Analyzing, Learning, and Shaping Planning Search Spaces

Blazej Bulka
Department of Computer Science
University of Maryland, Baltimore County
bulka1@umbc.edu

Introduction

The complexity of a task faced by a planner depends strongly on the search space. The importance of the planner's search space is reflected in the two main classes of classical AI planners: *state-space* planners, which explore the search space by considering totally ordered sequences of directly neighboring states of the world, and *plan-space* planners, which explore the search space by considering incomplete plans consisting of partially ordered actions. The search space also depends on the structure of subgoals (their interactions) and on the representation of knowledge used by the planner (e.g., the level of abstraction at which the planning is done or the expressiveness of the description language). Taking these factors into account can lead to finer-grained classification of planning domains and search spaces.

Knowing the properties of a search space is useful when a human designer creates a new planner or decides which of the existing planners to use. However, knowledge of the properties of a search space may also be useful to the planner itself. In my research, I plan to explore which features of the search spaces can be automatically determined or learned by planners (during repeated executions), and how such information may be used by a planner to improve the search process (e.g., by choosing an appropriate planning technique, abstraction level, or representation).

This research is still in the idea stage: I am in the process of developing my dissertation proposal, and plan to take the preliminary exam (proposal defense) in July, after the conference. Attending the doctoral consortium will give me the ideal opportunity to receive early feedback that can help guide my dissertation research. I also look forward to the opportunity to attend the conference in order to get a better sense of the current state of the field, and to meet other graduate students and senior researchers.

Overview of the approach In my approach, the planner will discover (either through a preliminary analysis or learning) features of the structure of the planning domain. Information about these features can improve the planning process by using them as heuristics, selecting a more appropriate search method, tuning the parameters of the planner, or by changing to a different representation of the domain knowledge (shaping the search space).

In this paper, I propose two approaches to discover features of a planning domain. The first one is a preliminary analysis of the domain description. Although the cost of such analysis would be high, it can be amortized over multiple planning attempts in the future that would benefit from it. The second approach includes observing multiple valid plans (either produced by the planner itself, other planners or humans), and learning the features from the observations. This approach may be particularly effective in discovering statistical features of the search space for the domain (e.g., intermediate states that are often included in a plan).

Detection of search space features

Analysis In my approach, the planner will run a preliminary analysis of the planning domain (i.e., actions, their preconditions, and their effects) in order to discover dependencies among the actions. Such dependencies can later be used to make planning decisions for any specific problem (i.e., any initial state and goals). This approach is inspired by work of Kambhampati, Parker, and Lambrecht (1997) and Hoffmann, Porteous, and Sebastia (2004).

Kambhampati, Parker, and Lambrecht analyzed Graphplan. They point out that the planning graph created by Graphplan is a way of representing the search tree in a compact but approximate manner. Each level represents a set of states in the search space, and the exact path in the state-space can be retrieved, which happens during the extraction of the plan produced by the planner. The representation of the search tree is approximate because it includes only information for certain goals interactions (*mutexes*—i.e., sets of size of two containing mutually exclusive propositions). They also show that there are domains where this information is not sufficient, yet looking for the additional information during each generation of a planning graph is not feasible. This would justify preprocessing, which needs to be done only once since it does not depend on the particular initial state and goals. An open question remains: how to automatically detect whether a particular domain has the property that information in a planning graph created by Graphplan is not sufficient.

Hoffman, Porteous, and Sebastia also did preprocessing in order to find ordering of subgoals, as described later. Their research, however, was dependent on the initial and goal states, which means that the preprocessing has to be executed during each run of the planner. If a more general analysis of the search space's structure (not dependent on initial and goal states) could be performed, the preprocessing could be performed only once. Moreover, the information obtained during such an analysis could be used in multiple ways. The first one would be to perform reachability analysis of a search space (which would allow the planner to reject some paths in the search space and to declare some goals not achievable from certain starting points). Another way would be through the identification of *hub states*: states or preconditions that *always* or *often* appear in valid plans in the domain. This would improve planning by adding hubs as subgoals, or by precomputing actions that lead from one hub state to another one. An extension of this idea would be treating a group of states as a hub (a minimal group of states such that at least one of the states appears in almost all valid plans).

Hoffmann, Porteous, and Sebastia also used heuristics based on parsing techniques (e.g., lookahead). I would also like to explore the similarities between planning, given a domain representation, and parsing given a grammar. Many efficient parsers do not analyze the grammar while parsing: they usually use precomputed information such as a parser table. Most parsers, however, cannot process arbitrary grammars. Instead, they are limited to a few basic classes like context-free or LALR(n) grammars. It would be interesting to identify and analyze analogous classes of planning spaces. Such classes of spaces could either be based on specific structures of subgoals or relations among planning states (especially in state-space search). (The relation of planning to context-grammars has already been noticed by Erol, Hendler and Nau (1994), but they mostly focused on HTN planning, which has an explicit structure among actions.)

Learning The search space analysis described above may be most efficient for features that are present in every valid plan, but an approach that considers only such features may be too restrictive (especially if we consider preprocessing of the space for all initial states). However, detection of features that are often (but not always) included in plans may require enumerating a large part of the search space. Therefore, it could be feasible to learn the existence of such hub nodes by observing multiple generated plans for the same domain, instead of analyzing.

Many domains (including benchmark domains for planning) are reported to have regularities in their local search topology. For example, Hoffmann (2003) analyzed heuristics that ignore delete lists of operations in the context of phenomena that occur in the local search topology. Similarly, Haslum and Geffner (2000) showed that the successful use of heuristics to guide the planning process can be linked to the regularities in the domains. Learning the local features and regularities of the search space can possibly lead to the development of good, domain-specific heuristics. Moreover, for a search guided by a heuristic, it could also be possible to incrementally learn a better heuristic.

Shaping the search space

Having discovered the features of a planning domain (e.g., by analyzing the search space or by learning the features from observation of valid plans), a planner may take advantage of this knowledge by reasoning about planning in this domain (meta-planning) and shaping the search space.

One way to change the shape of the search space is to change the representation of facts. My idea is partially inspired by work by Haslum and Jonsson (2000), who focused on the idea of removing redundant operators *given an initial state*. I believe that planning could be improved by removing particular effects of the operators, while still preserving correctness comparing to the plan with non-modified operators (*redundant effects*). It may also be possible to remove operators that are rarely used in plans (as learned by the planner). As a result, removing some operators would allow easier preprocessing of the domain regardless of the initial state (e.g., if it would reduce the class of a search space to a simpler one) at the expense of producing less optimal plans in terms of the plan's length.

Another possible way of changing the search space is the selection of an alternative search method for the whole plan or part thereof. (A similar approach was used in the FLECS algorithm (Veloso & Stone 1995), but the selection condition in that case was given by the designer, not learned by the planner.) For example, it may be possible to learn which search method performs the best given the set of detected features of the planning domain. Alternatively, the planner could learn a hierarchical representation of the problem (inspired by work by Knoblock (1994)), decide to first solve the planning problem at a higher level, and then solve the subproblems independently (similar to HTN planning).

Related work

The problem of structure and interactions among the goals has previously been analyzed by Barrett and Weld (1994). They described different classes of planning domains, and tested the behavior of both total-order and partial-order planners on these domains. Hoffmann, Porteous, and Sebastia (2004) described different kinds of possible ordering relations between subgoals. They also introduced the concept of *landmarks*, which can be perceived as a particular type of hub states mentioned earlier in this paper. (A landmark is a subgoal that must be satisfied at some point in *every* plan in the domain; my definition of a hub state is slightly broader and also includes the goals or states that are included in *most* plans in the domain.) Additionally, Hoffmann in his earlier work described how local features of the topology of the search space (as opposed to "global" landmarks and orderings) may influence planning, and how such features of the space can be detected and used in FF planner (Hoffmann 2001; 2003). The work by Smith and Peot (1996) described analyzing the search space by using *operator graphs* to avoid recursion and prune it. Preanalysis of the search space is also used in the work by Fox and Long (1998), which finds state invariants using type inference. Their later work (Fox & Long 1999; Porteous, Long, & Fox 2004) focuses on finding regularities in planning problems (symmetry and almost symmetry).

Other work has explored the appropriate level of abstraction during planning. This issue is the focal point of HTN planning (Sacerdoti 1975), where different levels of abstraction are explicitly represented. Nevertheless, information about an abstract hierarchy may also be used in planners that do not support hierarchy explicitly: for example, Veloso and Stone (1995) mention treating some intermediate goals as *milestones*, which divide states in a plan into independent groups. Each such group can be treated as a goal at a higher level of abstraction. Information about these groups is obtained from an external source. (Veloso and Stone mention work by Knoblock (1994) as a method to generate abstractions automatically.) In fact, Kambhampati (1995) presents a comparison of "pure" partial-order planners (no use of information regarding hierarchy) and HTN planning, and discusses the advantages of having an explicit representation for abstract or higher-level goals.

Different ways of representing the same domain are also a popular research topic, especially the tradeoff between the complexity of planner's data structures and the size of the search space (Kambhampati & Yang 1996; Kambhampati, Parker, & Lambrecht 1997).

There is also a body of work on learning in planning. Learning appropriate heuristics by planners based on their previous experiences in planning can be found in work by Likhachev and Koenig (2005). Botea *et al.* (2005) presented an approach that exploits the underlying domain structure, and learns ordering of operators (actions) and combining them into groups (macros) by observing plans in the domain.

Status of the work

This paper presents preliminary work done under supervision of my advisor, Prof. Marie desJardins. I plan to have the ideas further extended by July 2006 by providing details of the proposed methods and examples of planning domains where these methods are applicable. At this time, this work should be developed far enough to form a Ph.D. thesis proposal.

References

Barrett, A., and Weld, D. S. 1994. Partial-Order Planning: Evaluating Possible Efficiency Gains. *Artificial Intelligence* 67(1):71–112.

Botea, A.; Müller, M.; and Schaeffer, J. 2005. Learning Partial-order Macros from Solutions. In *Proceedings of the Fifteenth International Conference on Automated Planning and Scheduling (ICAPS-05)*, 231–240.

Erol, K.; Hendler, J.; and Nau, D. S. 1994. HTN Planning: Complexity and Expressivity. In *Proceedings of Eleventh National Conference on Artificial Intelligence (AAAI-94)*, 1123–1128.

Fox, M., and Long, D. 1998. The Automatic Inference of State Invariants in TIM. *Journal of Artificial Intelligence Research (JAIR)* 9:367–421.

Fox, M., and Long, D. 1999. The Detection and Exploitation of Symmetry in Planning. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99)*.

Haslum, P., and Geffner, H. 2000. Admissible Heuristics for Optimal Planning. In *Proceedings of the Fifth International Conference on Artificial Intelligence Planning and Scheduling (AIPS-2000)*, 140–149.

Haslum, P., and Jonsson, P. 2000. Planning with Reduced Operator Sets. In *Proceedings of the Fifth International Conference on Artificial Intelligence Planning and Scheduling (AIPS-2000)*, 150–158.

Hoffmann, J.; Porteous, J.; and Sebastia, L. 2004. Ordered Landmarks in Planning. *Journal of Artificial Intelligence Research (JAIR)* 22:215–278.

Hoffmann, J. 2001. Local Search Topology in Planning Benchmarks: An Empirical Analysis. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-01)*, 453–458.

Hoffmann, J. 2003. Where Ignoring Delete Lists Works: Local Search Topology in Planning Benchmarks. Technical Report 185, Institute for Computer Science, Albert-Ludwigs-University, Freiburg, Germany.

Kambhampati, S., and Yang, X. 1996. On the Role of Disjunctive Representations and Constraint Propagation in Refinement Planning. In *Proceedings of the Fifth International Conference on Principles of Knowledge Representation and Reasoning (KR-96)*, 135–146.

Kambhampati, S.; Parker, E.; and Lambrecht, E. 1997. Understanding and Extending Graphplan. In *Recent Advances in Planning: Fourth European Conference on Planning (ECP-97)*, 260–272.

Kambhampati, S. 1995. A Comparative Analysis of Partial Order Planning and Task Reduction Planning. *SIGART Bulletin* 6(1):16–25.

Knoblock, C. A. 1994. Automatically Generating Abstractions for Planning. *Artificial Intelligence* 68(2):243–302.

Likhachev, M., and Koenig, S. 2005. A Generalized Framework for Lifelong Planning A*. In *Proceedings of the Fifteenth International Conference on Automated Planning and Scheduling (ICAPS-05)*.

Porteous, J.; Long, D.; and Fox, M. 2004. The Identification and Exploitation of Almost Symmetry in Planning Problems. In *Proceedings of the 23rd UK Planning and Scheduling SIG*.

Sacerdoti, E. 1975. The Nonlinear Nature of Plans. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 206–214.

Smith, D. E., and Peot, M. A. 1996. Suspending Recursion in Causal-link Planning. In *Proceedings of the Third Artificial Intelligence Planning Systems Conference (AIPS-96)*, 182–190.

Veloso, M., and Stone, P. 1995. FLECS: Planning with a Flexible Commitment Strategy. *Journal of Artificial Intelligence Research (JAIR)* 3:25–52.