Proceedings of the Ninth International
Symposium on Combinatorial Search (SoCS 2016)

An Empirical Comparison of the Hardness of Multi-Agent Path Finding under the Makespan and the Sum of Costs Objectives

Pavel Surynek
Charles University Prague
Malostranské náměstí 25
11800, Praha, Czech Republic
pavel.surynek@mff.cuni.cz

Ariel Felner **Roni Stern**
Ben Gurion University
Beer-Sheva, Israel 84105
felner,sternron@bgu.ac.il

Eli Boyarski
Bar-Ilan University
Ramat-Gan, Israel
eli.boyarski@gmail.com

Abstract

In the *multi-agent path finding* (MAPF) the task is to find non-conflicting paths for multiple agents. Recently, existing *makespan optimal* SAT-based solvers for MAPF have been modified for the sum-of-costs objective. In this paper, we empirically compare the hardness of solving MAPF with SAT-based and search-based solvers under the makespan and the sum-of-costs objectives in a number of domains. The experimental evaluation shows that MAPF under the makespan objective is easier across all the tested solvers and domains.

1 Introduction and Background

The *multi-agent path finding* (MAPF) problem consists a graph, $G = (V, E)$ and a set $A = \{a_1, a_2, \ldots a_m\}$ of m agents. Time is discretized into time steps. The arrangement of agents at time-step t is denoted as α_t. Each agent a_i has a start position $\alpha_0(a_i) \in V$ and a goal position $\alpha_+(a_i) \in V$. At each time step an agent can either *move* to an adjacent empty location[1] or *wait* in its current location. The task is to find a sequence of move/wait actions for each agent a_i, moving it from $\alpha_0(a_i)$ to $\alpha_+(a_i)$ such that agents do not *conflict*, i.e., do not occupy the same location at the same time.

MAPF has practical applications in video games, traffic control, robotics etc. (see (Sharon *et al.* 2015) for a survey). The scope of this paper is limited to the setting of *fully cooperative* agents that are centrally controlled. We focus here on two objective functions used in MAPF:
(1) sum-of-costs (denoted ξ) is the summation, over all agents, of the number of time steps required to reach the goal location (Standley 2010; Sharon *et al.* 2013; 2015).
(2) makespan (denoted μ) is the total time until the last agent reaches its destination (i.e., the maximum of the individual costs) (Surynek 2012; 2014).

Both objectives have important real-life applicability when for example the execution of an action corresponds to consumption of a unit of energy. The *sum-of-cost objective* is then applicable if the total energy consumption is a concern while the makespan objective is an option if it is permissible to reduce the total time at a cost of higher energy consumption. Hence the optimization of either the makespan and the sum-of-costs inherently leads to different solutions - see Figure 1.

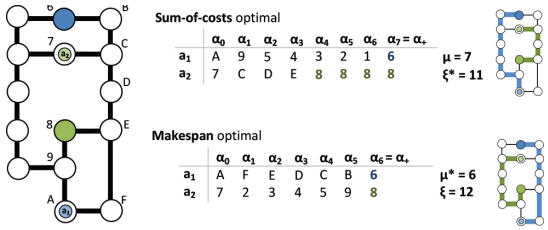

Figure 1: An instance of MAPF where *makespan* optimal and *sum-of-costs* optimal solutions differ (that is, any makespan optimal solution is (strictly) sum-of-costs suboptimal and vice versa).

2 Optimal MAPF Solvers

Optimal MAPS solvers can be divided into two main classes:
(1) Reduction-based solvers that reduce MAPF to known problems such as CSP (Ryan 2010) or SAT (Surynek 2012). The advantage of these solvers is a possibility of using efficient off-the-shelf solvers for the target formalism.
(2) Search-based solvers are often variants of the A* algorithm on a global *search space* – all different ways to place m agents into V vertices, one agent per vertex (Standley 2010). Other sloves, like ICTS (Sharon *et al.* 2013), EPEA* (Sharon *et al.* 2015) and ICBS (Boyarski *et al.* 2015), employ novel search trees.

Most search-based solvers were implemented for the sum-of-cost objective while the reduction-based solveres are often designed for the makespan optimization. However, at least in case of search-based solvers it is easy to modify them to other objectives.

3 Using MDDs in Makespan Optimal Case

Recently *cardinality constraints* and search space reduction technique of MDD (Sharon *et al.* 2013) have been used to build and optimal SAT-based sum-of-costs solver optimal called *MDD-SAT* (Surynek *et al.* 2016).

Previous makespan optimal SAT-based solvers like DIRECT-SAT and MATCHING-SAT (Surynek 2012; 2014)

[1] Some variants of MAPF relax the empty location requirement by allowing a chain of neighboring agents to move, given that the head of the chain enters an empty locations.

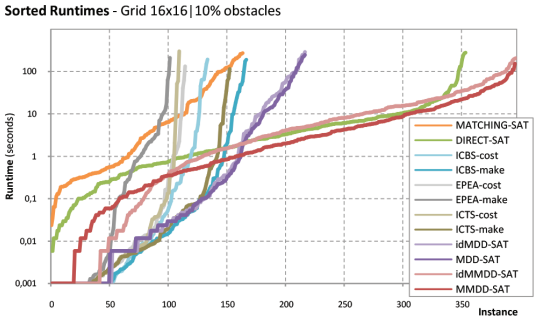

Figure 2: Runtimes on 4-connected grid 16×16 with 10% of nodes occupied by obstacles.

use s technique which is called *reachability heuristic*. These solvers first generate Boolean variables that represent positions of all agents at all time steps. Then, the occurrence of agents at unreachable positions in time/space are forbidden by extra constraints.

MDD-SAT is different. MDDs representing time/space reachable positions are generated first and then Boolean variables are introduced only for nodes appearing in MDDs. In this paper we also implemented an analogue idea for the makespan cost function. This new solver is called MMDD-SAT.

4 Experimental Evaluation

We compared the sum-of-costs and makespan variants of search-based solvers ICTS, EPEA, and ICBS with SAT-based makespan optimal solvers DIRECT-SAT, MATCHING-SAT, and MMDD-SAT. We also included the sum-of-costs optimal SAT-based MDD-SAT. Finally, we tried versions of MDD-SAT and MMDD-SAT enhanced with the *independence detection* - *ID* heuristic (Standley 2010) - the idMDD-SAT and idMMDD-SAT solvers.

All the solvers were executed on 10 random MAPF instances over 4-connected grids with 10% of nodes occupied with obstacles (Silver 2005). We varied the number of agents from 1 to 64. We randomly placed starts for each number of agents. Goal positions were generated by running a long random walk from starts to ensure solvability of each instance. Runtimes (y-axis) obtained on grid of size 16×16 sorted in the ascending order of instances per solver (x-axis) are shown in Figure 2. For example around 100 instances are solvable by the DIRECT-SAT solver below 1 second - tested on CPU Xeon 2.0.Ghz, 12 Gb RAM.

In addition, we tested SAT-based solvers on grids and on *hypercubes* where edges were substituted with paths consisting of a fixed number of internal vertices - the former called *networks*. These instances contain subgraphs (similar to that of Figure 1) that cause differences between makespan and sum-of-costs optimal solutions. Sorted runtimes for a network instance derived from the 6×6-grid and 4-dimensional hypercube are shown in Figure 3.

5 Discussion and Conclusions

Results show that the makespan optimal variant tend to

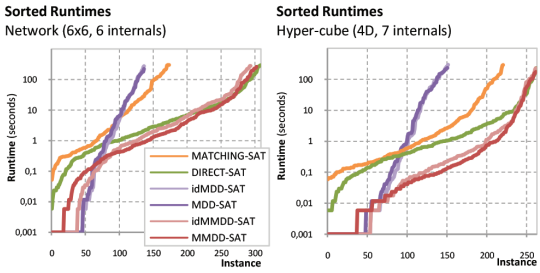

Figure 3: Runtimes on *network* and *hypercube* instances.

be easier (except for the EPEA* solver), all the other solvers are faster in their makespan optimal configuration. Moreover, it can be observed that SAT-based solvers perform better on harder instances than search-based ones. Particularly, MMDD-SAT solver turned out to be the new state-of-the-art makespan optimal solver.

Integrating ID into the SAT-based solvers is helpful in easier instances while in harder ones it represents an overhead. Hence one of the future directions is an investigation of how to integrate ID to be more beneficial.

6 Acknowledgments

This research was funded by the joint program for cooperation of the Israeli and Czech ministries of science.

References

E. Boyarski, A. Felner, R. Stern, G. Sharon, D. Tolpin, O. Betzalel, and S. Shimony. ICBS: improved conflict-based search algorithm for multi-agent pathfinding. In *IJCAI*, pages 740–746, 2015.

M. Ryan. Constraint-based multi-robot path planning. In *ICRA*, pages 922–928, 2010.

G. Sharon, R. Stern, M. Goldenberg, and A. Felner. The increasing cost tree search for optimal multi-agent pathfinding. *Artificial Intelligence*, 195:470–495, 2013.

G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant. Conflict-based search for optimal multi-agent pathfinding. *Artif. Intell.*, 219:40–66, 2015.

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

T. Standley. Finding optimal solutions to cooperative pathfinding problems. In *AAAI*, pages 173–178, 2010.

P. Surynek, A. Felner, R. Stern, and E. Boyarski. Boolean satisfiability approach to optimal multi-agent path finding under the sum of costs objective. In *AAMAS*, 2016.

P. Surynek. On propositional encodings of cooperative pathfinding. In *ICTAI*, pages 524–531, 2012.

P. Surynek. Compact representations of cooperative pathfinding as SAT based on matchings in bipartite graphs. In *ICTAI*, pages 875–882, 2014.