On Improving the Quality of Solutions in Large-Scale Cooperative Multi-Agent Pathfinding

Ko-Hsin Cindy Wang and **Adi Botea** and **Philip Kilby**

NICTA* and The Australian National University
{cindy.wang|adi.botea|philip.kilby}@nicta.com.au
http://users.rsise.anu.edu.au/~cwang/

Introduction

Scaling up the number of simultaneously moving units in pathfinding problems to hundreds, or even thousands, is well beyond the capability of theoretically optimal algorithms in practice, which is consistent with existing intractability results (Surynek 2010). However, significant scalability can be achieved by trading off solution optimality. This motivates evaluating the quality of suboptimal solutions, especially in instances much larger than can be handled by optimal algorithms.

In this work, we consider pathfinding in uniform cost grid maps. We study the solution quality using the three most common quality criteria. The *total travel distance* is the sum of distances covered by all units, measuring the total cost. The *sum of actions* includes moves and also wait actions that appear in a plan, providing an indication of the aesthetic quality of a plan, as long or frequent wait intervals are undesirable. The *makespan* measures the total number of time steps in a global plan when steps can be run in parallel.

We focus on MAPP (Wang and Botea 2009; 2010), which has been shown as state-of-the-art in terms of scalability and success ratio (i.e., percentage of solved units) on realistic game grid maps. Until now, the quality of MAPP's solutions had not been as extensively analyzed. Our analysis indicates that solutions computed with existing versions of MAPP can often suffer in terms of quality. We introduce enhancements that significantly improve MAPP's solution quality. For example, the sum of actions is cut to half on average.

MAPP becomes competitive in terms of solution quality with FAR (Wang and Botea 2008) and WHCA* (Silver 2005), two successful algorithms from the literature. MAPP provides a formal characterization of problems it can solve, and low-polynomial upper bounds on the resources required, which are not reported for FAR and WHCA*. Our extensions bring MAPP's solution quality to a state-of-the-art level, while maintaining its advantages over FAR and WHCA* on the performance criteria of scalability, success ratio, and ability to tell a priori if it will succeed in the instance at hand. One of our extensions, which spreads out the precomputed paths, can be more generally applied to other decentralised methods to reduce waiting time and collisions.

To evaluate the quality of suboptimal solutions in instances beyond the capability of optimal algorithms, we use lower bounds of optimal values to show our solutions have a reasonable quality. A simple and computationally cheap lower bound uses a set of shortest paths between all start–target pairs. For example, MAPP's average total travel distance is 19% longer than the total length of all shortest paths.

Improving MAPP's Solution Quality

In brief, MAPP precomputes a path, π, from the start to the target of each unit, satisfying three well-defined conditions that guarantee conflicts can be resolved online. MAPP's solution plans consist of alternating progression and repositioning stages. Units advance along their π-paths during progression, and attempt to push aside blocking units to clear the way. Repositioning brings unsolved units back on their π-paths, ensuring more units will be solved in the next progression stage, and the algorithm eventually terminates. For more details, we guide the reader to the original publications (Wang and Botea 2009; 2010). We identified two causes affecting MAPP's solution quality.

Firstly, as units prefer shorter paths, a large number of π-paths overlapped, creating traffic jams online and affecting the solution quality. A different choice of paths can reduce this problem. Hence, our first enhancement, called SP, encourages units to *spread out the paths* by avoiding already busy locations when searching for π-paths. Units are informed by a global traffic report, which is computed incrementally. With a hash map implementation and lazy instantiation, the memory overhead is very reasonable. SP does not affect MAPP's completeness range or low-polynomial time and memory upper bounds.

Secondly, since a repositioning stage starts only after the previous progression stage completes (and vice-versa), when a unit is pushed off its π-path, it needs to wait until the end of the current progression stage, before attempting to reposition back on its π-path. Hence, we modify the algorithm to allow an off-track unit to make a *dynamic repositioning* move, interleaved on the fly with progression moves of other units, under well specified conditions that do not create cycles between units. Repositioning stages can still occur globally, but are significantly reduced. We call this

*NICTA is funded by the Australian Government's *Backing Australia's Ability* initiative.

	Dr	Sp	Sp+Dr
Repositioning stages	0.24	0.79	0.18
Undo moves	0.71	1.22	0.41
Total distance (TD)	0.98	0.97	0.9
Makespan (MS)	0.52	0.81	0.43
Sum of actions (SA)	0.59	0.8	0.5
Elapsed real time	0.98	1.05	1.08
Additional nodes expanded	1	1.02	1.02

Table 1: Ratios of Dr, Sp, and Sp+Dr Mapp's solutions compared to old Mapp. A ratio in column x is the value of x Mapp divided by the value of old Mapp.

	Far	Whca*	Whca*+d	A* lb	A*+d lb
TD	1.01	0.95	1.12	1.19	1.43
MS	1.06	1.24		3.47	4.38
SA	0.94	0.98		1.83	2.21
Time	1.46	0.15	0.16	n/a	n/a

Table 2: Ratios of Sp+Dr Mapp's solutions compared to Far, Whca*, and lower bounds of optimal solutions ("+d" means diagonals enabled). A ratio in column x is the value of Sp+Dr Mapp divided by the value of algorithm x.

enhancement Dr. It also preserves previous low-polynomial upper bounds on resource requirements and solution length.

Experimental Results

We used input data from previous published work (Wang and Botea 2010; 2008), consisting of the 10 largest maps from the game Baldur's Gate, with 13765 to 51586 traversable tiles, non trivial configurations of obstacles, and 100 to 2000 mobile units (in increments of 100). For each value of the number of units on each map, 10 instances were generated with random start and target locations. Unless stated otherwise, the maps are assummed to be 4-connected.

Our new versions, Sp Mapp, Dr Mapp, and Sp+Dr Mapp, are compared with the best existing Mapp (Wang and Botea 2010), as shown in Table 1. Since Sp+Dr Mapp produced the best results, we also compared it with Far and Sturtevant and Buro's Whca*(w, a) (2006), with spatial abstraction (but with no unit priority system for replanning). All parameters are set as recommended in the original works. As these algorithms have different success ratios (Mapp (2010) solved 98.8% of all units, Far and Whca* solved 81.9% and 80.9%, respectively), we plot the subset of problem instances fully solved by all algorithms. Furthermore, Far and Whca* cannot a priori identify units that they are guaranteed to solve, thus a timeout limit of 10 minutes per instance is set, as used in previously work. Figure 1 illustrates an average behaviour from one sample map. Table 2 shows that Sp+Dr Mapp is at least as competitive as Far and Whca* on average in terms of solution quality.

The lower bounds for optimal solutions are obtained by assuming no unit will interfere with another. The A* lower bound, A* lb, is computed with cardinal moves only. Notice the sum of actions and makespan lower bounds become significantly smaller than the optimal value.

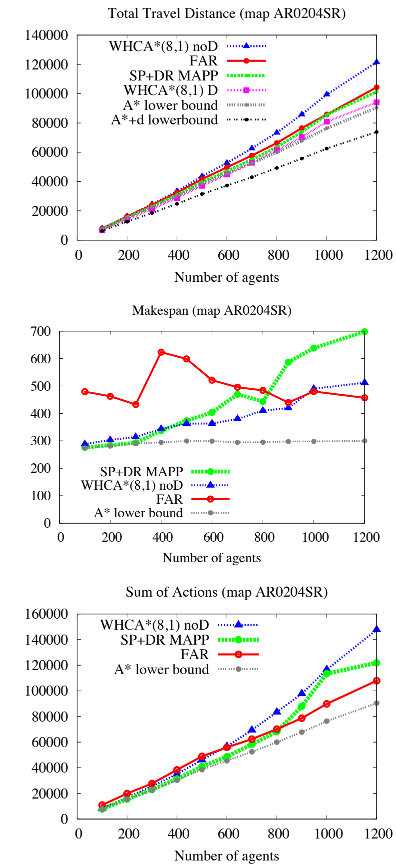

Figure 1: Sp+Dr Mapp's solution quality compared with Far, and Whca* with or without diagonals.

Conclusion

Suboptimal multi-agent pathfinding algorithms scale well beyond the capabilities of optimal methods. In terms of scalability, success ratio, and ability to provide formal completeness guarantees and low polynomial upper bounds on resource requirement, Mapp dominates convincingly benchmark algorithms such as Far and Whca*. In this work, we have also improved Mapp's solution quality to a state-of-the-art level. In future work, we plan to investigate extensions to units of different speed or size.

References

Silver, D. 2005. Cooperative Pathfinding. In *AIIDE*, 117–122.

Sturtevant, N. R., and Buro, M. 2006. Improving collaborative pathfinding using map abstraction. In *AIIDE*, 80–85.

Surynek, P. 2010. An Optimization Variant of Multi-Robot Path Planning is Intractable. In *AAAI*, 1261–1263.

Wang, K.-H. C., and Botea, A. 2008. Fast and Memory-Efficient Multi-Agent Pathfinding. In *ICAPS*, 380–387.

Wang, K.-H. C., and Botea, A. 2009. Tractable Multi-Agent Path Planning on Grid Maps. In *IJCAI*, 1870–1875.

Wang, K.-H. C., and Botea, A. 2010. Scalable Multi-Agent Pathfinding on Grid Maps with Tractability and Completeness Guarantees. In *ECAI*, 977–978.