Our Thoughts (and Recent Results) on Multi-Agent Path Finding

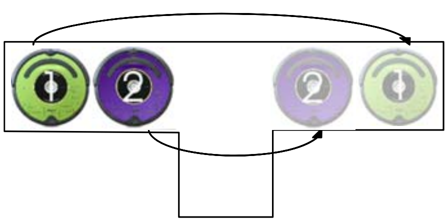
H. Ma, S. Koenig, N. Ayanian, L. Cohen, W. Hoenig,
S. Kumar, T. Uras, H. Xu, C. Tovey and G. Sharon

Department of Computer Science
University of Southern California
Los Angeles, CA, USA
+
Georgia Institute of Technology
University of Texas at Austin

Disclaimer

I will not point out much related work from other research groups since it will likely be presented at this workshop later.

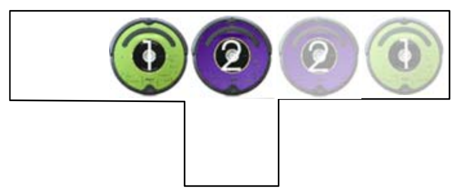
Multi-Agent Path Finding (MAPF)



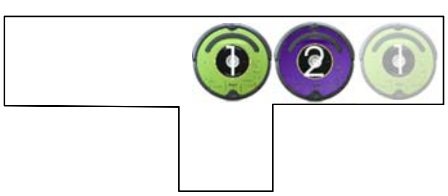
Flow time: the sum of the earliest times when each robot has reached its goal location and remains there

Makespan: the earliest time when all robots have reached their goal locations and remain there

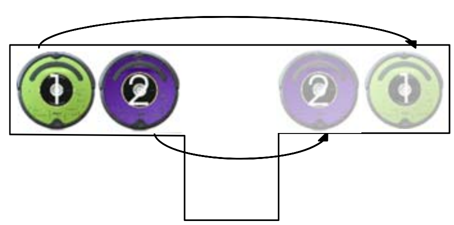
Multi-Agent Path Finding (MAPF)



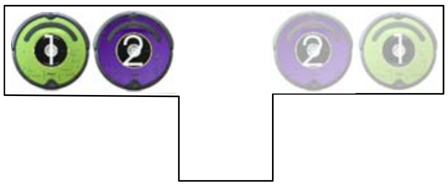
Multi-Agent Path Finding (MAPF)



Multi-Agent Path Finding (MAPF)



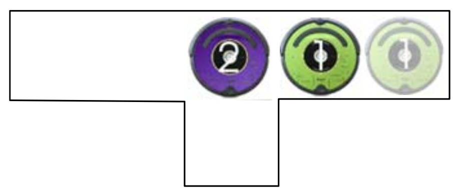
Multi-Agent Path Finding (MAPF)



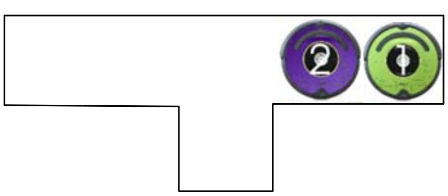
Multi-Agent Path Finding (MAPF)

Multi-Agent Path Finding (MAPF)

Multi-Agent Path Finding (MAPF)



Multi-Agent Path Finding (MAPF)



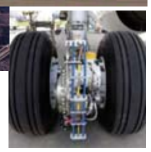
Multi-Agent Path Finding (MAPF)

[www.amazon.com]

[www.ilex-press.com]

Multi-Agent Path Finding (MAPF)

Application: Autonomous tug robots for aircraft joint exploration with Morris (Nasa Ames)



- Reduce pollution
- Reduce energy consumption
- Reduce congestion
- Reduce human workload

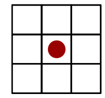
[Google Earth]

[Morris]

Multi-Agent Path Finding (MAPF)

Robot

Agent



- Simplifying assumptions
  - Point robots
  - No kinematic constraints
  - Discretized environment

Multi-Agent Path Finding (MAPF)

Theorem: MAPF is NP-hard to solve optimally for makespan and flow time minimization.

Multi-Agent Path Finding (MAPF)

Lots of good algorithms exist

The MAPF problem has been studied in artificial intelligence, robotics, and theoretical computer science, see (Wagner 2015) for an extensive overview. It can, for example, be solved by reductions to other well-studied problems, including satisfiability (Surynek 2015), integer linear programming (Yu and LaValle 2013), and answer set programming (Erdem et al. 2013). Optimal dedicated MAPF solvers include Independence Detection with Operator Decomposition (Standley and Korf 2011), Enhanced Partial Expansion A* (Goldenberg et al. 2014), Increasing Cost Tree Search (Sharon et al. 2013), Conflict-Based Search (Sharon et al. 2015), M* (Wagner 2015), and their variants. Dedicated suboptimal MAPF solvers include Push and Swap/Rotate (Sajid, Luna, and Bekris 2012; de Wilde, ter Mors, and Witteveen 2013), TASS (Khorshid, Holte, and Sturtevant 2011), BIBOX (Surynek 2009), and their variants. Other approaches, such as Windowed-Hierarchical Cooperative A* (Silver 2005; Sturtevant and Buro 2006), Flow Annotation Replanning (Wang and Botea 2008) and MAPP (Wang and Botea 2011), combine paths of individual agents.

Conflict-Based Search with Highways

Conflict-based search with highways (ECBS+HWY)

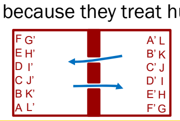
- Conflict-based search (Sharon, Stern, Felner and Sturtevant): Bounded-suboptimal MAPF solver that plans for each agent independently. If two agents A and B collide in cell x at time t, impose the constraint: Agent A and agent B cannot both be in cell x at time t.
- Highways in form of e-graphs (Phillips and Likhachev): Bounded-suboptimal single-agent search so that the resulting path uses edges in a given subgraph (the highway) as much as possible.

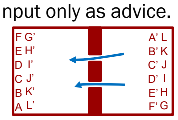
L. Cohen, T. Uras and S. Koenig.
Feasibility Study: Using Highways for Bounded-Suboptimal Multi-Agent Path Finding.
In *Proceedings of the Symposium on Combinatorial Search*, pages 2-8, 2015.

Conflict-Based Search with Highways

Conflict-based search with highways (ECBS+HWY)

- Highways (really: one-way streets) avoid head-to-head collisions, which decreases planning effort.
- Highways provide consistency and thus predictability of robot movement, which might be important for human co-workers.
- Highways do not make MAPF problems unsolvable because they treat human input only as advice.





Conflict-Based Search with Highways

Highways provide quality guarantees.

Theorem: ECBS(w)+HWY(w') is w·w'-suboptimal.

Conflict-Based Search with Highways

3	4	3
2		2
1	0	1

6	5	4
4		3
2	0	2

Conflict-Based Search with Highways

Highways provide quality guarantees.

Theorem: iECBS(w) is w-suboptimal.

L. Cohen, T. Uras, S. Kumar, H. Xu, N. Ayanian and S. Koenig.
Improved Solvers for Bounded-Suboptimal Multi-Agent Path Finding.
In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2016.

Conflict-Based Search with Highways

Rapid random restarts help to solve more multi-agent path finding problems within a given runtime limit.

Here: We randomize the ordering in which the agents plan their paths in the high-level root node.

runs	time limit	38 "easy"	12 "hard"	50 total
1	300 sec	100.00%	0.00%	76.00%
3	100 sec	97.65%	96.87%	97.60%
5	60 sec	98.57%	98.81%	98.70%

L. Cohen, T. Uras, S. Kumar, H. Xu, N. Ayanian and S. Koenig.
Improved Solvers for Bounded-Suboptimal Multi-Agent Path Finding.
In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2016.

Conflict-Based Search with Highways

Highways can be learned rather than designed.

Key idea:

- Plan a path for each agent individually for the given MAPF problem.
- Use the resulting information to heuristically generate highways.

Two approaches:

- Graphical-model based approach
- Heat-map based approach

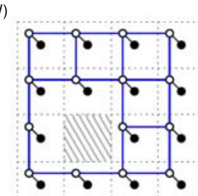
L. Cohen, T. Uras, S. Kumar, H. Xu, N. Ayanian and S. Koenig.
Improved Solvers for Bounded-Suboptimal Multi-Agent Path Finding.
In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2016.

Conflict-Based Search with Highways

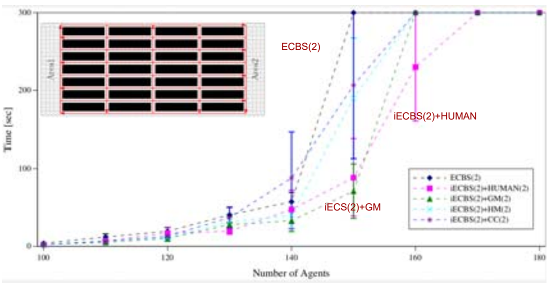
Direction vector of a cell: Average of entry and exit directions of each path for the given cell

Features:

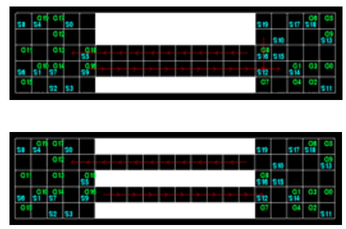
- Collision?
- Direction of direction vector (N, E, S, W)
- Magnitude of direction vector > 0.5?



Conflict-Based Search with Highways

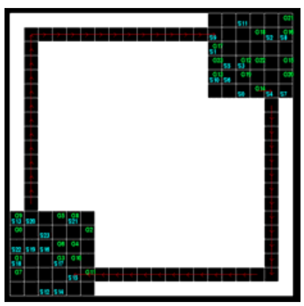


Conflict-Based Search with Highways



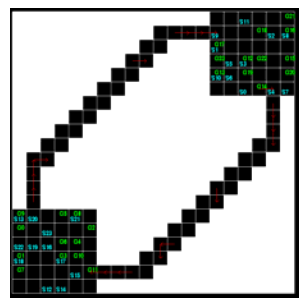
USC Viterbi School of Engineering University of Southern California

Conflict-Based Search with Highways



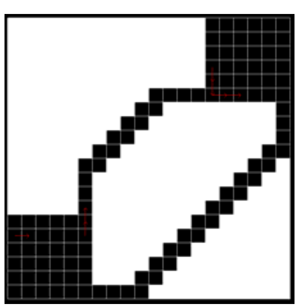
USC Viterbi School of Engineering University of Southern California

Conflict-Based Search with Highways



USC Viterbi School of Engineering University of Southern California

Conflict-Based Search with Highways



USC Viterbi School of Engineering University of Southern California

Multi-Agent Path Finding (MAPF)

Lots of good algorithms exist

The MAPF problem has been studied in artificial intelligence, robotics, and theoretical computer science, see (Wagner 2015) for an extensive overview. It can, for example, be solved by reductions to other well-studied problems, including satisfiability (Surynek 2015), integer linear programming (Yu and LaValle 2013), and answer set programming (Erdem et al. 2013). Optimal dedicated MAPF solvers include Independence Detection with Operator Decomposition (Standley and Korf 2011), Enhanced Partial Expansion A* (Goldenberg et al. 2014), Increasing Cost Tree Search (Sharon et al. 2013), Conflict-Based Search (Sharon et al. 2015), M* (Wagner 2015), and their variants. Dedicated suboptimal MAPF solvers include Push and Swap/Rotate (Sajid, Luna, and Bekris 2012; de Wilde, ter Mors, and Witteveen 2013), TASS (Khorshid, Holte, and Sturtevant 2011), BIBOX (Surynek 2009), and their variants. Other approaches, such as Windowed-Hierarchical Cooperative A* (Silver 2005; Sturtevant and Buro 2006), Flow Annotation Replanning (Wang and Botea 2008) and MAPP (Wang and Botea 2011), combine paths of individual agents.

The "STRIPS problem"

USC Viterbi School of Engineering University of Southern California

Multi-Agent Path Finding (MAPF)

Directions for making MAPF more application-relevant:

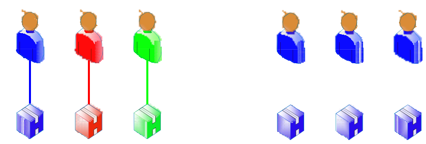
- Extend the functionality of the planning approaches
- Focus on plan execution as well

Idea:

- Consider a mix of anonymous and non-anonymous MAPF

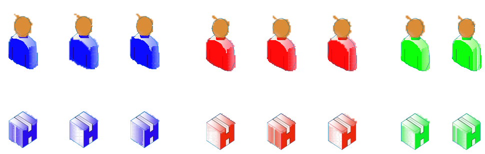
H. Ma and S. Koenig.
Optimal Target Assignment and Path Finding for Teams of Agents.
In *Proceedings of the Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*,2016.

Multi-Agent Path Finding (MAPF)



non-anonymous MAPF	anonymous MAPF
NP-hard	solvable in polynomial time
solved with A* approaches	solved with flow approaches
e.g. conflict-based search or M*	e.g. max-flow algorithm

Target Assignment and Path Finding (TAPF)



k Target Assignment and Path Finding (k-TAPF)
with k groups (here: 3), also called types

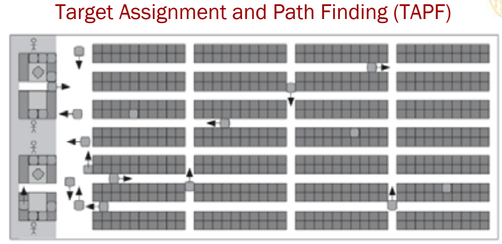
Target Assignment and Path Finding (TAPF)

Theorem: TAPF (for k>1) is NP-hard to solve optimally for makespan and flow time minimization.

Target Assignment and Path Finding (TAPF)

[www.ilex-press.com]

Target Assignment and Path Finding (TAPF)



[Wurman, D'Andrea and Mountz 2008]

Group 0: Robots that move from the packing stations to the storage locations
Group 1: Robots that move from the storage locations to Packing Station 1.
Group 2: Robots that move from the storage locations to Packing Station 2.
Group 3: Robots that move from the storage locations to Packing Station 3.

The Conflict-Based Min-Cost Flow Algorithm (CBM)

How do we solve TAPF (for makespan minimization)?

- We use a version of **conflict-based search** where each group is treated as a meta-agent.
- We plan for each group with a **max-flow approach** that assigns targets to the agents of the group and plans collision-free paths for them.

The details are important. For example, it is important to choose paths that have few collisions with agents from other groups, which is why we actually use a **min-cost max-flow approach**.

The Conflict-Based Min-Cost Flow Algorithm (CBM)

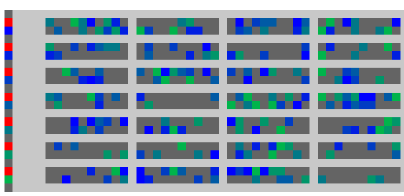
Theorem: CBM is complete and optimal.

	CBM		ILP	
Agents	Time	Success	Time	Success
10	0.34	100%	18.24	100%
20	0.78	100%	62.85	94%
30	1.71	100%	108.75	66%
40	2.95	100%	152.98	14%
50	5.32	100%	161.95	4%

30x30 4-neighbor grids with 10% randomly blocked cells and a 5-minute time limit

The Conflict-Based Min-Cost Flow Algorithm (CBM)

CBM solves 40 out of 50 TAPF instances with 420 agents and a 5-minute time limit each with an average run time of 92 seconds.



Multi-Agent Path Finding (MAPF)

Directions for making MAPF more application relevant:

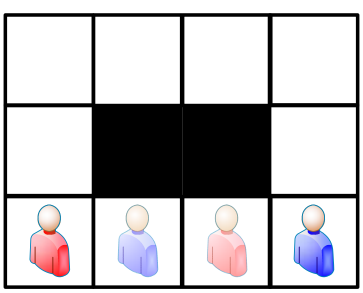
- Extend the functionality of the planning approaches
- Focus on plan execution as well

Idea:

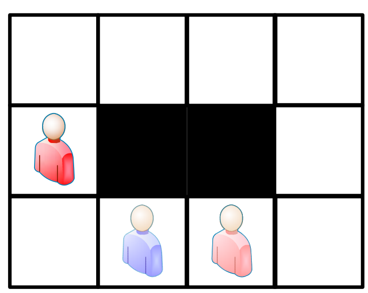
- Consider payload exchanges.

H. Ma, C. Tovey, G. Sharon, S. Kumar and S. Koenig.
Multi-Agent Path Finding with Payload Transfers and the Package-Exchange Robot-Routing Problem.
In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 3166-3173, 2016.

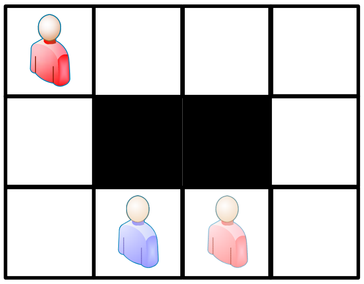
Multi-Agent Path Finding (MAPF)



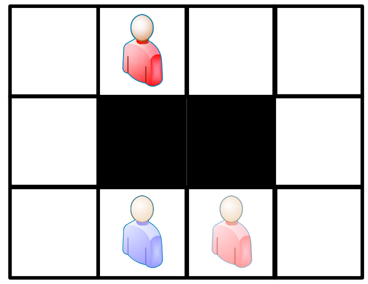
Multi-Agent Path Finding (MAPF)



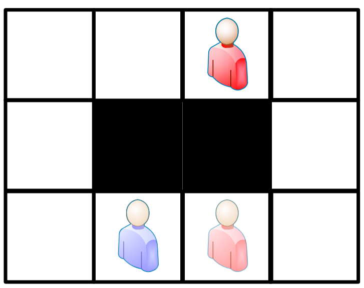
Multi-Agent Path Finding (MAPF)



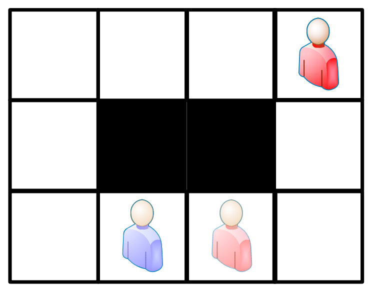
Multi-Agent Path Finding (MAPF)



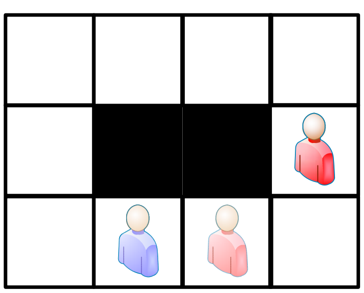
Multi-Agent Path Finding (MAPF)



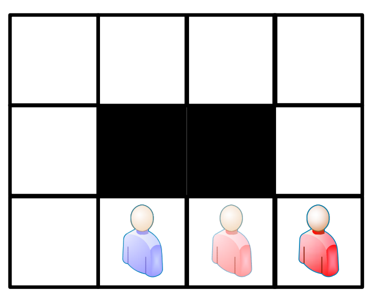
Multi-Agent Path Finding (MAPF)



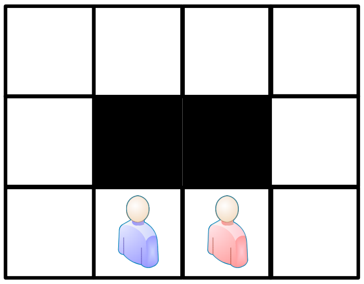
Multi-Agent Path Finding (MAPF)



Multi-Agent Path Finding (MAPF)



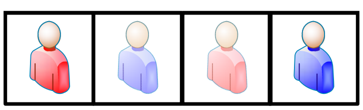
Multi-Agent Path Finding (MAPF)



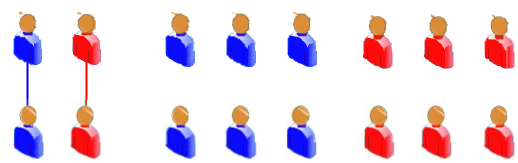
Multi-Agent Path Finding (MAPF)

Theorem: MAPF is NP-hard to solve optimally for makespan and flow time minimization.

Multi-Agent Path Finding (MAPF)



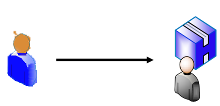
Multi-Agent Path Finding (MAPF)



(non-anonymous) MAPF

k Target Assignment and Path Finding (k-TAPF) with k groups (here: 2)

Package Exchange Robot Routing (PERR)

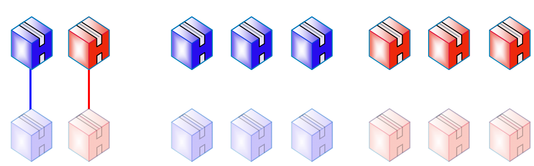


The Package Exchange Robot Routing problem (PERR):

- Each robot carries exactly one package.
- Each package needs to be delivered to a given destination.
- Two robots in adjacent locations can exchange packages.

H. Ma, C. Tovey, G. Sharon, S. Kumar and S. Koenig.
Multi-Agent Path Finding with Payload Transfers and the Package-Exchange Robot-Routing Problem.
In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 3166-3173, 2016.

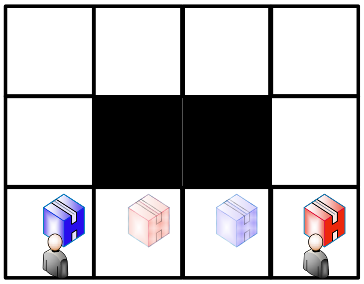
Package Exchange Robot Routing (PERR)



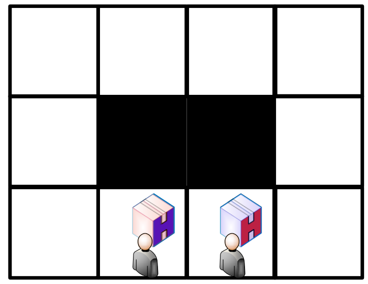
PERR

k-PERR with k-groups (here: 2)

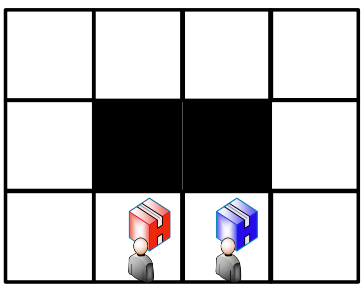
Package Exchange Robot Routing (PERR)



Package Exchange Robot Routing (PERR)



Package Exchange Robot Routing (PERR)



Package Exchange Robot Routing (PERR)

Theorem: All PERR and k-PERR instances are solvable (as long as every agent can reach its destination individually).

Theorem: Plans with polynomial makespans and flow times can be found in polynomial time.

Package Exchange Robot Routing (PERR)

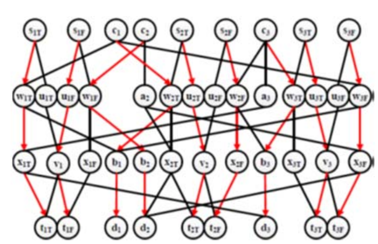
Theorem: PERR and k-PERR (for k>1) are NP-hard to solve optimally for makespan and flow time minimization.

Reductions are from NP-hard versions of SAT, namely ≤3,=3-SAT for PERR and $2/\overline{2}/3$ SAT for k-PERR.

≤3,=3-SAT: Each variable appears in exactly three clauses, uncomplemented at least once and complemented at least once. Each clause contains at most three literals.
$2/\overline{2}/3$ SAT: Each variable appears complemented in one clause of size two, appears uncomplemented in one clause of size two and appears a third time in a clause of size three.

Package Exchange Robot Routing (PERR)

$(X_1 \lor X_2 \lor \overline{X_3}) \land (\overline{X_1} \lor X_2 \lor \overline{X_3}) \land (X_1 \lor \overline{X_2} \lor X_3)$



Package Exchange Robot Routing (PERR)

Theorem: PERR and k-PERR (for k>1) are NP-hard to approximate within any factor less than 4/3 for makespan minimization.

Corollary: MAPF and TAPF for k>1 are NP-hard to approximate within any factor less than 4/3 for makespan minimization.

Package Exchange Robot Routing (PERR)

PERR instances can be solved with versions of conflict-based search and multi-commodity flow approaches.

Studying PERR is a first step toward understanding MAPF problems with more general payload exchanges and transfers.

Multi-Agent Path Finding (MAPF)

Directions for making MAPF more application relevant:

- Extend the functionality of the planning approaches
- Focus on plan execution as well

Idea:

- What if the previous suggestions appear too incremental?

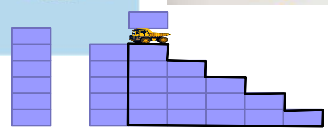
S. Kumar, S. Jung and S. Koenig.
A Tree-Based Algorithm for Construction Robots.
In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, 2014.

T. Cai, D. Zhang, S. Kumar, S. Koenig and N. Ayanian.
Local Search on Trees and a Framework for Automated Construction Using Multiple Identical.
In *Proceedings of the Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2016

Planning for the TERMES Robots

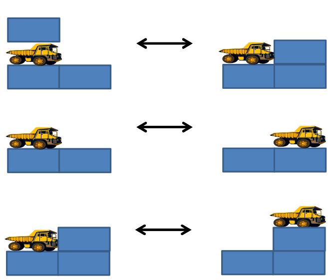
Consider the Harvard TERMES robots

[Harvard University]



Planning for the TERMES Robots

Capabilities of the Harvard TERMES robots



Planning for the TERMES Robots

Tree-based dynamic programming

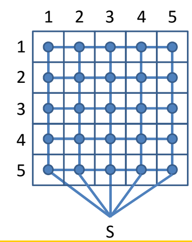
Planning for the TERMES Robots

Tree-based dynamic programming

1	1	1	1	1
1				1
1		3		1
1				1
1	1	1	1	1

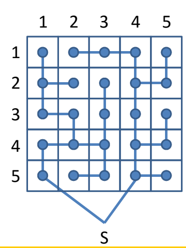
Planning for the TERMES Robots

Tree-based dynamic programming



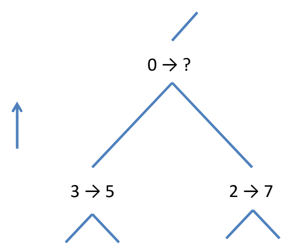
Planning for the TERMES Robots

Tree-based dynamic programming



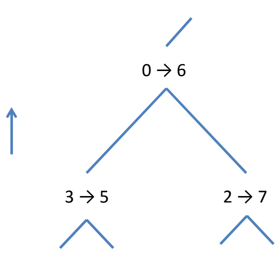
Planning for the TERMES Robots

Tree-based dynamic programming



Planning for the TERMES Robots

Tree-based dynamic programming

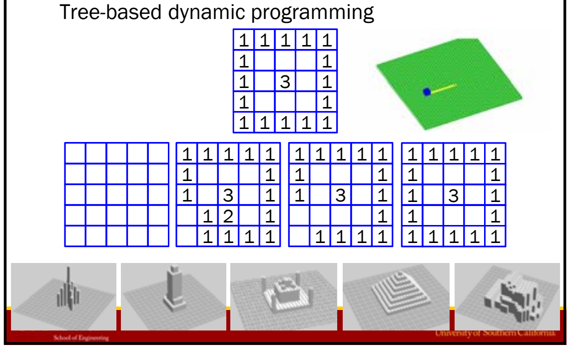


Planning for the TERMES Robots

Tree-based dynamic programming

Planning for the TERMES Robots

Tree-based dynamic programming



Multi-Agent Path Finding (MAPF)

Directions for making MAPF more application relevant:

- Extend the functionality of the planning approaches
- Focus on plan execution as well

W. Hoenig, S. Kumar, L. Cohen, H. Ma, H. Xu, N. Ayanian and S. Koenig.
Multi-Agent Path Finding with Kinematic Constraints.
In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*, 2016.

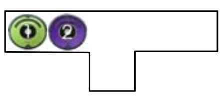
Multi-Agent Path Finding (MAPF)

Planning uses models that are not completely accurate

- Robots are not completely synchronized
- Robots do not move exactly at the nominal speed
- Robots have unmodeled kinematic constraints
- ...

Plan execution will therefore likely deviate from the plan

Replanning whenever plan execution deviates from the plan is intractable since it is NP-hard to find good plans



Idea: Plan Execution with MAPF-POST

MAPF-POST = a novel approach that makes use of a simple temporal network to post-process the output of a multi-agent path finding solver from AI in polynomial time to allow for plan execution on robots

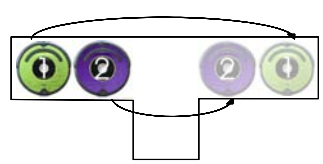
- Takes into account edge lengths
- Takes into account velocity limits (robots and edges)
- Guarantees safety distance
- Avoids replanning in many cases

Executive Summary

Idea: Plan Execution with MAPF-POST

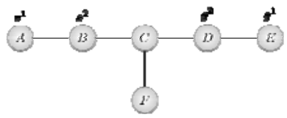
planning layer	any MAPF solver
	MAPF plan ↓ / ↑ re-plan request (future work)
plan-execution layer	MAPF-Post
	direction velocity ↓ / ↑ location
hardware layer	robot

Multi-Agent Path Finding (MAPF)



Step 1: (Discrete) MAPF problem

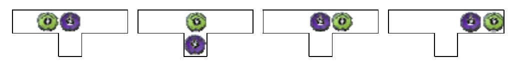
Create a graph representation of the MAPF problem



Step 2: (Discrete) MAPF problem

Solve using a MAPF solver

Agent	$t=1$	$t=2$	$t=3$	$t=4$
1	$A \rightarrow B$	$B \rightarrow C$	$C \rightarrow D$	$D \rightarrow E$
2	$B \rightarrow C$	$C \rightarrow F$	$F \rightarrow C$	$C \rightarrow D$

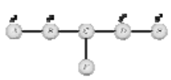


K: number of robots
T: makespan of MAPF plan

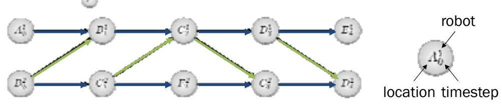
Step 3: Temporal Plan Graph (TPG)

Construct a precedence graph in $O(K^2T^2)$ time

- Vertices: event that a robot arrives at a location
- Edges: temporal precedences between events for MAPF Plan
  - Type 1: order in which the same robot arrives at locations
  - Type 2: order in which any two robots arrive at the same location

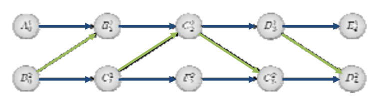


Agent	$t=1$	$t=2$	$t=3$	$t=4$
1	$A \rightarrow B$	$B \rightarrow C$	$C \rightarrow D$	$D \rightarrow E$
2	$B \rightarrow C$	$C \rightarrow F$	$F \rightarrow C$	$C \rightarrow D$



Step 3: Temporal Plan Graph (TPG)

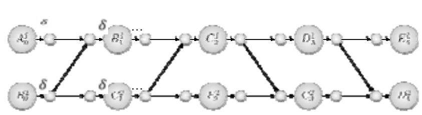
- T: makespan of MAPF plan (here: 4)
- K: number of robots (here: 2)
- TPG has $O(TK)$ vertices and $O(K^2T)$ edges
- TPG can be constructed in $O(K^2T^2)$ time
  - Type 1: straight-forward (skip wait actions)
  - Type 2: identical locations for different routes



Step 4: Augmented Temporal Plan Graph (TPG)

Augment TPG to guarantee a safety distance between robots

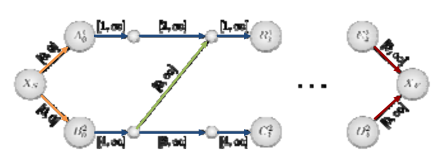
- Add additional vertices ○ (safety markers) to TPG at distance δ from all locations ○
- Change Type 2 edges to be between safety markers



Step 5: Simple Temporal Network (STN)

Add time intervals to encode kinematic constraints (here: a maximum velocity)

- Add two vertices: X_S (start event) and X_F (finish event)
- Annotate edges with upper and lower bounds
  - Type 1: bounds given by velocity limits and edge length $[\frac{edge\ length}{maximum\ velocity}, \infty]$
  - Type 2: $[0, \infty]$, Type 3: $[0, 0]$, Type 4: $[0, \infty]$



Step 6: Plan-Execution Schedule

Minimize flow time and makespan

- Use graph-based optimization
  (find the shortest path on the distance graph)
- Use linear programming

polynomial time

$$\begin{aligned}&\text{Minimize } \textstyle\sum_{j=1}^{K} t(v^j)\\ &\text{such that } t(X_S) = 0\\ &\text{and, for all } e = (v, v') \in \mathcal{E}',\\ &t(v') - t(v) \geq LB(e)\\ &t(v') - t(v) \leq UB(e)\end{aligned}$$

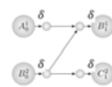
Step 6: Plan-Execution Schedule

Maximize the global minimum velocity limit

- Assume that every robot moves with
  constant velocity [v_{min}, v_{max}] along every Type 1 edge
- Calculate the safety distance as $2\delta v_{min}/v_{max}$

Theorem 1. *There always exists a plan-execution schedule that is consistent with the simple temporal constraints of the STN for a MAPF plan and assigns finite plan-execution times to all vertices in the augmented TPG.*

Theorem 2. *Consider a plan-execution schedule that is consistent with the simple temporal constraints of the STN for a MAPF plan and assigns finite plan-execution times to all vertices in the augmented TPG. Then, point agents always maintain a safety distance of at least $2\delta v_{\min}/v_{\max} > 0$ with respect to graph G (and thus do not collide) if they execute the plan-execution schedule under the uniform velocity model.*



Step 6: Plan-Execution Schedule

Maximize the global minimum velocity limit

- Assume that the robot moves with
  constant velocity [v_{min}, v_{max}] along every Type 1 edge
- Calculate the safety distance as $2\delta v_{min}/v_{max}$
- Maximize v_{min} to maximize the safety distance
- Use linear programming

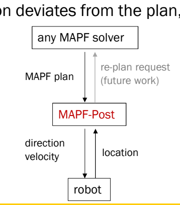
polynomial time

$$\begin{aligned}&\boxed{\text{Minimize } (v^*_{\min})^{-1}}\\ &\text{such that } t(X_S) = 0\\ &\text{and, for all } e = (v, v') \in \mathcal{E}',\\ &t(v') - t(v) \geq LB(e)\\ &t(v') - t(v) \leq UB(e)\\ &\boxed{t(v') - t(v) \leq l(e)(v^*_{\min})^{-1} \text{ if } e \text{ is a Type 1 edge}}\end{aligned}$$

MAPF-POST

Main loop

- Construct a simple temporal network for the given MAPF plan.
- If it is not solvable, replan and repeat loop for the new MAPF plan.
- Execute the given MAPF plan with the calculated arrival times.
- If plan execution deviates from the plan, repeat loop.



Experiments

MAPF solver: ECBS+HWY

MAPF-POST: C++, boost graph library, Gurobi LP solver

PC: i7-4600U 2.1 GHz, 12 GB RAM

Terrain: 5x4 gridworld with 1m^2 cells and δ = 0.4m

Architecture: ROS

Robot controller with state [x,y,Θ] (attempts to meet deadline)

PID controller (corrects for heading error and drift)

Robot simulator: V-REP

Robots: iRobot Create2 robots

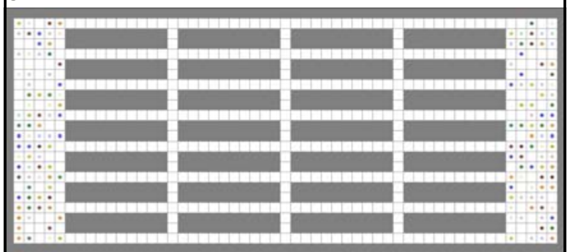
Test environment: VICON MX Motion Capture System

Experiments

8x

Experiments

8x



Conclusion

MAPF-POST = a novel approach that makes use of a simple temporal network to post-process the output of a multi-agent path finding solver from AI in polynomial time to allow for plan execution on robots

- Takes into account edge lengths
- Takes into account velocity limits (robots and edges)
- Guarantees safety distance
- Avoids replanning in many cases

Future work

- User-provided safety distances
- More kinematic constraints (maximum accelerations, …)
- Exploit slack for replanning

Conclusion

Thanks!

Our research was supported by ARL under grant number W911NF-14-D-0005, ONR under grant numbers N00014-14-1-0734 and N00014-9-1-1031, NASA via Stinger Ghaffarian Technologies, and NSF under grant numbers 1409987 and 1319966.