Exploration and Combination: Randomized and Multi-Strategies Search in Satisficing Planning

Fan Xie
Computing Science, University of Alberta
Edmonton, Canada
fxie2@ualberta.ca

Introduction

Heuristic (Informed) Search takes advantage of problem-specific knowledge beyond the definition of the problem itself to find solutions more efficiently than uninformed search, such as Breadth-First Search (BFS) and Depth-First Search (DFS). We design domain-dependent search algorithms to plan tasks. However, the domain-dependent design pattern cannot be applied to fully automatic domains, such as robots that need to plan unknown tasks. AI Planning, as a special case of heuristic search, emphasizes domain-independent approaches for solving such problems.

Satisficing planning generally can handle harder problems than optimal planning. Solution optimality is not required in satisficing planning: finding a good suboptimal solution suffices. Recently, there has been great progress in satisficing planning: in the three last international planning competitions (IPC), top planners solved more than 90% of the tasks. Many problems are inspired by applications, such as oil industry, transportation and molecular biology.

Considering that the top planners perform quite well over current IPC benchmarks, it is natural to check the scaling behavior of the current search algorithms as well as the reasons why the top planners do not perform well on these remaining unsolved problems. Most current state of the art satisficing planners use heuristic search techniques. During the past decades, the planning community spent a lot of effort in developing strong domain-independent heuristics to fit into the classical heuristic search algorithms. We now have many good domain-independent heuristics such as FF, causal graph (CG) and context-enhanced additive (CEA). However, because of the generality of domain-independent planning, it is hard to develop domain-independent heuristics as strong as domain-specific heuristics. It is not realistic to expect that all hard planning problems can be solved simply by combining domain-independent heuristics with classic search algorithms such as Hill-Climbing and Greedy Best First search. Since domain-independent heuristics might be uninformed or even ill-informed sometimes, we need more sophisticated search algorithms where heuristics are not reliable. Recently, more effort is spent in developing more powerful search algorithms, such as EHC (Hoffmann and Nebel 2001) and Random-Walk Planning (Nakhost and Müller 2009), as well as search enhancements such as preferred operators (Richter and Helmert 2009) and sophisticated explorative probes (Lipovetzky and Geffner 2011).

There is still a lot of space to improve for current state of the art search algorithms. A typical algorithm is Greedy Best First Search (GBFS). GBFS always expands a node n that is closest to a goal state according to a heuristic h. Misleading or uninformative heuristics can massively increase the time and memory complexity of such searches. In this abstract, we are going to use GBFS as a typical example to talk about two problems closely related to my research, which also commonly exist in heuristic search algorithms: early mistakes and uninformative heuristic regions.

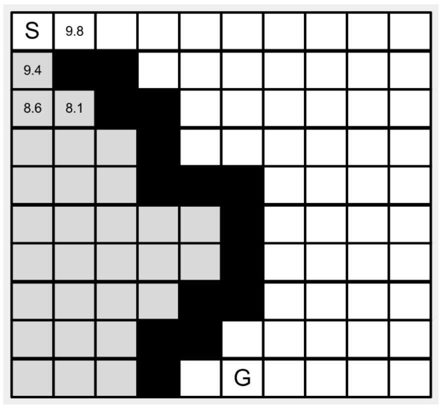
Early Mistakes (EM) and Uninformative Heuristic Regions (UHR)

Early mistakes (EM) are mistakes caused by heuristic functions in evaluating nodes at the same shallow levels of the search tree. It causes the root nodes of bad sub-trees (have no solution or only hard-to-find solutions) to have lower heuristic values than root nodes of good sub-trees, which efficiently lead to a solution.

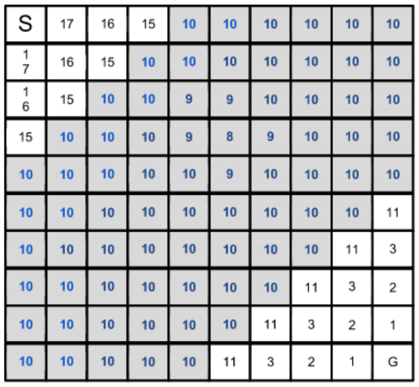
Figure 1 (a) is a typical illustration of what early mistake is. The task is to reach G from the start state S. Since GBFS with an euclidean distance heuristic will expand the h-value 9.4 state first, it has to expand the whole left sub-tree before moving to the correct right sub-tree. This example illustrates a costly early mistake of being greedy on heuristics. Similar costly behavior is not limited to GBFS. Other deterministic greedy algorithms such as Hill Climbing also have the same problem. Techniques addressing the early mistake problem all involve some non-greedy exploration. Some successful approaches include K-BFS (Felner, Kraus, and Korf 2003), which expands first k best nodes in the open list and inserts their successors into the open list, Diverse-BFS (Imai and Kishimoto 2011), which expands some nodes with non-best h-value with nonzero probability, and MRW (Nakhost and Müller 2009), which adopts random operator selecting and restarting from the initial state.

Uninformative heuristic region (UHR) includes local minima and plateaus. A local minimum is a state with minimum h-value within a local region, which is not a global minimum (which is h = 0 in the case of solvable planning problems). A plateau is an area of the state space where all states have the same heuristic value.

Consider Figure 1 (b). The grey area contains a local minimum with h = 8 and a plateau with h = 10. h provides no guidance within the grey area, so GBFS needs to visit all grey states, an inefficient breadth-first behaviour. Figure 2 (left) shows a typical BFS style search behavior in a local minimum using GBFS and h^{FF} (Hoffmann and Nebel 2001). While some branches are deeper than others, the search is basically exhaustive. Figure 2 (left) visualizes how the GBFS search tree grows in one uninformed heuristic region on the instance #02 from IPC domain 2011-barman while searching from h = 24 towards h = 23, and escaping from three uninformed heuristic regions including this takes 90% of the search time that GBFS spends in solving this planning instance. Because GBFS's very inefficient escaping from UHRs, it is a natural choice to switch to a secondary search strategy, which is better at escaping from uninformative heuristic regions.



(a) Example of early mistakes with GBFS.



S	17	16	15	10	10	10	10	10	10
17	16	15	10	10	10	10	10	10	10
16	15	10	10	9	9	10	10	10	10
15	10	10	10	9	8	9	10	10	10
10	10	10	10	10	9	10	10	10	10
10	10	10	10	10	10	10	10	10	11
10	10	10	10	10	10	10	10	11	3
10	10	10	10	10	10	10	11	3	2
10	10	10	10	10	10	11	3	2	1
10	10	10	10	10	11	3	2	1	G

(b) Example of an uninformative heuristic region.

Figure 1: The effect of early mistakes and uninformative heuristic regions. For all grid states generated by GBFS, h-values are shown in the center. (a) h-values are euclidean distances. GBFS makes an early mistake, expanding the 9.4 state, and needs to expand all grey states before visiting the correct subtree. (b) The grey squares contain a local minimum with $h = 8$ and a $h = 10$ plateau. GBFS needs to expand all grey squares to find an escape.

Objectives and Methods

The ultimate goal in my research is to scale the current state of the art search algorithms to larger problems and solve the current hard problems more efficiently. The classical search algorithms or strategies, such as GBFS, Hill-Climbing and Any-time weighted A*, are all deterministic and strongly depends on the evaluation functions. While algorithms with randomized exploration, such as Monte Carlo Tree Search (Kocsis and Szepesvári 2006), have been well studied and successfully applied in the fields of challenging game-playing and probabilistic planning, there is not yet enough attention to applying them in satisficing planning. The randomized algorithms certainly have some potential addressing problems such as early mistake in deterministic satisficing planning. We wish to apply some randomized methods to deterministic satisficing planning in order to build more robust and efficient planning systems. Since some satisficing planning instances can be very hard and very different from each other, it is not easy to develop one single algorithm to handle all cases. Another interesting direction in my research is to build loosely-coupled portfolio or tightly-coupled multi-strategies systems that combine search strategies/algorithms addressing different types of problems together. The main challenge is to balance the trade-off between the benefit that we gain from and the overhead that we pay for combining different algorithms into the system.

Recent Progress

In this section, we present two published work in ICAPS (one in 2012, one accepted for 2013) and one submitted work to IJCAI-2013.

Planning via Random Walk-Driven Local Search

Objective/Motivation: Most successful current satisficing planners combine several complementary search algorithms. Examples range from portfolio planners such as Fast Downward Stone Soup (Helmert, Röger, and Karpas 2011) and loosely coupled parallel planners such as ArvandHerd to systems which alternate several search strategies, such as Fast-Forward (Hoffmann and Nebel 2001) and Fast Downward (Helmert 2006). Using local random walks with jumping, as in Arvand, scales better to larger problems, and at the same time best-first search planners work much better in these domains where specific action sequences need to be discovered in order to make progress. Can we further improve both types of planners by combining the two different search strategies?

Method/Approach: We present the new algorithm Random Walk-Driven Local Search (RW-LS) (Xie, Nakhost, and Müller 2012), which is a mixing strategies algorithm that combines deterministic local GBFS and explorative random walk. Compared with the MRW (Nakhost and Müller 2009) algorithm, which does not conduct any systematic search, RW-LS uses a local Greedy Best-First Search driven by both direct node evaluation and random walks. RW-LS jumps to either the best state in the closed list or the best endpoint found by random walks to exploit advantages of

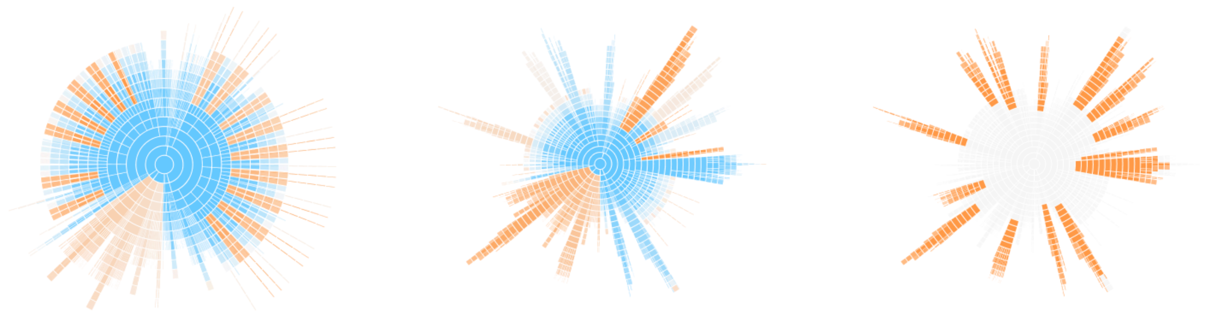


Figure 2: 20000 node search trees generated by GBFS (left) and LS-GBFS (center and right) while decreasing h_{min} from 24 to 23 in 2012-barman-02. In the left and center, colours correspond to node generation time: blue = early, orange = late. The right figure shows the subset of the LS-GBFS nodes generated by local search.

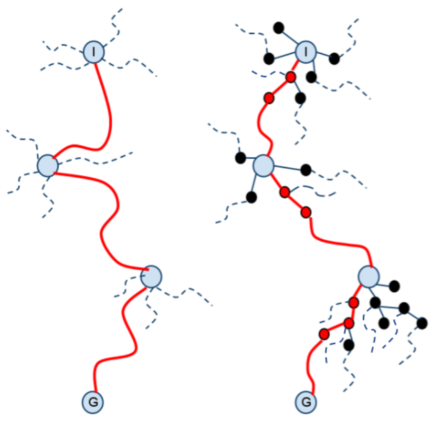


Figure 3: The search strategies of MRW (left) and RW-LS (right).

both search strategies. Figure 3 shows the difference between MRW and RW-LS. The planner based on RW-LS, Arvand-LS, improves both coverage[1] and plan quality significantly over the IPC-2011 version of Arvand.

Another contribution of this work is IPC-2011-LARGE, a set of scaled up test instances for several IPC-2011 domains. Such scaled up instances are useful since some of the current IPC benchmarks have become too easy for the best planners. They show some limits of the state-of-the-art planners and indicate that the proposed RW-LS algorithm scales to the IPC-2011-LARGE instances.

Better Quality Search via Randomization and Postprocessing

Objective/Motivation: Because plan quality is also an important metric in IPC-2008 and IPC-2011, planners usually keep searching for better solutions after finding the first solution. Using a post-processor system, such as ARAS (Nakhost and Müller 2010) which takes an existing plan and tries to find a higher quality one, is another option. One interesting phenomenon is that it is usually better to feed several low quality plans into Aras than to feed only one high-quality plan when we do experiments on Arvand. Figure 4 shows the intuition of why low quality input plans can lead to better quality output plans. If we apply the same method to more systematic any-time search planners, such as LAMA-2011, would the same phenomenon happen?

Most of the satisficing planners which are based on heuristic search iteratively improve their solution quality through an anytime approach. Typically, the lowest-cost solution found so far is used to constrain the search. This avoids areas of the state space which cannot directly lead to lower cost solutions. However, if the same phenomenon discussed above also exists in state of the art planners, in conjunction with a post-processing plan improvement system such as ARAS, this bounding approach can harm a planner's performance.

Method/Approach: We applied ARAS with LAMA-2011 on the latest IPC competition domains. 29% of final solutions from ARAS come from non-best solution generated by LAMA-2011. Based on this observation, we proposed the Diverse Any-time Search Meta-Algorithm (DAS) (Xie, Valenzano, and Müller 2013) for plan quality improvement using restarting with some randomization and post-processing. When adding both Diverse Any-Time Search and the ARAS post-processor to LAMA-2011 and AEES, the Anytime Explicit Estimation Algorithm (Thayer, Benton, and Helmert 2012), the performance on the 550 IPC 2008 and IPC 2011 problems is improved by almost 60 points according to the IPC metric, from 511 to over 570 on LAMA-2011, and 73 points from 440 to over 513 on AEES.

Improving Greedy Best First Search by Local Search

Objective/Motivation: Domain independent planning is a good test bed for deepening our understanding of classical search algorithms. Without strong domain-specific heuristics, some structure problems caused by uninformed or ill-informed heuristics, such as early mistake and uninformative heuristic region, become the performance bottleneck.

[1]: the number of problems solved

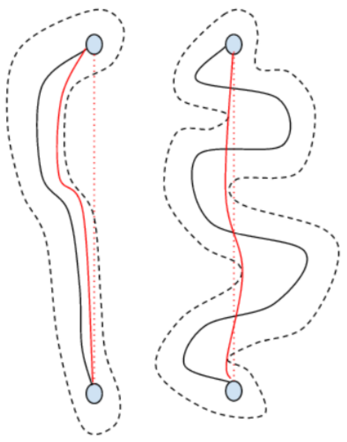


Figure 4: Why low quality plans into Aras can generate better final output plans. In the figure, red dot lines denote optimal path, black solid lines denotes input path, regions surrounded by dashed black lines denote the neighbour state space ARAS expanded from the input paths, and solid red lines denotes the shortest path ARAS can find inside the neighbour states space.

In this work, we explored how UHRs influence GBFS' performance. GBFS is an algorithm that strongly depends on the heuristic function. GBFS always expands a node that is closest to a goal state. The early mistake problem, where heuristic functions give wrong evaluations, is discussed in previous work on the K-BFS (Felner, Kraus, and Korf 2003) and Diverse-BFS (Imai and Kishimoto 2011). However, the uninformative heuristics region (UHR) problem, where a heuristic function provides no useful guidance, remains under-explored.

Method/Approach: In UHRs, Greedy Best First Search is as inefficient as uninformed Breadth First Search, as illustrated in Figure 2 (left). Based on that observation, we developed an algorithm called **Local Search based Greedy Best First Search (LS-GBFS)** which is a dual strategies algorithm that combines global GBFS and local GBFS in order to escape UHRs quickly. LS-GBFS, shown in Figure 2 (middle), searches much more deeply along some branches. Figure 2 (right) shows that these deep branches are mainly grown by the local search. The new planner LS-LAMA, which replaces the GBFS component of the top planner LAMA-2011 by LS-GBFS, solves more problems and reduces the search time on a common set of IPC problems by one third. Figure 5 shows the time usage for LS-LAMA and LAMA-2011 in solving the n easiest problems. We submitted this paper to IJCAI-2013 this year.

Short-term Objectives

This section proposes research that is already in progress with preliminary results, or is a clear next step of previous work.

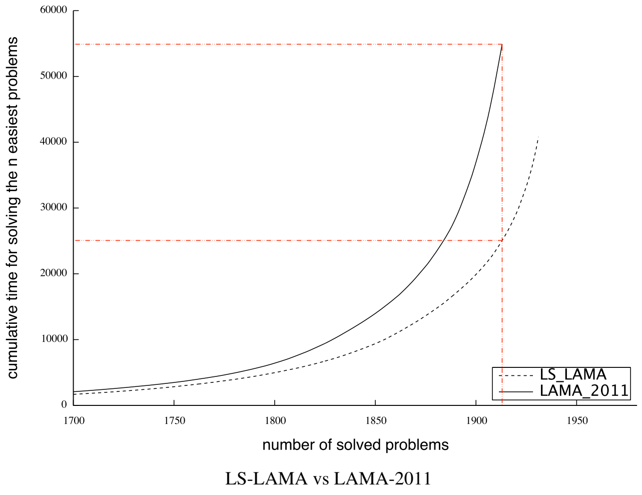


Figure 5: Time usage (in seconds) for solving the n easiest problems.

Preferred Operators Revisited

Objective/Motivation: Preferred Operator is a very strong search enhancement for current state of the art satisficing planners. The dual queue approach is empirically shown as the strongest way of using preferred operators by (Richter and Helmert 2009). Compared to pruning strategy introduced by (Hoffmann and Nebel 2001), they showed that pruning performs badly if the preferred operators are ill-informed. The dual queue approach of preferred operators contains two parts:

- information provided by preferred operators;
- an extra queue filled with a partial set (only preferred operators) of the all legal operators.

It is widely assumed that the majority or nearly all improvement come from the first point. What the extra queue does is mainly to provide a more smooth way of adding information provided by preferred operators than pruning. Is it really the case? We designed some controlled experiments on GBFS over all IPC domains by replacing the preferred operators with the same number of random operators. The experimental results show that the random operators dual queue approach can achieve roughly **50%** the improvement in coverage of the dual queue preferred operators approach!

Method/Approach: We recently start studying why a partial random operator extra queue can achieve such a big improvement. There are several possible reasons: 1), because we use a smaller number operators in the extra queue, the search can explore deeper search space quicker, which is helpful for escaping from uninformative heuristic regions; 2), because some best h-value nodes are ignored in the partial queue, the non-best h-values get more possibility to be expanded, in other words, it provides more exploration. Some preliminary results show that the extra partial queue has improvement on both aspects. Based on this observation, we develop a very simple three queue approach containing:

- a queue with all legal operators,

- a queue with only preferred operators,
- a queue with a partial set of randomly selected operators.

We test it on LAMA-2011. The new three queue approach decrease the search time on a common set of IPC-2011 problems by 20%. In short, the multi-queue open list approach provides a smooth way to combine different search strategies into the same search framework.

A General Variant of GBFS Handling both Early Mistake and UHR

Objective/Motivation: Another interesting future work is combining techniques for handling early mistakes and uninformative heuristic regions into a single algorithm. Since we have different algorithms handling early mistakes and uninformative heuristic regions, such as Diverse-BFS for early mistakes and LS-GBFS for uninformative heuristic regions, one simple approach is to build a portfolio system including both. Preliminary results show that a simple loosely-coupled portfolio planner of these two algorithms reduces the number of unsolved problems over all instances from past IPC competitions from 199 to 145. However, since early mistakes and UHRs appear together in many hard problems, an efficient search algorithm should be able to handle both at the same time.

Method/Approach: A tighter integrated combination of LS-GBFS and DBFS seems promising because of their complementary strengths. DBFS contains several significant innovations, such as expanding nodes with non-best h-value, and size-controlled local search. Studying these mechanisms deeply could be an important step towards building a strong hybrid algorithm. LS-GBFS and DBFS are very similar in algorithm structure. Though for different purposes, both algorithms use small local searches for exploration. It also seems possible to combine both algorithms into one single general framework and gradually switch between the two behaviours via some internal parameters.

Long-term Objectives

Classifying Planning Domains and Algorithms: Next Step towards Strong Portfolio or Multi-Strategies Systems

Objective/Motivation: It is hard to find one strong algorithm that can achieve state of the art performance in all domains. Even LAMA-2011, the clear winner of IPC-2011 single core track, is inferior to the state of the art in many domains, such as 2011-nomystery (Nakhost, Hoffmann, and Müller 2012), 2006-trucks (Imai and Kishimoto 2011) and large-2011-woodworking (Xie, Nakhost, and Müller 2012). Because there exist different types of planners with state of the art performance in some domains but inferior to other planners in other domains, building a portfolio system is a natural choice. Some successful approaches include Fast Downward Stone Soup (Helmert, Röger, and Karpas 2011), the second place in the IPC-2011 single core track, and ArvandHerd (Valenzano et al. 2012), the winner of the IPC-2011 multi-core track. Many promising search algorithms are proposed recently. We have more candidates to build loosely-coupled portfolio or tightly-coupled multi-strategies systems. How to select candidate search algorithms including the new proposed algorithms is still an open problem.

Method/Approach: One interesting closely related work is (Nakhost and Müller 2012) which analyzes search behavior in plateaus. A plateau is an area of the state space where all states have the same heuristic value while the real distance to goal states d varies. This paper proves that random walks are more efficient than traditional algorithms such as GBFS in escaping some plateaus. In order to analyze random walks, this work introduces an interesting parameter called *regress factor*, which is the probability that performing a random operator increases the goal distance divided by the chance that decreases the goal distance. Except the relationship between d and h, we also plan to investigate regress factor from small problems, whose state space can be enumerated explicitly, to see whether we can get some interesting results. Another direction is algorithm-dependent analysis. While the analysis mentioned above is more like an off-line approach, it is also interesting to investigate some online approaches such as recording numbers of node expansions and search tree structure over per heuristic value improvement.

Anticipated Significance

Work in applying randomized methods and developing portfolio or multi-strategies systems has already attracted a large amount of interest in deterministic satisficing planning. More general and efficient search methods over planning benchmarks would certainly have an impact on the general AI community and definitely have strong potential to propagate back to the classical heuristic search fields. Given the significant improvement of our current work over state of the art, we expect more follow-up research on randomized methods and mixed strategies hybrid/portfolio systems in deterministic satisficing planning. For the proposed work in classifying domains and search algorithms, we wish it can provide a fundamental basis for future research on building portfolio or multi-strategies systems.

References

Borrajo, D.; Felner, A.; Korf, R. E.; Likhachev, M.; López, C. L.; Ruml, W.; and Sturtevant, N. R., eds. 2012. *Proceedings of the Fifth Annual Symposium on Combinatorial Search, SOCS 2012, Niagara Falls, Ontario, Canada, July 19-21, 2012*. AAAI Press.

Felner, A.; Kraus, S.; and Korf, R. E. 2003. KBFS: K-best-first search. *Ann. Math. Artif. Intell.* 39(1-2):19–39.

Helmert, M.; Röger, G.; and Karpas, E. 2011. Fast Downward Stone Soup: A baseline for building planner portfolios. In *Proceedings of the ICAPS-2011 Workshop on Planning and Learning (PAL)*, 28–35.

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.

Imai, T., and Kishimoto, A. 2011. A novel technique for avoiding plateaus of greedy best-first search in satisficing planning. In Burgard, W., and Roth, D., eds., *AAAI*, 985–991. AAAI Press.

Kocsis, L., and Szepesvári, C. 2006. Bandit based Monte-Carlo planning. In Fürnkranz, J.; Scheffer, T.; and Spiliopoulou, M., eds., *ECML*, volume 4212 of *Lecture Notes in Computer Science*, 282–293. Springer.

Lipovetzky, N., and Geffner, H. 2011. Searching for plans with carefully designed probes. In Bacchus, F.; Domshlak, C.; Edelkamp, S.; and Helmert, M., eds., *Proceedings of the Twenty-First International Conference on Automated Planning and Scheduling (ICAPS-2011)*, 154–161. AAAI.

McCluskey, L.; Williams, B.; Silva, J. R.; and Bonet, B., eds. 2012. *Proceedings of the Twenty-Second International Conference on Automated Planning and Scheduling, ICAPS 2012, Atibaia, São Paulo, Brazil, June 25-19, 2012*. AAAI.

Nakhost, H., and Müller, M. 2009. Monte-Carlo exploration for deterministic planning. In Walsh, T., ed., *Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence (IJCAI'09)*, 1766–1771. IJCAI/AAAI.

Nakhost, H., and Müller, M. 2010. Action elimination and plan neighborhood graph search: Two algorithms for plan improvement. In Brafman, R.; Geffner, H.; Hoffmann, J.; and Kautz, H., eds., *Proceeedings of the 20th International Conference on Automated Planning and Scheduling (ICAPS-2010)*, 121–128. Toronto, Canada: AAAI Press.

Nakhost, H., and Müller, M. 2012. A theoretical framework for studying random walk planning. In Borrajo et al. (2012), 57–64.

Nakhost, H.; Hoffmann, J.; and Müller, M. 2012. Resource-constrained planning: A Monte Carlo random walk approach. In McCluskey et al. (2012), 181–189.

Richter, S., and Helmert, M. 2009. Preferred operators and deferred evaluation in satisficing planning. In Gerevini, A.; Howe, A. E.; Cesta, A.; and Refanidis, I., eds., *ICAPS*, 273–280. AAAI.

Thayer, J.; Benton, J.; and Helmert, M. 2012. Better parameter-free anytime search by minimizing time between solutions. In Borrajo et al. (2012).

Valenzano, R.; Nakhost, H.; Müller, M.; Schaeffer, J.; and Sturtevant, N. 2012. Arvandherd: Parallel planning with a portfolio. In Raedt, L. D.; Bessière, C.; Dubois, D.; Doherty, P.; Frasconi, P.; Heintz, F.; and Lucas, P. J. F., eds., *ECAI*, volume 242 of *Frontiers in Artificial Intelligence and Applications*, 786–791. IOS Press.

Xie, F.; Nakhost, H.; and Müller, M. 2012. Planning via random walk-driven local search. In McCluskey et al. (2012), 315–322.

Xie, F.; Valenzano, R.; and Müller, M. 2013. Better quality search via randomization and postprocessing under time constraint. In *ICAPS*, 9 pages. AAAI.