Randward and Lamar: Randomizing the FF Heuristic

Alan Olsen and **Daniel Bryce**
alan@olsen.org, daniel.bryce@usu.edu
Utah State University
Logan, UT

Abstract

We present two planners, Randward and Lamar, which are respectively built upon the Downward and LAMA planners. The technique developed in these planners is to randomize heuristic construction so that heuristic plateaus are less frequent and the bias introduced by poor, arbitrary nondeterministic choices in heuristic construction is removed. Specifically, we randomize the FF heuristic, naming the resulting planner Randward, and with the addition of the landmark count heuristic, name it Lamar. Within the FF heuristic, we randomize planning graph construction, which is akin to a random-walk in the relaxed planning space. From this randomized planning graph, we extract a relaxed plan heuristic (exactly as would FF).

Introduction

Randomization in planning has shown to be effective at overcoming bias in both search (Gerevini, Saetti, and Serina 2003; Nakhost and 0003 2009) and heuristics (Bryce, Kambhampati, and Smith 2008). We present a unique point of randomization in the FF heuristic (Hoffmann 2001), which is then used in two planners, Randward and Lamar. Both planners are built upon the Fast Downward (Helmert 2006) and LAMA (Richter and Westphal 2008) planners, and differ in that Randward does not use the landmark count heuristic, but Lamar does.

The FF heuristic involves constructing a planning graph to solve the relaxed planning problem, and the resulting relaxed plan is used to compute a search heuristic as well as identify preferred operators. The planning graph construction is deterministic in the FF heuristic, and resembles the forward chaining algorithm for Horn clause inference. The algorithm associates with each action a count of the number of unsatisfied preconditions (initially all of its preconditions), and initializes a list of propositions to process. Processing a proposition involves decrementing the number of unsatisfied preconditions for each action using it as a proposition. Once an action's count reaches zero, its effects are added to the proposition list. Processing the propositions as a FIFO queue resembles the traditional planning graph that inserts all actions at the same level, and then moves the next level in a breadth-first manner.

Randomizing the order in which propositions are processed will lead to a random walk in the relaxed planning space. While we do not extract relaxed plans by randomization, we extract a relaxed plan as soon as one exists in the planning graph. Random walks can thus randomize the heuristic and preferred operators. As we have seen in our prior work (Bryce, Kambhampati, and Smith 2008), randomization is useful for overcoming poor decisions (such as proposition order) in a heuristic because, for example, a parent node may have an uncharacteristically low heuristic value, and its child may suffer from the same bias. However by randomizing the heuristic, a child's heuristic is unlikely to suffer from the same bias as the parent, and the probability that an entire path suffers from the same bias is very low.

In the following, we describe the approach taken in Randward and Lamar through an example, discuss preliminary results on the IPC-2008 domains, and outline directions for future work.

Relaxed Planning Graph Expansion Order

Traditionally, relaxed planning graph generation has been explained in a breadth-first fashion, as in (Bryce and Kambhampati 2007). Each proposition in the initial layer is considered towards adding applicable relaxed-actions to the graph at that layer. The add-effects of those actions, as well as all propositions currently true, are added to the next layer. This new set of propositions is used to determine which relaxed-actions are applicable in that layer. This continues until all goal propositions are true, and then a relaxed plan is generated by tracing back through the graph. Figure 1 gives one example of a relaxed search space. Figure 2 shows the breadth-first expansion of propositions in layer P_0 yielding the actions in layer A_0 and ultimately reaching the goal g in layer P_1.

Alternatively, a relaxed planning graph can be generated in a random order. The only difference is which proposition to consider next. As soon as an action is added to the graph, its add-effects immediately become candidates for expansion. When expanding randomly, any random proposition is considered next towards applying actions in its layer. Figure 3 and Figure 4 show two random expansions of the same relaxed search space from the previous scenario. Notice how randomizing removes bias in preferred operators (i.e. which actions are first in the relaxed plan). Also, while the order in which propositions are listed can bias which plans are gen-

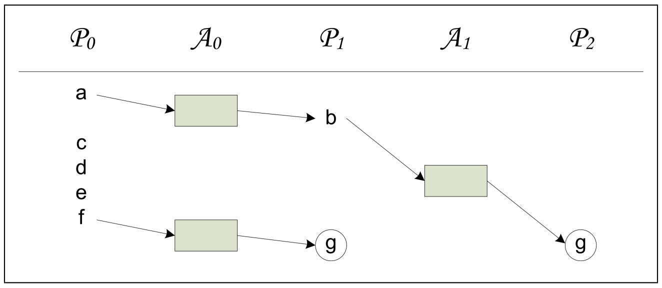

Figure 1: An example of a relaxed search space

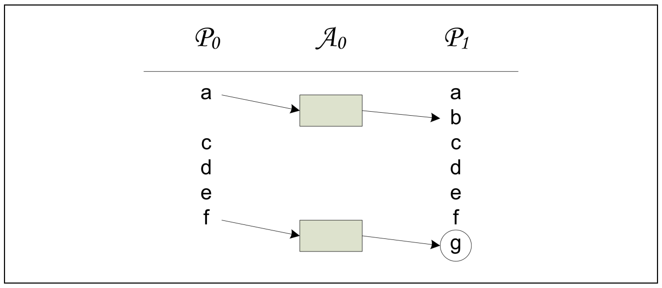

Figure 2: Breadth-first expansion of a relaxed planning graph

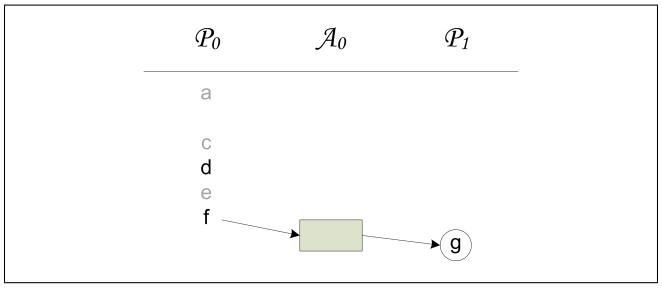

Figure 3: One possible random expansion of a relaxed planning graph (unexpanded propositions in gray)

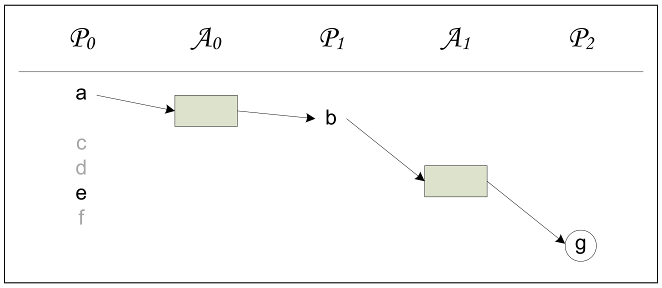

Figure 4: Another random expansion of a relaxed planning graph (unexpanded propositions in gray)

erated by breadth-first expansion, it does not bias those generated through random expansion.

Order of expansion does not effect reachability of goal propositions but it does effect the length of the relaxed plan. Because delete-effects are ignored, actions can only progress the graph towards the goal. Therefore, reaching the goal depends only on applying the right actions, not the order in which they are applied. However, some actions help progress the graph towards the goal sooner than others. Applying those actions first will generate a shorter relaxed plan. Breadth-first is guaranteed to find the step-optimal relaxed plan however random expansion is not guaranteed to. And neither approach is guaranteed to always return the most accurate approximate distance to the goal.

It would be ideal to know which proposition should be considered next to produce the relaxed plan that most accurately approximates the distance to the goal. However, without that knowledge, randomly picking the next proposition avoids the bias inherent in breadth-first expansion.

Results

Each of the IPC-2008 problems were run on a Linux machine with one dedicated 1.95 GHz processor, 2 GB memory, and a time limit of 30 minutes. LAMA and Lamar were also tested without the landmark count heuristic - in which case they are called FF and Randward, respectively. Randomized algorithms were tested on 5 different random seeds.

Table 1 shows that Randward is able to complete more problems than FF, particularly for the elevators and transport domains. This means that simply randomizing the expansion order of the FF heuristic is an improvement.

When landmarks are introduced, however, we see that Lamar completes less problems than LAMA. To expand propositions randomly, LAMA's list of propositions had to be replaced with a priority queue in Lamar. When Lamar's implementation is given breadth-first priorities for each proposition (listed as "LM-BFS" in the table) it performs worse than Lamar. Once again, random expansion is an improvement over breadth-first expansion. Therefore, it is likely that the difference between LAMA and Lamar is implementation related.

While LAMA performs the best, over all, it can still be seen between FF and Randward, and Lamar and LM-BFS, that randomly expanding the relaxed planning graph improves the FF heuristic.

Future Work

Each randomly generated relaxed plan is only one sample from the set of all valid relaxed plans in the relaxed search space and may not be sufficient to get an accurate heuristic. Aggregating over several samples, by taking the average or the minimum, may prove to give a better heuristic value.

Also, if a simple heuristic for relaxed plan expansion exists, then it could be used to perform an A* search over the relaxed planning space. This wouldn't solve the problem of finding the most accurate relaxed plan, however it would be able to return the step-optimal solution quicker than breadth-first.

Domain	**FF**	**Randward**	**LAMA**	**Lamar**	**LM-BFS**
cybersec	29	29 (1)	**30**	29.4 (0.5)	**30**
elevators	16	**26.6 (1.3)**	25	26.4 (1.1)	20
openstacks	30	30	30	30	30
openstacks-adl	30	30	30	30	30
parcprinter	12	14.6 (0.9)	**18**	16.2 (0.4)	16
pegsol	**30**	29 (0)	**30**	29.2 (1.1)	28
scanalyzer	28	27.4 (0.5)	**30**	**30**	**30**
sokoban	24	22.8 (1.1)	**25**	22.6 (0.5)	21
transport	21	25.4 (0.5)	**30**	27 (1)	29
woodworking	29	**30**	29	**30**	**30**
Total	249	264.8 (2.3)	**277**	270.8 (2.4)	264

Table 1: Number of IPC-2008 problems solved (30 instances per domain, totaling 300), each given a 30-minute time limit. Randomized algorithms are averaged over 5 seeds with standard deviation in parentheses.

Conclusion

A relaxed planning graph can be expanded in many different ways - two of which have been explained here. Traditionally only a breadth-first approach has been used, which finds the step-optimal relaxed plan but maintains a bias towards certain relaxed plans. A random approach does not assure step-optimality but has shown to produce a better heuristic than breadth-first alone.

Acknowledgments

This work was supported by DARPA contract HR001-07-C-0060.

References

Bryce, D., and Kambhampati, S. 2007. A tutorial on planning graph based reachability heuristics. *AI Magazine* 28(1):47.

Bryce, D.; Kambhampati, S.; and Smith, D. 2008. Sequential monte carlo in probabilistic planning reachability heuristics. *AIJ* 172(6-7):685–715.

Gerevini, A.; Saetti, A.; and Serina, I. 2003. Planning through stochastic local search and temporal action graphs in lpg. *J. Artif. Intell. Res. (JAIR)* 20:239–290.

Helmert, M. 2006. The fast downward planning system. *Journal of Artificial Intelligence Research* 26(1):191–246.

Hoffmann, J. 2001. FF: The fast-forward planning system. *AI magazine* 22(3):57.

Nakhost, H., and 0003, M. M. 2009. Monte-carlo exploration for deterministic planning. In Boutilier, C., ed., *IJCAI*, 1766–1771.

Richter, S., and Westphal, M. 2008. The lama planner using landmark counting in heuristic search. *Proceedings of the IPC*.