Analysis and Utilisation of Conflicts in Multi-Agent Path Finding

Avgi Kollakidou,[1] **Leon Bodenhagen,** [1]

[1] Mærsk Mc-Kinney Møller Institute, University of Southern Denmark
avko@mmmi.sdu.dk, lebo@mmmi.sdu.dk

Abstract

With the escalation in deployments of robotic fleets in unstructured environments, the need to address the increasing number of conflicts in Multi-Agent Path Finding also rises. We discuss the evident issue of conflicts and analyse their spatial relationships. A method to use previous missions and their resulting conflicts to extract highways is proposed. The highways facilitate a modified heuristic for Conflict Based Search allowing for fewer initial conflicts and thus decreasing the computational complexity of the search. The importance of the analysis of conflict patterns is displayed with real life experiments of a simulated assembly line workplace.

Introduction

Mobile robots are progressively employed in an ever expanding variety of applications and in larger fleets. With the increasing number of robots within a finite environment, navigation becomes an issue. Multi-Agent Path Finding (MAPF) describes the problem of finding paths for a set of agents, which can be carried out simultaneously without the agents coming in conflict. The search for a solution is time consuming, especially with the increasing number of agents required as well as the maps and environment complexity the agents have to navigate in.

Conflict Based Search (CBS) is a two stage solution to the MAPF problem. The solution revolves around the conflicts occurring during the search and the constraints they impose (Sharon et al. 2015). CBS has its limitations, as discussed in (Sharon et al. 2013). It is affected by the number of agents as well as the environment's topology and size. The iterative nature of CBS and its large state-space (exponential to agent number), create a high computational load and memory strain. This impedes the use of CBS in real-time.

In this work, we investigate the conflicts occurring during the CBS search and the possibility of their exploitation for improved future planning. The understanding of such conflicts will enable improved initial plans with fewer conflicts through virtual structure introduced in the environment.

Some CBS variations attempted to address the computational strain of CBS by imposing structure to the environment. Wang and Botea (2008) proposed Flow Annotation Replanning which introduces restraints similar to traffic rules, e.g. one way streets, to avoid head on conflicts between agents. A decentralised approach where agents share information about their previous states and direction of travel, thus encouraging other agents to follow the same direction, was also considered (Jansen and Sturtevant 2008). The use of highways as a means to decrease conflicts in CBS, with manually generated directed graph edges was introduced (Cohen, Uras, and Koenig 2015). Automatic generation of highways was later proposed (Cohen et al. 2016), determined by the initial conflicts detected in CBS. These highways are inferred before each CBS attempt and applied only on the upcoming search. The information is not utilised any further for future plans.

We extend on previous ideas, especially in (Cohen et al. 2016), with a method to extract machine generated highways and use knowledge of previous missions to inform the path search. These highways facilitate CBS attempts in the same environment without pre-processing, utilising observations of previous missions.

Problem Definition

Multi-Agent Path Finding

The input to the MAPF problem is an undirected graph $G = (V, E)$, in the form of a grid, and a set of k agents with a starting vertex s_i and target vertex t_i each. An agent can occupy a single vertex and each vertex can accommodate only a single agent at a certain time step. All actions are carried out in unit time. Two types of actions can be performed with unit cost: move to an adjacent vertex, or wait. The qualifications for adjacency vary in different versions of MAPF. In this work, a vertex is adjacent to the 4 vertices that border it (west, north, east, south). No diagonal movement is admissible (Sharon et al. 2015; Stern et al. 2019).

We consider two types of conflicts: vertex and edge conflicts. A vertex conflict arises when two agents are planned to occupy the same vertex during the same time step. An edge conflict occurs when two agents are planned to traverse the same edge during the same time step.

Conflict Based Search

The CBS solution to the MAPF problem is divided in two levels. The two levels have distinct tasks: individual path planning, and constraint definition from detected conflicts. On the low level each agent searches for a path individually and independently, ignoring all other agents' positions

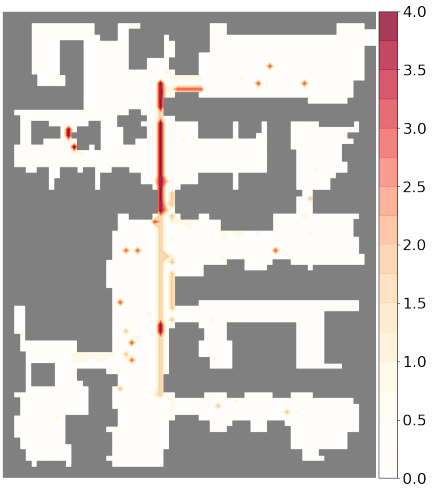

Figure 1: Accumulated conflict occurrences (25 scenarios of 30 missions each)

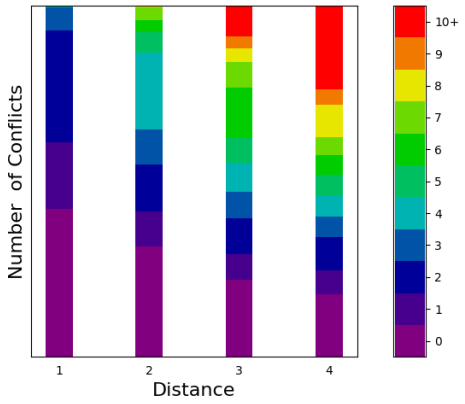

Figure 2: Spatial analysis of conflict occurrences with respect to other conflicts. The bars indicate the distribution of the conflicts according to the amount of other conflicts within a certain distance.

or plans, using the A* algorithm. A list of constraints, derived from previously existing conflicts, is provided by the upper-level. For the first low level search, an empty list of constraints exists. If a node is generated where agent α_i needs to occupy vertex v at time t and a constraint exists that involves the same configuration, that node is discarded. Likewise, if agent α_i is to traverse edge e at time t and a constraint exists involving the edge, the node is also dismissed.

On the high level, CBS creates a binary Constraint Tree (CT). The process begins with the investigation of the produced paths from the low level search for occurring conflicts. If a conflict arises, this is translated into constraints. We define one constraint for each agent involved in the form of $c = \langle \alpha_i,\ v/e,\ t \rangle$, where α_i is one of the involved agents, v/e is the involved vertex or edge and t the time step. The constraints are then added to the CT. Each CT node contains a set of constraints, a solution including a set of paths for all agents and the paths' total cost. The first node of the CT is initiated with an empty list of constraints. All new nodes inherit the list with previous constraints. Two child nodes are created from each conflict and both are explored. If more than two agents are involved in a conflict, these are resolved iteratively, two by two. With every new node, a low level search is triggered only for the agent involved in the constraint, with the rest of the paths remaining unchanged.

Conflict Analysis

A large number of agents combined with a rather narrow or complex environment leads to a sizable number of conflicts. This extensively increases the computational power and memory needed to solve the problem as well as the time required to reach the solution, making the algorithm unsuitable for real time applications. To explore possible conflict patterns and their potential applications for improving future path planning, we use an open-source MAPF benchmark for data generation and evaluation. The benchmark (Stern et al. 2019) consists of a set of maps and accompanying sets of agent missions, in the form of start and target locations. The maps have varying dimensions and complexities, e.g. real city maps, Dragon Age Origins (DAO video game) maps or simulated warehouse environments.

For each tested benchmark instance (map), multiple scenarios were used. The conflicts encountered in each scenario were combined and then analysed collectively. Fig. 1 shows the position and frequency of conflict occurrences in an example benchmark instance (DAO den312sd). Problematic areas can easily be seen along the corridor in the centre, even though the corridor is wide enough for three robots to traverse simultaneously (each vertex/cell fits exactly one agent). Reappearing conflicts are also visible next to obstacles, several corners and other narrow passages. Even though the topology of each map is different, a tendency for the conflicts to be more frequent in such areas was observed.

We analysed the spatial relationships of conflicts, to determine whether any patterns occur. Two inquiries were carried out. Firstly, the relationships of conflicts with each other were examined. For each recorded conflict, all other conflicts within a certain distance are tallied. The L1 norm distance is used, as it is also applied in A* for the heuristics. Fig. 2 shows the number of such conflicts in the vicinity with increasing distances (up to 4). A clustering of conflicts is evident. On average, conflicts have one additional conflict within a distance of one move. This is exacerbated in complicated maps. For visualisation purposes a threshold was defined, where all cases with more than 10 conflicts were accrued in one group. The maximum of 24 conflicts was recorded within a distance of 4 moves.

Furthermore, the conflict locations relative to obstacles were investigated. For all conflicts, we registered whether an obstacle is within a certain L1 norm distance. A fifth of the conflicts happened next to an obstacle (fig. 3). This suggests that the map's topology influences the frequency and locations of conflicts and should be considered during planning. An additional input of nearby obstacles and their location relative to the conflict could be for example utilised during

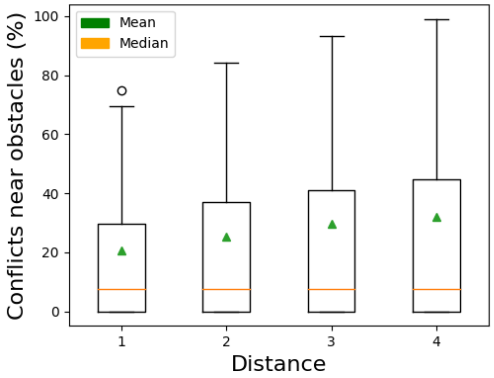

Figure 3: Percentage of conflicts occurring near obstacles.

the conflict resolution.

Highway Definition

Drawing from the patterns seen in the conflict analysis and expanding on the idea of (Cohen, Uras, and Koenig 2015), we propose the use of weighted edges, from here on mentioned as highways, used in the low level search graph to influence the path decisions, in order to minimise the number of initial conflicts. Similarly to (Cohen et al. 2016), machine generated highways are produced. While the previously suggested method, considers each MAPF instance individually, this work exploits knowledge acquired from previous missions to inform future path planning.

Within an environment, with every MAPF problem, all occurred conflicts are documented and analysed. In the case where a vertex conflict is detected at time t the states of the involved agents $s_i[t] = (\alpha_i, v, t)$ and $s_j[t] = (\alpha_j, v, t)$ as well as their states at the time steps immediately before $s_i[t-1],\ s_j[t-1]$ and after $s_i[t+1],\ s_j[t+1]$ are recorded. In the case of an edge conflict, the states of both agents at time t and $t-1$ are recorded.

The accumulated conflicts are analysed to distinguish problematic areas, where conflicts are frequent. A variable threshold can be used to determine whether a highway is defined. Initially, the threshold is set to 1, the threshold can however change according to the density and recurrence of conflicts, as well as the amount of data available.

A highway is defined as a weight on the edge connecting two nodes. The direction of the defined highway is determined by the directions the conflicting agents attempted to follow. For every vertex, 8 uni-directional vectors are defined: inbound $I = \{I_W, I_N, I_E, I_S\}$ and outbound $O = \{O_W, O_N, O_E, O_S\}$ for the four directions (west, north, east, south) connecting adjacent vertices (Jansen and Sturtevant 2008). With every recorded vertex conflict, for each agent, the value of the corresponding inbound (time t) and outbound (time $t+1$) vector is increased by 1. For every edge conflict, the score of the two vectors at time t is increased. The scores of each pair of vectors, e.g. I_W and O_W, are compared. The MAPF edge corresponding to the highest scoring vector is defined as a highway. Tie-breaker possibilities, such as preferred right-hand driving or specified distance from obstacles, are considered.

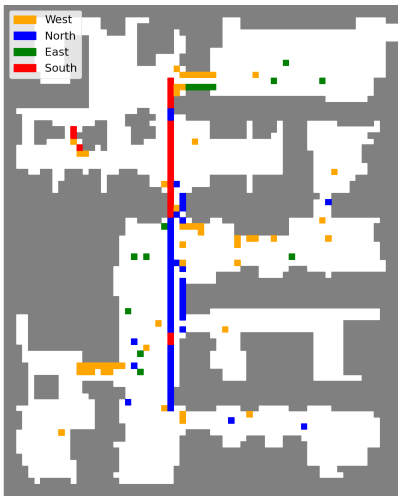

Figure 4: Highways and their directions

Highway Structure

The conflict analysis and highway extraction result to weights informed not only from one single instance but from multiple missions ran on the same environment thus recognising and addressing reoccurring conflicts. The highways form continuous patterns, as well as occasionally contradicting their neighbours, showing the need for post-processing to avoid multiple conflict occurrences at highway edges. Fig. 4 shows the defined highways after analysis of the conflicts seen previously in fig. 1. The long corridor has a continuous north bound direction changed to south bound when reaching the narrow passage. Post-processing should be considered to address such issues. A possible consideration is the inclusion of obstacle produced vectors to weigh in the direction decision and deduce structures such as driving on the right to follow crowd flow. This would also affect the predictability of agent movement, an important issue for mobile robots when navigating in areas occupied by humans.

Modified CBS with highways

Following their definition, the highways are embedded in the low level search of the CBS by updating the cost function. The revised cost expands on the classic A* function $f(n) = g(n) + h(n)$ with an extended heuristic. With the Manhattan distance as base , a highway penalty/reward constant is added. The heuristic is then updated to $h(n) = |t-n|_1 + (\chi \cdot p)$. The variable χ has 3 possible values and depends on the existence of a highway and its direction. If no highway is defined, $\chi = 0$ and therefore the heuristic is not affected. If a highway is defined, against the attempted direction, $\chi = 1$ and a penalty is applied. On the contrary, if it matches the highway direction, $\chi = -1$ and a reward is applied. The penalty/reward value can be adjusted accordingly, for example in the case of a large amount of conflicts. For this work, the χ variable is set equal to the unit action cost of 0.5.

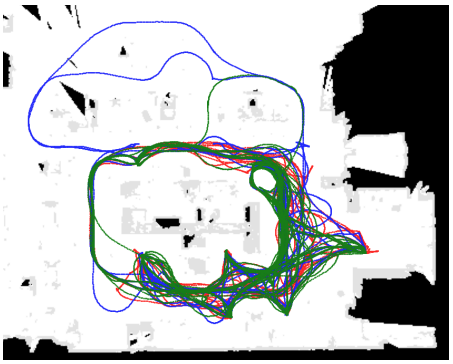

Figure 5: Robot movement during experiment

The highways are not enforced, thus traversing an edge in a direction dissimilar of the highway's, is not forbidden, it is only more costly. This ensures that, even though, the derived paths might not be optimal, they are bounded-suboptimal with a suboptimality factor proportional to the penalty value.

Conflicts in the wild

To explore the nature of conflicts in the real world, experiments were conducted in lab conditions to examine conflict occurrences in real life and investigate whether or how the difficulties transfer to the wild. The experiment was controlled. No outside interference was allowed, e.g. humans in the area. The scenario was synthesised with real cases used as inspiration and a strive for adhering to real conditions. The environment seen in fig. 6 contains a conveyor belt in the centre and spans approximately 20 m $\times$ 15 m. Four pick-up and four drop-off points were assigned around the belt and the surrounding area (indicated with green and magenta points respectively). Three MiR100[1] robots were given randomised missions and left to navigate freely in the environment with their on-board planning and collision avoidance. The paths of the robots can be seen in fig. 5. The conflicts were recorded using cross-correlations between a robot's need to replan, as its original plan was obstructed, and the location of another robot in the vicinity. The second robot's log was then consulted to identify whether both robots had a need of a new plan.

With a run-time of approximately 45 minutes, the robots completed collectively 112 missions and had 71 conflicts. Out of the 71 conflicts (blue), 5 resulted in a crash between two robots (red) (fig. 6). The conflict locations are more dense towards the right side of the map. This is correlated with more target points in close proximity as well as the more turbulent trajectories of the robots in contrast with the left side where the robots seem to follow one route. Analysing the spatial relationships between conflicts, we determined that each detected conflict, has on average 2.73 other conflicts happening within 0.5 m of its location, with a maximum of 9 instances and with the distance increased to 1 m, the numbers reach 6.62 in average and a maximum of 12 conflicts.

[1] www.mobile-industrial-robots.com/solutions/robots/mir100/

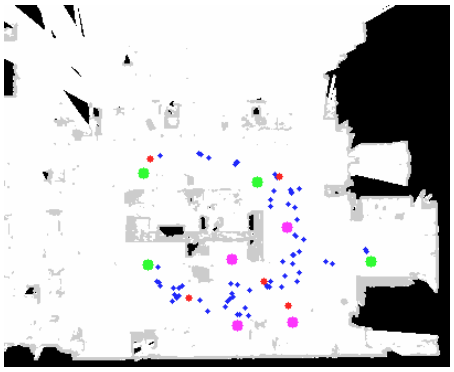

Figure 6: Map of real life experiments and extracted conflicts. Blue: ordinary conflicts; Red: robot crashes; Green: mission pick-up points; Magenta: mission drop-off points

This shows that conflicts arise often in the real world. The precise definition of conflicts in the wild is not outlined as they could appear in different forms e.g. deadlocks between robots, inability to reach their target or strongly affected by the map's topology and left to wander aimlessly. The application of highways could be beneficial in real life scenarios, in addressing the previously mentioned conflicts, with extensions to accommodate the resolution of the used maps and timing uncertainties.

Conclusion and Future Work

In this work, we explore the occurrences of conflicts in the MAPF problem and their exploitation for the production of informed future paths. The quantity and spatial relationships of conflicts, both with each other as well as the map, are analysed and discussed. A method to determine locations and directions of highways is presented along with an updated heuristic for the low-level search of CBS. The significance of conflict patterns is investigated in a controlled experiment of a simulated assembly line where robots come into multiple conflicts and several crashes in certain areas.

In future work, it is our intention to further investigate conflicts and occurring patterns. The highway definitions will be tested against a basic CBS method. Additionally, possible connections of highways with others in the vicinity is considered, as well as the influence the map topology should have on them.

Finally, to bridge the perfect world of simulation with the often precarious real world, we intend to adapt the suggested method for deployment on a fleet of mobile robots. Several adaptations and considerations need to be made for it to be possible. The definitions and nature of conflicts in the wild should be explored as the situations offer much more uncertainty both in timing as well as in localisation. Varying map resolutions and robot sizes also affect the applicability of the method in the real world. The temporal dimension of highways will also be examined to accommodate for periodical obstacle appearances in real world scenarios, such as weekly deliveries in a warehouse environment.

References

Cohen, L.; Uras, T.; and Koenig, S. 2015. Feasibility study: Using highways for bounded-suboptimal multi-agent path finding. In *International Symposium on Combinatorial Search*.

Cohen, L.; Uras, T.; Kumar, T. S.; Xu, H.; Ayanian, N.; and Koenig, S. 2016. Improved Solvers for Bounded-Suboptimal Multi-Agent Path Finding. In *IJCAI*, 3067–3074.

Jansen, R.; and Sturtevant, N. 2008. A new approach to cooperative pathfinding. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems*, 1401–1404.

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

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

Stern, R.; Sturtevant, N. R.; Felner, A.; Koenig, S.; Ma, H.; Walker, T. T.; Li, J.; Atzmon, D.; Cohen, L.; Kumar, T. S.; et al. 2019. Multi-agent pathfinding: Definitions, variants, and benchmarks. In *Twelfth Annual Symposium on Combinatorial Search*.

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