Anytime Multi-Agent Path Finding via Large Neighborhood Search: Extended Abstract

Jiaoyang Li (jiaoyanl@usc.edu),[1] Zhe Chen,[2] Daniel Harabor,[2] Peter J. Stuckey,[2] Sven Koenig[1]

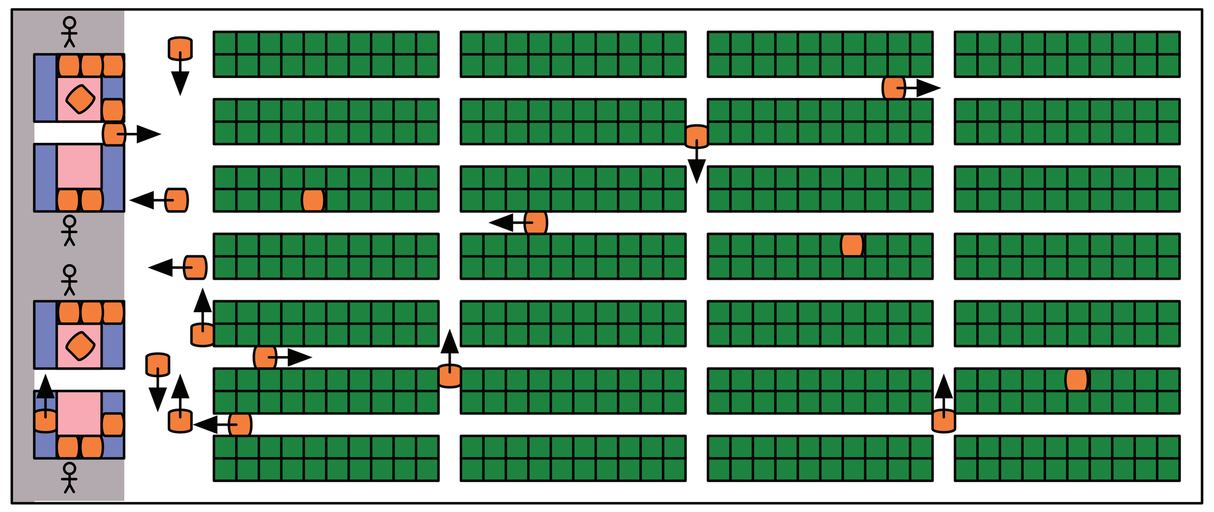
[1]University of Southern California, [2]Monash University

Abstract

Multi-Agent Path Finding (MAPF) is the challenging problem of computing collision-free paths for multiple agents. MAPF algorithms can be categorized on a spectrum. At one end are (bounded-sub)optimal algorithms that can find high-quality solutions for small problems. At the other end are unbounded-suboptimal algorithms that can solve very large practical problems but usually find low-quality solutions. In this paper, we consider a third approach that combines both advantages: anytime algorithms that quickly find an initial solution, including for large problems, and that subsequently improve the solution to near-optimal as time progresses. To improve the solution, we replan subsets of agents using Large Neighborhood Search. Empirically, we compare our algorithm MAPF-LNS to the state-of-the-art anytime MAPF algorithm anytime BCBS and report significant gains in scalability, runtime to the first solution, and speed of improving solutions.

1 Background

Multi-Agent Path Finding (MAPF)



[Picture credits: P. R. Wurman et al. Coordinating Hundreds of Cooperative, Autonomous Vehicles in Warehouses. AI Magazine 29, 1 (2008), 9-20.]

Input:

- A graph $G = (V, E)$.
- A set of agents $\{a_i | i = 1, \cdots, m\}$, each with a start location $s_i \in V$ and a target location $g_i \in V$.

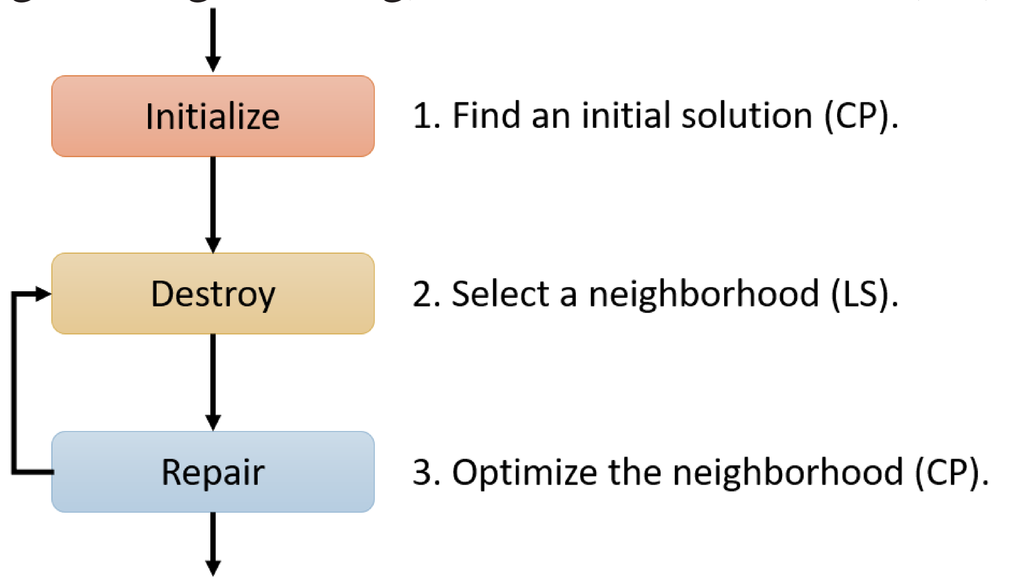
Output:

- A set of collision-free path, one for each agent, that minimizes the sum of the travel times.

2 MAPF-LNS

Large Neighborhood Search (LNS)

LNS [2] combines the power of Constraint Programming (CP) (or Mixed Integer Programming) and Local Search (LS).



Neighborhood: Fix a subset of variables to their values in the best solution found so far.

MAPF-LNS

MAPF-LNS is an anytime MAPF algorithm motivated by LNS.

- Initialize: Find a MAPF solution (by any non-optimal MAPF solver).
- Destroy: Select a subset of agents A_s.
- Repair:
  - Fix the paths for the agents not in A_s and plan collision-free paths for the agents in A_s (by a modified MAPF solver).
  - Replace the old paths if the new ones result in a smaller sum of the travel times.

Adaptive LNS (ALNS)

ALNS [1] makes use of multiple destroy heuristics by recording their relative success in improving solutions and choosing the next neighborhood to explore guided by the most promising heuristic.

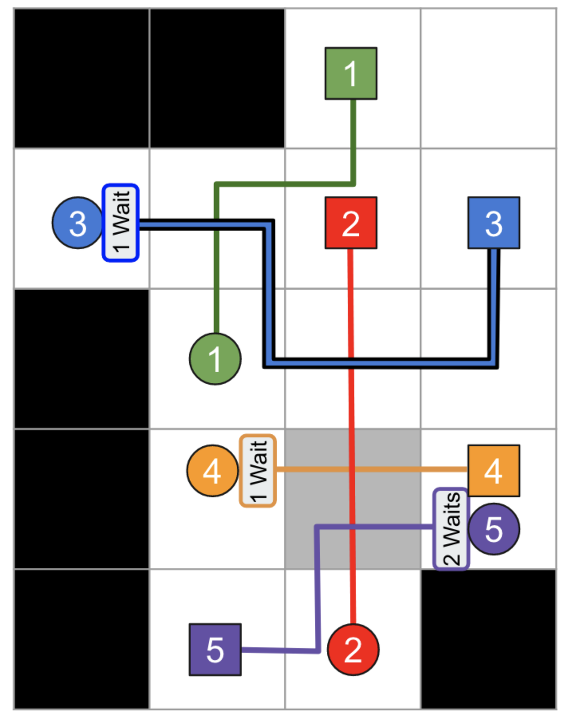
3 Neighborhood Selection

Agent-Based Neighborhood

Select the most delayed agent and the subset of agents that block this agent.

Map-Based Neighborhood

Select the agents that visit the same "intersection" location.

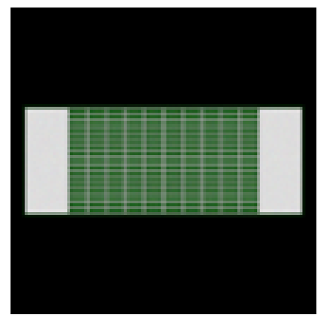


Select a subset of 3 agents from the left figure:

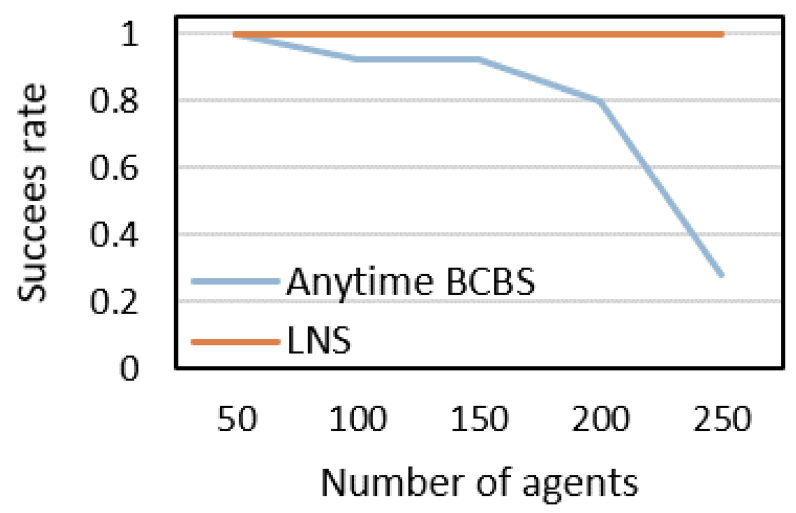
Agent-based method selects $\{1, 2, 3\}$, as agent 3 is delayed the most and blocked by agents 1 and 2.

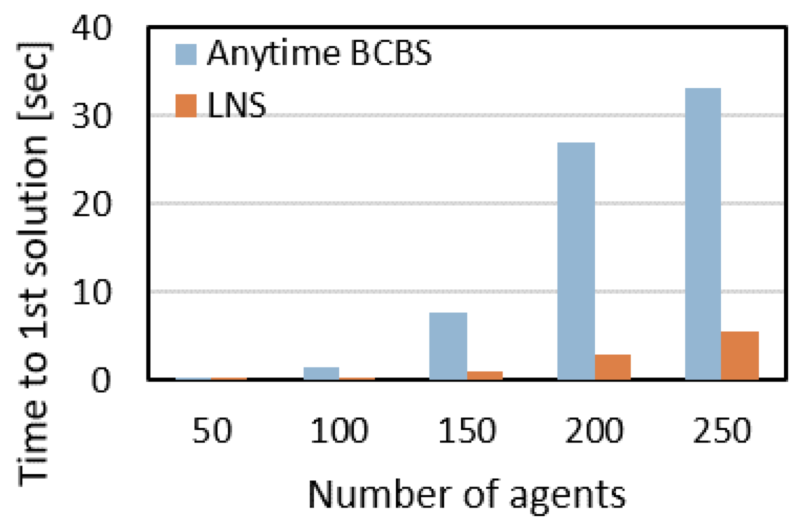
Map-based method selects $\{2, 4, 5\}$, if the grey tile is the selected intersection.

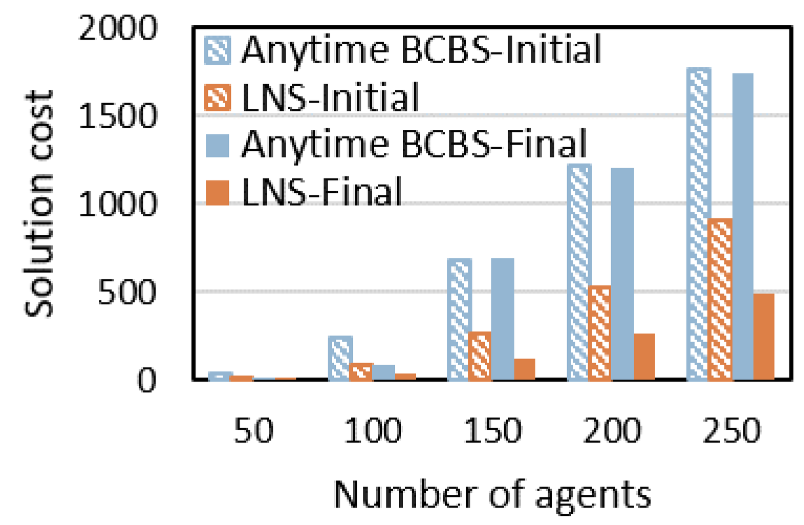
4 Empirical Evaluation



Warehouse



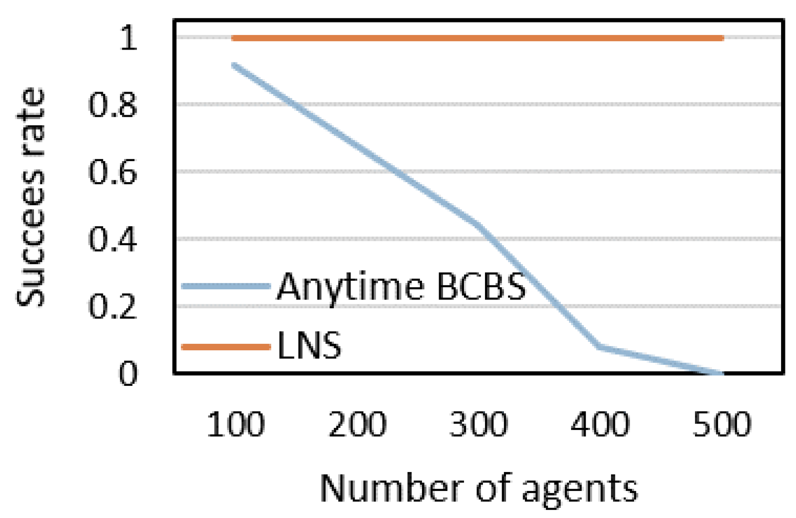


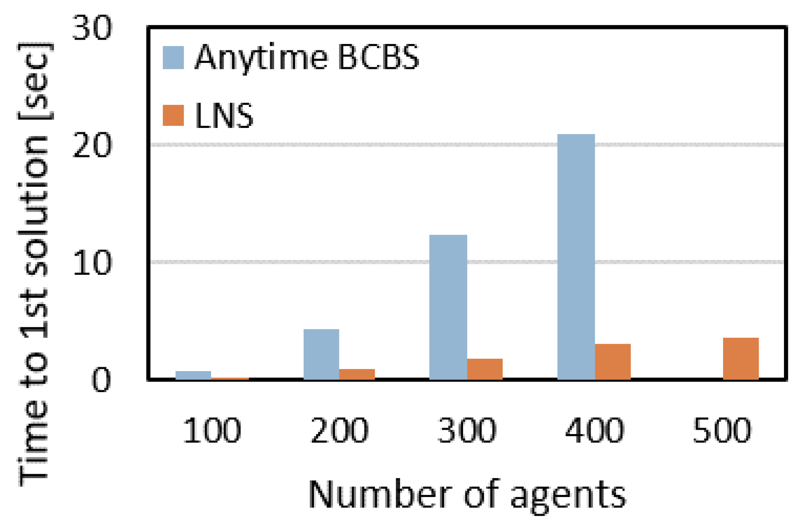


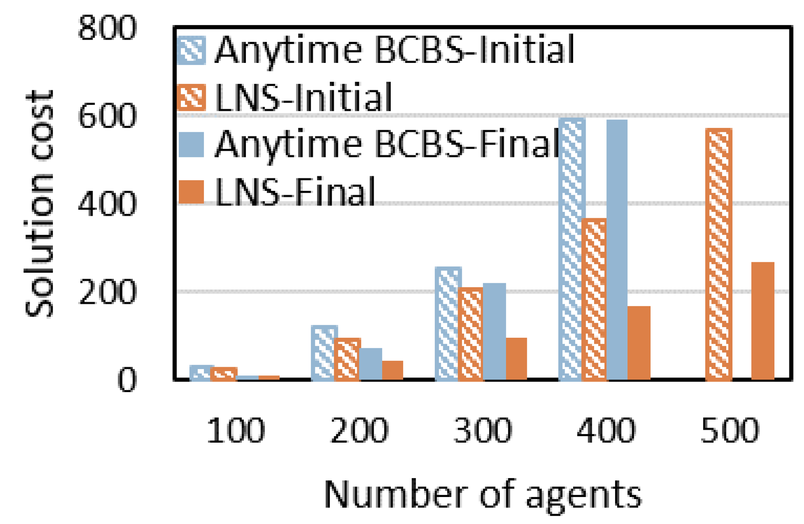
m	Solution cost		Suboptimality
	Initial	Final	
250	13,199	635	≤1.03
300	18,587	1,400	≤1.06
350	25,539	3,979	≤1.14

Results of MAPF-LNS on hard instances

Game







m	Solution cost		Suboptimality
	Initial	Final	
700	20,713	4,473	≤1.04
800	25,885	7,408	≤1.05
900	31,888	12,186	≤1.08

Results of MAPF-LNS on hard instances

Summary: On easy instances, that anytime BCBS can solve, MAPF-LNS has higher success rates, smaller runtimes to the first solution, and better final solutions than anytime BCBS. On hard instances, that anytime BCBS cannot solve, MAPF-LNS can rapidly improve the costly initial solution and quickly converge to a near-optimal solution.

Acknowledgement

Jiaoyang Li performed the research during her visit to Monash University. The research at the University of Southern California was supported by the National Science Foundation (NSF) under grant numbers 1409987, 1724392, 1817189, 1837779, and 1935712 as well as a gift from Amazon. The research at Monash University was supported by the Australian Research Council under Discovery Grant DP190100013 and DP200100025 as well as a gift from Amazon.

[1] Stefan Ropke and David Pisinger. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4):455–472, 2006.

[2] Paul Shaw. Using constraint programming and local search methods to solve vehicle routing problems. In *Proceedings of the International Conference on Principles and Practice of Constraint Programming (CP)*, pages 417–431, 1998.