Research Plan for Multi-Agent Path Finding with Matching using A* with OD and ID

Ivar de Bruin

April 21, 2021

Title:	Multi-Agent Path Finding with Matching using A* with OD and ID
Author:	Ivar de Bruin
Responsible Professor:	Mathijs de Weerdt
Other Supervisor:	Jesse Mulderij
Peer group members:	Robbin Baauw, Jonathan Dönszelmann, Jaap de Jong, Thom van der Woude

Contents

1 Background of the research

1.1 TUSS problem

The Dutch Railways (NS) is tasked with maintaining trains during the night. The trains are routed to shunting yards where they can be cleaned and receive maintenance. This problem was originally defined as the TUSP problem by Freling et al.[1]. However for this research we will be focusing more on the definition given by Mulderij et al. [2] for the Train Unit Servicing and Shunting (TUSS) problem. One of the main questions in this problems is with regards to the capacity of these shunting yards. To try and establish an upper bound to this capacity Mulderij et al propose to create a relaxation of the problem in the form of a Multi-Agent path finding problem (MAPF). For this problem to be a relaxation it should include matching[2].

1.2 Multi-Agent path finding

Multi-Agent path finding or MAPF is an NP-hard problem about getting a group of agents from their starting point to their assigned goal point without collisions. Stern provides a list of terminology and definitions for this problem in [3]. According to these definitions we can start defining our MAPF problem as a classical MAPF problem with n agents as a $\langle G, s, t\rangle$ where:

- $G = \langle V, E\rangle$ is an undirected graph of vertices V and edges E.
- s is a list of n starting vertices such that s_i denotes the starting vertex of agent a_i
- t is a list of n target vertices such that t_i denotes the target vertex of agent a_i

As the focus of this research is on introducing matching and comparing algorithms, G will be simplified to a 4-neighbor grid with walls. This allows for easier implementations while still giving a fair comparison.

A solution of a MAPF problem provides a path π_i for each agent a_i that defines the location of each agent at a moment x as $\pi_i[x]$ and ends at t_i. For this set of paths to be a solution they need to be non-conflicting. For this research we will be using the following conflicts as defined by Stern[3]:

- Edge conflict: An edge conflict occurs when two agents traverse the same edge at the same time.
- Vertex conflict: A vertex conflict occurs when two agents are at the same vertex at the same time.

With these conflicts defined we can clearly describe a solution, to also define an optimal solution we need a cost function. For this we will use the sum of individual paths.

As the final step according to Stern we need to define the behaviour at the target for which we will use stay at target which means the agents will remain at their target and as such can still cause vertex conflicts.

1.3 Matching

For a MAPF problem to be a relaxation of TUSS it needs to include matching. This means we have to redefine our problem tuple for a problem with n agents to: $\langle G, s, t, cs, ct\rangle$ where:

- G is a 4-neighbor grid.
- s is a list of n starting vertices such that s_i denotes the starting vertex of agent a_i
- t is a list of n target vertices
- cs is a list of n colors such that cs_i is the color of s_i (and as such of agent a_i)
- ct is a list of n colors such that ct_i is the color of t_i

In this definition agents can go to any target vertex that has the same color. Formally this means that an agent a_i has to end at a target vertex t_j such that $cs_i = ct_j$

1.4 Algorithms

To solve a basic MAPF instance a few algorithms are commonly used: A* with operator decomposition and independence detection (A*+ID+OD)[4], Branch and cut and price[5], Conflict based search[6], Increasing cost tree search[7] and M*[8].
This research will focus on adapting A*+ID+OD to include matching while each of my peers work on one of the other algorithms. This will allow us to determine which of these algorithms is more suited for matching.

1.5 A* with operator decomposition and independence detection

A*+ID+OD is defined by Standley in [4]. The idea of the algorithm is to improve the amount of nodes searched by A* in to major ways.

Operator decomposition When A* is used for MAPF it has a very large search space because when we expand a node a new node is created for every possible next move. With multiple agents this means we need a new node for every combination of moves of all the agents. This cause exponential node creation. Operator compositions reduces this branching factor by instead advancing one agents move at a time[4]. This is done by creating two types of states: intermediary states and standard states. In standard states all agents have yet to move for that timestep, while in intermediary states some agents have already moved while other have not. This creation of intermediary nodes reduces the branching factor from exponential to linear while only increasing the depth of the search tree in a linear manner. As such operator decomposition reduces the amount of nodes by a significant amount.

Independence detection The second technique used to reduce the size of the search tree is independence detection. The idea here is to group agents on their own and then for each group create a non-conflicting solution. When conflicts arise that cannot be fixed easily the two conflicting groups are combined and this groups solution is recomputed. Depending on the problem, independence detection can cause incredible speedups as no time is spent on preventing conflicts that will never occur.

2 Research Question

The main question that will be answered in this paper is:

> How can the MAPF algorithm A* with ID and OD be used to solve a relaxation of the TUSS problem when it is expanded with matching.

We can then look at the following sub-questions (which may change during the research project):

- Which matching algorithm performs best when combined with A* with OD and ID?
- Are these combined algorithms still optimal?
- How does this combined algorithm perform compared to other MAPF algorithms expanded with matching?
- Under which conditions should this algorithm be used?
- Under which conditions should this algorithm not be used?

3 Method

To complete this research the following things need to be accomplished:

- Implement A* with ID and OD to be extended with matching
- Extend this basic version with a matching method to solve MAPF-M
- Use created benchmarks to analyse the performance of the algorithm compared to other versions and other algorithms
- Use this analysis to improve performance and enhancements of the algorithm

These benchmarks will be run on a website maintained by one of my peers (Jonathan Dönszelmann). This will mean all algorithms are tested under the same conditions allowing for a fair comparison. All algorithms will also be developed in Python to further aid the fairness of this comparison.

4 Planning of the research project

Here I will explain what I plan to spent each week on, I will also include a table of the deadlines and other things from the university as a clear overview of deadlines.

Week 1: Orientation

The first week will be spent on orientation. What is the current state of the field? How does A* ID OD really work. How can I make it? etc.

During this week I will also orientate myself on the different deadlines and lectures we have. For example how to prepare for each one.

For this orientation I will also be reading quite a few papers starting with Standley's paper on A* with ID and OD [4], Stern et al. paper on Multi-Agent pathfinding[3] and Mulderij et al. paper on the TUSS problem[2]. As well as any papers that follow from that that seem relevant.

This orientation should then finally allow me to make the final version of this research plan at the end of the week.

Week 2: Developing a basic version of A*+ID+OD

This week will be spent on creating a basic MAPF solver in A*+ID+OD. This will make me more familiar with the algorithm as well as give me something to extend. This week there is also the research plan presentation that needs to be prepared.

Week 3: Research matching

This week I will finish the algorithm if I did not finish it last week. After creating a MAPF solver I will need to research how to introduce matching to it. For this I will look at some papers such as [9] but as matching with MAPF is not a research area with a lot of publications quite a bit of this research will be done through experimentation. Once I have an idea of what to do I will start extending the basic algorithm with an initial matching algorithm that can then be iteratively tested, compared and improved.

Week 4: Matching and preparing midterm

This week I will continue with the addition of matching to the algorithm. I will also start preparing for the midterm presentation on Wednesday of week 5

Week 5: Midterm presentation

This week is the midterm presentation. Besides this I will start on the paper as well as perhaps start comparing the benchmarks with my peers.

Week 6: Improve performance

This week will be about improving performance and running further comparisions with my peers. I will also continue using the lectures from university to continue working on the first draft of my paper.

Week 7: Start final rounds of improvement and finish draft v1

This week I will figure out what can still be improved in the algorithm as well as finish the draft of my paper.

Week 8: Draft reviews

This week I will need to review others papers as well as improve my own. I will also be running most of the final experiments so version 2 of the paper draft can contain this data.

Week 9: Draft version 2

This week draft version 2 needs to be finished. This week I will also finish the work on the code and the experiments.

Week 10: Work on final paper and poster

This week the paper should be finished and the poster mostly finished.

Week 11: Presentation

This week the poster will be handed in and the final presentation be prepared and given.

Planning overview

Legend	Deadline	Meeting	Lecture			
	Monday	**Tuesday**	**Wednesday**	**Thursday**	**Friday**	**End of week**
Q4 W1 (19-4)	First week plan; Responsible methods lecture	Information literacy		Supervisor meeting		Research plan
Q4 W2 (26-4)				Research plan presentation	Responsible research lecture	
Q4 W3 (3-5)				Supervisor meeting	Session ACS	
Q4 W4 (10-5)	Session responsible research			Supervisor meeting		
Q4 W5 (17-5)	Session ACS		Midterm presentation	Supervisor meeting, midterm feedback		
Q4 W6 (24-5)				Supervisor meeting, reflection discussion	Session ACS	
Q4 W7 (31-5)				Supervisor meeting		
Q4 W8 (7-6)	Paper draft v1			Supervisor meeting; Peer-review draft v1		
Q4 W9 (14-6)			Paper draft v2	Supervisor meeting		
Q4 W10 (21-6)				Supervisor meeting		Submit final paper
Q4 W11 (28-6)	Session ACS	Submit final poster		Poster presentation		

Figure 1: Draft planning overview

References

[1] L. G. K. D. H. Richard Freling Ramon M. Lentink, "Shunting of passenger train units in a railway station," *Transportation Science*, vol. 39, no. 2, pp. 261–272, 2005, ISSN: 1084-6654. DOI: 10.1287/trsc.1030.0076. [Online]. Available: https://doi.org/10.1287/trsc.1030.0076.

[2] J. Mulderij, B. Huisman, D. TÃ¶nissen, K. van der Linden, and M. de Weerdt, *Train unit shunting and servicing: A real-life application of multi-agent path finding*, 2020. arXiv: 2006.10422 [cs.MA].

[3] R. Stern, N. Sturtevant, A. Felner, S. Koenig, H. Ma, T. Walker, J. Li, D. Atzmon, L. Cohen, T. K. S. Kumar, E. Boyarski, and R. Bartak, *Multi-agent pathfinding: Definitions, variants, and benchmarks*, 2019. arXiv: 1906.08291 [cs.AI].

[4] T. Standley, "Finding optimal solutions to cooperative pathfinding problems," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 24, 2010.

[5] E. Lam, P. Le Bodic, D. D. Harabor, and P. J. Stuckey, "Branch-and-cut-and-price for multi-agent pathfinding.," in *IJCAI*, 2019, pp. 1289–1296.

[6] G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant, "Conflict-based search for optimal multi-agent pathfinding," *Artificial Intelligence*, vol. 219, pp. 40–66, 2015.

[7] G. Sharon, R. Stern, M. Goldenberg, and A. Felner, "The increasing cost tree search for optimal multi-agent pathfinding," *Artificial Intelligence*, vol. 195, pp. 470–495, 2013.

[8] G. Wagner and H. Choset, "M*: A complete multirobot path planning algorithm with performance bounds," in *2011 IEEE/RSJ international conference on intelligent robots and systems*, IEEE, 2011, pp. 3260–3267.

[9] H. Ma and S. Koenig, *Optimal target assignment and path finding for teams of agents*, 2016. arXiv: 1612.05693 [cs.AI].