Solving the Non-Crossing MAPF for non point-sized robots

Xiao Peng[1], Olivier Simonin[1], Christine Solnon[1]

CITI, INRIA, INSA Lyon, F-69621 Villeurbanne, France

`{xiao.peng, olivier.simonin, christine.solnon}@insa-lyon.fr`

Keywords : *Constraint Programming (CP), Variable Neighborhood Search (VNS), Assignment Problems, Multi-Agent Path Finding (MAPF).*

1 Introduction

Multi-agent path finding (MAPF) is a very active research topic which has important applications for robotics in industrial contexts (e.g., transport in fulfillment centers, autonomous tug robots). The goal of MAPF is to find a set of paths from starting points to target points while minimizing the *makespan* (length of the longest path). In [2], we introduced an extension of MAPF for tethered robots, where robots are attached with flexible cables to anchor points (which are starting points), and the main difficulty comes from the fact that robots are not able to cross cables. We have shown that this problem is related to an Euclidean bipartite matching problem, and that we can compute feasible solutions that provide upper bounds in polynomial time, by solving the *Linear Sum Assignment Problem (LSAP)*. An approach based on the sequential combination of *Variable Neighborhood Search* (VNS) and *Constraint Programming* (CP) was introduced to solve the problem to optimality.

In [2], we considered point-sized robots, so that two paths may share a same subpath provided that they do not cross. This simplifying assumption on the physical form of the robots is far from reality. In this paper, we extend this work by studying the effect of robots' size on the makespan in a real-world application case. In this case, a robot cannot travel between an obstacle and a cable if the distance between the cable and the obstacle is smaller than its size. As a consequence, we must ensure a minimal safety delay between two robots when traversing a same subpath, as shown in Figure 1.

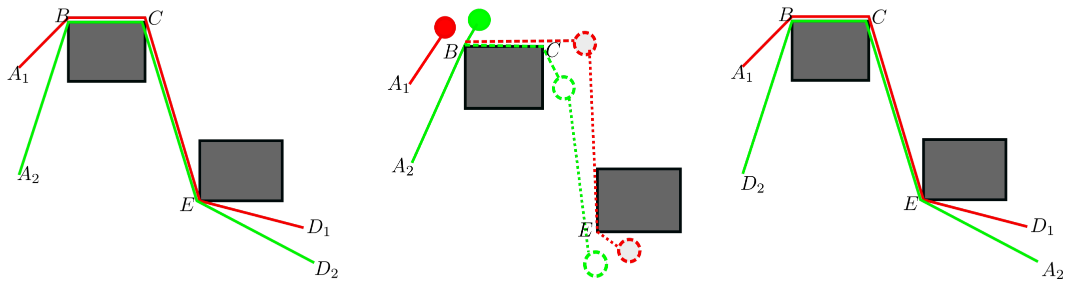

FIG. 1: Left: A_i and D_i with $i = 1, 2$ represent the anchor points and the destinations. In previous work, the makespan is defined as the maximum length between the red path π_1 and the green path π_2. Middle: when robots' physical shape is considered, we impose a time delay dt when two robots share a same subpath. In this example, the green path has higher priority at point B and C, while the red path is prioritized at E. Hence the *makespan* can be noted as $max\{|A_1B| + dt, |A_2B|\} + |BC| + |CE| + max\{|ED_2| + dt, |ED_1|\}$. Right: a deadlock occurs if we exchange the positions of A_2 and D_2: the red robot should wait the green one to pass C first, while the green one cannot pass E until the red robot has passed it.

2 Contribution

We follow most of the notions from previous work and impose a priority relationship between two robots, assuming that the robot with lower priority must wait before the one with higher priority passes first. This new setting can definitely change the definition of *makespan*, as robot's motion should be synchronized with repect to their priority. The solution we are looking for is a set of non-crossing paths that is optimal in terms of both path distance and waiting time.

As a first contribution, we study the topological relationship of two polygonal lines in a 2D Euclidean plane in order to decide whether two paths cross as well as their prority from a geometrical point of view. This work was also investigated in [1, 3] and it is remarked that deadlocks might appear if we adopt a sequential motion mode between robots (as illustrated on the right part of Fig. 1). We consider implementing a topological sorting among these selected paths and posting a *no-circuits* constraint on the constructed graph to avoid such deadlocks.

As a second contribution, we prove that the optimal solution of *LSAP* cannot contain deadlocks and thus provides an upper bound for the problem.

As a third contribution, we introduce a VNS algorithm to improve the LSAP solution and a CP model to improve the VNS solution and prove optimality. The VNS algorithm is similar to the approach introduced in [2], but it enlarges the neighborhood by taking into account non-shortest paths, as the optimal solution could contain non-shortest paths in order to avoid collision. The CP model is a refinement of the CP model of [2] where waiting times are computed for each new solution (assuming infinite waiting times in case of circuits). An experimental analysis is driven by measuring how much the upper bound is reduced and the impact on the runtime cost of the solver.

3 Conclusion

We extended previous work on non-crossing MAPF by considering the impact of robots's size. Our findings show that a new definition of *makespan* should be introduced to cope with the real-world constraints. In these new settings, we follow the principe of combining the local search method with CP model to resolve the problem to optimality on randomly generated instances. The results show that our approach scales well enough to solve realistic instances within a few seconds.

References

[1] Susan Hert and Vladimir Lumelsky. Planar curve routing for tethered-robot motion planning. *International Journal of Computational Geometry & Applications*, 7(03):225–252, 1997.

[2] Xiao Peng, Christine Solnon, and Olivier Simonin. Solving the Non-Crossing MAPF with CP. In *CP 2021 - 27th International Conference on Principles and Practice of Constraint Programming*, LIPIcs: Leibniz International Proceedings in Informatics, pages 1–17, Montpellier (on line), France, October 2021.

[3] Xu Zhang and Quang-Cuong Pham. Planning coordinated motions for tethered planar mobile robots. *Robotics and Autonomous Systems*, 118:189–203, 2019.