Proceedings of the Twenty-Sixth International Conference on
Automated Planning and Scheduling (ICAPS 2016)

An Analysis of Merge Strategies for Merge-and-Shrink Heuristics

Silvan Sievers, Martin Wehrle, Malte Helmert
University of Basel, Switzerland
{silvan.sievers,martin.wehrle,malte.helmert}@unibas.ch

Abstract

The merge-and-shrink framework provides a general basis for the computation of abstraction heuristics for factored transition systems. Recent experimental and theoretical research demonstrated the utility of non-linear merge strategies, which have not been studied in depth. We experimentally analyze the quality of state-of-the-art merge strategies by comparing them to random strategies and with respect to tie-breaking, showing that there is considerable room for improvement. We finally describe a new merge strategy that experimentally outperforms the current state of the art.

Introduction

Heuristic search is a common approach for finding shortest paths in large state spaces, for example in optimal classical planning. Several recently proposed heuristics use the *merge-and-shrink* framework (Dräger, Finkbeiner, and Podelski 2006; 2009; Helmert, Haslum, and Hoffmann 2007; Helmert et al. 2014), where *atomic abstractions* of a planning task are incrementally combined (*merging* two abstract transition systems) and simplified (*shrinking* an abstract transition system) until a single abstraction is left, whose goal distances then induce a heuristic for the planning task. Throughout the paper, we assume basic familiarity with classical planning and the merge-and-shrink framework. A self-contained introduction to the most recent evolution of the merge-and-shrink framework is provided by Sievers, Wehrle, and Helmert (2014).

An important aspect of merge-and-shrink is the *merge strategy*, which determines which two intermediate abstractions to combine in each merge step. We will use the following terminology: A merge strategy for a task is defined by a binary tree over the state variables of the task. If this tree degenerates to a list, the merge strategy is called *linear*, otherwise *non-linear* (Fig. 1). More generally, when speaking of a merge strategy from the literature, we refer to the specific (domain-independent) algorithm that generates the merge strategies for the given planning tasks. Such an algorithm is called a linear merge strategy if and only if the merge strategies it produces are linear for all planning tasks. In other words, non-linear merge strategy algorithms are not constrained to produce linear merge trees for all tasks, but still may produce linear merge trees for *some* tasks.

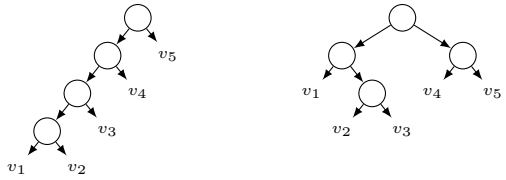

Figure 1: Linear (left) and non-linear (right) merge strategy for a planning task with five state variables.

Early merge-and-shrink heuristics in planning used linear merge strategies because only these could be implemented efficiently. This has only changed recently with the introduction of *generalized label reduction* (Sievers, Wehrle, and Helmert 2014), which led to the proposal of several non-linear merge strategies, most notably *DFP* (Sievers, Wehrle, and Helmert 2014), *MIASM* (Fan, Müller, and Holte 2014) and a framework for enhancing existing merge strategies based on symmetries (Sievers et al. 2015). All these experimentally outperformed the earlier best linear strategies. Moreover, Helmert, Röger, and Sievers (2015) proved that non-linear merge strategies are more powerful than linear ones in the sense that compiling non-linear strategies into linear ones incurs an unavoidable super-polynomial blowup of runtime and memory usage for certain problems.

These experimental and theoretical results motivate a deeper investigation of merge strategies in general, which are currently quite poorly understood. For example, the literature contains no systematic study of merge strategies, no comparison to simple baselines, and no investigation of tie-breaking effects. We make the following contributions.

Firstly, we compare state-of-the-art merge strategies to *all* merge strategies on a small planning task and to a large set of *random* merge strategies on the IPC benchmark set, showing that the current state of the art leaves significant room for improvement. Secondly, we show that the DFP strategy is highly sensitive to tie-breaking, which can change its behavior from mostly linear to highly non-linear. Thirdly, we introduce a simple merge strategy based on the same principles as the complex MIASM strategy and show that it of-

fers comparable (slightly worse) performance to MIASM. Finally, we define a simple new merge strategy that outperforms previous merge strategies.

Throughout the paper, the only aspect of the merge-and-shrink heuristic we vary is the merge strategy. All experiments use the existing implementation of merge-and-shrink in the Fast Downward planner (Helmert 2006), the state-of-the-art bisimulation-based shrink strategy by Nissim, Hoffmann, and Helmert (2011) with a size limit of 50 000 states per abstraction, and exact generalized label reduction.[1] Unless noted otherwise, we use a 30 minute time limit and 2 GiB memory limit. The benchmark set used for all experiments stems from IPCs 1998–2014 with 1667 tasks in total.

Evaluating Merge Strategies

One deficiency of the current merge-and-shrink literature is that all existing evaluations of merge strategies are *relative comparisons* of the form "strategy X solves more tasks than Y" or "strategy X requires fewer expansions than Y". While certainly useful, this tells us little about the quality of X and Y in absolute terms. Is X strong and Y poor? Is Y strong, but X even stronger? Can we do better than X?

The only "baseline comparison" of merge strategies we could find is an experiment reporting coverage results for a single *random linear* merge strategy, which is shown to be worse than state-of-the-art strategies (Sievers et al. 2015). We found no information on random non-linear strategies or on the variance of performance with random merges.

We consider the linear merge strategies *CGGL* (*causal graph/goal/level*), *RL* (*reverse level*) and *L* (*level*), the non-linear merge strategies DFP and MIASM, and all symmetry-enhanced variants *SYMM* of these strategies. CGGL (Helmert, Haslum, and Hoffmann 2007) and RL (Nissim, Hoffmann, and Helmert 2011) are the linear merge strategies that have been recommended in the literature; L is the inverse order to RL. These are all based on variations on the idea of topologically sorting causal graphs (Knoblock 1994); we refer to the original papers for details. We describe DFP and MIASM in more detail in the following sections, which focus on these two strategies. The SYMM-variants of the merge strategies stem from the symmetry-enhancement framework by Sievers et al. (2015), which prefers merging transition systems so that factored symmetries arise (to increase information-preserving shrinking based on full bisimulation) whenever the task exhibits such symmetries, and falling back onto the original merge strategy otherwise.[2]

Table 1 shows overall coverage (number of tasks solved) for the five merge strategies and their symmetry-enhanced variants (SYMM, second row; they correspond to the configurations "symm" of Sievers et al. (2015)) on the benchmark set.[3] Despite the large overall difference, no strategy dominates another when considering per-domain coverage.

	CGGL	**DFP**	**L**	**MIASM**	**RL**
Coverage	710	745	704	**757**	725
SYMM	**CGGL**	**DFP**	**L**	**MIASM**	**RL**
Coverage	747	752	742	749	749

Table 1: Coverage for merge strategies from the literature.

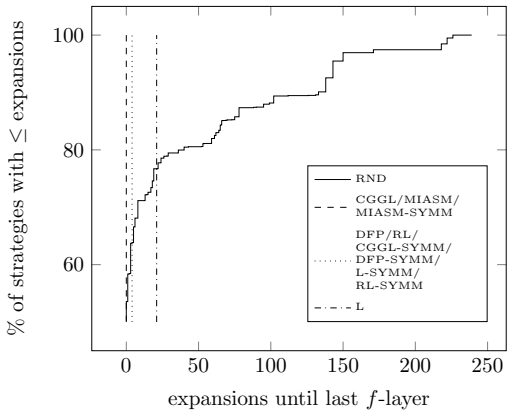

Figure 2: Expansions until last f-layer for Zenotravel #5 with all 1587600 possible merge strategies.

Considering All Merge Strategies

In our first experiment, we consider small planning tasks where we can compare the strategies from the literature to *all possible* merge strategies. This is only feasible for tasks with up to 8 state variables (which already give rise to 1587600 merge strategies). Most such tasks are so simple that all merge strategies result in a perfect heuristic, but there are exceptions. Here, we report results for *Zenotravel #5*.

Figure 2 shows the quality of all merge strategies for this instance as a cumulative distribution function. For example, a data point at $(62, 83.0\%)$ means that 83% of all merge strategies require 62 or fewer expansions to reach the final f-layer of an A* search. The curve starts at $(0, 50.2\%)$, showing that 50.2% of *all* merge strategies reach the final f-layer immediately. Given this, the results for the merge strategies from the literature may appear somewhat disappointing: only CGGL, MIASM and MIASM-SYMM have 0 expansions until the final f-layer; DFP, RL and all other SYMM-variants require 4, and L requires 21.

Sampling Random Merge Strategies

For larger planning tasks, evaluating all merge strategies is infeasible. However, it is still possible to assess how strong a given merge strategy is in absolute terms by *sampling* a large subset of random merge strategies. We conducted experiments with 1000 merge strategies per task on the entire benchmark set. This showed that the existing merge strategies are quite well suited for many planning domains: The expected coverage of random merge strategies is 680.107,

[1] Changing the shrink strategy or even only its parameters clearly can have an impact on the results reported in this paper, but varying shrink strategies goes beyond the scope of this paper.

[2] Note that this modification makes only limited sense for statically precomputed merge strategies such as MIASM (see also footnote 2 in the paper by Sievers et al. (2015)).

[3] Per-domain results for this and all further coverage tables are reported in a technical report (Sievers, Wehrle, and Helmert 2016).

worse than any strategy from the literature we consider. Moreover, we found 72 tasks in 19 domains which were solved by at least one merge strategy from the literature but by *none* of the 1000 random merge strategies.

However, interestingly, we also found 21 tasks in 9 domains solved by at least one random merge strategy but by no strategy from the literature. For example, in the NoMystery-2011 domain, only 18 tasks are solved by existing strategies, but all 20 tasks are solved by some random strategy. Moreover, solving these tasks is not a rare occurrence, with most of them solved by all 1000 random strategies and the hardest one solved by 26.4%. Another domain with clear room for improvement is Elevators-2008, where only 18 tasks are solved by any of the existing strategies, but 22 tasks are solved by some random strategy.

In these two domains, we performed additional experiments with the merge strategies from the literature with no time limit and a 64 GiB memory limit in order to determine how much better the good random strategies (with the regular limits of 2 GiB and 30m) are compared to the state of the art. Figures 3 and 4 show the results (again as cumulative distribution function of expansions) for two of these tasks.[4] For example, we see that in NoMystery-2011 #9, the best existing merge strategies require roughly 1000 times as many expansions as the best random ones. The results look similar for other instances of these two domains (not shown). In particular, the best random merge strategy *always* performs strictly better on all non-trivial tasks which are not perfectly solved by any merge strategy.

The experiment also showed that in 8 of the 11 tasks that the RL strategy can solve in Elevators-2008, its heuristic quality is equal to the *worst* of the 1000 random merge strategies. Similarly damning, the L strategy is consistently as bad as the worst 5% of random strategies in the NoMystery-2011 domain.

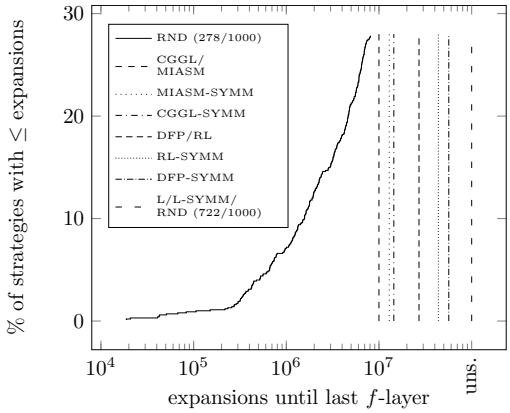

Figure 3: Expansions for NoMystery-2011 #9 with 1000 random merge strategies and those from the literature.

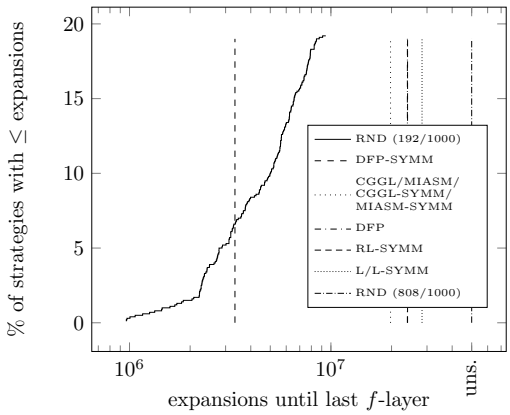

Figure 4: Expansions for Elevators-2008 #7 with 1000 random merge strategies and those from the literature.

The DFP Merge Strategy

We now take a closer look at the merge strategy introduced by Dräger, Finkbeiner, and Podelski in the original papers on merge-and-shrink for model checking (Dräger, Finkbeiner, and Podelski 2006; 2009). Dräger et al. consider a process model and describe a "composition strategy" used for merging processes. This strategy has been adopted and implemented by Sievers, Wehrle, and Helmert (2014) as the first non-linear merge strategy in planning (called the DFP strategy in the planning setting). To date, DFP is a state-of-the-art merge strategy, outperformed only by the much more complex MIASM strategy (see next section) among the five (non-symmetry enhanced) merge strategies we consider.

Whenever DFP is asked to select the next two abstract transition systems to merge, it computes a *score* for every possible merge (i.e., for every pair of two transition systems). It then performs the merge with the best score. A potential merge receives a good score if there exist labels (operators) that are relevant for both transition systems and that occur in transitions close to the abstract goal states. However, it turns out that there are frequently many possible merges that score perfectly, with synchronization required right at transitions reaching goal states. In this case, a tie-breaking criterion is needed to choose among the best-scoring merges, and we found that this tie-breaking criterion has significant impact. The conceptual description of DFP in the planning literature does not include any tie-breaking criteria. In the following, when talking about DFP's tie-breaking, we refer to the particular *implementation* of the DFP merge strategy in the Fast Downward planning system.

Table 2 shows the result of an experiment with various tie-breaking strategies. *Prefer atomic* prefers atomic transition systems for breaking ties, while *Prefer composite* prefers composite transition systems. Each of these additionally use a variable ordering to break ties further. These are the RL and L orderings that we also considered as linear merge strategies, and a random ordering *RND*. Finally, *Random* breaks ties fully randomly, preferring neither atomic nor composite transition systems. The combination *Prefer*

[4]The y-axis only goes up to 28% (19%) because only 278 (192) out of 1000 random merge strategies solved these instances. However, also note that *none* of the considered merge strategies from the literature (except DFP-SYMM in Elevators-2008 #7) can solve these instances within the regular time and memory limits.

	Prefer atomic			Prefer composite			Ran-
	RL	**L**	**RND**	**RL**	**L**	**RND**	dom
Coverage	726	**760**	723	745	729	697	*706*
Cons. (avg)	**60.1**	61.4	65.0	60.4	*89.1*	74.9	60.2
Lin ord (%)	10.8	10.9	**10.6**	81.7	*86.5*	84.3	13.2
50th perc:	6776	**2862**	7340	3324	4008	13760	*15873*
75th perc:	596k	**274k**	536k	390k	488k	1332k	*1370k*

Table 2: Results for DFP merge strategy. Reported are: coverage, average heuristic construction time (sec), fraction of tasks for which the construction of the abstraction finished and the merge strategy is linear, and 50th percentile (median) and 75th percentile of number of expansions to reach the last f-layer. Best performance in bold, worst in italics.

composite / RL corresponds to the implementation of DFP as reported in the literature.

The table shows a huge variability in performance, ranging from 697–760 solved tasks, from *worse* than the worst merge strategies in the literature to *better* than the best. The strategies that prefer merging composites result in a linear merge strategy in the vast majority (more than 80%) of cases where the abstraction computation completes. The reason for this is that after the initial merge (which necessarily merges two atomic transition systems), *Prefer composite* tends to repeatedly include the only existing composite in the next merge, which leads to a linear merge strategy. In contrast, the *Prefer atomic* or purely random tie-breaking strategies very rarely lead to linear merge strategies.

These results clearly indicate that there is more to the DFP merge strategy than initially meets the eye and that a better understanding of the approach is needed.

The MIASM Merge Strategy

To complete our discussion of existing merge strategies, we turn to MIASM (Fan, Müller, and Holte 2014). MIASM is a rather complex merge strategy whose main aim is to perform merges that allow a high amount of *pruning* in the resulting product systems. (After each merge, merge-and-shrink prunes so called "unnecessary " abstract states, i.e. abstract states that do not lie on any path from the abstract initial state to any abstract goal state.) Roughly speaking, MIASM partitions the planning task's variable set based on a search in the space of sets of variable subsets. In contrast to DFP, MIASM is a *static* non-linear merge strategy in the sense that the merge tree is precomputed entirely before the actual computation of the merge-and-shrink abstractions starts.

To provide a baseline for MIASM, we implemented a much simpler strategy, *DYN-MIASM*, which follows perhaps the simplest possible approach to facilitate "merge to prune". Like DFP, DYN-MIASM computes a score for every candidate merge and performs a merge with maximal score. The score of a candidate merge is simply the percentage of states in the resulting product system that can be immediately pruned. As in DFP, identical scores are quite common (e.g., in some domains no pruning ever occurs), although somewhat less common than in DFP. We consider the same tie-breaking strategies as in the previous section.

	Prefer atomic			Prefer composite			Ran-
	RL	**L**	**RND**	**RL**	**L**	**RND**	dom
Coverage	743	746	745	**747**	724	730	*726*
Cons. (avg)	**137.5**	143.0	141.9	194.1	*236.9*	234.0	169.0
Lin ord (%)	**10.4**	10.5	11.9	45.2	*53.2*	51.2	11.8
50th perc:	383	412	641	**67**	370	397	*1282*
75th perc:	231k	231k	231k	**185k**	231k	279k	*359k*

Table 3: DYN-MIASM, a dynamic version of MIASM. Results reported in the same way as in Table 2.

The results in Table 3 show similar trends as for DFP, but linear strategies with *Prefer composite* are somewhat less common (45%–53%) because there are more cases where merging is driven by differences in scores rather than tie-breaking. Note that DYN-MIASM requires (tentatively) performing all possible merges in order to compute the pruning ratios, and therefore the time to compute the heuristic is much higher than for other merge strategies. Still, results show that this effort is usually not prohibitive. The best variant of DYN-MIASM achieves a coverage of 747, close to the 757 achieved by the full MIASM strategy (Table 1). We believe that these results show that simple, score-based merge strategies still offer much potential for further research.

A New Merge Strategy

For our final experiment, we introduce a new merge strategy that combines some ideas of the existing approaches while also hinting at some avenues for future investigation.

All current merge strategies are linked to the concept of the *causal graph* of a planning task (Knoblock 1994). This is true for the linear merge strategies as discussed earlier, but it also applies to DFP and MIASM. With DFP, only transition systems that require non-trivial synchronization of transitions can be candidates for merging. For atomic transition systems, this can only be the case if they occur together in some operator. Similarly, in DYN-MIASM, transition systems that are unrelated in the causal graph cannot lead to pruning and hence receive the worst possible score.

However, while both DFP and DYN-MIASM indirectly incorporate information from the causal graph, they are fairly myopic in the sense that they do not *plan ahead* to maximize the amount of causal relatedness that they can capture. They are both based on greedy maximization of scores and do not look beyond the next merge.

The new merge strategy we suggest combines a more global picture of the causal graph with the score maximization of DFP. It works as follows. First, we compute the strongly connected components (SCCs) of the causal graph. Secondly, for each SCC, we apply the DFP merge strategy to the atomic transition systems in this SCC, resulting in one transition system per SCC. Finally, we merge these transition systems by again applying the DFP merge strategy.

Table 4 shows the results of this merge strategy, which we call SCC-DFP. We again use the same tie-breaking strategies as previously. Considering each SCC separately leads to a larger fraction of non-linear merge strategies than with DFP. (Note that SCC-DFP can never produce a linear merge strat-

	Prefer atomic			Prefer composite			Ran-
	RL	**L**	**RND**	**RL**	**L**	**RND**	dom
Coverage	751	760	*732*	**776**	751	741	736
Cons. (avg)	61.7	61.7	66.5	**59.8**	*86.0*	73.8	60.7
Lin ord (%)	8.2	8.4	**8.2**	58.2	58.7	*61.6*	11.5
50th perc:	2252	1796	*2649*	**350**	1410	2288	2352
75th perc:	349k	258k	370k	**221k**	362k	409k	*410k*

Table 4: SCC-DFP, a strategy using SCCs of the causal graph. Results reported in the same way as in Table 2.

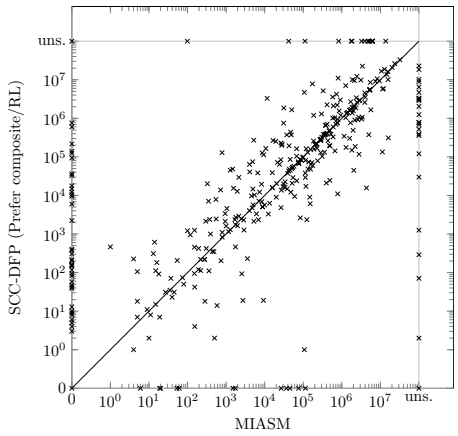

Figure 5: Expansions until last f-layer for the best SCC-DFP strategy (Prefer composite/RL) and MIASM.

egy for a planning task which has at least two SCCs consisting of at least two variables. However, a considerable number of planning tasks has strongly connected causal graphs, in which case SCC-DFP becomes DFP.)

In terms of coverage, all variants are at least as good as the corresponding DFP variant (Table 2), and in most cases they are considerably better. Coverage ranges from 732–776 for SCC-DFP compared to 697–760 for DFP. In particular, the best result of 776 solved tasks is better than any other merge strategy reported in earlier work or in this paper.

To compare the new best variant against the previous best merge strategy, MIASM, Figure 5 shows a scatter plot of the expansions for those two strategies. We observe that SCC-DFP and MIASM have orthogonal strengths, showing that there is room for further improvements of either strategy.

Conclusion

Merge strategies for merge-and-shrink heuristics are an underexplored topic. Keeping all other aspects of merge-and-shrink invariant, we have compared current merge strategies to *all* (for one small example task) and to *random* merge strategies (on a larger set of tasks), showing that there is considerable scope for improvement.

We have seen that DFP, a state-of-the-art merge strategy, is strongly susceptible to tie-breaking and that the performance of the complex state-of-the-art MIASM strategy can nearly be reached with a much simpler greedy approach. We also introduced a new strategy that integrates a simple "global analysis" (partitioning the causal graph into SCCs) with DFP, setting a new high water mark for merge-and-shrink heuristic performance.

The swings in performance between good and bad tie-breaking can be dramatic for the studied merge strategies, in particular for DFP. On the negative side, this means that these strategies are fragile. On the positive side, this means that there is still considerable room for stronger merge-and-shrink heuristics and that investigating better merge strategies appears to be a fruitful direction for future research.

Acknowledgments

This work was supported by the Swiss National Science Foundation (SNSF) as part of the project "Reasoning about Plans and Heuristics for Planning and Combinatorial Search" (RAPAHPACS).

References

Dräger, K.; Finkbeiner, B.; and Podelski, A. 2006. Directed model checking with distance-preserving abstractions. In *Proc. SPIN 2006*, 19–34.

Dräger, K.; Finkbeiner, B.; and Podelski, A. 2009. Directed model checking with distance-preserving abstractions. *Software Tools for Technology Transfer* 11(1):27–37.

Fan, G.; Müller, M.; and Holte, R. 2014. Non-linear merging strategies for merge-and-shrink based on variable interactions. In *Proc. SoCS 2014*, 53–61.

Helmert, M.; Haslum, P.; Hoffmann, J.; and Nissim, R. 2014. Merge-and-shrink abstraction: A method for generating lower bounds in factored state spaces. *JACM* 61(3):16:1–63.

Helmert, M.; Haslum, P.; and Hoffmann, J. 2007. Flexible abstraction heuristics for optimal sequential planning. In *Proc. ICAPS 2007*, 176–183.

Helmert, M.; Röger, G.; and Sievers, S. 2015. On the expressive power of non-linear merge-and-shrink representations. In *Proc. ICAPS 2015*, 106–114.

Helmert, M. 2006. The Fast Downward planning system. *JAIR* 26:191–246.

Knoblock, C. A. 1994. Automatically generating abstractions for planning. *AIJ* 68(2):243–302.

Nissim, R.; Hoffmann, J.; and Helmert, M. 2011. Computing perfect heuristics in polynomial time: On bisimulation and merge-and-shrink abstraction in optimal planning. In *Proc. IJCAI 2011*, 1983–1990.

Sievers, S.; Wehrle, M.; Helmert, M.; Shleyfman, A.; and Katz, M. 2015. Factored symmetries for merge-and-shrink abstractions. In *Proc. AAAI 2015*, 3378–3385.

Sievers, S.; Wehrle, M.; and Helmert, M. 2014. Generalized label reduction for merge-and-shrink heuristics. In *Proc. AAAI 2014*, 2358–2366.

Sievers, S.; Wehrle, M.; and Helmert, M. 2016. An analysis of merge strategies for merge-and-shrink heuristics: Additional data. Technical Report CS-2016-001, University of Basel, Department of Mathematics and Computer Science.