Learning to Select Promising Initial Solutions for Large Neighborhood Search-Based Multi-Agent Path Finding

Marc Huber

Institute of Logic and Computation, TU Wien, Vienna, Austria

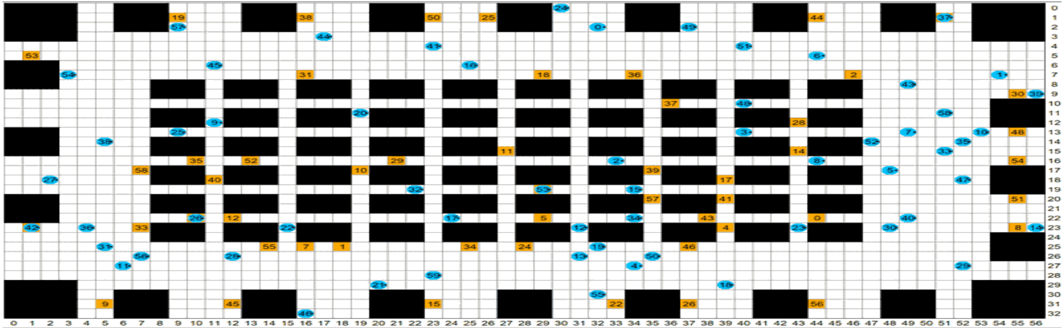
mhuber@ac.tuwien.ac.at

VGSCO Seminar

November 17, 2023

Introduction

What is multi-agent path finding (MAPF)?



Moving robots on a warehouse grid.[1]

Optimization problem with the objective to minimize the sum of task-completion time or the sum of travel times

[1] https://www.leagueofrobotrunners.org

Introduction

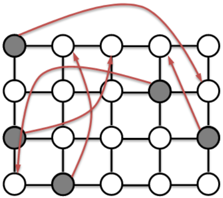
Applications

- Unmanned aerial vehicle traffic management
- Warehouse logistics
- Airport operations
- Game characters in video games

Introduction

Formulation

- Graph
- Set of agents, each with unique start- and goal vertices



Plans

- Sequence of actions
- Length of a plan: time when the agent reaches its goal and does not leave it anymore

Introduction

Task

- Find plans for all agents such that the plans do not collide in time and space

Objectives

- Makespan: number of time steps until all agents reach their goals
- Sum of costs: sum of time steps required by each agent to reach its goal

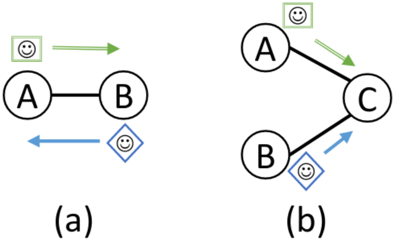
Complexity

- NP-hard to find a makespan or- sum of costs optimal MAPF plan [Surynek, 2010, Yu and LaValle, 2013]

Introduction

Project assumptions [Stern et al., 2019]

- No swapping and vertex conflicts

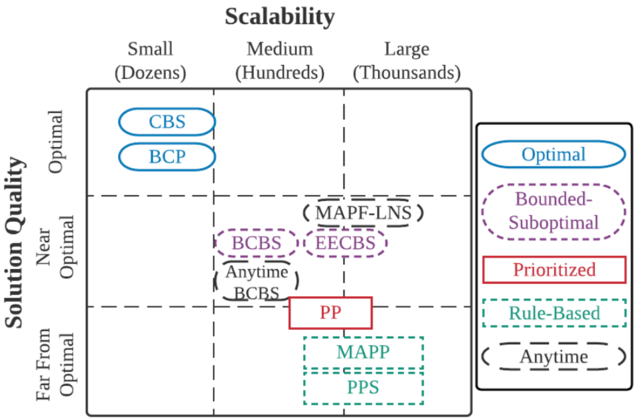


Swapping conflict. Vertex conflict.

- Stay at target
- Objective: minimize sum of costs

Related Work

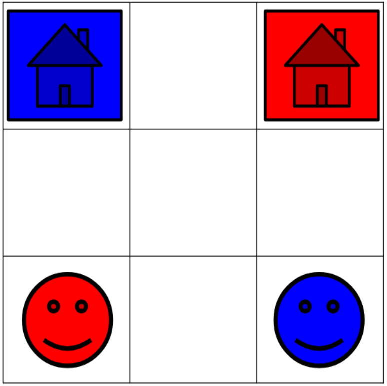
Comparison of scalability versus solution quality tradeoffs of existing algorithms [Li et al., 2021]



Related Work

Prioritized planning (PP) [Silver, 2005]

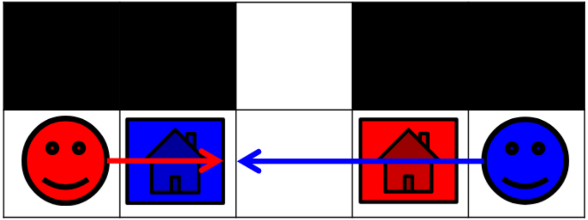
- Plans paths for each agent one after the other
- Avoids collisions with the already planned paths



Related Work

Prioritized planning (PP) [Silver, 2005]

- Challenges: maintaining **soundness**, **completeness**, and **optimality**
  - Complexity: **polynomial** in the grid size and max time
  - Soundness: **Yes!**
  - Complete and Optimal? **No!**



Exemplary illustration of a scenario where PP fails to find a solution.

Related Work

Anytime Multi-Agent Path Finding via Large Neighborhood Search (MAPF-LNS) [Li et al., 2021]

1. Invokes a MAPF solver to **quickly** find a feasible initial solution
2. Proceeds to improve the solution by **repeatedly replanning a subset of agent paths** selected by **randomized destroy heuristics**

Anytime Multi-Agent Path Finding via Machine Learning-Guided Large Neighborhood Search (MAPF-ML-LNS) [Huang et al., 2022]

- Learns a ranking function for a collection of agent sets generated by the destroy heuristics in MAPF-LNS, such that replanning increases the solution quality more

Related Work

MAPF-LNS2: Fast Repairing for Multi-Agent Path Finding via Large Neighborhood Search [Li et al., 2022][2]

1. Calls PP to find an initial solution quickly
2. Plans paths for agents that do not have a plan yet such that the number of collisions with the existing paths are minimized
3. Repeats a repairing procedure using PP until the plan becomes feasible
4. Proceeds with MAPF-LNS

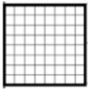
[2] `https://github.com/Jiaoyang-Li/MAPF-LNS2`

Learning to Select Promising Initial Solutions for LNS-Based MAPF

Benchmark set[3]

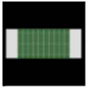
- 25 instances for each map and each number of agents

Map	Map Size	Number of Agents
empty-8-8	8×8	$\{16, 24, 32, 40, 48\}$
empty-32-32	32×32	$\{300, 350, 400, 450, 500\}$
random-32-32-20 (random)	32×32	$\{50, 100, 150, 200, 250\}$
warehouse-10-20-10-2-1 (warehouse)	161×63	$\{150, 200, 250, 300, 350\}$
ost003d	194×194	$\{100, 200, 300, 400, 500\}$
den520d	256×257	$\{500, 600, 700, 800, 900\}$


empty-8-8


empty-32-32


random


warehouse

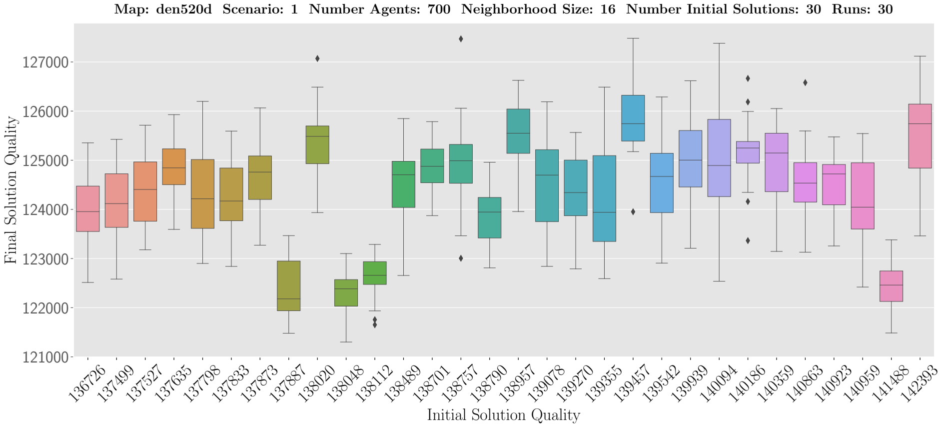

ost003d


den520d

[3] `https://movingai.com/benchmarks/mapf`

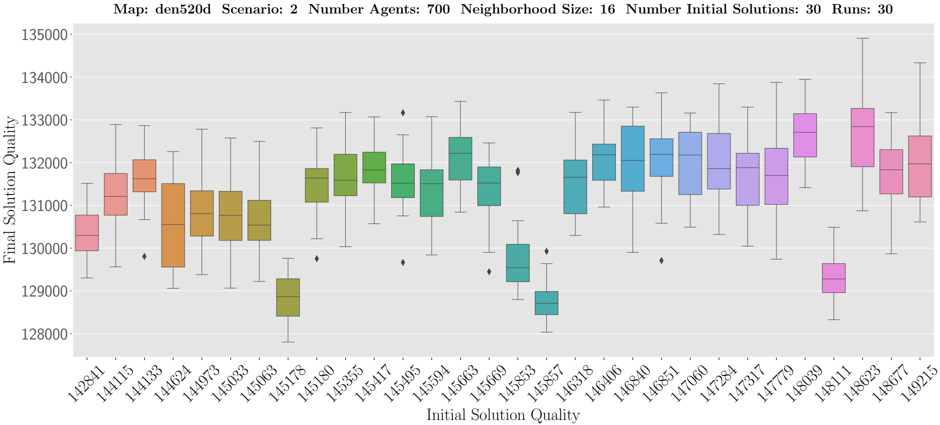
Learning to Select Promising Initial Solutions for LNS-Based MAPF

MAPF-LNS2 observation (time limit 60s): final solution quality depends on the initial solution independently from its solution quality



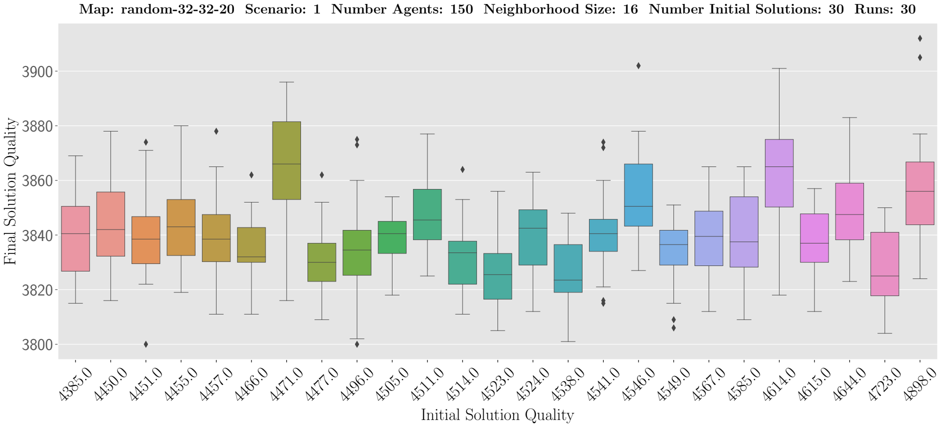
Learning to Select Promising Initial Solutions for LNS-Based MAPF

MAPF-LNS2 observation (time limit 60s): final solution quality depends on the initial solution independently from its solution quality



Learning to Select Promising Initial Solutions for LNS-Based MAPF

MAPF-LNS2 observation (time limit 60s): final solution quality depends on the initial solution independently from its solution quality



Learning to Select Promising Initial Solutions for LNS-Based MAPF

Key idea

- Train an ML model that ranks a set of initial solutions generated by PP
- Use the trained model in MAPF-LNS2 to select the most promising initial solution that leads to the smallest sum of costs after applying the destroy- and repair procedure

Training instance generation

- Given a map M and n agents $\mathcal{A} = \{a_1, \ldots, a_n\}$
  1. Select n start and goal vertices randomly to generate a training instance I_M^n
  2. Repeat 1. for i times to obtain a set of i training instances $\mathcal{I}_M^n$

Learning to Select Promising Initial Solutions for LNS-Based MAPF

Training data generation (time limit 60s)

Features

1. Execute PP with randomized ordering p times on each training instance $I_M^n \in \mathcal{I}_m^n$ to collect p different initial solutions for each training instance $I_M^n \in \mathcal{I}_m^n$
2. Extract from each initial solution 16 agent features for each agent $a_i \in \mathcal{A}$
3. Compute the minimum, maximum, sum, and average over all agent features of $\mathcal{A}$
4. Normalize the set of features for each initial solution to the range $[0, 1]$ using min-max normalization

Labels

- After PP execution, the final solution obtained by the destroy and repairing process is used to rank the p initial solutions for each training instance

Learning to Select Promising Initial Solutions for LNS-Based MAPF

Agent features of agent $a_i \in \mathcal{A}$

Feature Description	Count
Distance between agent a_i's start and goal vertices	1
Row and column numbers of agent a_i's start and goal vertices	4
Degree of agent a_i's goal vertex	1
Delay of agent a_i	1
Ratio between the delay of agent a_i and the distance between a_i's start and goal vertices	1
Minimum, maximum, sum, and average of the heat values of the vertices on agent a_i's path	4
Number of time steps that agent a_i is on a vertex with degree j $(1 \leq j \leq 4)$ before reaching its goal vertex	4

Evaluation

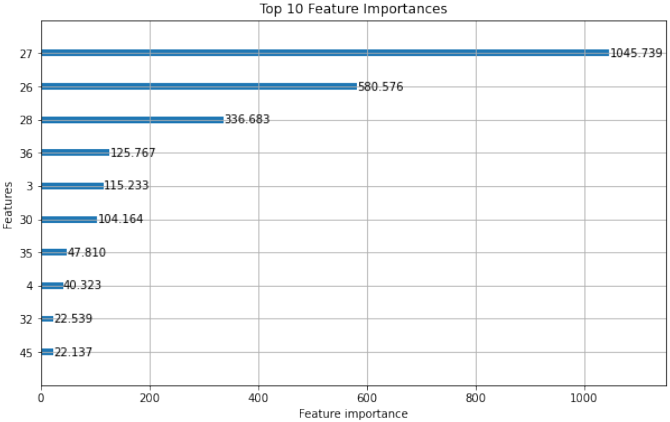
Setup

- Generated 1000 training instances with 50 agents and 30 initial solutions on map *random*
- Used LambdaMART for training – a learning-to-rank algorithm, implemented in the LightGBM framework[4]
- Set a time limit of 60 seconds in the tests
- Tested trained model on map *random* with 50 and 200 agents
- Evaluated approach 1000 times on each of the 25 benchmark instances

[4] `https://lightgbm.readthedocs.io/en/stable`

Evaluation

Feature importance



- Feature 27: average delay Feature 26: sum of delay
- Feature 28: ratio between the delay and the distance between start and goal vertices

Evaluation

Map random with 50 agents

MAPF-LNS2 results averaged over 25 instances.

	Runtime	Solution cost	Initial solution cost	Sum of distance	Runtime of initial solution	Runtime of initial solutions gen.	Runtime of best initial sol. prediction	Area under curve
Mean of Instances Means	60.000424	1136.702480	1232.382360	1112.48	0.004554	0.000000	0.000000	1465.547897
Mean of Instances Medians	60.000376	1136.560000	1230.540000	1112.48	0.004094	0.000000	0.000000	1460.193880
Medians of Instances Means	60.000412	1128.149000	1224.240000	1099.00	0.003655	0.000000	0.000000	1273.298360
Mean of Instances SDs	0.000318	0.368144	30.586329	0.00	0.001745	0.000000	0.000000	25.136322

MAPF-ML-LNS2 results averaged over 25 instances.

	Runtime	Solution cost	Initial solution cost	Sum of distance	Runtime of initial solution	Runtime of initial solutions gen.	Runtime of best initial sol. prediction	Area under curve
Mean of Instances Means	60.000458	1136.687680	1229.629760	1112.48	0.005115	0.135314	0.028795	1464.903052
Mean of Instances Medians	60.000412	1136.560000	1229.840000	1112.48	0.003812	0.137877	0.011753	1457.538420
Medians of Instances Means	60.000432	1128.093000	1240.000000	1099.00	0.004799	0.113115	0.029514	1281.361670
Mean of Instances SDs	0.000346	0.363907	5.395529	0.00	0.006240	0.035441	0.036013	24.859422

Solution cost mean: statistical test

Shapiro-Wilk test p-value: 0.0

Wilcoxon signed-rank test p-value: 0.00021633718179290347

Evaluation

Map random with 200 agents

MAPF-LNS2 results averaged over 25 instances.

	Runtime	Solution cost	Initial solution cost	Sum of distance	Runtime of initial solution	Runtime of initial solutions gen.	Runtime of best initial sol. prediction	Area under curve
Mean of Instances Means	60.002219	5289.469120	6407.723960	4444.40	0.281363	0.000000	0.000000	56767.171536
Mean of Instances Medians	60.001848	5287.580000	6400.780000	4444.40	0.269738	0.000000	0.000000	56641.414000
Medians of Instances Means	60.002306	5260.244000	6397.289000	4445.00	0.276701	0.000000	0.000000	57158.726000
Mean of Instances SDs	0.001712	48.162960	143.263315	0.00	0.110180	0.000000	0.000000	3083.067946

MAPF-ML-LNS2 results averaged over 25 instances.

	Runtime	Solution cost	Initial solution cost	Sum of distance	Runtime of initial solution	Runtime of initial solutions gen.	Runtime of best initial sol. prediction	Area under curve
Mean of Instances Means	60.002303	5286.731320	6369.296080	4444.40	0.259941	8.425453	0.052537	56728.993300
Mean of Instances Medians	60.001892	5285.480000	6366.120000	4444.40	0.273025	8.846618	0.038068	56730.228000
Medians of Instances Means	60.002400	5260.668000	6394.196000	4445.00	0.259011	8.142390	0.053548	57315.850200
Mean of Instances SDs	0.001793	39.812404	63.321988	0.00	0.073720	1.103487	0.037090	2538.797399

Solution cost mean: statistical test

Shapiro-Wilk test p-value: 0.0

Wilcoxon signed-rank test p-value: 2.2227957933236724e-12

Conclusion and Future Work

- Presented MAPF-ML-LNS2, an approach that utilizes ML to select the most promising initial solution generated by PP in MAPF-LNS2
- Showed that MAPF-ML-LNS2 yields superior results than the original MAPF-LNS2 on map *random*
- Demonstrated that the trained ML model generalizes well

Future work

- Feature engineering
- Learning destroy operator in MAPF-LNS2

References I

Huang, T., Li, J., Koenig, S., and Dilkina, B. (2022).
Anytime multi-agent path finding via machine learning-guided large neighborhood search.
In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 9368–9376.

Li, J., Chen, Z., Harabor, D., Stuckey, P., and Koenig, S. (2022).
Mapf-lns2: fast repairing for multi-agent path finding via large neighborhood search.
In Honavar, V. and Spaan, M., editors, *36th AAAI Conference on Artificial Intelligence (AAAI-22)*, number 9 in 36th AAAI Conference on Artificial Intelligence (AAAI-22), pages 10256–10265. Association for the Advancement of Artificial Intelligence (AAAI).

References II

Li, J., Chen, Z., Harabor, D., Stuckey, P. J., and Koenig, S. (2021).
Anytime multi-agent path finding via large neighborhood search.
In Zhou, Z.-H., editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4127–4135. International Joint Conferences on Artificial Intelligence Organization.

Silver, D. (2005).
Cooperative pathfinding.
In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE), 117–122.*

References III

Stern, R., Sturtevant, N., Felner, A., Koenig, S., Ma, H., Walker, T., Li, J., Atzmon, D., Cohen, L., Satish Kumar, T., Boyarski, E., and Barták, R. (2019).
Multi-agent pathfinding: Definitions, variants, and benchmarks.
In Surynek, P. and Yeoh, W., editors, *Proceedings of the 12th International Symposium on Combinatorial Search, SoCS 2019*, Proceedings of the 12th International Symposium on Combinatorial Search, SoCS 2019, pages 151–158. AAAI press.

Surynek, P. (2010).
An optimization variant of multi-robot path planning is intractable.
AAAI'10, page 1261–1263. AAAI Press.

Yu, J. and LaValle, S. M. (2013).
Structure and intractability of optimal multi-robot path planning on graphs.
In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence*, AAAI'13, page 1443–1449. AAAI Press.