Scaling up WA* with Commitment and Diversity[*]

David Furcy
University of Wisconsin Oshkosh
Computer Science Department
Oshkosh, WI 54901-8643
furcyd@uwosh.edu

Sven Koenig
University of Southern California
Computer Science Department
Los Angeles, CA 90089-0781
skoenig@usc.edu

Abstract

Weighted A* (WA*) is a popular search technique that scales up A* while sacrificing solution quality. Recently, researchers have proposed two variants of WA*: KWA* adds diversity to WA*, and MSC-WA* adds commitment to WA*. In this paper, we demonstrate that there is benefit in combining them. The resulting MSC-KWA* scales up to larger domains than WA*, KWA* and MSC-WA*, which is rather surprising since diversity and commitment at first glance seem to be opposing concepts.

1 Introduction

Weighted A* (WA*) is a popular search technique that scales up A* while sacrificing solution quality, by weighing the heuristic values more strongly than A*. In this paper, we study how to scale up WA* to even larger domains, building on two variants of WA* that have been proposed recently: First, WA* expands only one state per iteration. KWA* is a variant of WA* that expands several states per iteration, which adds diversity to WA* to give it a stronger breadth-first component. Second, WA* considers all states in the OPEN list as candidates for expansion. Multi-State Commitment WA* (MSC-WA*) is a variant of WA* that considers only a small number of states in the OPEN list as candidates for expansion, which adds commitment to WA* to give it a stronger depth-first component. In this paper, we demonstrate that there is benefit in combining them. Our experimental results, for example, show that the resulting MSC-KWA* with the Manhattan heuristic scales up to the 48-Puzzle, whereas WA*, KWA* and MSC-WA* do not.

2 WA*, KWA* and MSC-WA*

WA*: WA* is a variant of A* that scales it up by making its search more greedy and thus more focused [Pohl, 1970]. While A* calculates the f-value of a state s as $f(s) = g(s) + h(s)$, WA* calculates it as $f(s) = (1-W)g(s) + W \times h(s)$, where $1 \geq W \geq 0.5$ is the only parameter of WA*. **KWA*:** Due to its strong focus, WA* is likely to be led into goal-free regions of the search space by misleadingly small heuristic values. KWA* tackles this problem by adding a breadth-first component to WA* (diversity) to make its search less focused. It expands the $K \geq 1$ states with the smallest f-values in the OPEN list (rather than just one) in parallel during each iteration [Felner *et al.*, 2003]. Thus, KWA* has two parameters, namely W and K. If $K = 1$, then KWA* reduces to WA*. Larger values of K increase the probability that WA* selects a state on the shortest path from the start state to the goal state during the next iteration at the risk of generating more states than necessary during each iteration. **MSC-WA*:** WA* focuses the search more than A*. However, the state that WA* expands next can be any state in the OPEN list (depending on the f-values). Multi-State Commitment WA* (MSC-WA*) increases the focus of WA* even more by focusing the search on a subset of the states in the OPEN list (commitment) [Kitamura *et al.*, 1998]. MSC-WA* splits the OPEN list into a COMMIT list of at most C states and a RESERVE list. Thus, MSC-WA* has two parameters, namely W and C. The COMMIT list plays the role of a smaller OPEN list, whereas the RESERVE list stores the remaining states of the original OPEN list. MSC-WA* expands the state with the smallest f-value in the COMMIT list. MSC-WA* never re-expands a state, even if it finds a shorter path from the start state to the state in question. The number of states in the COMMIT list can thus get smaller than C. In this case, MSC-WA* uses the states with the smallest f-values in the RESERVE list to refill the COMMIT list. If $C = \infty$, then the RESERVE list remains empty and MSC-WA* reduces to a variant of WA* that never re-expands a state. Smaller values of C make MSC-WA* focus the search on a smaller subset of the states in the original OPEN list.

3 Comparison

Table 1 reports the results of our experiments with WA*, KWA* and MSC-WA* on the N-Puzzle with $N = 15, 24, 35$ and 48, using the Manhattan distance as the heuristic function. The goal state has the blank in the upper left corner. For each combination of parameters, we average over 50 experiments (100 experiments for $N = 15$) with a memory capacity of six million states. (For W, for example, we use the values 0.50, 0.56, 0.60, 0.67, 0.75, 0.80, 0.86, 0.90, 0.95 and 0.99.) Each row of the table then reports the parameter combinations that optimize the average performance for each one of four performance measures, namely the solution cost, the num-

[*]We thank Richard Korf for providing us with WA* code and acknowledge funding from NSF under contracts IIS-0098807 and IIS-0350584. The views and conclusions contained in this document are those of the authors and not of NSF.

Table 1: Comparison of WA*, KWA*, MSC-WA*, and MSC-KWA* for the N-Puzzle

N	Performance Measure	WA*			KWA*				MSC-WA*				MSC-KWA*			
		Value	W	Best	Value	W	K	Best	Value	W	C	Best	Value	W	$K=C$	Best
15	Solution Cost	63.51	0.67		53.85	0.67	50,000	✓	56.29	0.60	80,000		53.89	0.99	50,000	✓
	Stored States	6,050	0.99		6,028	0.99	8		4,113	0.95	20		3,223	0.99	5	✓
	Generated States	6,972	0.99		6,704	0.99	8		4,191	0.95	20		3,259	0.99	5	✓
	Runtime (Seconds)	0.003	0.99		0.003	0.99	5		0.002	0.99	6		0.001	0.95	3	✓
24	Solution Cost	165.16	0.75		113.56	0.99	20,000	✓	164.56	0.75	90,000		116.32	0.99	20,000	
	Stored States	44,097	0.99		32,567	0.99	5		36,907	0.99	300		16,178	0.99	6	✓
	Generated States	56,070	0.99		43,578	0.99	4		37,832	0.99	300		16,331	0.99	6	✓
	Runtime (Seconds)	0.027	0.99		0.021	0.99	4		0.021	0.99	50,000		0.007	0.99	6	✓
35	Solution Cost				236.50	0.99	7,000	✓	472.10	0.90	3,000		244.14	0.99	7,000	
	Stored States				417,675	0.95	20		456,777	0.99	90		56,807	0.99	5	✓
	Generated States				652,100	0.95	500		467,586	0.99	90		57,291	0.99	5	✓
	Runtime (Seconds)				0.377	0.95	500		0.297	0.99	90		0.033	0.99	5	✓
48	Solution Cost												18,379.32	0.60	5	✓
	Stored States												275,293	0.80	4	✓
	Generated States												277,282	0.80	4	✓
	Runtime (Seconds)												0.181	0.80	4	✓

ber of stored states, the number of generated states, and the actual runtime. A check mark indicates that the algorithm is within one percent of the optimal performance reported in the row. An empty cell indicates that the algorithm does not solve all of our random instances. The table shows the following trends: First, all runtimes are small because the algorithms quickly either solve a search problem or run out of memory. Second, KWA* has a smaller solution cost than WA* and MSC-WA*. The solution cost of MSC-WA* remains roughly the same as the one of WA*, whereas the one of KWA* improves with N compared to the one of WA*, which can be attributed to the breadth-first component (diversity) of KWA*. Third, MSC-WA* has a smaller number of generated states than WA* and KWA*, which can be attributed to the depth-first component (commitment) of MSC-WA*. Fourth, both KWA* and MSC-WA* store a smaller number of states than WA* although neither KWA* nor MSC-WA* dominates the other one in this respect. (KWA* stores a smaller number of states than WA* only for large values of N.) The small memory consumption of KWA* and MSC-WA* allows them to solve all of our random instances of the 35-Puzzle, while WA* solves only 98 percent of our random instances, even for the best possible value of W. For KWA*, this result is partly due to our improved implementation of KWA* that stores each state at most once on the OPEN and CLOSED list. However, it turns out that the low memory consumption of our implementation of KWA* is *not* a systematic effect but due to noise since it requires one to hit the correct value of K precisely, which is not possible in practice.

4 MSC-KWA*

Although the concepts of diversity and commitment seem to be opposing at first glance, we now combine them: MSC-WA* splits the OPEN list into a COMMIT list of C states and a RESERVE list. This makes MSC-WA* even more focused than WA*, and thus makes it more likely that MSC-WA* is led into goal-free regions of the search space by misleadingly small heuristic values. We alleviate this problem by adding to MSC-WA* the mechanism used by KWA*. MSC-KWA*, the resulting algorithm, expands in parallel during each iteration all $K \leq C$ states with the smallest f-values in the COMMIT list (rather than just one). Thus, MSC-KWA* has three parameters, namely W, C and K. We eliminate one of them by setting $K = C$ to simplify the experimental conditions, which is reasonable since decreasing K and C both increase the focus of MSC-KWA*. Table 1 shows that MSC-KWA* has a smaller number of stored states, a smaller number of generated states, and a smaller actual runtime than WA*, KWA* and MSC-WA*. The savings are significant and increase with N. For example, the number of stored states of MSC-KWA* is at least a factor of seven smaller for the 35-Puzzle than the ones of WA*, KWA* and MSC-WA*. The small memory consumption of MSC-KWA* allows it to solve all of our random instances of the 48-Puzzle, which WA*, KWA* and MSC-KWA* cannot. (KWA* solves only 76 percent and MSC-WA* solves only 78 percent of our random instances, even if their parameter values are optimized.) Furthermore, the solution cost of MSC-KWA* is always within about three percent of the solution cost of KWA*, the algorithm with the smallest solution cost. Additional experiments are reported in [Furcy, 2004]. For example, MSC-KWA* (with $K \neq C$) solves all of our random instances of the 4-Peg Towers of Hanoi, which WA*, KWA* and MSC-WA* cannot. Note that beam search is a special case of MSC-KWA* (namely, with $K = C$ and no RESERVE list) and thus can be understood as a variant of best-first search with commitment and diversity, not just commitment (the current view). It is future work to determine the best parameters values of MSC-KWA* automatically.

References

[Felner *et al.*, 2003] A. Felner, S. Kraus, and R. Korf. KBFS: K-best-first search. *Annals of Mathematics and Artificial Intelligence*, 39:19–39, 2003.

[Furcy, 2004] D. Furcy. *Speeding Up the Convergence of Online Heuristic Search and Scaling Up Offline Heuristic Search.* PhD thesis, College of Computing, Georgia Institute of Techmology, Atlanta (Georgia), 2004.

[Kitamura *et al.*, 1998] Y. Kitamura, M. Yokoo, T. Miyaji, and S. Tatsumi. Multi-state commitment search. In *Proceedings of the International Conference on Tools for Artificial Intelligence*, pages 431–439, 1998.

[Pohl, 1970] I. Pohl. First results on the effect of error in heuristic search. In B. Meltzer and D. Michie, editors, *Machine Intelligence*, volume 5, pages 219–236. American Elsevier, New York, 1970.