Boolean Satisfiability Approach to Optimal Multi-agent Path Finding under the Sum of Costs Objective

(Extended Abstract)

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

This paper focuses on finding optimal solutions to the *multi-agent path finding* (MAPF) problem over undirected graphs where the task is to find non-colliding paths for multiple agents, each with a different start and goal position. An encoding of MAPF to Boolean satisfiability (SAT) is already known to the *makespan optimal* variant of the problem. In this paper we present the first SAT-solver for minimizing the *sum of costs* enabled by introducing *cardinality constraints* into the SAT encoding. An experimental evaluation on grid graphs indicate promising performance of the new SAT-based method in comparison with the best variants of previous sum-of-costs search solvers.

Keywords

Multi-agent path finding (MAPF), Boolean satisfiability (SAT), sum of costs objective, makespan objective

1. INTRODUCTION AND BACKGROUND

The *multi-agent path finding* (MAPF) problem consists a graph, $G = (V, E)$ and a set of m agents labeled $a_1, a_2, \ldots a_m$, where each agent a_i has a start position in V and a goal position in V. Time is discretized into time points. At each time step an agent can either *move* to an adjacent empty location or *wait* in its current location. The task is to find a *valid solution*, i.e., a sequence of move/wait actions for each agent a_i, moving it from the start to the goal position such that agents do not *conflict*, i.e., do not occupy the same location at the same time.

MAPF has many practical applications [3], and can model many real-world problems in games, traffic control, robotics, aviation and more. The scope of this paper is limited to the *centralized setting*, in which we assume that the agents are *fully cooperative* and centrally controlled.

MAPF is usually solved aiming at minimizing a global cumulative cost function. Two such cost functions that are commonly used in the literature are the *sum of costs* [8, 5, 4] and the *makespan* [9, 10]. The *sum of costs* is the summation, over all agents, of the number of time steps required to reach the goal location while the *makespan* is the maximum of the number of times steps over all agents.

Finding makespan optimal and sum-of-cost optimal solutions are both NP-Hard [11, 9] as the state-space grows exponentially with m (# of agents). Therefore, many suboptimal solvers based on rules or relaxed search were developed and are usually used when m is large [6, 1].

The focus of this paper is within the class of *optimal MAPF solvers*. Research on optimal solutions sheds light on the theoretical hardness of the problem and on its attributes and characteristics. Optimal solvers are divided into two main classes.

(1) Reduction-based solvers which reduce MAPF to problems such as CSP [3], SAT, Inductive Logic Programming [10] (ILP) [11] and Answer Set Programming (ASP) [2]. These papers mostly provide and prove a polynomial-time reduction from MAPF to these problems typically for the makespan variant.

(2) Search-based solvers consider all the different ways to place m agents into V vertices, one agent per vertex. Many enhancements of the basic A*-based approach have been developed like *Independence Detection* and *Operator Decomposition* [8]. Another approach is used in the *increasing cost tree search* (ICTS) [5] and *conflict based search* algorithms (CBS, MA-CBS, and ICBS [4]) - they use a two-level framework where the search is divided between a more abstract high-level and low-level which places agents to vertices according to constraints from the high-level.

1.1 Contributions

Most of the search-based algorithms can be easily modified to the makespan variant by modifying the cost function and the way the state-space is represented. By contrast, the reduction-based algorithms are not trivially modified to the sum-of-costs variant and sometimes a completely new reduction is needed.

In this paper we address this challenge and develop the first SAT-based solvers for the sum-of-costs variant which is enabled by integrating so called *cardinality constraint* [7] for bounding the sum-of-costs.

2. OPTIMAL COST MAPF SOLVING

A significant difficulty in applying SAT-based approach to the *sum of costs* metric is the numeric character of this objective function, which contrasts with the modeling abilities of the SAT paradigm where no numeric but only Boolean

Appears in: *Proceedings of the 15th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2016), J. Thangarajah, K. Tuyls, C. Jonker, S. Marsella (eds.), May 9–13, 2016, Singapore.*

variables are available. Fortunately, the SAT modeling practice offers another technique that can be used for the purpose of calculating and bounding the cost within SAT - *cardinality constraints* [7]. The cardinality constraint bounds the number of propositional variables from the given set that can be set to `TRUE`.

We integrated cardinality constraints into existing SAT-approaches to *makespan optimal* MAPF solving to obtain a *cost optimal* approach. Particularly, we adopted the core concept of a *time expansion graph* (TEG) which is a graphical representation of agents' arrangements at individual time-steps. In TEG, the underlaying graph G is duplicated for all time-steps from 0 up to the given bound. Each *time expansion* in TEG corresponds to adding propositional variables and constraints for a single time-step into the formula so that any of possible arrangement of agents at that time step can be represented. Constraints added as clauses into the formula ensure that the formula is satisfiable if and only if there is a solution of the given MAPF of makespan equal to the number of time expansions.

By the simple strategy, that iteratively increases the number of time expansions [10] (that is, makespans $0, 1, 2, ...$ are tried) until a satisfiable formula is obtained, the optimal makespan can be calculated (the corresponding makespan optimal solution can be extracted from the formula as well).

Furthermore, agent's actions that have non-zero cost are mapped to positive valuations of certain propositional variables. Bounding the number of these variables that are assigned `TRUE` by the cardinality constraint then leads to the desirable bounding of the *sum of costs*. The search for optimal sum-of-costs is done in a similar way as in the case of makespan optimal approach - cost bounds $\xi_0, \xi_0+1, \xi_0+2, ...$ are tried, where ξ_0 is the lower bound of the sum-of-costs calculated as the sum of lengths of shortest path between agent's initial positions and goals. The cost bound increasing strategy needs to be accompanied with increasing the number of time expansions in the TEG (makespan bound) so that any solution of the given cost can be represented in the corresponding TEG.

3. EXPERIMENTAL EVALUATION

We performed experiments on 4-connected grids with randomly placed obstacles, which are a standard MAPF benchmark [6]. Specifically, we experimented on 8×8, 16×16, and 32×32 grids, in which 10% of the vertices are occupied by an obstacle. The initial position of the agents were randomly selected (all vertex had uniform probability).

Two variants of our novel encoding called BASIC-SAT and MDD-SAT with several state-of-the-art search-based techniques, namely the ICTS [5] and ICBS [4] were compared. The MDD-SAT encoding uses the technique of MDD [5] to reduce the size of TEG which consequently reduces the size of the Boolean formulae.

[1] The presented SAT-based approach was implemented in C++. As a SAT solver, we used `Glucose 3.0`. The cardinality constraints were encoded using a standard circuit based encoding called *sequential counter* in [7]. ICTS and ICBS were implemented in C#, based on their original implementation.

For each number of agents, we generated 10 random instances as explained above. Experimental results achieved for the grid 32×32 are presented in Figure 1. We measured the number of problem instances solved by each algorithm under a timeout of 512 seconds and the average runtime over the cases that all the algorithms solved all the 10 instances.

[1]All experiments were performed on a computer with CPU Xeon 2Ghz, 12 Gb of memory, and Linux kernel 3.5.0-48.

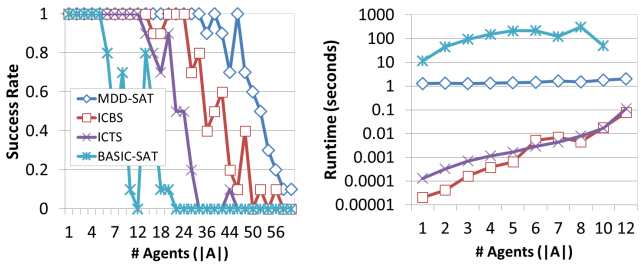

Figure 1: Success rate and runtime on 32×32 grid.

4. DISCUSSION AND CONCLUSIONS

A trend observed in grid experiments is that our MDD-SAT encoding solves more instances and for more agents than all other algorithms. It indicates that SAT-based solvers would be able to scale to denser and harder problems than the search-based algorithms.

The runtime results suggest that while search-based solver may be faster for sparse domains with a small number of agents, the SAT encodings, and in particular MDD-SAT, is expected to be the algorithm of choice for complicated scenarios.

REFERENCES

[1] B. de Wilde, A. W. ter Mors, and C. Witteveen. Push and rotate: cooperative multi-agent path planning. In *AAMAS*, pages 87–94, 2013.

[2] E. Erdem, D. G. Kisa, U. Oztok, and P. Schueller. A general formal framework for pathfinding problems with multiple agents. In *AAAI*, 2013.

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

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

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

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

[7] C. Sinz. Towards an optimal CNF encoding of boolean cardinality constraints. In *CP 2005*, pages 827–831, 2005.

[8] T. S. Standley. Finding optimal solutions to cooperative pathfinding problems. In *AAAI*, 2010.

[9] P. Surynek. An optimization variant of multi-robot path planning is intractable. In *AAAI*, 2010.

[10] P. Surynek. Compact representations of cooperative path-finding as SAT based on matchings in bipartite graphs. In *ICTAI*, pages 875–882, 2014.

[11] J. Yu and S. M. LaValle. Structure and intractability of optimal multi-robot path planning on graphs. In *AAAI*, 2013.