at most $O(n)$ validations in the plan, and the constraints required to preserve each validation can be computed in polynomial time, the precondition generalization itself can be done in polynomial time in the length of the plan. Including the cost of initial explanation construction, the complexity of precondition generalization can be shown to be $O(n^3)$ [10].

3.2 Explanation-based Order Generalization

Just as precondition generalization eliminates any initial state conditions (effects of t_I), and variable bindings that are not required to make the plan correct according to the given validation structure, we can also eliminate any orderings that are redundant. In particular, if $\mathcal{V}$ is the validation structure of the plan $\mathcal{P}$ (see Section 2), then we can justify the ordering relations among the steps of the plan $\mathcal{P}$ with respect to the validation structure. The key insight is that the only ordering constraints that are necessary to ensure correctness of the plan are those that are required to maintain the individual validations (as given by equation 3).

Let $o : t_i \prec t_j$ be an ordering relation on the plan $\mathcal{P}$. Then, from the semantics of the validation given in equation 3, it can easily be shown that o is justified with respect to the validation structure $\mathcal{V}$ (or $justified(o : (t_i \prec t_j), \mathcal{P}, \mathcal{V})$) if and only if at least one of the following cases hold:

case 0. Either $t_i = t_I$ or $t_j = t_G$ (all plan steps follow the the start node and precede the end node).

case 1. $\exists v : \langle e, t_i, C, t_j \rangle \in \mathcal{V}$

case 2. $\exists v : \langle e, t', C, t_i \rangle \in \mathcal{V}$ and t_j has a delete list literal d such that $d \approx C$

case 3. $\exists v : \langle e, t_j, C, t' \rangle \in \mathcal{V}$ and t_i has a delete list literal d such that $d \approx C$.

Additionally, any ordering relation that belongs to the transitive closure of all the justified ordering relations is also justified.

If an ordering relation o is not either directly justified or transitively justified, then o can be safely removed from $\mathcal{P}$ without affecting the correctness of the plan according the explanation $\mathcal{V}$. Since there are $O(n^2)$ ordering relations and $O(n)$ validation links in n-step plan, order generalization (i.e., removing unjustified orderings) can be done in $O(n^3)$ time.

Note that order generalization method discussed above can be used to generalize partially ordered or totally ordered plans with equal ease. For example, if a (total ordering) planner produced the plan $4BS_c$ discussed in Section 1.1, as a plan for the 4 blocks problem, then the order generalization can generalize it to 4BS plan, by removing the unjustified ordering $t_1 \prec t_2$. In this sense, it subsumes the plan order optimization algorithms described in [19] and [3].

3.3 Combining order generalization and precondition generalization

Given a plan $\mathcal{P} : \langle T, O, \mathcal{B} \rangle$ and a validation structure $\mathcal{V}$, there may typically be more than one way of specializing (adding constraints to) $\mathcal{B}$ and O to make $\mathcal{V}$ a validation structure of $\mathcal{P}$. In particular, there are essentially two types of ordering and binding constraints that $\mathcal{P}$ has to satisfy so as to maintain $\mathcal{V}$ as its validation structure [13]:

Causal constraints: For each validation $v : \langle E, t_s, C, t_d \rangle \in \mathcal{V}$, $(t_s \prec t_d) \in O$, and $(E \approx C) \in \mathcal{B}$.

Safety constraints: For each validation $v : \langle E, t_s, C, t_d \rangle \in \mathcal{V}$, foreach node $t' \in T$ such that t' has a delete list literal d that can possibly codesignate with C, either of the following constraints must be satisfied: (i) $(t_d \prec t') \in O$ or $(t' \prec t) \in O$ **or** (ii) $d \not\approx C \in \mathcal{B}$

Although the causal constraints are fixed once a validation structure has been chosen, the safety constraints can be satisfied in more than one way -- either by adding additional ordering constraints or by adding additional codesignation/non-codesignation constraints. Thus, given a validation structure, we can have multiple minimal $\mathcal{B} - O$ pairs. We consider two specific hybrid generalizations:

Sequential generalization: The iterative generalization uses the strategy of first pegging $\mathcal{B}$ to one value, and generalizing O with respect to it, and then pegging O to the generalized value and generalizing $\mathcal{B}$. Such a process can, for example, take a totally ordered totally instantiated plan, such as one generated by STRIPS [4] or PRODIGY [15] and generalize it by first removing unnecessary orderings and then generalizing the variable bindings. As an example, suppose we started with a completion of the 4BS plan with $t_1 \prec t_2$. The order generalization process will realize that $t_1 \prec t_2$ is unjustified and will remove it, to give rise to 4BS plan, which can then be precondition generalized to give 4BS^p discussed in Section 3.1. Since both the order and precondition generalizations w.r.t. MTC1 can be computed in $O(n^3)$ time, sequential generalizations can be done in $O(n^3)$ time.

Disjunctive Generalization: It is also possible to compute all minimal $\mathcal{B}$ and O combinations and consider the resulting collection of plans as a single disjunctive generalization of the original plan. For the 4BS example, a disjunctive generalization process could provide three generalized plans: (i) the plan 4BS^p discussed in 3.1, (ii) a variant of 4BS^p with the constraint $q \approx r$ replacing $q \not\approx r$, and the additional ordering constraint $t_2 \prec t_1$ and finally (iii) a variant of 4BS^p with codesignation constraint $s \approx p$ replacing $s \not\approx p$, and the ordering $t_1 \prec t_2$. The last two plans correspond to a 3 block, 3 stack problems, (such as the one to achieve $On(A, B) \wedge On(B, C)$ starting from a state where A, B and C are all clear and on table). This process is identical to the generalization algorithm proposed in [17]. Computing all the consistent $\mathcal{B} - O$ combinations for a given plan $\mathcal{P}$ and validation structure $\mathcal{V}$ is costlier than sequential generalization; Mooney's algorithm takes $O(n^5)$ in the length of the plan [17].

4 Extending the generalization framework

4.1 Utilizing more general truth criteria

It is possible to use truth criteria more general than MTC1, as a basis for explanation construction and generalization. MTC2 below is one possibility:

Definition 3 (MTC2) *A precondition C of a step s in plan $\mathcal{P} : \langle T, O, \mathcal{B} \rangle$ is said to be necessarily true according to MTC2 if and only if there is some step $s' \in T$ with an effect E such that $\Box(E \approx C)$, and for all $s'' \in T$ such that $\Diamond(s' \prec s'' \prec s)$,* **if D is a delete list literal of s'', such that $\Diamond(D \approx C)$, then there exists some step $w \in T$ (called a white-knight step) such that $\Box(s'' \prec w \prec s)$ and w has an effect E_w such that $\Box[(D \approx C) \Rightarrow (E_w \approx C)]$**

MTC2 is equivalent to Chapman's [1] necessary and sufficient truth criterion for TWEAK plans. (Note that the part in bold lettering, called the white-knight clause, is the only difference between MTC1 and MTC2.) The explanation of correctness of a plan with respect to MTC2 can also be represented in the form of validation links that are similar to those described in Section 2, but have the more general semantics that for every possibly intervening deleter, there must be a white-knight step that necessarily re-asserts the condition. Constructing an explanation of correctness with respect to MTC2 will be costlier --- $O(n^4)$ for an n-operator plan, instead of $O(n^3)$ for the algorithm in Figure 1. Once such a validation structure is found, straightforward variants of the precondition and order generalization algorithms described in the previous sections can be developed for MTC2. These algorithms will still be of polynomial time complexity.[4]

The difference between generalizations based on MTC1 and MTC2 is that the former provides only sufficient conditions for necessary truth, while the latter provides both sufficient and necessary conditions. Thus, there may be some POPI plans which are necessarily correct, and have a proof of correctness with respect to MTC2, but not with respect to MTC1.[5] For those plans which can be handled by both MTC1 and MTC2, the latter returns a slightly more general (disjunctive) preconditions (set of codesignation/non-codesignation constraints among variables). Just as we can use the order generalization to generalize plans generated by a total ordering planner, we can also use MTC2 to further generalize plans generated by a planner using a less general truth criterion such as MTC1.

[4] When we go for action representations more expressive than those allowed by TWEAK, interpreting a necessary and sufficient truth criterion becomes intractable. It is however possible to come up with tractable sufficient truth criteria [16]

[5] The fact that MTC1 cannot explain the correctness of some partially ordered plans is not as limiting as it may seem. In particular, since every totally ordered plan *can* be handled by MTC1, it is always possible to take a topological sort of the original plan, use that as a basis to construct MTC1 based order and precondition generalizations. Further more, most existing partial order planners do not use white-knights in planning, and thus generate partial orderings that are correct with respect to MTC1 as well as MTC2

4.2 Generalization based on Multiple Explanations and ''Weakest Preconditions''

The precondition generalizations discussed in preceding sections work with respect to a particular chosen validation (explanation) structure based on a particular modal truth criterion. Unless the plan has only a single validation structure, these conditions will not in general be the weakest such conditions for guaranteeing correct execution of the plan with respect to the modal truth criterion being used.

To compute weakest preconditions with respect to a truth criterion, we have to compute all possible explanation structures of the plan with respect to that truth criterion, and do generalizations based on each of them. The disjunction of all these generalizations represents the weakest preconditions with respect to the truth criterion. (If the truth criterion under consideration is both necessary and sufficient for the class of plans under consideration, then this computation will give the absolute weakest preconditions for the success of the plan). The explanation construction algorithms in Figure 1 can be modified in a straightforward fashion to return *all* rather than one explanations structure.

Not surprisingly, finding weakest preconditions with respect to multiple explanations, is in general costlier than finding an explanation based generalization of the plan. In particular, we note that in an n-action plan, there are at most $O(\xi n^2)$ different validations, where ξ is the upper bound on the number of preconditions per step.[6] Since finding a validation link with respect to TWEAK truth criterion takes $O(n^3)$ (see Section 4.1), the total cost of finding all possible validations is $O(\xi n^5)$. Since computing the codesignation and non-codesignation constraints imposed by a schematized validation can be done in $O(n)$ time (see Section 3.1), the cost of generalization is $O(\xi n^3)$, making the overall process $O(n^5)$.

4.3 Generalized Preconditions for Possible Correctness

Until now, we have looked at generalization algorithms that use the target concept of necessary correctness. It is also possible to construct generalization algorithms based on possible correctness. For example, precondition generalizations based on possible correctness would compute weakest preconditions under which at least some completion of the plan can possibly execute. During reuse, they can be used to select a generalized plan that is possibly applicable in a given situation, and make

[6] To visualize this worst case situation, consider an n action, n goal totally ordered plan, with each step of the plan providing one of the goals. Suppose that each step has p preconditions, and they can be supplied by *any* of the steps that precede it. Thus, for the ith step, each of its preconditions can be supported by $(i-1)$ different validations, giving rise to $\xi(i-1)$ validations. Summing this over all the steps, we get $O(\xi n^2)$ different validations.

it necessarily applicable by allowing the planner to add only additional ordering and binding constraints *without ever backtracking over the plan*. Such a mechanism would, for example, allow us to store a possible correctness generalization of 4BS plan, and reuse it both in another four block 2-stack stacking scenario and in a three-block 3-stack stacking scenario (such as achieving $On(A, B) \wedge On(B, C)$). Since such a mechanism allows a stored plan to be used in a much larger set of situations, it improves storage and retrieval, and provides an interesting memory vs. planning tradeoff.

Unfortunately, unlike necessary correctness, checking possible correctness is NP-hard even for plans in TWEAK representations [12]. Even more importantly, the resulting preconditions could be highly disjunctive and thus the match cost might outweigh their expected savings [15]. One compromise would be to look for necessary *or* sufficient conditions for possible correctness which will be easier to compute and match against. Since $\Box P \supset \Diamond P$, the generalized preconditions for necessary correctness (see Section 3.1 already provide a set of sufficient conditions for possible correctness.

A way of computing necessary conditions for possible correctness, without having to reason with individual completions, is to use the modal truth criterion for *possible truth*[7] of a proposition in a POPI plan. The basic idea is to use a possible truth criterion, which is obtained by reversing the modalities in any necessary truth criterion (such as MTC1 or MTC2), to explain the possible correctness of each precondition and goal of the plan individually, and base the generalization on this explanation. Since $\Diamond P \wedge \Diamond Q$ does not imply $\Diamond(P \wedge Q)$, the conditions that ensure that all preconditions individually possibly hold may not in general guarantee that there exists a completion in which they all hold simultaneously.[8] They do however constitute a set of *necessary* conditions for ensuring possible correctness of the plan --- if the generalized preconditions computed by possible correctness *do not* hold in a problem situation, then we know for sure that no specialization of that plan will ever solve that problem. In [11] we describe an $O(n^3)$ algorithm for doing this, that is a straightforward adaptation of precondition generalization scheme discussed in Section 3.1.

5 Tradeoffs between the spectrum of generalizations

Table 1 summarizes the comparative costs of producing, and modes of reusing each of the generalizations discussed in the preceding sections. The various necessary correctness generalizations developed in this paper all fall within a lattice of generalizations shown in Figure 2. From this, we see that there are a variety of factors influencing the generalizations of a POPI plan. These include the generality of the truth criterion used as the basis for generalization; the particular constraints on the POPI plan that are

[7] Strictly speaking, possible truth conditional on executability; see [12]

[8] In other words, even if its preconditions for possible correctness, as described above, hold in that situation. there may not be any completion of the plan that will work in a given situation,

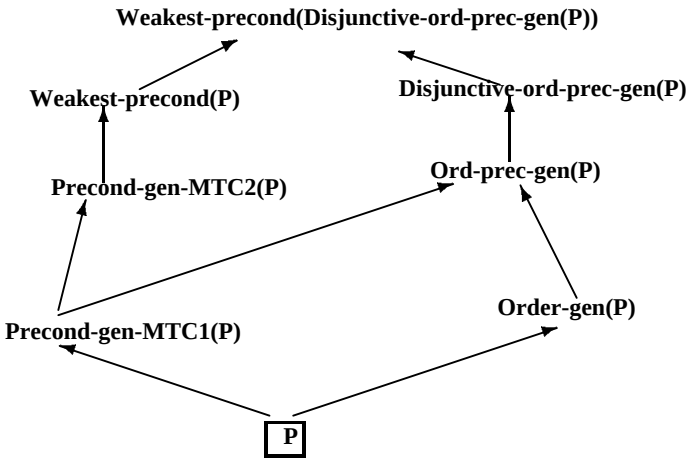

Figure 2: A lattice showing the spectrum of generalizations of POPI plans. Dashed lines show the possible correctness generalizations

being generalized (viz., codesignation constraints, ordering constraints, steps); whether the generalization is based on necessary correctness or possible correctness; and finally whether the generalization is based on a single explanation or multiple explanations.

An important issue is the cost of using each type of generalization during planning. The necessary correctness generalizations of POPI plans developed in this paper can be used during planning either as (nonlinear) macro-operators [4] or as generalized plans to be reused [9]. In the former case, the stored plans can be instantiated even in cases where only some parts of the plan are applicable. In particular, the validation structure can be used to *edit* the macro operator (c.f. [4]) and find sub-parts of it that are applicable in the current situation [9]. The cost of instantiating a macro-operator depends on the number of preconditions that need to be checked. It is thus typically costlier to instantiate generalizations with disjunctive sets of preconditions (such as the ones described in Sections 3.3 and 4.1).

In addition to being used as macro operators, the stored plans can also be instantiated in new situations through plan refinement and modification [9]. This latter process involves using the modal truth criterion to establish additional propositions within the context of the retrieved plan. This option is particularly relevant in the case of possible correctness generalizations, since these generalizations cannot be instantiated as macro operators. The planner will have to specialize them by adding additional ordering and binding constraints.

From the preceding discussion, we see that the spectrum of generalizations offer a corresponding spectrum of tradeoffs for plan reuse; these can be summarized as follows:

Type of Generalization	Cost	Usage
Precondition Generalization with MTC1	$O(n^3)$	M,A
Order Generalization	$O(n^3)$	M,A
Precondition generalization with MTC2	$O(n^4)$	M,A
Disjunctive order generalization [17]	$O(n^5)$	M,A
Precondition Generalization via multiple explanations (Weakest preconditions w.r.t. Tweak MTC)	$O(n^5)$	M,A
weakest preconditions for poss. corr.	NP-hard	A
necessary preconditions for poss. corr.	$O(n^3)$	A

Table 1: Comparison between generalization algorithms. The entrees M and A in the usage field stand for reuse as macrops and reuse by adaptation, respectively

- The different truth criteria can be seen as providing differing biases for EBG. In particular, the more general a truth criterion, the more general the generalizations based on it.
- The more general a necessary correctness generalization, the costlier it is to compute.
- The more general a generalization, the costlier it is to instantiate during planning.
- Necessary correctness generalizations are cheaper to instantiate (use) during planning, than possible correctness generalizations. This is because after checking the preconditions of the generalized plan, the planner has to do additional work to see if any completion of the generalized plan can be applied in the given problem situation.
- Possible correctness generalizations may provide more compact generalizations (and thus tradeoff instantiation cost to storage cost)

6 Conclusions and Related Work

The main contribution of this paper is a unified approach for plan generalization based on the modal truth criteria, which have been originally proposed as a way of formalizing generative planning [1, 6]. The approach can be summarized as follows:

1. Compute the explanation of correctness of a (POPI) plan with respect to the given truth criterion. Represent the explanation in terms of a validation structure
2. Relax the orderings, initial state conditions, and/or bindings of the plan by removing any constraint that is not justified with respect to the validation structure

We showed that this framework gives rise to a spectrum of generalizations including such novel ones as generalizations based on multiple explanations, stronger truth criteria, and truth criteria for possible truth. We have also characterized the spectrum of tradeoffs offered by the various generalizations, including the costs of producing and using them. Although this paper is not the first to explore generalizations of POPI plans (see [2] and [10]), it subsumes and unifies previous work by providing a larger set of generalizations for POPI plans with interesting storage, retrieval and instantiation tradeoffs. In this paper we avoided the issue of whether partial order or total order planning provides the best framework for plan reuse. Both types of planners can benefit from storing and reusing generalized plans discussed in this paper. In [8], we discuss the orthogonal issue of relative merits of basing reuse in a partial order vs. total order planning framework.

Fink and Yang [5] focus on a closely related notion of plan minimization (removing unnecessary sets of steps from the plan, without affecting its correctness), and discuss a different, explanation independent, notion of justification (especially of plan steps) to aid this minimization. An interesting conclusion of their work is that perfect minimization of plans a an NP-complete problem. Our generalization algorithms, in constrast, depend on justification with respect to a specific explanation structure. An interesting avenue of future investigation would involve first minimizing the plan, and then generalizing it. However, the utility of such an approach depends on the likeliness of getting a non-minimal plan to begin with. In [6], we discuss a more general notion of validation structure, that allows multiple supports for each condition, and explore notions of justification with respect to it. Such generalized validation structures may become necessary to represent proofs of correctness of plans with expressive actions that facilitate synergistic effects. Finally, while this paper focused on generalizations based on explanations of plan correctness, we are currently developing similar truth criterion based frameworks for generalizing explanations of other types of target concepts, such as search failures during planning.

References

[1] D. Chapman. Planning for conjunctive goals. *Artificial Intelligence*, 32:333--377, 1987.

[2] S.A. Chien. *An Explanation-Based Learning Approach to Incremental Planning*. PhD thesis, TR. UIUCDCS-R-90-1646 (Ph.D. Thesis). Dept. of Computer Science, University of Illinois, Urbana, IL, 1990.

[3] B. Fade and P. Regnier. Temporal optimization of linear plans of action: A strategy based on a complete method for the determination of parallelism. Technical report, IRIR-Laboratorie Langages et Systemes Informatiques, Universite Paul Sabatier, France, 1990.

[4] R. Fikes, P. Hart, and N. Nilsson. Learning and executing generalized robot plans. *Artificial Intelligence*, 3(4):251--288, 1972.

[5] E. Fink and Q. Yang. A Spectrum of Plan Justifications. In *Working Notes of the AAAI Spring Symposium on Foundations of Automatic Planning: Classical Approach and Beyond, March 1993*

[6] S. Kambhampati. Characterizing multi-contributor causal structures for planning. In *Proceedings of 1st Intl. Conf. on AI Planning Systems*, 1992 (Extended version to appear in *Artificial Intelligence*, Fall 94).

[7] S. Kambhampati. Utility tradeoffs in incremental plan modification and reuse. In *Working Notes of the 1992 AAAI Spring Symposium on Computational Considerations in Supporting Incremental Modification and Reuse*, March, 1992.

[8] S. Kambhampati and J. Chen. Relative utility of basing ebg based plan reuse in partial ordering vs. total ordering planning. Submitted to AAAI-93, 1993.

[9] S. Kambhampati and J.A. Hendler. A validation structure based theory of plan modification and reuse. *Artificial Intelligence*, 55(2-3), June 1992.

[10] S. Kambhampati and S. Kedar. Explanation based generalization of partially ordered plans. In *Proceedings of 9th AAAI*, 1991.

[11] S. Kambhampati and S. Kedar. A Unified framework for Explanation based generalization of partially ordered and Partially Instantiated plans *Artificial Intelligence*, to appear in Fall 1994.

[12] S. Kambhampati and D.S. Nau. On the Nature and Role of Modal Truth Criteria in Planning Tech. Report. ISR-TR-93-30, Inst. for Systems Research, University of Maryland, March, 1993.

[13] D. McAllester and D. Rosenblitt. Systematic nonlinear planning. In *Proceedings of 9th AAAI*, 1991.

[14] T. M. Mitchell, R. M. Keller, and S. T. Kedar-Cabelli. Explanation-based learning: A unifying view. *Machine Learning*, 1(1):47 -- 80, 1986.

[15] S. Minton. Quantitative results concerning the utility of explanation-based learning. In *Artificial Intelligence*, volume 42, pages 363--392, 1990.

[16] S. Minton, M. Drummond, J. Bresina and A. Philips. Total Order vs. Partial Order Planning: Factors Influencing Performance In *Proc. KR-92*, 1992.

[17] R. J. Mooney. Generalizing the order of operators in macro-operators. In *Proceedings of the 5th International Conference on Machine Learning*, pages 270--283, June 1988.

[18] R. J. Mooney and S. W. Bennett. A domain independent explanation-based generalizer. In *Proceedings of the Fifth National Conference on Artificial Intelligence*, pages 551--555, Philadelphia, PA, 1986. Morgan Kaufmann.

[19] M. M. Veloso, M. A. Perez, and J. G. Carbonell. Nonlinear planning with parallel resource allocation. In *Proceedings of the Workshop on Innovative Approaches to Planning, Scheduling and Control*, pages 207--212, November 1990.