To Reopen or Not To Reopen in the Context of Weighted A*? Classifications of Different Trends

(Extended Abstract)

Vitali Sepetnitsky **Ariel Felner**
Information Systems Engineering
Ben-Gurion university
{sepetnit, felner}@bgu.ac.il

Weighted A*

The most famous, and simple suboptimal search algorithm is *Weighted A** (WA*) (Pohl 1970). Similar to A*, WA* uses two data structures: *OPEN* and *CLOSED*. Initially, *OPEN* contains only the start state and *CLOSED* is empty. At every iteration, the algorithm chooses the state in *OPEN* that minimizes the cost function $f(n) = g(n) + W \cdot h(n)$, where $g(n)$ is the cost of the lowest cost path found so far, from the start state to n, $h(n)$ is an admissible heuristic estimate of the cost of reaching the goal from n, and W is a parameter.

It is well-known that A* with a *consistent* heuristic (and thus the f-cost is *monotonically increasing*) expands a state only after the lowest cost path to it was found. This is not the case for *non-monotonic* f-functions such as the one used by WA*. In this case, a state n may be generated with a smaller g-value, after it has been already expanded. At this point, it can be removed from *CLOSED* and put back in *OPEN* – an action called *reopening*. Reopening is optional; for every closed state which is seen with a smaller g-value, a decision needs to be made whether to reopen it or not. In the case of not reopening, the newly seen state is not placed back in *OPEN* but remains in *CLOSED*. However, it is a common practice to update its g-value as well as its parent pointer. In this paper we focus on the two extreme reopening polices: *always reopen* (AR) and *never reopen* (NR).

Related work

The notion of reopening has been first introduced by (Pohl 1970) and discussed in a variety of papers thereafter (Likhachev, Gordon, and Thrun 2003; Hansen and Zhou 2007; Thayer and Ruml 2008). A number of authors discussed the influence of AR and of NR on the path returned and on the number of expanded nodes (Thayer and Ruml 2010; Malima and Sabanovic 2007; Valenzano, Sturtevant, and Schaeffer 2014). For example, (Hansen and Zhou 2007) discuss some reasons for why AR may require fewer node expansions than NR in some cases, but did not discuss the case where AR finds a worse solutions than NR. We consider the full spectrum of the relative performance of AR and NR.

Ranges of W with different behavior

Any possible relation between AR and NR can occur regarding the path returned and the number of nodes expanded. The different trends can occur by only modifying the weight W but without changing the structure of the graph.

Let $P(AR)$ and $P(NR)$ denote the length of the path returned by the AR and NR policies, respectively. Let $P^+, P^=, P^-$ denote a win for AR ($P(AR) < P(NR)$), a tie ($P(AR) = P(NR)$) and a win for NR ($P(NR) < P(AR)$), respectively. Similarly, let $N(AR)$ and $N(NR)$ denote the number of nodes expanded by the AR and NR policies, respectively. Finally, let $N^+, N^=, N^-$ denote a win for AR ($N(AR) < N(NR)$), a tie ($N(AR) = N(NR)$) and a win for NR ($N(NR) < N(AR)$), respectively. There are $3 \times 3 = 9$ combinations and we use a four character notation P^*N^* to denote these cases. For example, P^+N^- denotes the case where in the path aspect (P), AR is the winner, while in the nodes-expanded aspect (N), NR is the winner.

Syntectic example

In the syntectic graph given in Figure 1 the task is to go from node S to node G. For now, assume that states E, X_1, X_2, X_3, X_4 and their connecting edges (all colored in gray) are not in the graph. The graph has four paths from S to G: $P_1 = \{S, A, C, G\}$ of cost 24, $P_2 = \{S, B, C, G\}$ of cost 22 and $P_3 = \{S, A, G\}$ of cost 24 and the optimal path, $P_{OPT} = \{S, A, D, G\}$, of cost 11.

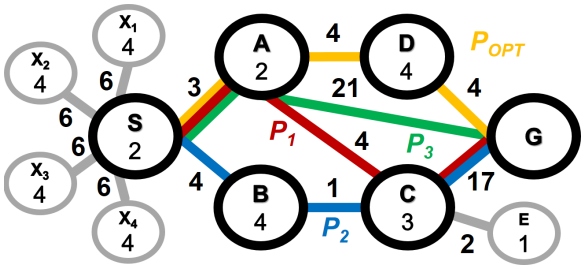

Figure 1: Reopening example

For different ranges of W different trends are observed as seen in the left part of Table 1. For very large values (*range 4*), reopening will never happen, hence both policies

		Path returned			Nodes Expansions					
Range	W	AR	NR	P	Subrange	Case	W	X_i	AR	NR
1	$1 \leq W < 3.75$	P_{OPT}	P_{OPT}	$P^{=}$	1a	$P^{=}N^{=}$	$1 \leq W < 3$	both	11	11
					1b	$P^{=}N^{-}$	$3 \leq W < 3.75$	both	13	11
2	$3.75 \leq W < 4.25$	P_2	P_{OPT}	P^{-}	2a	$P^{-}N^{-}$	$3.75 \leq W < 4$	both	12	11
					2b	$P^{-}N^{+}$	$4 \leq W < 4.25$	NR	8	11
3	$4.25 \leq W < 5$	P_2	P_3	P^{+}	3a	$P^{+}N^{+}$	$4.25 \leq W < 4.5$	NR	8	10
					3b	$P^{+}N^{-}$	$4.5 \leq W < 5$	None	8	6
4	$5 \leq W$	P_3	P_3	$P^{=}$	4	$P^{=}N^{=}$	$5 \leq W$	None	5	5

Table 1: The path and expansions for different values of W in Figure 1

tie. For very small values (*range 1*), reopening will make no difference as the optimal path (P_{OPT}) will be returned anyway – again a tie. The interesting cases are in the middle range. For *range 3*, AR will result in a better path. This is the intuitive case where AR is expected to be better, or at least no worse than NR (if ranges *1* and *4* are considered too). The somewhat counter-intuitive case is *range 2*, where AR finds a better path (P_2) and halts, while the path that NR has (P_3) is not strong enough, forcing NR to continue searching and eventually finding the optimal path (P_{OPT}).

By considering the grayed states and edges of the graph we can enrich the ranges of W to also have different tendencies in the node expansions. This update allows us to divide the four ranges mentioned above into subranges, depicted in the right part of Table 1. Thus, we have 6 out of the 9 possible cases in P^*N^*.

Experimental results

We experimented with the 8-puzzle, and the 4x4 15-puzzle with the Manhattan distance heuristic. For each puzzle we generated 1000 random instances and used a range of values for W. Figure 2 shows the percentage of 15-puzzle instances, partitioned into five disjoint groups according to the cost of the found path. The figure highlights the "anomalies" (bottom three slices), or the *counter intuitive* cases – when NR finds a *strictly* better path than AR (P^-) and when AR expands *strictly* less nodes than NR (N^+). Our results show that for some values of W these full-dominance cases occur in a non-negligible percentage of the instances. This calls for a method for predicting which reopening policy to use. We also experimented with 4-connected grid-based maps and observed similar trends.

Better polices

AR and NR are extreme policies – either *always* or *never* reopen nodes. One can implement a hybrid policy that chooses to reopen some nodes while bypassing reopening in other nodes. To demonstrate the potential of such hybrid policies we implemented a *reopening oracle*, as follows. Given an instance to be solved by WA*, for every state in *CLOSED*, which is generated again with a smaller g-value, a decision needs to be made whether to reopen it or not. By marking positive decision (to reopen) by 1 and negative decision (not to reopen) by 0, a sequence of reopening choices can be described as a binary vector. An oracle simulates the decisions imposed by *all* the possible binary vectors and then returns the sequence of reopening choices that results in finding a

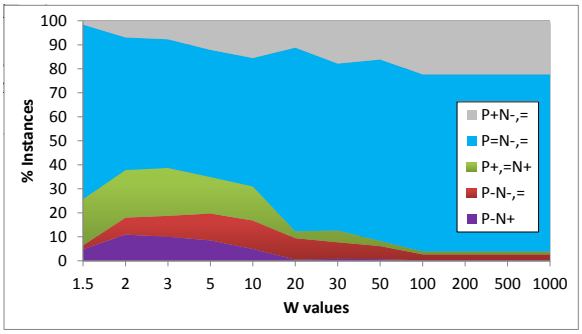

Figure 2: % instances per class, 15-puzzle

solution of lowest cost (min-cost oracle) or alternatively of the minimal number of nodes (min-expansions oracle). In our experiments the min-cost oracle found shorter solutions of length which was up to 17% shorter than the best solution found by the best extreme policy. Similarly, we have found instances where the min-expansions oracle expanded only 60% of the nodes expanded by the best extreme policies.

Future work

This is a work in progress. Based on our oracle experiments our research now focuses on finding reopening policies smarter than the extreme AR and NR. Such policies probably should use the current state of the search (e.g. the g and h value of the state being reopened, the current number of expansions etc.) and try to predict at each decision point, whether reopening should be done.

References

Hansen, E. A., and Zhou, R. 2007. Anytime heuristic search. *Journal of Artificial Intelligence Research (JAIR)* 28:267–297.

Likhachev, M.; Gordon, G. J.; and Thrun, S. 2003. ARA*: Anytime A* with provable bounds on sub-optimality. In *NIPS*.

Malima, A., and Sabanovic, A. 2007. Motion planning and assembly for microassembly workstation. In *ISCO*, 467–474.

Pohl, I. 1970. Heuristic search viewed as path finding in a graph. *Artificial Intelligence* 1(3-4):193–204.

Thayer, J. T., and Ruml, W. 2008. Faster than weighted A*: An optimistic approach to bounded suboptimal search. In *ICAPS*.

Thayer, J. T., and Ruml, W. 2010. Anytime heuristic search: Frameworks and algorithms. In *SOCS*.

Valenzano, R. A.; Sturtevant, N. R.; and Schaeffer, J. 2014. Worst-case solution quality analysis when not re-expanding nodes in best-first search. In *AAAI*, 885–892.