Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence

Semi-Relaxed Plan Heuristics

Emil Keyder
INRIA
Nancy, France
emilkeyder@gmail.com

Jörg Hoffmann
Saarland University
Saarbrücken, Germany
hoffmann@cs.uni-saarland.de

Patrik Haslum
The Australian National University, NICTA
Canberra, Australia
patrik.haslum@anu.edu.au

Abstract

The currently dominant approach to domain-independent planning is planning as heuristic search, with most successful planning heuristics being based on solutions to delete-relaxed versions of planning problems, in which the negative effects of actions are ignored. We introduce a principled, flexible, and practical technique for augmenting delete-relaxed tasks with a limited amount of delete information, by introducing special fluents that explicitly represent conjunctions of fluents in the original planning task. Differently from previous work, conditional effects are used to limit the growth of the task to be linear in the number of such conjunctions, making its use for obtaining heuristic functions feasible. The resulting heuristics are empirically evaluated, and shown to be sometimes much more informative than standard delete-relaxation heuristics.[1]

Introduction

Planning as heuristic search is currently the dominant approach to domain independent planning. In both satisficing and optimal planning, the most informative heuristics are obtained from the simplified delete-relaxation task, in which negative effects of actions are ignored, and conditions that have been achieved are assumed to stay true throughout the execution of the plan (Helmert and Domshlak 2009; Bonet and Geffner 2001; Hoffmann and Nebel 2001). Even more informative heuristics could be obtained, however, if it were possible to also account for a limited amount of delete information concerning the planning task.

Recent work in this area has shown that this can be done by considering the costs of modified planning tasks with *no deletes*, that nevertheless encode delete information about the original task in their fluents and operators (Haslum 2009). These formulations explicitly represent the truth value of conjunctions c in the original task with new fluents π_c, called π-*fluents*, and modify the initial state, goal, and operators of the planning task to contain these fluents as appropriate. The Π^C formulation (Haslum 2012) makes use of this idea and represents only a *chosen* set C of conjunctions of *arbitrary size*, rather than the set of *all* conjunctions of size $\leq m$ for some m as in the previously proposed Π^m compilation (Haslum 2009). This more fine-grained formulation allows greater efficiency, avoiding wasting resources on the representation of conjunctions that are unnecessary in practice. The drawback to Π^C however, is that it grows exponentially in the number of conjunctions that are considered, and in practice quickly becomes large.

Here, we introduce a similar construction Π^C_{ce} that makes use of conditional effects to limit the growth of the task to be *linear* in $|C|$. While this gain comes at the price of some information loss relative to Π^C, Π^C_{ce} preserves the important property that there exists C such that the optimal delete relaxation cost of Π^C_{ce} is a perfect heuristic for the original problem Π, and can be used to obtain informative heuristics in practice. We discuss the issues that arise in using Π^C_{ce} to compute heuristics, and evaluate the resulting *partial relaxation heuristics*, showing that they improve on the state of the art for satisficing planning.

Background

Our planning model is based on the propositional STRIPS formalization, to which we add action costs and conditional effects. States and operators are defined in terms of a set of propositional variables, or fluents, with a *state* $s \subseteq F$ given by the set of fluents that are true in that state. A *planning task* is described by a 4-tuple $\Pi = \langle F, A, I, G\rangle$, where F is a set of such variables, A is the set of actions, $I \subseteq F$ is the initial state, and $G \subseteq F$ describes the set of goal states, given by $\{s \mid G \subseteq s\}$. Each action $a \in A$ consists of a 4-tuple $\langle \mathsf{pre}(a), \mathsf{add}(a), \mathsf{del}(a), \mathsf{ce}(a)\rangle$ and a cost $\mathsf{cost}(a) \in \mathbb{R}^+_0$. Here, $\mathsf{pre}(a)$, $\mathsf{add}(a)$, and $\mathsf{del}(a)$ are subsets of F; $\mathsf{ce}(a) = \{\mathsf{ce}(a)_1, \ldots, \mathsf{ce}(a)_n\}$ denotes a set of conditional effects, each of which is a triple $\langle \mathsf{c}(a)_i, \mathsf{add}(a)_i, \mathsf{del}(a)_i\rangle$ of subsets of F. If $\mathsf{ce}(a) = \emptyset$ for all $a \in A$, we say that Π is a *STRIPS planning task*.

An action a is *applicable* in s if $\mathsf{pre}(a) \subseteq s$. The result of applying it is $s[a] = (s \setminus (\mathsf{del}(a) \cup \bigcup_{\{i|\mathsf{c}(a)_i \subseteq s\}} \mathsf{del}(a)_i)) \cup (\mathsf{add}(a) \cup \bigcup_{\{i|\mathsf{c}(a)_i \subseteq s\}} \mathsf{add}(a)_i)$, in other words, all conditional effects whose conditions hold are applied to s. A plan is a sequence of actions $\sigma = a_1, \ldots, a_n$ such that applying it in I results in a goal state. The *cost* of σ is $\sum_{i=1}^n \mathsf{cost}(a_i)$, with an *optimal* plan σ^* being a plan with minimal cost.

A *heuristic* for Π is a function h mapping states of Π

[1] This is a short version of the paper of the same name published at ICAPS 2012 (Keyder, Hoffmann, and Haslum 2012).

into $\mathbb{R}_0^+$. The *perfect heuristic* h^* maps each state s to the cost of an optimal plan for s. A heuristic h is *admissible* if $h(s) \leq h^*(s)$ for all s. By $h(\Pi')$, we denote a heuristic function for Π whose value in s is given by estimating the cost of the corresponding state s' in a modified task Π'. We specify Π' in terms of the transformation of $\Pi = \langle F, A, I, G\rangle$ into $\Pi' = \langle F', A', I', G'\rangle$; s' is obtained by applying to s the same transformation used to obtain I' from I. To make explicit that h is a heuristic computed on Π itself, we write $h(\Pi)$.

The delete relaxation Π^+ of a planning task is obtained by discarding the delete effects in all actions and conditional effects. Formally, $\Pi^+ = \langle F, A^+, I, G\rangle$, where $A^+ = \{\langle \mathsf{pre}(a), \mathsf{add}(a), \emptyset, \mathsf{ce}^+(a)\rangle \mid a \in A\}$, where $\mathsf{ce}^+(a) = \{\langle \mathsf{c}(a)_i, \mathsf{add}(a)_i, \emptyset\rangle \mid \mathsf{ce}(a)_i \in \mathsf{ce}(a)\}$, and each action $a^+ \in A^+$ has the same cost as the corresponding action a in A. The optimal relaxation heuristic h^+ for Π is defined as the cost $h^*(\Pi^+)$ of an optimal plan for Π^+.

We denote the powerset of F with $\mathcal{P}(F)$. As in the introduction, in the context of Π^C and Π^C_{ce} we often refer to the fluent subsets $c \in C$ as *conjunctions*.

The Π^C and Π^C_{ce} Compilations

Π^C is constructed from the original planning task Π given a set of subsets C of the set of fluents of the planning task. Each of the sets $c \in C$ is represented in Π^C by a special fluent π_c whose truth value represents whether all of the fluents $p \in c$ are *simultaneously* true. These π-fluents are added to the initial state and goal of the problem, and to the preconditions and effects of all actions, when they appear as subsets of these sets. Furthermore, *representatives* of actions are created that model the situation in which some subset of a set of fluents c is already true, and the application of an action makes the remaining fluents in c true while deleting none of them, thereby also making π_c true.

To ensure that $h^+(\Pi^C)$ is admissible, Π^C must contain action representatives that are capable of making multiple different π-fluents true at once. Π^C therefore constructs an action representative for each possible subset $C' \subseteq C$, leading to an exponential number of action representatives being created. In contrast, Π^C_{ce} achieves the same outcome by creating for each $c' \in C$ a *single* conditional effect. In other words, while Π^C creates an action representative for each possible situation in which a different set of π-fluents is made true by an action, Π^C_{ce} implicitly describes the conditions under which each individual π-fluent becomes true.

Theoretical Properties of Π^C_{ce}

We state some theoretical properties of Π^C_{ce}, considering its optimal delete-relaxation cost $h^+(\Pi^C_{\mathsf{ce}})$ instead of more practical approximations:

Theorem 1 (Consistency and admissibility) *$h^+(\Pi^C_{\mathsf{ce}})$ is consistent and admissible.*

Theorem 2 ($h^+(\Pi^C_{\mathsf{ce}})$ dominates $h^+(\Pi)$) *Given a planning task Π and a set of conjunctions C, $h^+(\Pi^C_{\mathsf{ce}}) \geq h^+(\Pi)$. There are cases where the inequality is strict.*

Though $h^+(\Pi^C_{\mathsf{ce}})$ dominates $h^+(\Pi)$, Π^C_{ce} incurs a slight loss of information compared to Π^C, as there are certain π-fluents that appear as preconditions of representatives of actions in Π^C yet do not appear in the condition of any conditional effect in Π^C_{ce}. For the same set of conjunctions C, we then have that the optimal delete relaxation heuristic for Π^C strictly dominates that obtained with Π^C_{ce}. However, Π^C_{ce} preserves an important property of Π^C, which is that there exists a set C such that the optimal delete relaxation cost of Π^C_{ce} is a perfect heuristic for Π:

Theorem 3 ($h^+(\Pi^C_{\mathsf{ce}})$ is perfect in the limit) *Given a planning task Π, there exists C such that $h^+(\Pi^C_{\mathsf{ce}}) = h^*(\Pi)$.*

Practical Aspects of Using Π^C_{ce}

In order to use Π^C_{ce} in practice, we must develop strategies for choosing the set C. Here we follow a strategy that is similar to that previously proposed in the context of Π^C (Haslum 2012). This strategy attempts to detect ways in which the relaxed plan generated for the initial state of the task will fail in the real problem, and to add conjunctions to C that prevent valid relaxed plans from failing in the same way in the original problem Π. As an example, consider a planning task with goal $G = \{p, q\}$, and a relaxed plan that uses an action a_p to achieve p that deletes q. The addition of $\{p, q\}$ to C will then introduce a π-fluent $\pi_{\{p,q\}}$ which will be part of the goal, and $\pi_{\{p,q\}}$ will be unachievable by a_p since it deletes q. The relaxed plan generated for Π^C_{ce} will then have to achieve p with a different action. This process is iterated until either the relaxed plan obtained for Π^C_{ce} is also a plan for Π, or some resource bound is reached that prevents further conjunctions from being added to C.

A second issue that is peculiar to Π^C_{ce} is the problem of generating high-quality relaxed plans in the presence of conditional effects. To address this problem, we use a graph structure that represents dependencies between the effects included in a relaxed plan, and use it to generalize a previous technique (Hoffmann and Nebel 2001) for detecting when two conditional effects of the same action can be obtained with a single application of the action.

Experimental Results

We evaluated the non-admissible heuristic obtained as the cost of a suboptimal plan for Π^C_{ce} in the context of a state-of-the-art variant of greedy best-first search (Helmert 2006). We varied a parameter x that measures the size of the Π^C_{ce} problem compared to the size of the original delete relaxation problem Π^+. When $x = 1$, no conjunctions are added and Π^C_{ce} is equivalent to Π^+. As x grows, Π^C_{ce} becomes more informative, however the size of the problem representation and the difficulty of computing good relaxed plans in each state grow as well. We tested values of x between 1 and 3.

We ran the resulting planner on problems from the most recent International Planning Competition (IPC), the biennial forum in which top planners compete to solve a fixed set of planning tasks. Using Π^C_{ce} rather than Π^+ results in a more informative heuristic in several sets of tasks (domains), most strikingly in the "Floortile" domain, in which a

robot must paint the floor of a room with alternating colors. The delete relaxation ignores the fact that the robot cannot pass through tiles that it has previously painted, leading to poor performance. Using Π^C_{ce} however, all 20 Floortile IPC benchmark tasks are solved within 5 seconds for $x = 2.5$. In comparison, no other planner participating in the most recent planning competition solves more than 9 instances within the 30 minute time bound. When the available time is shared between our planner and LAMA, the winner of the most recent competition, a total of 264 of the 280 tasks in all the domains can be solved, compared to 250 for LAMA.

Conclusion

Π^C_{ce} is a principled and flexible method for improving delete-relaxation heuristics with a limited amount of delete information, providing a smooth continuum of tradeoffs between problem size and informativeness. Its linear, rather than exponential, growth in the size of C means that it is possible to use Π^C_{ce} in practice to obtain partial delete relaxation heuristics. Our results show that the computational investment is worthwhile in certain domains, where search with heuristics based on Π^C_{ce} very effectively finds solutions for tasks that are not solved by any other planner.

Acknowledgments. Work performed while Jörg Hoffmann was employed by INRIA, Nancy, France. NICTA is funded by the Australian Government as represented by the DBCDE and the ARC (DP0985532).

References

Bonet, B., and Geffner, H. 2001. Planning as heuristic search. *Artificial Intelligence* 129(1–2):5–33.

Bonet, B.; McCluskey, L.; Silva, J. R.; and Williams, B., eds. 2012. *Proceedings of the Twenty-second International Conference on Automated Planning and Scheduling (ICAPS 2012)*. AAAI Press.

Gerevini, A.; Howe, A.; Cesta, A.; and Refanidis, I., eds. 2009. *Proceedings of the Nineteenth International Conference on Automated Planning and Scheduling (ICAPS 2009)*. AAAI Press.

Haslum, P. 2009. $h^m(P) = h^1(P^m)$: Alternative characterisations of the generalisation from $h^{\max}$ to h^m. In Gerevini et al. (2009), 354–357.

Haslum, P. 2012. Incremental lower bounds for additive cost planning problems. In Bonet et al. (2012).

Helmert, M., and Domshlak, C. 2009. Landmarks, critical paths and abstractions: What's the difference anyway? In Gerevini et al. (2009), 162–169.

Helmert, M. 2006. The Fast Downward planning system. *Journal of Artificial Intelligence Research* 26:191–246.

Hoffmann, J., and Nebel, B. 2001. The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research* 14:253–302.

Keyder, E.; Hoffmann, J.; and Haslum, P. 2012. Semi-relaxed plan heuristics. In Bonet et al. (2012).