Integration of Independence Detection into SAT-based Optimal Multi-Agent Path Finding
A Novel SAT-based Optimal MAPF Solver

Pavel Surynek[1,2], Jiří Švancara[2], Ariel Felner[3] and Eli Boyarski[4]
[1]*National Institute of Advanced Industrial Science and Technology (AIST), Tokyo, Japan*
[2]*Faculty of Mathematics and Physics, Charles University, Prague, Czech Republic*
[3]*Ben Gurion University, Beer-Sheva, Israel*
[4]*Bar-Ilan University, Ramat-Gan, Israel*

Keywords: Multi-Agent Path-Finding (MAPF), Independence Detection (ID), Propositional Satisfiability (SAT), Cost Optimality, Makespan Optimality, Sum-of-costs Optimality, SAT Encodings, Path-Finding on Grids.

Abstract: The problem of optimal multi-agent path finding (MAPF) is addressed in this paper. The task is to find optimal paths for mobile agents where each of them need to reach a unique goal position from the given start with respect to the given cost function. Agents must not collide with each other which is a source of combinatorial difficulty of the problem. An abstraction of the problem where discrete agents move in an undirected graph is usually adopted in the literature. Specifically, it is shown in this paper how to integrate independence detection (ID) technique developed for search based MAPF solving into a compilation-based technique that translates the instance of the MAPF problem into propositional satisfiability formalism (SAT). The independence detection technique allows decomposition of the instance consisting of a given number of agents into instances consisting of small groups of agents with no interaction across groups. These small instances can be solved independently and the solution of the original instance is combined from small solutions eventually. The reduction of the size of instances translated to the target SAT formalism has a significant impact on performance as shown in the presented experimental evaluation. The new solver integrating SAT translation and the independence detection is shown to be state-of-the-art in its class for optimal MAPF solving.

1 INTRODUCTION

Multi-Agent Path Finding (MAPF) is the task is of finding collision free paths for a set of mobile agents so that each agent can reach its goal position from given start by following its path (Kornhauser et al., 1984, Silver, 2005, Surynek, 2009, Sharon et al. 2013). The MAPF problem recently attracted considerable attention from the research community and many concepts and techniques have been devised to address this problem.

An abstraction in which an environment with agents is represented by *undirected graph* is used in the literature (Wilson, 1974, Ryan, 2008). Agents in this abstraction are items placed in vertices of the graph. Edges represent passable regions. Physical space occupancy of agents is represented by the restriction that at most one agent can be placed in each vertex. The time is discrete which means that agents can do a single move in a time step.

Various movement schemes exist for this MAPF abstraction graph. Usually an agent can move into an unoccupied neighbor vertex not entered by another agent at the same time – this will be called *move-to-unoccupied* variant. Obviously, this variant requires at least one vertex in the graph unoccupied to be able to perform some movements at all.

But other variants like chain movement of agents where a chain of agents moves all at once with only the leader entering the unoccupied vertex exist (Surynek, 2010). Even cases with no unoccupied vertex in the graph were described in the literature (Yu, LaValle, 2013a). These usually allow movements of agents by rotating them along non-trivial cycles in graph (that is, cycles containing at least 3 vertices. Otherwise, allowing rotation over a trivial cycle consisting of a single edge would simplify the problem to

Surynek P., Åăvancara J., Felner A. and Boyarski E.
Integration of Independence Detection into SAT-based Optimal Multi-Agent Path Finding - A Novel SAT-based Optimal MAPF Solver.
DOI: 10.5220/0006126000850095
In *Proceedings of the 9th International Conference on Agents and Artificial Intelligence (ICAART 2017)*, pages 85-95
ISBN: 978-989-758-220-2

a practically not useful variant as arbitrary swaps of pairs of agents would then be possible).

The techniques shown in this paper are generic across all these variants although we base our presentation just on the basic variant *move-to-unoccupied*.

The MAPF problem and its variants are strongly practically motivated. Applications range from navigation of multiple mobile robots (Berg et al., 2010, Čáp et al., 2013), through traffic optimization (Michael et al., 2010, Kim et al., 2014), to movement planning in computer games (Wang, Botea, 2008). We refer the reader to various studies such as (Sharon et al. 2013, 2015) for the detailed survey of applications.

1.1 Optimality in MAPF

In this paper, we specifically address *optimal* MAPF in which paths that are optimal with respect to a given objective are searched. The two basic objectives studied in the literature are *makespan* (Surynek, 2014) and *sum-of-costs* (Standley & Korf, 2011, Sharon et al., 2013).

Under the **makespan objective** the aim is to obtain a plan that can be executed in as short as possible time while each movement consumes 1 unit of time. In the terms of agents /paths, we need the longest path out of all the paths to be as short as possible.

The **sum-of-costs objective** assumes that unit costs are assigned to actions agents can do where action is either a movement or a wait action. The cost of plan is the sum of action costs along all the paths and over all the agents. The aim is to obtain a plan with the minimum cost. Intuitively, the sum-of-costs objective corresponds to the energy consumed by agents when moving.

As we will show later, there may be situations where the increase in the sum-of-costs leads to a shorter makespan. This has practical/physical analogy where sometimes time can be saved at the cost of higher energy consumption.

Finding a feasible solution of MAPF can be done in polynomial time (Kornhauser et al., 1984, de Wilde et al., 2014). Adding any of the discussed objectives renders the decision version of MAPF (that is, we ask a *yes/no question* if a given MAPF has a solution of specified makespan/sum-of-costs) to be NP-complete (Ratner, Warmuth, 1990, Surynek, 2010).

We will keep the further description around the sum-of-costs variant but it is important to note that the presented techniques apply for the makespan variant as well.

1.2 Contributions to SAT-based MAPF

One of successful approaches for solving MAPF optimally is to translate the decision version into *propositional formula* (Kautz, Selman, 1999, Huang et al. 2010). The formula is *satisfiable* if and only if the instance of MAPF is solvable for a given value of the objective function. Assuming that satisfiability of such formula is a non-decreasing function of the value of objective function, it is easy to obtain the optimum by querying the satisfiability multiple times. A trivial strategy of increasing the value of objective function by one turned out to be the most efficient so far (Surynek et al., 2016) – this is mostly because of the non-uniform difficulty of each query.

Satisfiability of the formula can be decided by an off-the-shelf SAT solver (Biere et al., 2009, Audemard, Simon, 2009) which is one of the advantages of the SAT-based approach. All the very advanced techniques developed in recent decades for SAT solving are employed for solving MAPF - SAT Competitions (Balint et al., 2015) refers nicely about the huge progress in SAT solvers.

The most significant bottleneck of all the existing SAT-based algorithms for MAPF is the large size and combinatorial difficulty of the target propositional formula that grow significantly with the increasing number of agents as well as with growing size of the underlying graph.

This kind of growth of combinatorial difficulty has already been addressed by Standley (2010) in his search-based optimal MAPF solving algorithm. Standley described a method called *independence detection* (ID) that tries to determine the smallest possible groups of agents for which paths can be found independently of other groups. The ID technique turned out to be extremely beneficial when integrated with an algorithm for finding paths that is exponential in the number of agents. This is also the case of SAT-based MAPF solving.

Our contribution is integrating ID with MDD-SAT the most recent SAT-based MAPF solver (Surynek et al., 2016). As there are differences in how the original Standley's search-based algorithm and SAT-based approach work we suggested modifications to ID to be compatible with the SAT-based approach. Our new solver is called MDD-SAT+ID following the notation of (Standley, 2010). Conducted experiments demonstrate similar performance benefit as in the case of original application of ID. Considering that MDD-SAT has been state-of-the-art for a certain class of MAPF instances, the new MDD-SAT+ID represents new progress.

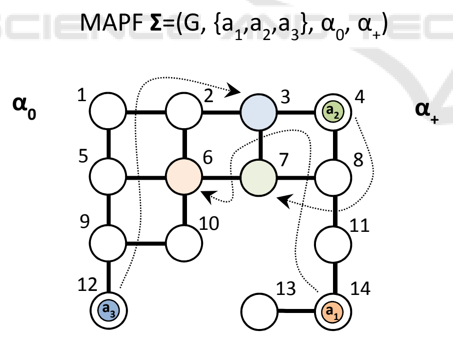
The paper is organized as follows. After the formal introduction of the MAPF problem a brief exposition of related work is done. Then, the original ID is recalled and integration of ID with the SAT-based approach is presented. Finally, an experimental evaluation with grids and large maps is presented.

2 DEFINITIONS

An arbitrary **undirected graph** can be used to model the environment where agents are moving. Let $G = (V, E)$ be such a graph where $V = \{v_1, v_2, \dots, v_n\}$ is a finite set of vertices and $E \subseteq \binom{V}{2}$ is a set of edges.

The placement of agents in the environment is modeled by assigning them vertices of the graph. Let $A = \{a_1, a_2, \dots, a_m\}$ be a finite set of *agents*. Then, an arrangement of agents in vertices of graph G will be fully described by a *location* function $\alpha: A \longrightarrow V$; the interpretation is that an agent $a \in A$ is located in a vertex $\alpha(a)$. At most **one agent** can be located in each vertex; that is α is uniquely invertible.

Definition 1 (MAPF). An instance of *multi-agent path-finding* problem is a quadruple $\Sigma = [G = (V, E), A, \alpha_0, \alpha_+]$ where location functions α_0 and α_+ define the initial and the goal arrangement of a set of agents A in G respectively.

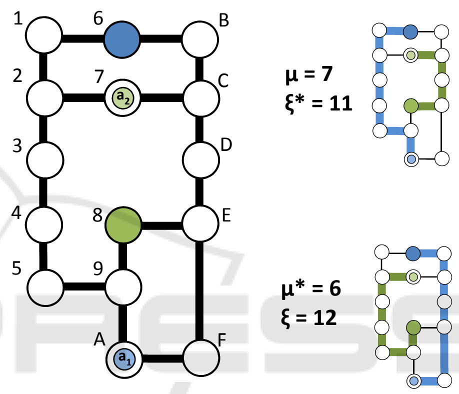


	α_0	α_1	α_2	α_3	$\alpha_4 = \alpha_+$
a_1	v_1	v_1	v_2	v_5	v_8
a_2	v_2	v_3	v_3	v_3	v_3
a_3	v_7	v_4	v_4	v_1	v_2

Figure 1: An example of a MAPF instance with three agents a_1, a_2, and a_3. A solution of the instance is shown below.

The dynamicity of the model assumes a discrete time divided into time steps. An arrangement α_i at the i-th time step can be transformed by a transition action which instantaneously moves agents in the non-colliding way to form a new arrangement α_{i+1}. The transition between α_i and α_{i+1} must satisfy the following *validity conditions*:

- $\forall a \in A$ either $\alpha_i(a) = \alpha_{i+1}(a)$ or $\{\alpha_i(a), \alpha_{i+1}(a)\} \in E$ holds (1)
  (agents move along edges or wait at their current location),
- $\forall a \in A \;\; \alpha_i(a) \neq \alpha_{i+1}(a) \Rightarrow \alpha_i^{-1}(\alpha_{i+1}(a)) = \perp$
  (agents move to vacant vertices only), and (2)
- $\forall a, b \in A \;\; a \neq b \Rightarrow \alpha_{i+1}(a) \neq \alpha_{i+1}(b)$ (3)
  (no two agents enter the same target/unique invertibility of resulting arrangement).



Optimal **makespan μ***
Sub-optimal sum-of-costs **ξ**

	α_0	α_1	α_2	α_3	α_4	α_5	$\alpha_6 = \alpha_+$
a_1	A	F	E	D	C	B	**6**
a_2	7	2	3	4	5	9	**8**

Optimal **sum-of-costs ξ***
Sub-optimal makespan **μ**

	α_0	α_1	α_2	α_3	α_4	α_5	α_6	$\alpha_7 = \alpha_+$
a_1	A	9	5	4	3	2	1	**6**
a_2	7	C	D	E	**8**	**8**	**8**	**8**

Figure 2: An instance of the MAPF problem in which no *makespan* optimal solution is sum-of-costs optimal and no *sum-of-costs* optimal solution is makespan optimal.

The task in MAPF is to transform α_0 using above valid transitions to α_+. An illustration of MAPF and its solution is depicted in Figure 1.

Definition 2 (MAPF solution). A *solution* MAPF instance $\Sigma = [G, A, \alpha_0, \alpha_+]$ is a sequence of arrangements $[\alpha_0, \alpha_1, \alpha_2, \dots, \alpha_\mu]$ where $\alpha_\mu = \alpha_+$ and α_{i+1} is a result of valid transition from α_i for every $= 1,2,\ldots,\mu-1$. □

Makespan μ is the total number of time steps until the last agent reaches its destination. *Sum-of-costs* denoted ξ is the sum of path costs per individual agents. Each action (including wait) of an agent before it reaches its goal has unit cost.

2.1 Makespan vs. Sum-of-costs

There exists an instance in which all the sum-of-costs optimal solutions are not makespan optimal. Similarly, none of the makespan optimal solution is sum-of-costs optimal there (see Figure 2 for illustration).

In the SAT-based optimal MAPF solver described below, a proper relation between makespan and sum-of-costs need to be found as both objectives are bounded during search. We need to ensure that smallest cost found under the given makespan bound is optimal (see (Surynek et al., 2016) for more detailed discussion).

3 RELATED WORK

Many other successful algorithms exist for the optimal MAPF solving. The state-of-the-art search-based algorithms (though there is no universal winner) include *increasing cost tree search* - ICTS (Sharon et al., 2013), *conflict base search* - CBS (Sharon et al., 2015), and *improved CBS* – ICBS (Boyarski et al., 2015). These algorithms excel in setups with relatively few agents on large maps.

Another research direction is represented by methods based on reduction of the MAPF problem to another formalism. Except the SAT as a target formalism, successful attempts to reduce MAPF to *constraint optimization problem* (Ryan, 2010), *inductive logic programming* (Yu, LaValle, 2013b), and *answer set programming* (Erdem et al., 2013) have been made. These approaches (the SAT approach including) can be generally characterized by a high performance in MAPFs with small underlying graph densely populated with agents. This is a natural outcome of the maturity of solvers used to solve hard combinatorial problems in the target formalism.

Recently new research directions driven by applications have been identified in the MAPF context. For example it is not always necessary to distinguish between individual agents – see (Ma, 2016) for detailed survey.

4 INDEPENDENCE DETECTION

Our major aim is to increase performance of the SAT-based MAPF solver by reducing the number of agents need to be considered at once. This has been successfully done in search based methods via a technique called *independence detection*.

In this section, we will describe the original method of independence detection proposed by Standley (2010). The main idea behind this technique is that difficulty of MAPF solving optimally grows exponentially with the number of agents. It would be ideal, if we could divide the problem into a series of smaller sub problems, solve them independently at low computational effort, and then combine them.

The simple approach, called *simple independence detection* (SID), assigns each agent to a group so that every group consists of exactly one agent. Then, for each of these groups, an optimal solution is found independently. Every pair of these solutions is evaluated and if the two groups' solutions are in conflict (that is, when collision of agents belonging to different group occurs), the groups are merged and replanned together. If there are no conflicting solutions, the solutions can be merged to a single solution of the original problem. This approach can be further improved by avoiding merging of groups.

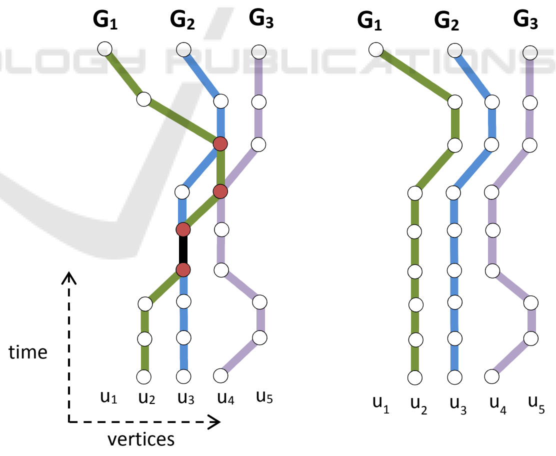


Figure 3: A schematic illustration of path replanning within the independence detection technique. A path for the group G_1 conflicted with paths of other two groups (left part). Then path for G_1 has been successfully replanned (right part).

Generally, each agent has more than one possible optimal path. However, SID considers only one of these paths. The improvement of SID known as *independence detection* (ID) is as follows. Let's have two conflicting groups G_1 and G_2. First, try to replan G_1

so that the new solution has the same cost and the steps that are in conflict with G_2 are forbidden. If no such solution is possible, try to similarly replan G_2. If this is not possible, merge G_1 and G_2 into a new group. In case either of the replanning was successful, that group needs to be evaluated with every other group again. This can lead to infinite cycle. Therefore, if two groups were already in conflict before, merge them without trying to replan.

Standley uses ID in combination with the A* algorithm. While planning, it is preferred to find paths that create the least possible amount of conflicts with other groups that have already planned paths. For this purpose, the *conflict avoidance table* is created (see Algorithm 1 for pseudo-code).

Algorithm 1: MAPF solving algorithm based on **independence detection** technique. Planning for groups is always done to have least number of conflicts with respect to conflict avoidance table.

```
assign each agent to a group;
plan a path for each group by A*;
fill conflict avoidance table;
while conflicting groups exist
   G1, G2 = conflicting groups;
   if G1, G2 not conflicted before
      replan G1 by A* with illegal
         moves based on G2;
      if failed to replan G1
         replan G2 by A* with illegal
            moves based on G1;
      endif
   endif
   if no alternate paths for G1, G2
      merge G1 and G2;
      plan a path for new
         group by A*;
   endif
   update conflict avoidance table;
end
return combined paths of all groups;
```

The table stores moves of agents in other groups. In case A* has a choice between several nodes with the same minimal $f()$ cost, the one with least amount of conflicts is expanded first. This technique yields an optimal solution that has a minimal number of conflicts with other groups. This property is useful when replanning of a group's solution is needed.

Both SID and ID do not solve MAPF on their own, they only divide the problem into smaller subproblems that are solved by any possible MAPF algorithms. Thus, ID and SID are general frameworks which can be executed on top of any MAPF solver.

5 INTEGRATING ID INTO SAT

We describe how to integrate ID into SAT-based solving of MAPF in this section. Recently, SAT-based MAPF solving has been shown to be considerably successful technique for obtaining both makespan or sum-of-costs optimal solutions.

The basic idea of SAT-based MAPF solving is to reduce the decision question of whether there exists a solution of a given MAPF of a given value of objective (makespan or sum-of-costs) into a propositional formula. This process of reduction is called *encoding* in the literature. There exist many types of encodings of MAPF to propositional formulae (Surynek et al., 2016).

The resulting propositional formula is then solved by the off-the-shelf SAT solver (Audemard, Simon, 2013). If the formula is satisfiable, a solution to MAPF is obtained by interpreting back the meaning of propositional variables in the satisfying valuation of the formula. Unsatisfiable formula means that the given MAPF has no solution under the given value of objective function.

Encodings are generally based on time expansion of the underlying graph G, that is, the graph is copied for each time step up to the given limit. In such expanded graph, we are able to record positions of all the agents at individual time-steps which is represented as a standard placement of agents at the given level of the time expanded graph. This is the common feature of all existing encodings for optimal MAPF solving. They however differ in how the positions of agents within levels of the time expanded graph are represented. Some encodings use binary representation of positions (sometimes called also log-space representation as a vector of $\lceil \log_2 n \rceil$ propositional variables is used to represent possible n positions of an agent).

Other representations use a single propositional variable per time/space position within the time expanded graph. These are sometimes called *direct representations*.

Although binary representations result in smaller formulae in terms of the number of variables and constraints they provide limited search space pruning and propagation. Hence, they are usually outperformed by encodings that use direct representation where the benefit of propagation outweighs larger size of the formulae.

The optimal solution in the SAT-based approach is searched by a sequence of queries to the SAT solver with increasing values of the objective encoded in formulae. Assuming that the solvability of MAPF (satisfiability of a formula) is a non-decreasing function of the value of the objective, the optimum can be found easily. The most efficient strategy is to simply start from the lower bound of the objective and increment it until the first satisfiable formula is encountered. The satisfying valuation of the formula represents an optimal solution. This strategy is applicable both to makespan to sum-of-costs optimization.

5.1 Multi-value Decision Diagrams

As discussed in (Surynek et al., 2016), the limitation of many existing encodings is their size which is implied by the size of the time expanded graph. To mitigate this limitation Surynek et al. took inspiration from another successful search-based solver called *increasing cost tree search* (ICTS) (Sharon et al., 2013).

ICTS uses a data structure called *multi-value decision diagram* (MDD) that is very similar to *time expanded graph* considered for a single agent (TEG). But unlike TEG, only those nodes that can be actually visited by the agent under the given value of objective function are included in MDD.

For example, assume that the value of objective function requires agent a makes no more than t moves. Then MDD for agent a will contain only vertices on paths connecting $\alpha_0(a)$ with $\alpha_+(a)$ of lengths at most t.

Using MDDs can rule out many vertices that would be normally considered in standard time expansions. Experiments confirmed that MDDs enabled using the SAT-based approach even for large MAPF instances for which the size of encodings without MDD was prohibitive.

5.2 MDD-SAT+ID

We will now describe integration of a variant of independence detection into the SAT-based solver. This represents the main contribution of this paper.

The standard SAT-based approach that uses MDDs, called MDD-SAT, (Surynek et al., 2016) has still considerable limitation when compared to existing search based techniques.

MDD-SAT considers the entire MAPF instance as a whole which significantly limits the scalability of this method. With large instances and many agents, MDD-SAT will eventually encounter formula of prohibitive size even with the use of MDDs. In all the other optimal search-based solvers some variant of ID is used to further mitigate the size of the instance needed to be tackled at once.

Algorithm 2: Independence detection in the **SAT-based framework**. Conflict aviodance is strictly required.

```
assign each agent to a group;
plan a path for each group
   $G_1,…,G_k$ by MDD-SAT;
fill conflict avoidance table;
while conflicting groups exist
   $G_1$, $G_2$ = conflicting groups;
   if $G_1$, $G_2$ not conflicted before
      replan $G_1$ by MDD-SAT with
         illegal moves
         based on $\{G_1,…,G_k\}-G_1$;
      if failed to replan $G_1$
         replan $G_2$ by MDD-SAT with
            illegal moves
            based on $\{G_1,…,G_k\}-G_2$;
      endif
   endif
   if no alternate paths for $G_1$, $G_2$
      merge $G_1$ and $G_2$;
      plan a path for
         new group by MDD-SAT;
   endif
   update conflict avoidance table;
end
return combined paths of all groups;
```

The logical step is hence to integrate a variant of ID into the SAT-based approach. We decided to do that for MDD-SAT as it is currently the state-of-the-art SAT-based solver for MAPF.

The SAT-based approach however requires modification of the original ID since in the propositional formula it is not possible to express preference that individual paths of groups of agents should avoid occupied positions in the *conflict avoidance table*. In the yes/no SAT environment we either manage to avoid occupied positions or not while in the negative case there is no easy tool how to control the number of conflicts.

The SAT-based version of ID works in similar way to the original version of Standley but instead of resolving conflicts between a pair of conflicting groups G_1 and G_2 it resolves conflict of group G_1 with all other groups. If this attempt is successful, G_1 is independent on others and the process can continue with resolving conflicts between remaining groups (see Figure 3 where G_1 has been made independent).

If the attempt to resolve conflict between G_1 and G_2 by making G_1 independent fails, the same is tried for G_2. If the attempt for G_2 fails too groups are merged. The pseudo-code is shown as Algorithm 2.

In contrast to original ID we strictly require avoidance with respect to the conflict avoidance table instead of stating it as a preference only. This is technically done by omitting the conflicting vertices in the MDD. The SAT approach does not allow to express a preference like in the search based algorithm. This is the reason why ID in the SAT-based solver differs from the original one.

6 EXPERIMENTS

We performed experimental comparison of the suggested MDD-SAT+ID solver with other state-of-the-art solvers – namely with the previous best SAT-based solver MDD-SAT and also with search-based algorithms ICTS and ICBS.

The MDD-SAT+ID has been implemented in C++ as an extension of an existing implementation of the MDD-SAT solver. A couple of minor improvements have been done in the original MDD-SAT encoding – some auxiliary propositional variables have been eliminated which reduced the size of the encoding and consequently saved runtime while generating formulae (this improvement affects both MDD-SAT and new MDD-SAT+ID used in presented experiments).

We used `Glucose 3.0` (Audemard, Simon, 2013) in MDD-SAT and MDD-SAT+ID which is a top performing SAT solver according to the recent SAT Competitions (Balint et al., 2015).

The complete implementation of the MDD-SAT+ID solver is available on-line to allow reproducibility of the presented results: ktiml.mff.cuni.cz/~surynek/research/icaart2017 .

ICTS and ICBS have been implemented in C#. The original implementations of these algorithms have been used.

All the tests were run on Xeon 2Ghz, and on Phenom II 3.6Ghz, both with 12 Gb of memory.

The experimental setup followed the scheme used in the literature (Silver, 2005) which tests MAPF algorithms on 4-connected grids. Let us note however that all the suggested algorithms are designed and implemented for general undirected graphs (the fact that grids are used in the experiments is not exploited to increase efficiency of solving in any way).

6.1 Small Grids Evaluation

The first series of experiments takes place on small square grids of sizes 8×8, 16×16, and 32×32 with 10% of vertices occupied by obstacles. In this setup of the environment, we increased population of agents from 1 and observed the runtime of all the solvers until no solver was able to solve the instance within the given time limit of 300 seconds (this was 20 agents for 8×8 grid, and 40 and 60 for 16×16 and 32×32 girds respectively).

Ten randomly generated instances per number of agents were used. The initial positions were generated by choosing a subset of vertices randomly. The goal arrangement has been generated as a long random walk from the initial state following valid moves – this ensured solvability of all the tested instances.

To be able to communicate results of experiments more easily we intuitively distinguish three different categories of instances with respect to the density of agents as follows. The behavior of solvers is then discussed with respect to these categories:

- **Low density** – few interactions among agents, paths for individual agents can be planned independently.

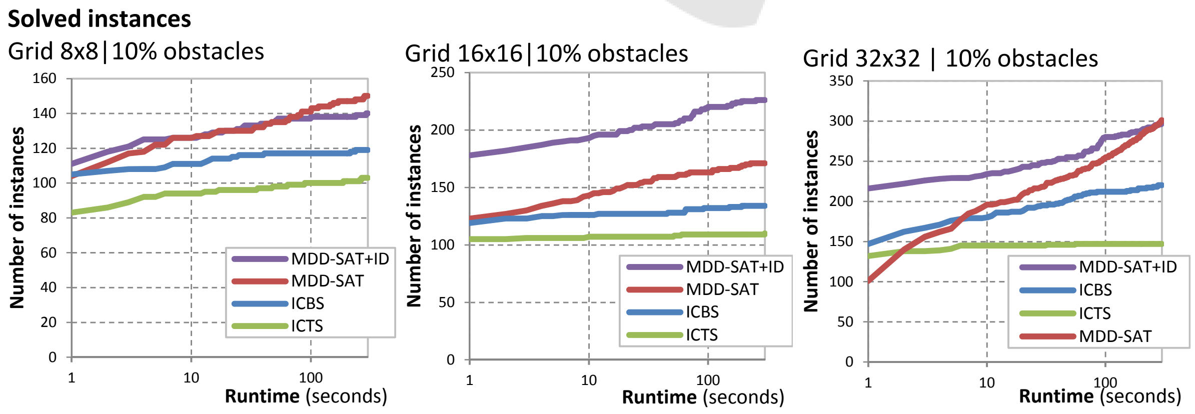


Figure 4: Results of experiments on **small grid maps** of sizes 8×8, 16×16, and 32×32. Figures show how many instances were solved within the given runtime. Clearly MDD-SAT and MDD-SAT+ID dominate in the test over search based algorithms ICTS and ICBS except few quickly solvable cases. Moreover, MDD-SAT+ID outperforms MDD-SAT in cases with low to medium density of agents.

- **Medium density** – some interaction among agents are inevitable but there exist multiple groups of agents that are independent of each other.
- **High density** – majority of agents are interdependent and form one large group.

The small grid experiment contains instances from all these three cases.

The hypothesis is that the ID technique will be helpful in instances with medium density of agents. We also expect that in the case of low density of agents there will be some benefit of ID since many agents will just follow their shortest paths towards goals in such a case. In low and medium density cases the complexity of the formula is not proportional to the difficulty of the instance.

Furthermore, we expect rather negative effect of using ID in instances with high density of agents. This is because of the fact that most agents will be gradually merged into a large group while the process of merging represents an overhead in such a case.

Experimental result for the small grids (see Figure 4) confirmed the hypothesis. MDD-SAT+ID clearly wins in low to medium density of agents. For the higher density, it tends to be outperformed by the original MDD-SAT.

6.2 Large Maps – Dragon Age

We also experimented on three structurally different large maps from Dragon Age: Origins (Sturtevant, 2012) – ost003d, den520d, and brc202d (see Figure 5). Our choice of maps is driven by the choice of authors in the previous literature (Sharon et al., 2015, Surynek et al. 2016).

We used setup with 16 and 32 agents randomly paced agents which represents low to medium density. Let us note that a case with high density of agents in the map of that size currently out of reach of any existing algorithm.

To obtain problems of various difficulties the distance of agents from initial positions to their goals has been varied in the range 8, 16, 24, …, 320.

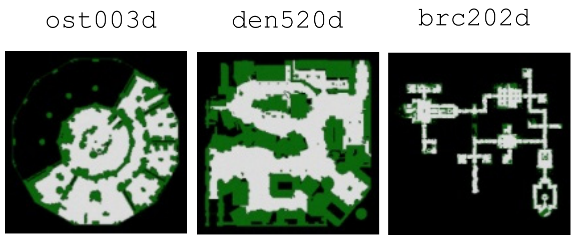


Figure 5: Illustration of large Dragon Age maps ost003d (size 194×194), den520d (size 257×256), and brc202d (size 481×530).

For each distance 10 random instances were generated in which initial positions were selected randomly and then random walk has been performed until all the agents reach at least the given distance from its initial position.

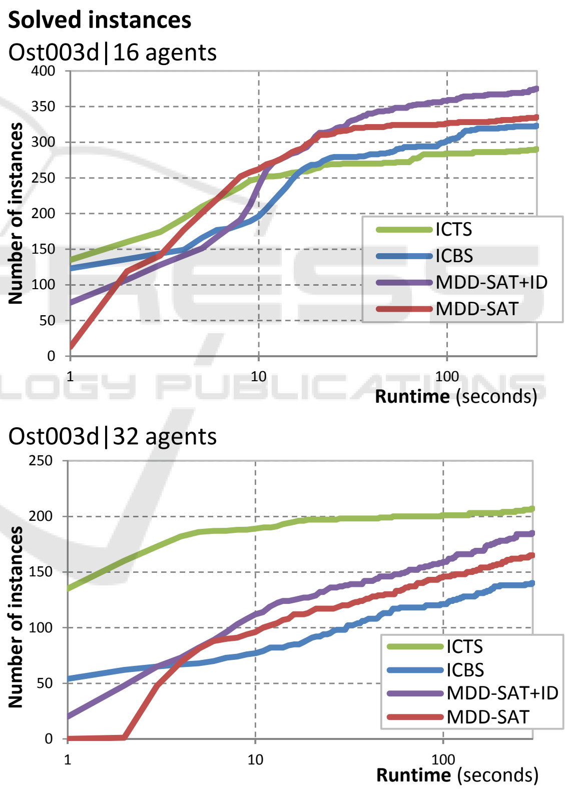


Figure 6: Results of experiments on Dragon Age map ost003d. MDD-SAT+ID outperforms MDD-SAT in harder instances but both are dominated by ICTS.

The hypothesis for large maps is that MDD-SAT+ID should dominate generally which in fact is the same hypothesis as in the case of small grids because here we have only the low-medium density case. However, as there are important structural differences between the three tested maps which impact is hardly predictable. Intuitively, ID should have been more beneficial in `ost003d` and `den520d` maps since in these maps there is more room to find alternative paths.

Results for the three Dragon Age maps are shown in figures 6, 7, and 8. Again the number of instances solved in the given runtime is shown. The difficulty (runtime) grows with the growing distance of agents from their goals in this setup.

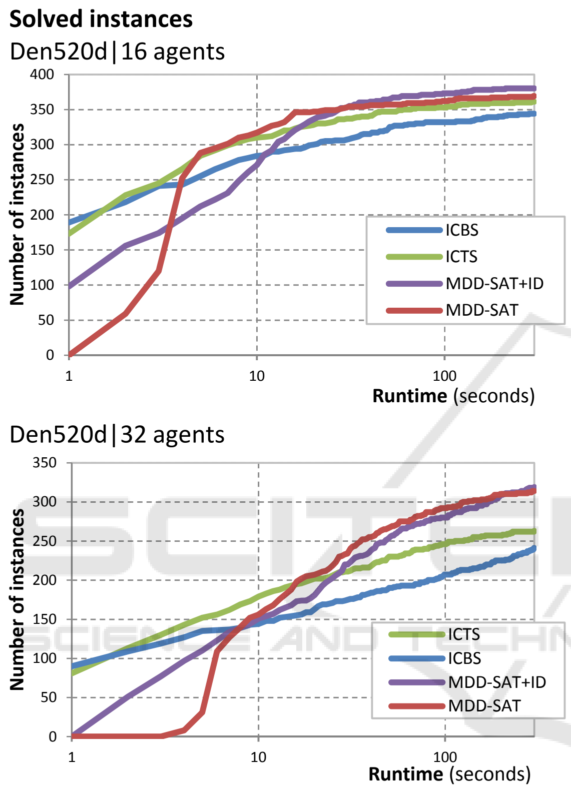


Figure 7: Results of experiments on Dragon Age map `den520d`. ID brings minor benefit in harder instances.

It can be read from these results that MDD-SAT+ID tends to outperform MDD-SAT in more difficult instances. In these instances, the interaction among agents in non-trivial but on the other hand the interdependence among agents is tractable by ID.

The intuitive hypothesis was not confirmed completely since surprisingly MDD-SAT is better than MDD-SAT+ID in easier instances of medium density category usually. Our initial intuitive hypothesis did not count with the fact that merging groups represents a big overhead in case of large maps. Hence, MDD-SAT+ID can show its benefit after the difficult of the formula modeling the entire instance prevails over the difficulty of group merging.

A surprising result was obtained in `brc202d` map where MDD-SAT+ID was a very clear winner in harder instances with 32 agents.

Moreover, we cannot say that SAT-based approach represented by MDD-SAT and MDD-SAT+ID is a universal winner as there are cases where ICTS and ICBS dominate (`ost003d` with 32 agents is such an example).

7 DISCUSSION

It can be generally observed that ID brings worthwhile improvement to MDD-SAT solver which by itself performs very well.

Experimental results indicate that there is a certain range of the density of agents though not precisely determined in our evaluation in which ID is beneficial while outside this range it cases an overhead.

The implementation of ID within the MDD-SAT+ID solver did not use any special reasoning about what groups of agents should be merged or not. The groups were processed in the ordering given by the original ordering of agents. We expect that more careful reasoning about merging can bring yet more improvements.

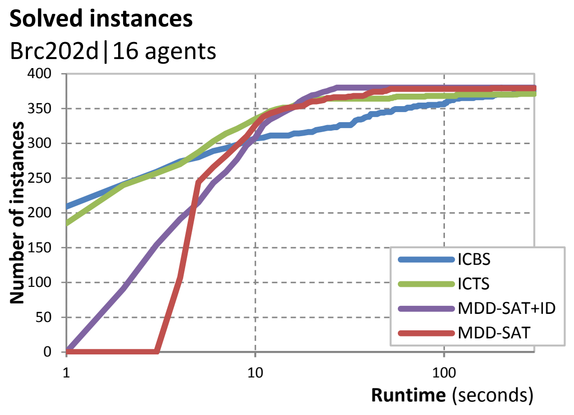
8 CONCLUSIONS

We described how to integrate existing technique of independence detection (ID) developed originally for search-based MAPF solver into the SAT-based approach to MAPF.

Experimental results confirm significant benefit of using ID within the SAT-based approach to optimal MAPF solving. The benefit is especially evident in instances with medium density of agents where interactions among agents are non-trivial but there exist group of agents that are independent of each other.

The suggested MDD-SAT+ID solver which is the result of integration of ID into an existing SAT-based MAPF solver MDD-SAT became a new state-of-the-art in optimal SAT-based MAPF solving. Moreover, the new MDD-SAT+ID performs well with respect to best search based solvers ICTS and ICBS though we cannot say there is a universal winner.

There are important future research directions which we just touched in this work. First, the performed experimental evaluation indicates the need to develop concepts for more precise classification of density and interaction among agents. Such a classification should ultimately lead to determining automatically in which cases ID would be beneficial and in which cases not.



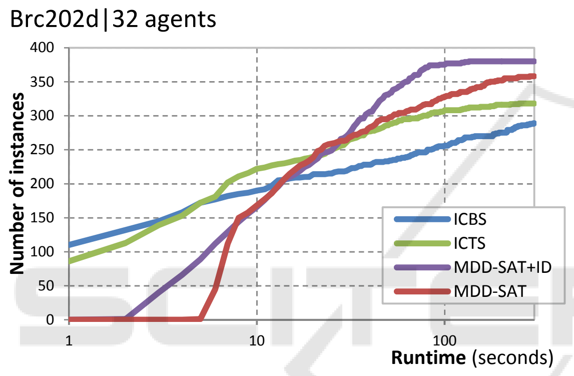


Figure 8: Results of experiments on Dragon Age map `brc202d`. ID brings significant improvement in harder instances with 32 agents.

The second future direction would become very apparent after a close look at the implementation. Currently we take groups of agents to be merged in the same order as they appear in the input. A more informed consideration which groups of agents should be merged may bring further reduction of the size of groups of agents.

ACKNOWLEDGEMENTS

This paper is supported by a project commissioned by the New Energy and Industrial Technology Development Organization Japan (NEDO), joint grant of the Israel Ministry of Science and the Czech Ministry of Education Youth and Sports number 8G15027, and Charles University under the SVV project number 260 333.

We would like thank anonymous reviewers for their constructive comments.

REFERENCES

Audemard, G., Simon, L., 2013. The Glucose SAT Solver. http://labri.fr/perso/lsimon/glucose/, 2013, [accessed in October 2016].

Audemard, G., Simon, L., 2009. Predicting Learnt Clauses Quality in Modern SAT Solvers. *Proceedings of the 21st International Joint Conference on Artificial Intelligence* (IJCAI 2009), pp. 399-404, IJCAI.

Balint, A., Belov, A., Heule, M., Järvisalo, M., 2015. SAT 2015 competition. http://www.satcompetition.org/, 2015, [accessed in October 2016].

Berg, J. van den, Snoeyink, J., Lin, M. C., Manocha, D., 2010. Centralized path planning for multiple robots: Optimal decoupling into sequential plans. *Proceedings of Robotics: Science and Systems V*, University of Washington, 2009, The MIT Press.

Biere, A., Heule, M., van Maaren, H., Walsh, T., 2009. Handbook of Satisfiability. IOS Press.

Boyarski, E., Felner, A., Stern, R., Sharon, G., Tolpin, D., Betzalel, O., Shimony, S.: ICBS: Improved Conflict-Based Search Algorithm for Multi-Agent Pathfinding. *Proceedings of the 24th International Joint Conference on Artificial Intelligence* (IJCAI 2015), pp. 740-746, IJCAI.

Čáp, M., Novák, P., Vokřínek, J., Pěchouček, M., 2013. Multi-agent RRT: sampling-based coop-erative pathfinding. International conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2013), pp. 1263-1264, IFAAMAS.

Erdem, E., Kisa, D. G., Öztok, U., Schüller, P., 2013. A General Formal Framework for Pathfinding Problems with Multiple Agents. *Proceedings of the 27th AAAI Conference on Artificial Intelligence (AAAI 2013)*, AAAI Press.

Huang, R., Chen, Y., Zhang, W., 2010. A Novel Transition Based Encoding Scheme for Planning as Satisfiability. Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI 2010), AAAI Press.

Kautz, H., Selman, B., 1999. Unifying SAT-based and Graph-based Planning. Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI 1999), pp. 318-325, Morgan Kaufmann.

Kim, D., Hirayama, K., Park, G.-K, 2014. Collision Avoidance in Multiple-Ship Situations by Distributed Local Search. *Journal of Advanced Computational Intelligence and Intelligent Informatics (JACIII)*, Volume 18(5), pp. 839-848, Fujipress.

Kornhauser, D., Miller, G. L., Spirakis, P. G, 1984. Coordinating Pebble Motion on Graphs, the Diameter of Permutation Groups, and Applications. *Proceedings of the 25th Annual Symposium on Foundations of Computer Science (FOCS 1984)*, pp. 241-250, IEEE Press.

Ma, H., Koenig, S., Ayanian, N., Cohen, L., Hoenig W., Kumar, T.K.S., Uras, T., Xu. H., Tovey, C., Sharon, G., 2016. Overview: Generalizations of Multi-Agent Path Finding to Real-World Scenarios. *IJCAI-16 Workshop on Multi-Agent Path Finding (WOMPF).*

Michael, N., Fink, J., Kumar, V., 2011. Cooperative manipulation and transportation with aerial robots. *Autonomous Robots*, Volume 30(1), pp. 73-86, Springer.

Ratner, D. and Warmuth, M. K., 1990. NxN Puzzle and Related Relocation Problems. *Journal of Symbolic Computation*, Volume 10 (2), pp. 111-138, Elsevier.

Ryan, M. R. K., 2008. Exploiting Subgraph Structure in Multi-Robot Path Planning. *Journal of Artificial Intelligence Research (JAIR)*, Volume 31, 2008, pp. 497-542, AAAI Press.

Ryan, M. R. K., 2010. Constraint-based multi-robot path planning. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2010)*, pp. 922-928, IEEE Press.

Sharon, G., Stern, R., Goldenberg, M., Felner, A., 2013. The increasing cost tree search for optimal multi-agent pathfinding. Artificial Intelligence, Volume 195, pp. 470-495, Elsevier.

Sharon, G., Stern, R., Felner, A., Sturtevant, N. R., 2015. Conflict-based search for optimal multi-agent pathfinding. *Artificial Intelligence*, 219, 40-66, Elsevier.

Silver, D., 2005. Cooperative Pathfinding. *Proceedings of the 1st Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE 2005)*, pp. 117-122, AAAI Press.

Standley, T., 2010. Finding Optimal Solutions to Cooperative Pathfinding Problems. *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI-2010)*, pp. 173-178, AAAI Press.

Standley, T., Korf, R. E., 2011. Complete Algorithms for Cooperative Pathfinding Problems. *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI 2011)*, pp. 668-673, IJCAI.

Sturtevant, N. R., 2012. Benchmarks for Grid-Based Pathfinding. *IEEE Transactions on Computational Intelligence and AI in Games*, Volume 4(2), pp. 144-148, IEEE Press.

Surynek, P., 2009. *A Novel Approach to Path Planning for Multiple Robots in Biconnected Graphs*. Proceedings of the 2009 IEEE International Conference on Robotics and Automation (ICRA 2009), pp. 3613-3619, IEEE Press.

Surynek, P., 2010. An Optimization Variant of Multi-Robot Path Planning is Intractable. *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI 2010)*, pp. 1261-1263, AAAI Press.

Surynek, P., 2014. Compact Representations of Cooperative Path-Finding as SAT Based on Matchings in Bipartite Graphs. *Proceedings of the 26th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2014)*, pp. 875-882, IEEE Computer Society.

Surynek, P., Felner, A., Stern, R., Boyarski, E., 2016. Efficient SAT Approach to Multi-Agent Path Finding Under the Sum of Costs Objective. *Proceedings of 22nd European Conference on Artificial Intelligence (ECAI 2016)*, pp. 810-818, IOS Press.

Yu, J., LaValle, S. M., 2013a. Structure and intractability of optimal multirobot path planning on graphs. *Proceedings of the 27th AAAI Conference on Artificial Intelligence (AAAI 2013)*, AAAI Press.

Yu, J., LaValle, S. M., 2013b. Planning optimal paths for multiple robots on graphs. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2013)*, pp. 3612-3617, IEEE Press.

Wang, K. C., Botea, A., 2008. Fast and memory-efficient multi-agent pathfinding, *Proceedings of the 18th International Conference on Automated Planning and Scheduling (ICAPS 2008)*, pp. 380-387, AAAI Press.

de Wilde, B., ter Mors, A., Witteveen, C.: Push and Rotate: a Complete Multi-robot Pathfinding Algorithm. *Journal of Artificial Intelligence Research (JAIR)*, Volume 51, pp. 443-492, AAAI Press, 2014.

Wilson, R. M., 1974. Graph Puzzles, Homotopy, and the Alternating Group. *Journal of Combinatorial Theory, Ser. B 16*, pp. 86-96, Elsevier.