# A Comparison of Knowledge-Based GBFS Enhancements and Knowledge-Free Exploration

**Richard Valenzano**
University of Alberta
valenzan@ualberta.ca

**Nathan R. Sturtevant**
University of Denver
sturtevant@cs.du.edu

**Jonathan Schaeffer, Fan Xie**
University of Alberta
{jonathan, fxie2}@ualberta.ca

**Abstract**

GBFS-based satisficing planners often augment their search with knowledge-based enhancements such as preferred operators and multiple heuristics. These techniques seek to improve planner performance by making the search more informed. In our work, we will focus on how these enhancements impact coverage and we will use a simple technique called $\epsilon$-greedy node selection to demonstrate that planner coverage can also be improved by introducing knowledge-free random exploration into the search. We then revisit the existing knowledge-based enhancements so as to determine if the knowledge these enhancements employ is offering necessary guidance, or if the impact of this knowledge is to add exploration which can be achieved more simply using randomness. This investigation provides further evidence of the importance of preferred operators and shows that the knowledge added when using an additional heuristic is crucial in certain domains, while not being as effective as random exploration in others. Finally, we demonstrate that random exploration can also improve the coverage of LAMA, a planner which already employs multiple enhancements. This suggests that even future knowledge-based enhancements need to be compared to appropriate knowledge-free random baselines so as to ensure the importance of knowledge being used.

## 1 Introduction

*Greedy Best-First Search* (GBFS) is a popular algorithm that is used in many heuristic search-based satisficing planners including LAMA (Richter and Westphal 2010) and Fast Downward (Helmert 2006). In its basic form, GBFS iteratively selects the most promising node for expansion, as suggested by a single heuristic which, along with a tie-breaking policy, will completely determine how the search progresses through the state-space. If a GBFS instance does not work well on a given problem, it means that the heuristic and the tie-breaking policy are not effectively guiding the search.

So as to improve the performance of GBFS-based planners on such problems, the standard algorithm is often enhanced with techniques such as *preferred operators* or *multi-heuristic best-first search*. These enhancements use automatically generated knowledge as an alternative source of guidance. The goal of using such enhancements is to improve the way that the state-space is examined by making the search more informed. As the variation induced into the search by these enhancements is based on knowledge, we refer to such enhancements as being *knowledge-based*.

Variation can also be introduced into GBFS by using *random exploration*. For example, we can change the way that nodes are iteratively selected for expansion by making the algorithm occasionally select a random node from the open list instead of the node suggested by the heuristic. This technique, which we refer to as *$\epsilon$-greedy node selection*, is easy to implement, has a low execution overhead, and does not require additional knowledge about the problem.

In this paper, we will compare and contrast the impact on coverage of the variation added by knowledge-based enhancements and knowledge-free random exploration. We begin by using $\epsilon$-greedy node selection to demonstrate that there is substantial value in adding random exploration to GBFS. As this value is gained by decreasing the algorithm's sensitivity to the heuristic, it suggests a need to revisit the existing knowledge-based enhancements so as to ensure they are not simply taking on a similar role. As such, we will then evaluate the impact of the knowledge being used in the knowledge-based enhancements by comparing these enhancements to equivalent systems in which the knowledge has been replaced by randomness. This investigation confirms that preferred operators are offering much more to the search than simply adding random variation, and that the knowledge employed when using secondary heuristics is crucial in some domains while not offering enough variation in others. Finally, we will show that random exploration can even benefit a fully enhanced planner like LAMA, and argue that knowledge-based planning enhancements need to be compared against stochastic alternatives so as to better understand the importance of the added knowledge.

## 2 The Value of Knowledge-Free Exploration

In this section we will demonstrate that there is value to adding knowledge-free random exploration to GBFS. We do so using *$\epsilon$-greedy node selection*, which is a simple modification to GBFS that allows for random exploration to be explicitly added to the algorithm. This technique is inspired by the $\epsilon$-greedy policies often used for multi-armed bandit problems (Sutton and Barto 1998). It requires the user to set a parameter $\epsilon$ with some value in the range $[0, 1]$. With probability $(1 - \epsilon)$, $\epsilon$-greedy node selection uses the same

| Planner | Coverage | # Domains Better than Baseline | # Domains Worse Than Baseline |
|---|---|---|---|
| Baseline | 528.0 | 0 | 0 |
| $\epsilon = 0.05$ | 578.0 | 13 | 1 |
| $\epsilon = 0.1$ | 581.4 | 14 | 1 |
| $\epsilon = 0.2$ | 585.1 | 14 | 1 |
| $\epsilon = 0.3$ | 584.7 | 13 | 2 |
| $\epsilon = 0.5$ | 574.5 | 12 | 3 |
| $\epsilon = 0.75$ | 546.3 | 7 | 5 |

Table 1: Adding $\epsilon$-greedy node selection to GBFS.

rule as GBFS to select a node for expansion: it selects the node with the lowest heuristic value. However, with probability $\epsilon$ this technique selects a node uniformly at random from amongst all nodes in the open list. This means that the value of $\epsilon$ determines how often this node selection policy chooses greedily according to the heuristic, and how often the algorithm explores randomly.

Since $\epsilon$-greedy node selection only changes GBFS by introducing random node selection, we can use this technique to evaluate the impact of adding knowledge-free random exploration to a GBFS-based planner. For the evaluation of this technique and the others considered below, we will only consider planner coverage due to space constraints. However, these techniques could be used as only the first iteration of a restarting weighted A* search (Richter, Thayer, and Ruml 2010), or with a post-processor in Diverse Any-time Search (Xie, Valenzano, and Müller 2013). We expect that when doing so, the quality of solutions found would be similar.

The experimental setup for both this section and the rest of this paper was then as follows. All experiments are performed using the Fast Downward planning system. The test set is composed of the 790 problems from IPCs 2006, 2008, and 2011, and these tasks will be treated as being unit-cost since our focus is on coverage. The experiments were performed on a cluster of 8-core machines, each with two 4-core 2.8 GHz Intel Xeon E546s processors and 6 MB of L2 cache. Planners were run with a 4 GB per-problem memory limit, and a 30 minute per-problem time limit which did not include the time for translation from PDDL to SAS+. All tested planners were set to break ties in a first-in first-out manner, do not re-open closed nodes, and use deferred heuristic evaluation (Helmert 2006) unless otherwise stated. For stochastic planners, the coverage shown is the average coverage seen over 10 runs on each problem.

Table 1 shows the coverage of a baseline GBFS planner and instances of that baseline planner that are enhanced with $\epsilon$-greedy node selection. The baseline runs standard GBFS with the FF heuristic (Hoffmann and Nebel 2001). The third column shows on how many of the 30 domains the enhanced planner could solve at least one more problem than the baseline, on average. The fourth column shows on how many domains the enhanced planner solved at least one fewer problem on average. The table shows that by adding random exploration to a GBFS search, we can substantially improve the coverage on the test set. This is true for a wide range of values of $\epsilon$ and in many domains. In some of these domains, the magnitude of this increase is also quite high. For example, for all values of $\epsilon$ tested, $\epsilon$-greedy was able to solve no less than an average of 29.4 of 30 2008 cybersecurity problems, 22.8 of 30 2008 woodworking problems, and 16.8 of the 20 2011 barman problems, while the baseline solved 20, 15, and 12, respectively. In the few domains in which $\epsilon$-greedy node selection decreased the coverage, the effect was minimal unless $\epsilon$ was high. For example, for all values of $0.05 \leq \epsilon \leq 0.3$, the coverage never decreased in any domain by more than an average of 2.5 problems.

To ensure this behaviour is not specific to GBFS when using the FF heuristic and deferred heuristic evaluation, we also tested $\epsilon$-greedy node selection with standard heuristic evaluation and a with a different heuristic. In these cases, the results were similar. For example, the performance of GBFS using the FF heuristic and standard heuristic evaluation improved from a total of 533 problems when not using $\epsilon$-greedy node selection to an average of 596.3 when $\epsilon = 0.3$. Similarly, when using a GBFS using deferred heuristic evaluation that is guided by the context-enhanced additive (CEA) heuristic (Helmert and Geffner 2008) solves 491 problems when not using $\epsilon$-greedy node selection and an average of 536.0 solved when $\epsilon = 0.3$.

These results indicate that while the popular heuristics offer effective guidance for standard GBFS in many cases, there is significant value in adding variation through knowledge-free random exploration. When GBFS fails, it is because it is too sensitive to the errors in the heuristic. The results suggest $\epsilon$-greedy can help to decrease this sensitivity.

## 3 Knowledge-Based GBFS Enhancements

In this section, we describe how preferred operators and multi-heuristic best-first search use additional knowledge to add variation to GBFS.

### 3.1 Preferred Operators

The *preferred operators* of a node $n$ are operators that are applicable to $n$ and which are identified — typically as a byproduct of heuristic calculation — as being more likely to be part of a solution path. Preferred operators were first introduced under the name of *helpful actions* by Hoffmann and Nebel (2001) who used them for pruning actions in an *enhanced hill-climbing search*. They were then adapted for use in GBFS by Helmert (2006) in the Fast Downward planning system. In this system and other GBFS-based planners like LAMA, preferred operators are most often used in a dual-queue search (Richter and Helmert 2009). The first queue contains the entire open list, while the second queue only contains those nodes achieved with a preferred operator. The simplest approach to using these two queues is to alternate between them when selecting a node to be expanded, with the heuristic being used to determine the most promising node in each queue. Some GBFS-based planners also use *boosting*, which increases the proportion of time in which the preferred operator queue is used to select the next node to be expanded (Richter and Westphal 2010).

In a GBFS that is enhanced by preferred operators, at least every second node expanded will have been achieved

| Planner | Operator Ordering | | | | |
|---|---|---|---|---|---|
| | Stand | Rev | RO | P 1st | P 1st RO |
| FF | 528.0 | 543.0 | 526.9 | NA | NA |
| FF & LM | 604.0 | 599.0 | 587.4 | NA | NA |
| FF, Prefs | 616.0 | 615.0 | 606.8 | 616.0 | 613.8 |
| FF, BP | 654.0 | 665.0 | 656.3 | 675.0 | 657.5 |
| FF & LM, BP | 680.0 | 692.0 | 676.0 | 713.0 | 677.7 |

Table 2: The impact of knowledge-based enhancements.

by a preferred operator. The enhancement is thereby using the knowledge given by the preferred operators to introduce variation into GBFS by putting a higher priority to those nodes which are the result of these operators. Doing so may also help in *heuristic plateaus*, which are contiguous regions of the state-space in which all nodes have the same heuristic value. In standard GBFS, the typical first-in first-out tie-breaking scheme will result in a breadth-first search being performed in the plateau. By putting priority on a subset of the open list, the use of preferred operators can help the search reach deeper parts of the plateau sooner. If they offer good guidance within the plateau, the preferred operators may also help the search to quickly find an exit.

### 3.2 Multi-Heuristic Best-First Search

*Multi-heuristic best-first search* (Helmert 2006) is another popular knowledge-based GBFS enhancement that has been shown to be an effective way to simultaneously use multiple heuristics (Röger and Helmert 2010). Given a set of $k$ heuristics, a GBFS instance using multi-heuristic best-first search will take turns using each of the $k$ heuristics for identifying the most promising node for expansion. This technique can be seen as enhancing a single-heuristic GBFS with an additional $k - 1$ heuristics, which represent the additional knowledge which this enhancement relies on. Variation will be added to the search by the additional heuristics whenever they disagree with the initial heuristic over which node is most promising. For example, while the initial heuristic may view a portion of the state-space as a plateau, one of the other heuristics may not and may instead offer guidance in that region of the state-space. However, if the additional heuristics are too similar to the initial heuristic or also offer poor guidance, they will not effectively vary the search.

## 4 Comparing Knowledge-Free and Knowledge-Based Variation

Having shown that GBFS can be improved by adding random exploration, we will evaluate the usefulness of the knowledge employed by the knowledge-based enhancements as compared to adding random variation. To do so, we will compare these enhancements to random baselines that are created by replacing the knowledge used in the enhancements with randomness. We begin by showing that the knowledge-based enhancements do improve coverage when they use their usual knowledge.

### 4.1 Knowledge-Based Enhancement Performance

Table 2 shows the coverage of the single-heuristic baseline planner used above and the coverage achieved when the knowledge-based enhancements are added. The secondary heuristic used is the landmark count (LM) heuristic (Richter and Westphal 2010). Each planner was tested with different *operator orderings*. The tested configurations are GBFS using the FF heuristic (FF), GBFS using multi-heuristic best-first search (FF & LM) with the FF and LM heuristics, GBFS using the FF heuristic and preferred operators (FF, Prefs), GBFS using the FF heuristic and boosted preferred operators (FF, BP), and GBFS using multi-heuristic best-first search with the FF and LM heuristics and boosted preferred operators (FF & LM, BP). The last of these configurations, "FF & LM, BP", corresponds to the first iteration of LAMA (Richter and Westphal 2010). As the subsequent iterations can only begin once the first iteration successfully finds an initial solution, 'FF & LM, BP" and LAMA are equivalent in the context of planner coverage.

The operator orderings tested are the standard ordering (Stand.), the reverse of the standard ordering (Rev), random operator ordering (RO), preferred operators first (P 1st), and random operator ordering with preferred operators first (P 1st RO). *Random operator ordering* means that the successors of a node are randomly shuffled before they are added to a queue, while *preferred operators first* means that the preferred operators are put at the front of the generated successor list. By default, Fast Downward and LAMA use "P 1st" when using preferred operators , and "Stand." otherwise.

Operator ordering has previously been shown to have a substantial effect on planner performance due to its impact on the way ties are broken between successors of the same node (Howe and Dahlman 2002). The table shows that this is also true of Fast Downward, though the best operator ordering changes depending on the planner configuration. For example, the standard ordering is better than the reverse ordering for "FF & LM", while the opposite is true for other configurations like "FF". The table also shows that the knowledge-based enhancements are able to improve the coverage regardless of the operator ordering. While the magnitude of this improvement changed depending on the operator ordering, the relative ordering of the planners did not.

### 4.2 Evaluating the Variation Added by Preferred Operators

It was shown in the previous section that enhancing GBFS with preferred operators substantially improved planner coverage. Recall that the knowledge being exploited by this enhancement is given by the preferred operators suggested by the heuristic, and that the variation introduced by using these operators is the result of putting a higher priority on nodes that are achieved using a preferred operator. As the search would most likely vary if any proper subset of the open list was prioritized using a second queue, we can evaluate the effectiveness of this knowledge by populating the second queue using randomly selected nodes as opposed to those corresponding to preferred operators. Note that we will not use boosting in this section for the sake of simplicity.

| Planner | No Prefs RO | No Prefs Stand | Boosted Prefs RO | Boosted Prefs P 1st |
|---|---|---|---|---|
| FF | 526.9 | 528.0 | 657.5 | 675.0 |
| FF & Random | 586.0 | 584.6 | 686.8 | 698.0 |
| FF & LM | 587.4 | 604.0 | 676.0 | 713.0 |

Table 3: Knowledge-based and knowledge-free multi-heuristic best-first search.

For this experiment, we ensured that the number of random successors of a given node that are put in the second queue was equal to the actual number of preferred operators suggested by the heuristic. We also use random operator ordering so as to avoid the inherent bias introduced through the use of a static operator ordering with first-in first-out tie-breaking. When the preferred operator queue is populated with random operators, the average number of problems solved is 554.1. While this is not as high as the average of 606.8 solved when the actual preferred operators are used, it is higher than the average of 526.9 problems solved when only a single queue is used. We also considered restricting the randomly selected operators to only include those which were not identified as preferred operators. Such a system solved an average of 531.6 problems.

These results suggest that preferred operators are an important source of knowledge that is adding substantially better guidance than randomly selected operators, though even populating a second queue with randomly selected operators offers useful variation. The use of the actual preferred operators was only worse than preferring randomly selected operators on 5 domains, in which the coverage never differed by more than an average of 1.4 problems per domain. In contrast, there were 6 domains in which the actual preferred operators were better than the randomly selected operators by an average of at least 4.5 problems.

Similar behaviour was also seen when this experiment was repeated using the CEA heuristic. For example, a GBFS which uses the CEA heuristic solves an average of 491.9 problems when not using preferred operators, 534.6 when preferring randomly selected operators, and 583.6 when using the actual preferred operators.

### 4.3 Evaluating the Variation Added by Multi-Heuristic Best-First Search

Multi-heuristic best-first search was also shown to lead to coverage improvements in Section 4.1, with some of those improvements coming in domains in which preferred operators were not as effective. For example, when using random operator ordering, "FF" solved an average of 3.8 of the 20 problems in the 2011 visitall domain, while "FF, BP" solved an average of 4.2 and "FF & LM" solved an average of 18.4. As with preferred operators, we will evaluate the importance of the extra knowledge used in this technique by replacing that knowledge with randomness. This means that we will still use a second heuristic, but it will be a purely random heuristic. For the experiments below, this was done by defining the second heuristic so that the heuristic value of a node was given by a random integer in the range from 1 to 100.

The coverage of the planner using this random heuristic is shown in Table 3 in the row labelled "FF & Random" both when using boosted preferred operators and when not using preferred operators. The table shows that the variation added by the random heuristic leads to substantially better coverage than the single-heuristic baseline planner. We include the results over different operator orderings since this attribute did affect the relative ordering of the planners tested. When using random operator ordering, the use of a random heuristic led to better coverage than the knowledge-based heuristic, though the opposite is true with the other orderings. When using the random heuristic, the planner is less sensitive to the operator ordering because when selecting nodes for expansion according to this heuristic, the randomly assigned heuristic value will matter more for tie-breaking between children of the same node than will the operator ordering. However, this also means that if the operator ordering is introducing a beneficial bias, the planner using the random heuristic is less able to take advantage.

Despite the similarity in the total coverage results, domain-by-domain analysis showed that using the knowledge-based heuristic is resulting in variation that is quite different than random exploration. For example, consider the results when using standard operator ordering and no preferred operators. Just as with $\epsilon$-greedy node selection, the random exploration added when using the random heuristic increased coverage in the 2008 cybersecurity and 2008 woodworking domains, from 20 and 15 respectively when using a single heuristic, to averages of 30.0 and 25.4 when using the random heuristic. In contrast, the use of the knowledge-based LM heuristic as a secondary heuristic actually hurts coverage, as the resulting planner solves only 12 and 19 problems respectively. However, the LM heuristic does add important guidance in the 2008 transport, 2011 parking, and the 2011 visitall domains. In these domains, the single heuristic planner solved 34 of the 70 total problems, while adding the random heuristic improved coverage to 45.7 which was still not as much as the 67 solved when using the LM heuristic.

The use of a random heuristic was also an effective way to increase coverage when used alongside the CEA heuristic. For example, when using random operator ordering and boosted preferred operators, CEA solved an average of 617.3 problems, while the addition of a random heuristic increased coverage to an average of 650.9. The use of the random heuristic even compares well to a multi-heuristic best-first search that uses both the CEA and FF heuristics when using random operator ordering and boosted preferred operators, as such a system solves an average of 646.5 problems.

## 5 Related Work

Diverse best-first search (DBFS) is a search algorithm that also stochastically selects nodes from the open list (Imai and Kishimoto 2011). The execution of this algorithm consists of two phases. First, a node is randomly selected from the open list according to a distribution which favours nodes with a low $g$-cost and a low heuristic value. Secondly, a node-limited local GBFS search is initiated from the selected node. This process repeats until a solution is found.

| | Operator Ordering | | | |
|---|---|---|---|---|
| Planner | Stand | RO | P 1st | P 1st RO |
| Baseline | 680.0 | 676.0 | 713.0 | 677.7 |
| $\epsilon = 0.05$ | 706.4 | 704.8 | 723.0 | 706.7 |
| $\epsilon = 0.1$ | 704.4 | 706.5 | 720.8 | 706.0 |
| $\epsilon = 0.2$ | 703.8 | 705.9 | 720.3 | 703.9 |
| $\epsilon = 0.3$ | 702.9 | 705.1 | 716.4 | 704.2 |

Table 4: Adding $\epsilon$-greedy node selection to the first iteration of LAMA. The default LAMA configuration corresponds to the baseline with the P 1st operator ordering.

Though $\epsilon$-greedy node selection may not increase coverage as much as DBFS when added to an otherwise unenhanced planner, it still increases coverage substantially and it is considerably simpler. In any case, the main purpose of $\epsilon$-greedy node selection is not to compete with DBFS, but to isolate the impact of knowledge-free random exploration on GBFS and to clearly demonstrate its positive impact. This is also the case when comparing $\epsilon$-greedy node selection to other systems which search for solutions using stochastic techniques. These include Arvand (Nakhost and Müller 2009), which uses a random-walk based search, Roamer (Lu et al. 2011), which adds random-walks to GBFS, and Lamar and Randward (Olsen and Bryce 2011), which use a stochastic version of the FF heuristic that is constructed by adding randomness into the way in which the heuristic is computed.

## 6 Conclusion and Discussion

In this paper, we have demonstrated through a simple technique called $\epsilon$-greedy node selection that there is substantial value in adding variation to GBFS through random exploration. This means that GBFS can be improved by both better guidance and by adding random exploration. It is therefore necessary to compare knowledge-based enhancements to proper random baselines to ensure that they are actually adding better guidance into the search instead of merely adding exploration which can be achieved in simpler ways. We performed such a comparison between appropriate randomized baselines and two existing enhancements: preferred operators and multi-heuristic best-first search. Our results indicate that both of these techniques are adding important guidance, particularly in certain domains in which random exploration is not enough when used on its own.

We have also tried adding $\epsilon$-greedy node selection to the first iteration of LAMA, so as to test if random exploration is unnecessary in such a fully enhanced planner. For this experiment, each queue was set to individually use $\epsilon$-greedy node selection. For example, if the next node expanded is to be selected from a preferred operator queue, the search will expand the most promising preferred operator with probability $1 - \epsilon$, and a random preferred operator with probability $\epsilon$. The results of this experiment are given in Table 4, which shows that the added exploration increases the coverage of LAMA over a wide range of $\epsilon$ values and over all the operator orderings considered. As there is still room for gains through random exploration, it remains necessary to ensure that any newly developed knowledge-based enhancements are also compared to appropriate random baselines.

## Acknowledgments

We would like to thank Martin Müller, Robert Holte, and the reviewers for their advice on this paper. This research was supported by GRAND and the Natural Sciences and Engineering Research Council of Canada (NSERC).

## References

Helmert, M., and Geffner, H. 2008. Unifying the Causal Graph and Additive Heuristics. In *Proceedings of the Eighteenth International Conference on Automated Planning and Scheduling*, 140–147.

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.

Howe, A. E., and Dahlman, E. 2002. A Critical Assessment of Benchmark Comparison in Planning. *Journal of Artificial Intelligence Research* 17:1–33.

Imai, T., and Kishimoto, A. 2011. A Novel Technique for Avoiding Plateaus of Greedy Best-First Search in Satisficing Planning. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, 985–991.

Lu, Q.; Xu, Y.; Huang, R.; and Chen, Y. 2011. The Roamer Planner: Random-Walk Assisted Best-First Search. *The 2011 International Planning Competition* 73–76.

Nakhost, H., and Müller, M. 2009. Monte-Carlo Exploration for Deterministic Planning. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, 1766–1771.

Olsen, A., and Bryce, D. 2011. Randward and Lamar: Randomizing the FF Heuristic. *The 2011 International Planning Competition* 55–57.

Richter, S., and Helmert, M. 2009. Preferred Operators and Deferred Evaluation in Satisficing Planning. In *Proceedings of the 19th International Conference on Automated Planning and Scheduling*, 273–280.

Richter, S., and Westphal, M. 2010. The LAMA Planner: Guiding Cost-Based Anytime Planning with Landmarks. *Journal of Artificial Intelligence Research* 39:127–177.

Richter, S.; Thayer, J. T.; and Ruml, W. 2010. The Joy of Forgetting: Faster Anytime Search via Restarting. In *Proceedings of the 20th International Conference on Automated Planning and Scheduling*, 137–144.

Röger, G., and Helmert, M. 2010. The More, the Merrier: Combining Heuristic Estimators for Satisficing Planning. In *Proceedings of the 20th International Conference on Automated Planning and Scheduling*, 246–249.

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press.

Xie, F.; Valenzano, R.; and Müller, M. 2013. Better Time Constrained Search via Randomization and Postprocessing. In *Proceedings of the Twenty-Third International Conference on Automated Planning and Scheduling*, 269–277.