A Local Monte Carlo Tree Search Approach in Deterministic Planning

Fan Xie and **Hootan Nakhost** and **Martin Müller**
University of Alberta
Athabasca Hall, University of Alberta
Edmonton, AB, CA T6G 2E8

Abstract

Much recent work in satisficing planning has aimed at striking a balance between coverage - solving as many problems as possible - and plan quality. Current planners achieve near perfect coverage on the latest IPC benchmarks. It is therefore natural to investigate their scaling behavior on more difficult instances. Among state of the art planners, LAMA (Richter, Helmert, and Westphal 2008) is able to generate high quality plans, but its coverage drops off rapidly with increasing problem complexity. The Arvand planner (Nakhost and Müller 2009) scales to much harder instances but generates lower quality plans. This paper introduces a new algorithm, Monte Carlo Random Walk-based Local Tree Search (MRW-LTS), which uses random walks to selectively build local search trees. Experiments demonstrate that MRW-LTS combines a scaling behavior that is better than LAMA's with a plan quality that is better than Arvand's.

Introduction

Arvand (Nakhost and Müller 2009) was the first planning system that applied Monte Carlo Random Walks (MRW) to deterministic classical planning. In Arvand, random walks are used to explore the neighborhood of a search state before jumping to the best found endpoint according to an heuristic evaluation function. Exploration helps to overcome the problem of local minima and plateaus. Jumping greedily exploits the knowledge gained by the random walks. Using the combination of Monte Carlo random walks and jumping, Arvand can often escape from local minima and plateaus and quickly solve even problems that are hard for other current planners such as LAMA and FF (Hoffmann and Nebel 2001) which use more systematic search methods. However, one disadvantage of Arvand's random walk and jump algorithm compared to more classical planners is that it often produces long, low quality plans.

The main motivation of this paper is to find an intermediate approach between global systematic search, as in LAMA or FF, which has higher plan quality but poorer scaling, and using local random walks with jumping, as in Arvand, which scales better to larger problems, but often at the cost of poor plan quality. *Monte Carlo Random Walks based Local Tree Search (MRW-LTS)* is a combination of systematic local tree search with Monte Carlo random walks. Local tree search is used to improve the quality of possible plan continuations before jumping to the next endpoint of a random walk as in Arvand. The aim of this local tree search is to prevent the overall search process from veering too far away from a good plan. The method sacrifices much of the planning speed of Arvand for the goal of achieving higher plan quality.

Monte Carlo Random Walks based Local Tree Search (MRW-LTS)

Local Tree Search (LTS)

The main motivation of the local tree search is to optimize the paths generated by random walks before each jump. The search tree grows selectively towards states where the h-values returned from random walks are more promising. Further random walks are biased towards starting from promising nodes in the search tree. LTS consists of three parts: selection, expansion and backpropagation.

The algorithm selects a leaf node by following an ϵ-greedy strategy in each node. However, new nodes are visited once to gain some initial knowledge. If all children of a node have had one visit, then the algorithm selects the child with minimum h-value with probability $1 - \epsilon$, and a random child with probability ϵ. LTS grows a very selective, unbalanced tree from the current search state. To save time and memory, the algorithm only expands nodes when they are visited the second time. After each random walk, the heuristic value reached at its end state is propagated up towards the root node as long as the new value improves the evaluation of those nodes.

Jump and Restart Policy

Arvand jumps after at most 2000 random walks, or as soon as *acceptable progress* (Nakhost and Müller 2009) is made. Jumping after "acceptable progress" makes Arvand faster in finding goal states. In contrast, MRW-LTS never jumps early. There are two reasons for disabling such an early jump: first, since the search tree grows towards more promising states, the probability for random walks to hit even better states often increases over time. The second reason is that the maximum length of random walks is decreased to a small

value after a lower h-value (Nakhost and Müller 2009) is found. Through the combination of the search tree and short random walks, MRW-LTS very often finds shorter paths to states with the same or better h-value.

MRW fails and restarts when the minimum obtained h-value does not improve over several search steps, or when it reaches a dead-end state. The restart policy of MRW-LTS is more strict: If the minimum h-value does not improve within a single search step, the search restarts from s_0.

Experiments

Experiments include tests on all IPC-2008 benchmarks, and on scaled harder problems generated by IPC-2008 domain generators. Experiments on IPC-2008 were run on a 3 GHz machine and experiments on scaled hard problems were run on a 2.5 GHz machine. The runtime limit was 30 minutes and the memory limit 2 GB for each problem. Results for planners which use randomization are averaged over 10 runs per instance.

Experiments compare four planners: LAMA, the winner of IPC-2008, as well as LAMA with Aras (Nakhost and Müller 2010), Arvand and a preliminary planner Arvand-LTS based on MRW-LTS. For improved performance, the latter three programs run alternately with Aras to improve each plan that is found.

In Arvand and LAMA, the cost of the best known plan before using Aras is used for pruning the search. For Arvand-LTS, increased diversity is more useful, so it uses a different bounding strategy. Arvand-LTS can run either with or without pruning based on the cost of the best plan. It first runs 3 times with and without pruning respectively, and prefers the version which computed the smallest cost plan, however, with with probability 0.2 randomly choose in each run.

In the experiment on IPC-2008 benchmarks, the total scores of LAMA, LAMA with Aras, ARVAND and ARVAND-LTS are 233.96, 240.92, 228.4 and 234.2 respectively. Both ARVAND and ARVAND-LTS are competitive against LAMA. Most of the original problems in IPC-2008 are easy for these planners, which solve around 90%. The following four domains of IPC-2008 have scalable problem generators which allow the creation of harder instances: Elevator, Openstacks, Transport and Woodworking. Figure 1 shows the average coverage and IPC-2008 score of LAMA, Arvand and Arvand-LTS on the scaled Elevator problems. The scores are calculated by averaging solved problem scores, and unsolved runs are ignored here to highlight plan quality. For large problems, Monte-Carlo Random

Figure 1: Average coverage (up) and scores (bottom) of LAMA, Arvand and Arvand-LTS on scaled hard problems of Elevator. The x-axis value is the number of passengers, which is gradually increased to make problems harder.

Walk based approaches have much better coverage because of the random walk local search which scales better than systematic search. Systematic search can be slow to escape from extensive local minima/plateaus, while random walks lead to bigger jumps that help get away from such traps. However, LAMA, if it solves the problem, generally creates a better quality plan.

The plan produced by local search in Arvand-LTS alternates between sequences generated systematically by local tree search, and random walks. Arvand-LTS balances coverage and plan quality by combining the benefit of Monte Carlo random walks in solving problems with the ability of systematic search algorithms of producing shorter paths.

Conclusion

Planning algorithms must address tradeoff between coverage and plan quality. Current state of the art planners seem to emphasize either one or the other. The latest International Planning Competition, IPC-2008, focused on plan quality while the Arvand system focused on coverage. Arvand-LTS represents a balanced approach by combining random walks for increased coverage with systematic local search for plan quality.

Important directions for future work are: 1. to increase speed by introducing a measure such as "acceptable progress" 2. To generate better quality plans, action costs should also be considered in the child selection function of local tree search.

References

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

Nakhost, H., and Müller, M. 2009. Monte-Carlo exploration for deterministic planning. In *Proc. IJCAI 09*, 1766–1771.

Nakhost, H., and Müller, M. 2010. Action elimination and plan neighborhood graph search: Two algorithms for plan improvement. In *Proc. ICAPS-2010*, 121–128. Toronto, CA: AAAI Press.

Richter, S.; Helmert, M.; and Westphal, M. 2008. Landmarks revisited. In *Proc. AAAI-08*, 975–982.