On Non-Conservative Plan Modification

Paolo Liberatore[1]

Abstract. The problem of minimally modifying a plan in response to changes in the specification of the planning problem has already been investigated in the literature. In this paper we consider the problem of *non-minimal modification* of plans from a computational point of view. We prove that while the general problem is intractable (that is, replanning from scratch is equally hard), there are scenarios in which the new plan can be built more efficiently (in polynomial time).

1 INTRODUCTION

The STRIPS formalism [2] has been used in the past years to carry on a computational-theoretical analysis of the problem of planning. The basic instance of a problem of planning in STRIPS is defined as: a description of the initial state of the world, a specification of the goal, and a set of *operators* that can be performed in order to achieve the goal. A *plan* is a sequence of operators that, when executed from the initial state, leads to a state where the goal is satisfied. The problem of planning is to find a plan, once known the initial state, the goal and the operators.

It is possible that the specification of the problem is changed during the execution of the plan, and thus a new plan is needed. The following two examples illustrate this issue.

Example 1 *The planning scenario is that of a robot that delivers mail to the various offices on a floor of a building. In the initial state the robot is in the office where the incoming mail is stored. The goal is the state in which all the letters have been delivered and the robot is again in the office where it started.*

The robot bases its planning on the addresses of the letters and on the layout of the floor. For example the plan may be to go first to Office 202, then trough a door to Office 203, and then again in the corridor to Office 209. This may be the shortest plan. However, if someone closes the door between Office 202 and Office 203, the robot needs to build a new plan that takes into account such a change.

In the above scenario, the state of the world is modified both by the actions of the robot and by external causes, whereas an assumption in STRIPS is that the world is modified only by the robot. The following example illustrates a different kind of situation where the plan still needs to be modified.

[1] Dipartimento di Informatica e Sistemistica, Università di Roma "La Sapienza", Via Salaria 113, I-00198, Roma, Italy.
Email: liberato@dis.uniroma1.it
WWW: http://www.dis.uniroma1.it/~liberato

ECAI 98. *13th European Conference on Artificial Intelligence*
Young Researcher Paper
Edited by Henri Prade
Published in 1998 by John Wiley & Sons, Ltd.

Example 2 *This example still involves the deliverer robot. Suppose that, when it arrives to Office 203 to deliver one of the letters, it is informed that its reading unit has made a mistake, and the address on the envelope is that of Office 208. In this case, the robot has a new initial state and a new goal: the current position is Office 203 (and not the room where all the incoming mail is stored), and the goal is to deliver one letter to Office 208 and one to Office 209.*

This situation is different: in this case, the world is not modified, but the plan is no more valid because the robot's knowledge was partially incorrect. Despite the differences, the problem is exactly the same as in the previous example: the plan does not work any more, so a new one is needed. This problem is known as *replanning*, and can be formalized as follows: given an instance of a planning problem, a plan for it, and an update of the instance, find a new plan for the updated instance. The problem has already been addressed in the literature, and a common hypothesis is that of minimal modification of the plan, or conservativism [3]:

> By "conservativism" we mean that the strategy should modify the plan minimally (i.e. salvage as much of the old plan as possible) while accommodating the changes in the specification. [...] [this] is needed to ensure efficiency.

Conservativism means that the new plan should not be built from scratch. The rationale is that part of the old plan should be encoded in the new one, if possible. This way, part of the new plan is found without any additional effort. From an empirical point of view this may look promising. However, the theoretical analysis of the costs of conservative replanning shows that in general it is not more efficient, since the problem may be even more complex that planning from scratch. For example, Nebel and Koehler claim [4]:

> As it turns out, modifying a plan is not easier than planning from scratch. On the positive side, we show that modification does not add any complexity to planning if we consider the *general* case. However, there exists special cases when modifying a plan *conservatively*, i.e. *by using as much of the old plan as possible*, can be harder than creating one from scratch, as we will show. This means that plan modification *is not uniformly as easy as plan generation*. Further, we show that these results also hold if we assume that the old and the new planning situations are similar.

Example 1 and Example 2 can be encoded in a framework of plan modification as follows: there is an instance of a planning

problem, of which we already have a solution (i.e. a plan). Then we modify the initial state and/or the goal, and we need to find a new plan for the modified instance. In Example 1, the new initial state is that in which the robot is in the Office 202 and the door between Office 202 and Office 203 is closed, and the goal is to deliver a letter to Office 203 and one to Office 209. Example 2 is similar, as in the new initial state the robot is in Office 203, while the goal is to deliver one letter to Office 208 and one to Office 209. The part of the old plan that has already been executed is taken into account in the new initial state and in the new goal.

Clearly, without the assumption of conservativism, replanning is at most as hard as planning from scratch, as we can just disregard the old plan. On the contrary, with the assumption of conservativism, replanning may be harder than planning from scratch (see [4] for details). What we prove in this paper is that there are scenarios in which non-conservative replanning is indeed easier than planning from scratch.

2 UPDATES

In this paper we consider scenarios in which an instance of a STRIPS problem may be modified. The simplest case is when the change consists of adding or removing a condition from the initial state or the goal.

Definition 1 *A* simple update *is a condition that is either added or removed from the conditions (either positive or negative) of the initial state or the goal.*

This notion of update takes into account only scenarios in which the initial state or the goal are modified *exactly for one condition*, which is either added or removed. In the more general case, the change may include an arbitrarily large number of elementary updates. A *complex update* is the result of any number of elementary updates. Intuitively, planning in presence of complex updates is a very complex task.

3 RESULTS

The basic scenarios of the examples seen above is: there is an instance of a planning problem, and a plan for it. Then, we face an update. Our task is to build a new plan for the updated instance. The instance, the update, and the plan are known. What is needed is a modified plan. This may be simply done by updating the instance and then determining a plan for it with classical techniques. What we do here is to investigate the usefulness of the knowledge of the old plan.

Note that we do not impose the new plan to be as similar to the old one as possible. Our idea can be intuitively explained as follows. The robot is in an initial state and has a goal. It builds a plan, but in the middle of the execution, it discovers that something in the world has been changed (or there is no change, but its initial knowledge was wrong). Then, it has a new initial state (the state in which it currently is), and possibly a new goal.

In this paper we consider the restriction of STRIPS in which all the operators have only positive postconditions. The problem of planning under this restriction is NP complete [1]. As a result, the problem of updating a plan is in NP, since in the worst case we can just update the original instance and find a new plan completely disregarding the old one. Indeed, it is possible to prove that the old plan does not help in finding the new one, even if we allow only two simple updates.

Theorem 1 *Deciding if there is a plan for an updated instance is* NP *complete, even if we know the old plan and the update is composed only of two elementary updates.*

Informally, the reason is that the result of two elementary updates may be an instance that has no similarities with the old one. In this case, the old plan is completely useless.

This result holds only if the definition of plan is simply "a sequence of operators whose execution from the initial state leads to a state satisfying the goal". This definition of plan can be extended as follows.

Definition 2 *Given an instance of a* STRIPS *planning problem, a* general plan *is a polynomial data structure such that determining the plans for the instance itself and for updated instances is a polynomial-time problem.*

The following theorem proves that general plans do exist, and can be determined at the same cost of regular plans, if we allow only a constant number of updates.

Theorem 2 *Determining a general plan is* NP*, under the hypothesis that only a constant number of elementary updates are allowed.*

This theorem proves that updates can be done efficiently, if the changes are not too big and the fact that updates are possible is taken into account before building the original plan. This is the positive result of this work: it is possible to build a new plan efficiently (in polynomial time), under the conditions above. Finding a general plan is an NP problem, but a. this is equal to the complexity of building regular plans, and b. building a plan for an updated instance becomes a polynomial time problem.

General plans are useful in the case of "small" updates but do not help if updates composed of an arbitrary number of elementary updates are allowed.

Theorem 3 *If there exist general plans for all the complex updates, then* $\Sigma_2^p = \Pi_2^p$*, i.e. the polynomial hierarchy collapses.*

REFERENCES

[1] T. Bylander, 'Complexity results for planning', in *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI-91)*, pp. 274–279, (1991).

[2] R. Fikes and N. Nilsson, 'STRIPS: a new approach to the application of theorem proving to problem solving', *Artificial Intelligence*, **2**, 189–208, (1971).

[3] S. Kambhampati and J. Hendler, 'A validation-structure-based theory of plan modification and reuse', *Artificial Intelligence*, **55**, 193–258, (1992).

[4] B. Nebel and J. Koehler, 'Plan modification versus plan generation: a complexity-theoretic perspective', in *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence (IJCAI-93)*, pp. 1436–1440, (1993).