Multi-Objective Path Planning for Safe Navigation

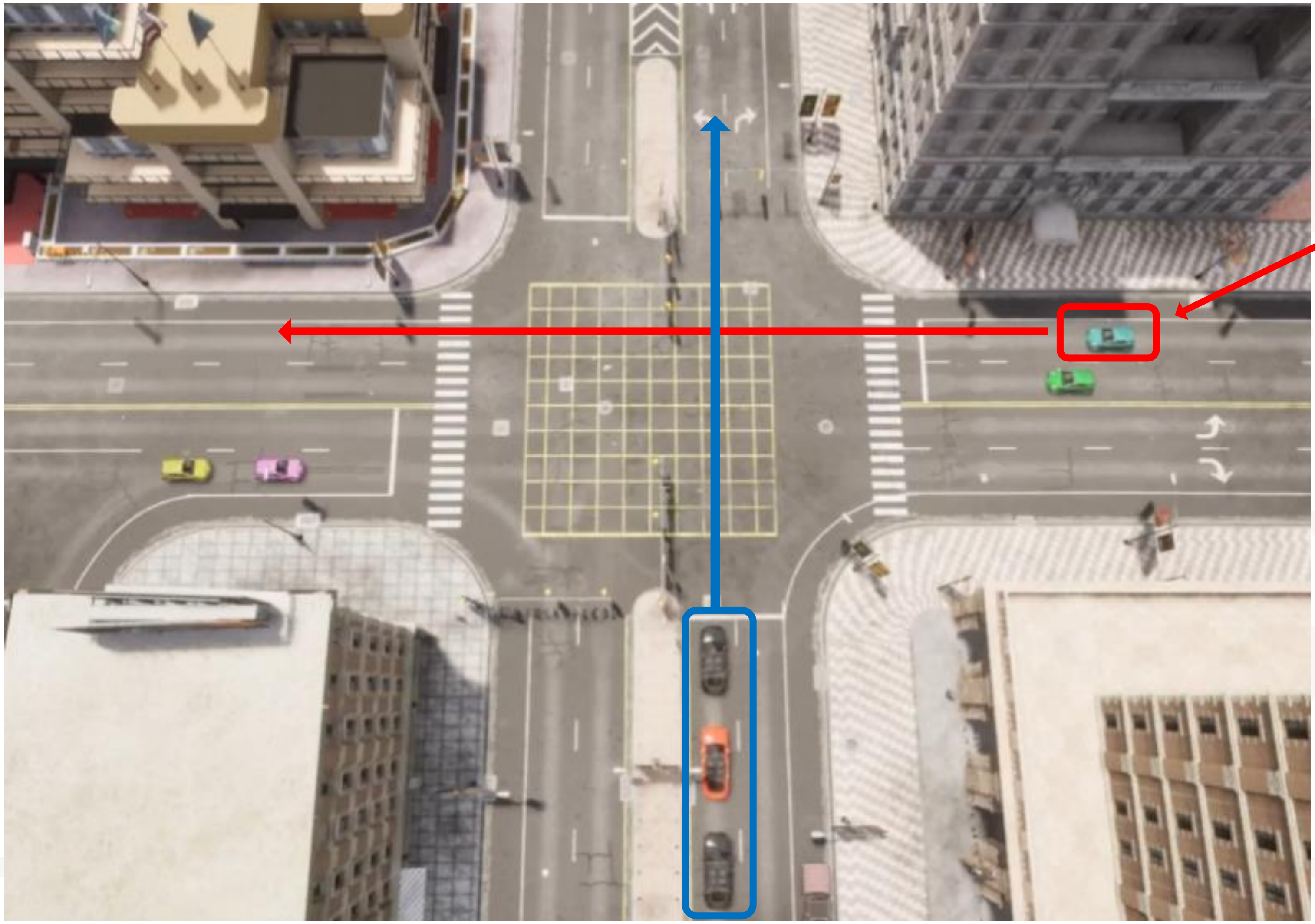
Zhongqiang Ren, Assistant Professor,

UM-SJTU Joint Institute, Shanghai Jiao Tong University

JOINT INSTITUTE
交大密西根学院

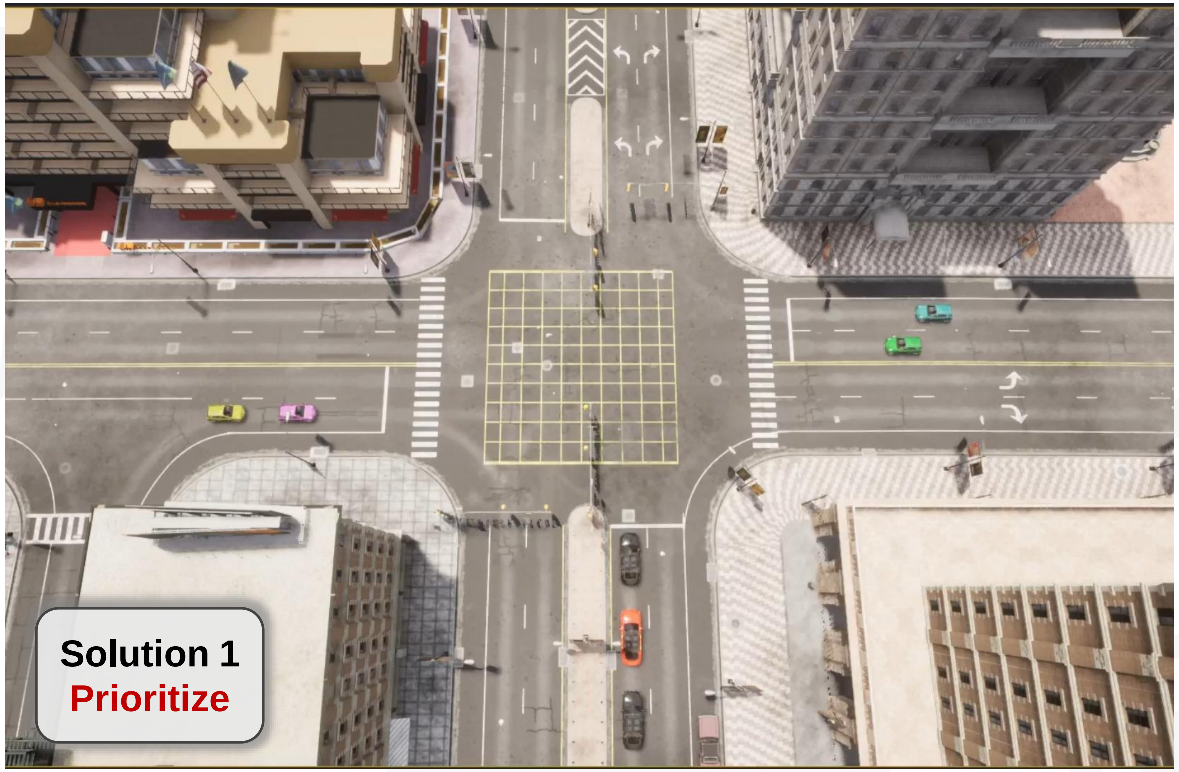
Robotics, Autonomy & Planning Lab

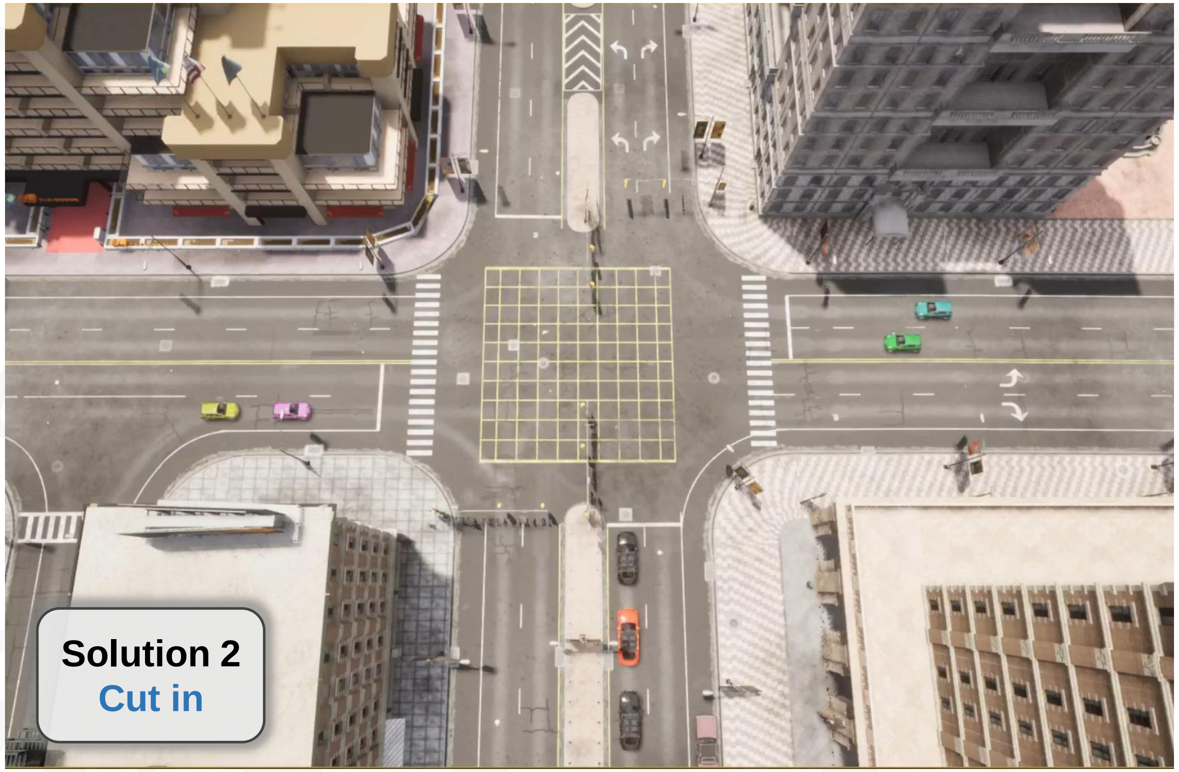
Background



This is your autonomous car.

Background

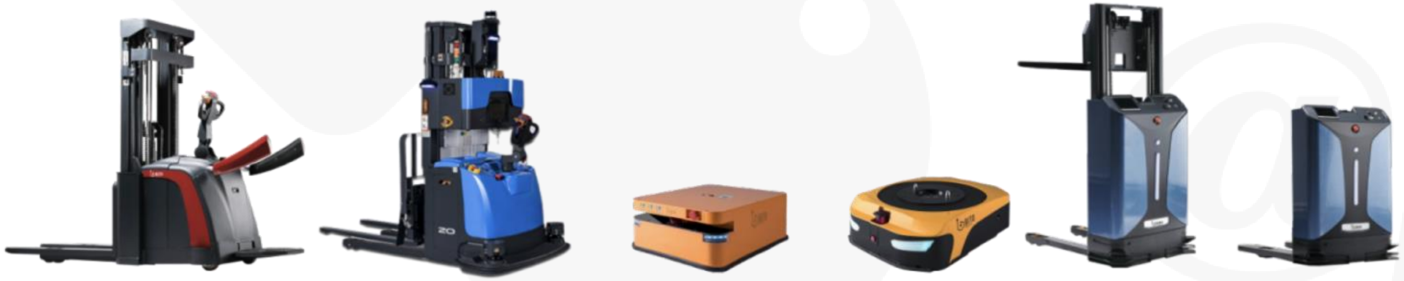


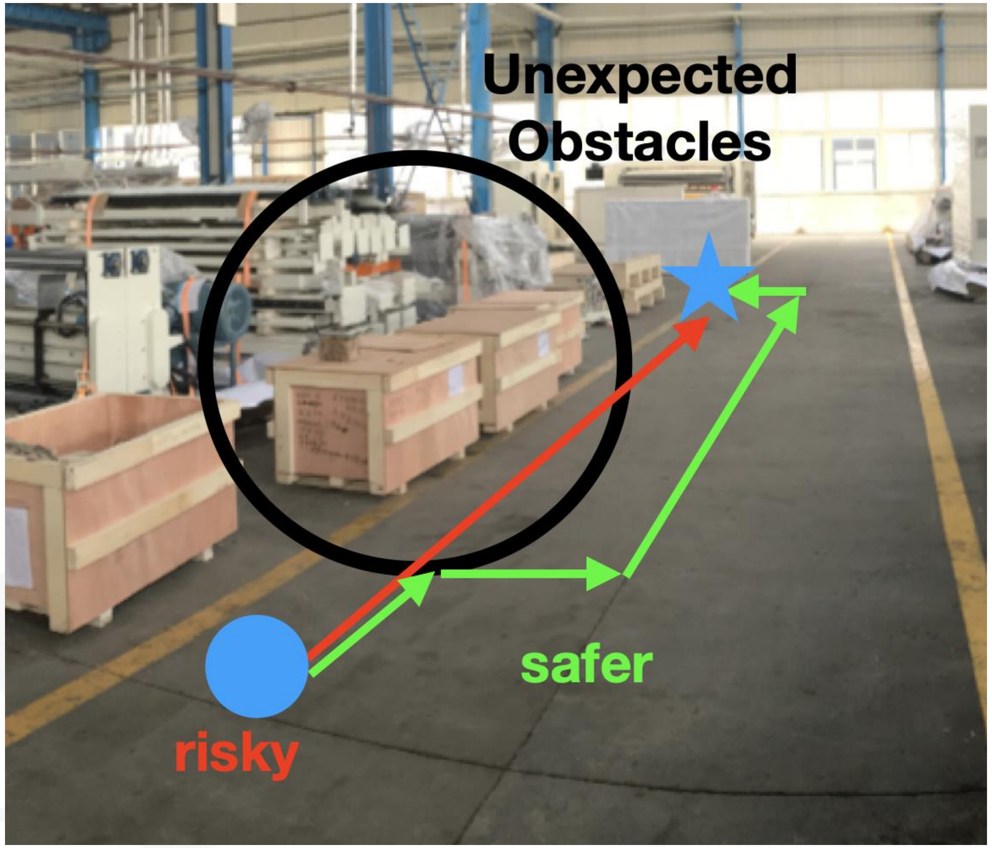


Wait till other cars leave
Safer, slower

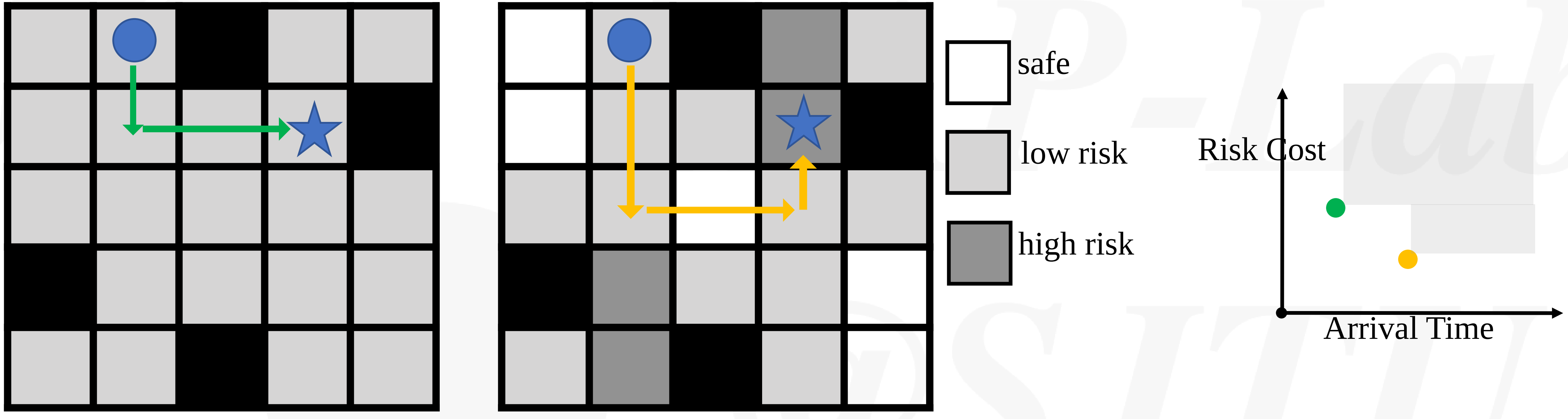
Cut in and pass
Risky, faster

Background





Problem



Uniform traversal cost from one cell to another

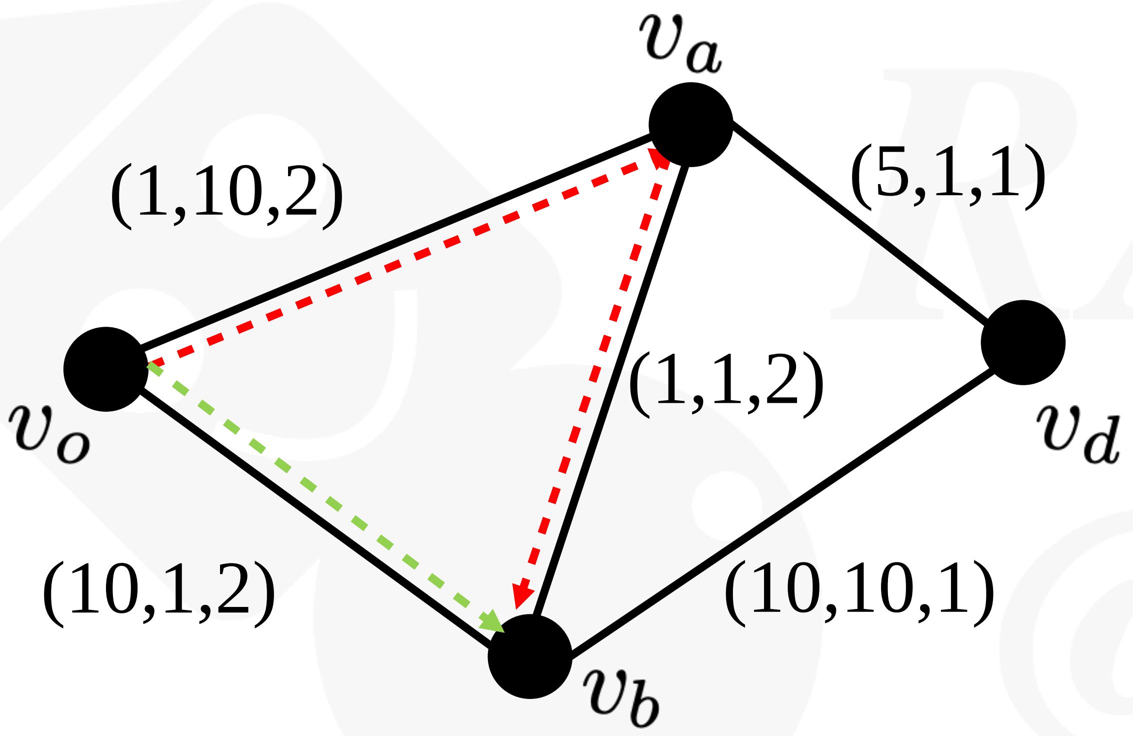
Risk cost map

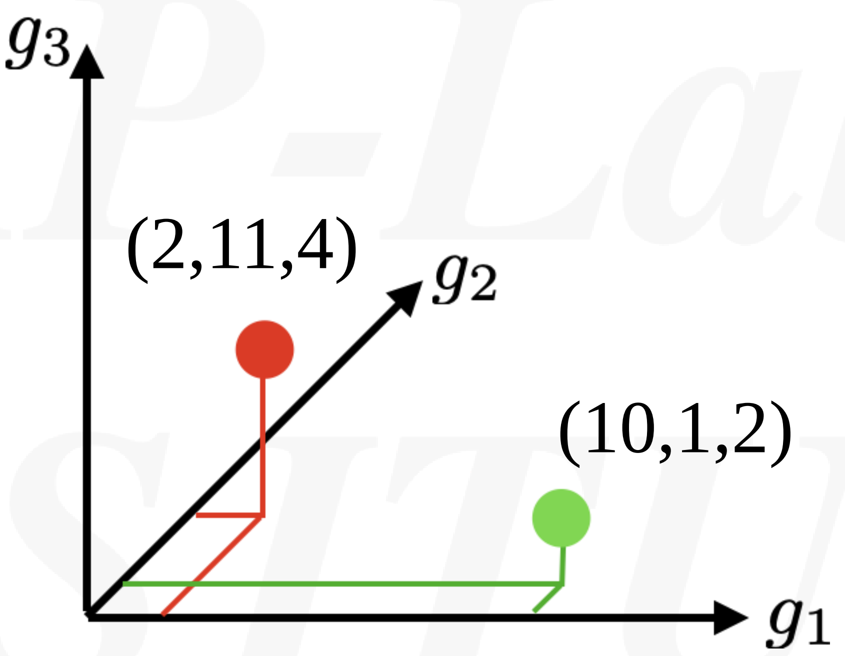
Multi-Objective Path Planning
- have to trade-off between multiple desired properties.

Contents

- (Single-Agent) Multi-Objective Path Planning
- Multi-Agent Multi-Objective Path Planning
- Discussion and Future Work

Multi-Objective Path Planning



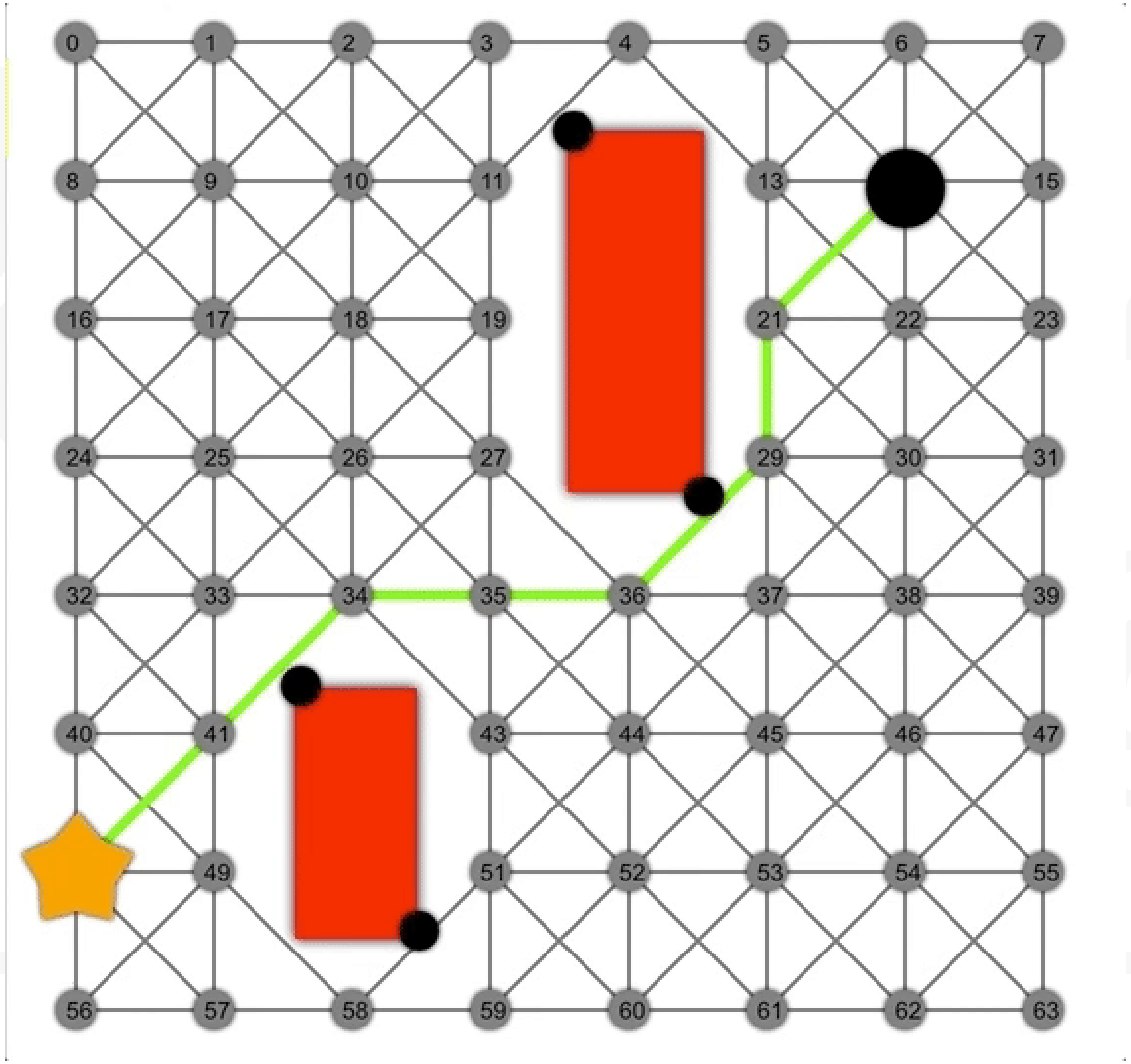


$G=(V,E)$

$$\vec{c}(e) = (c_1, c_2, \cdots, c_M) \in (R^+)^M$$

Stewart, B.S. and White III, C.C., 1991. Multiobjective a. *Journal of the ACM (JACM)*, *38*(4), pp.775-814.

A*-Based Path Planning Methods

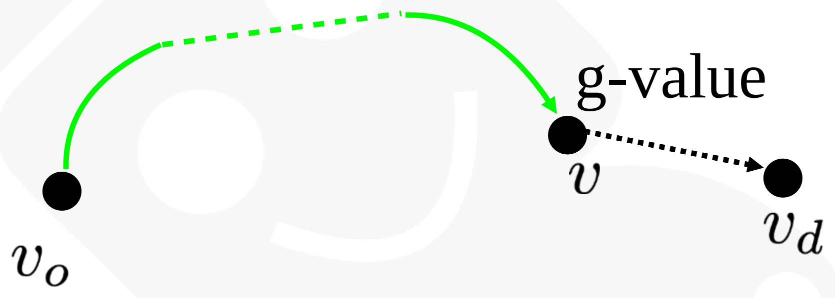


https://github.com/npretto/pathfinding

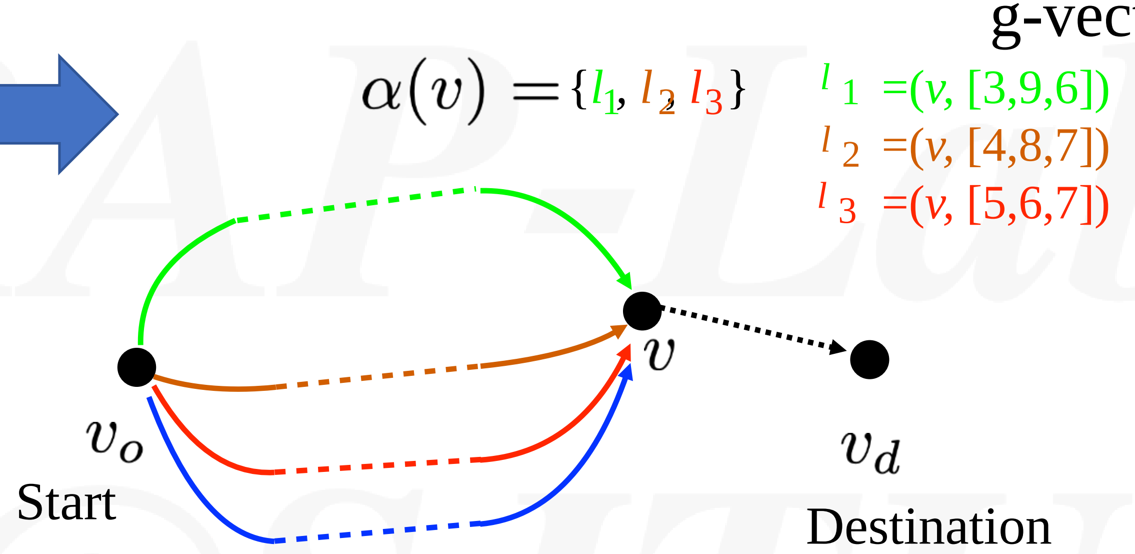
- Extend paths from the starting vertex towards the goal
- Systematically explore all possible paths
  - Always select the most promising path and extend it to all possible neighbors.
  - Keep the best path to reach each vertex
  - Discard the un-promising path
- Terminate when reach the goal

Challenge

Single Objective



Multi-Objective



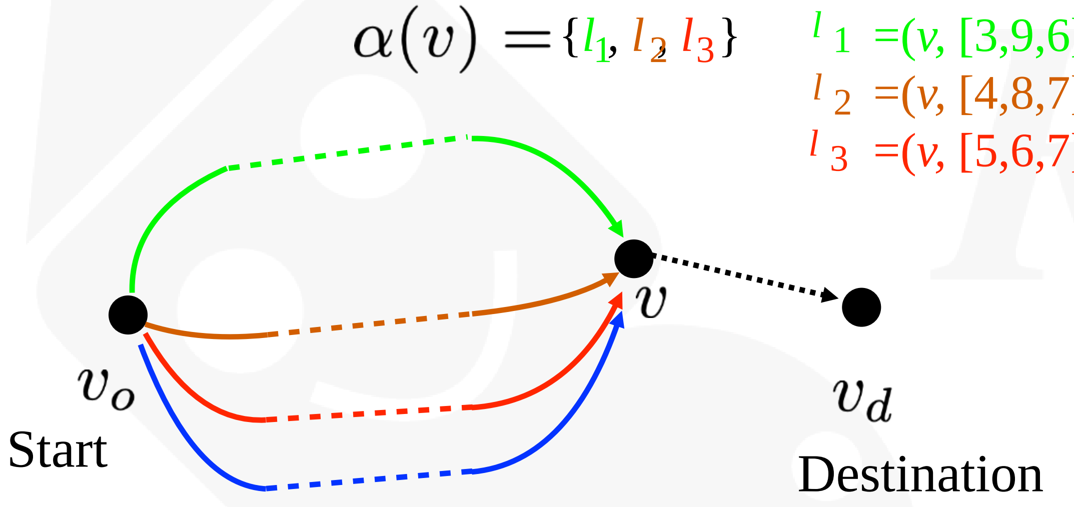
Scalar comparison

Against one existing scalar

Vector comparison

Against multiple non-dominated vectors

Fast Dominance Check



g-vector

l_1 =(v, [3,9,6])
l_2 =(v, [4,8,7])
l_3 =(v, [5,6,7])

(Existing) 1. By sorting the vectors in lex order, the first component of the vectors can be ignored [1].

Original	{(3,9,6), (4,8,7), (5,6,7)}
Truncated	$\{(9,6),(8,7),(6,7)\}$
Nondom. Trun.	$\{(9,6),(6,7)\}$

(Existing) 2. When there are only two objectives, all dominance checks are be converted to a scalar comparison in a lazy fashion [2].

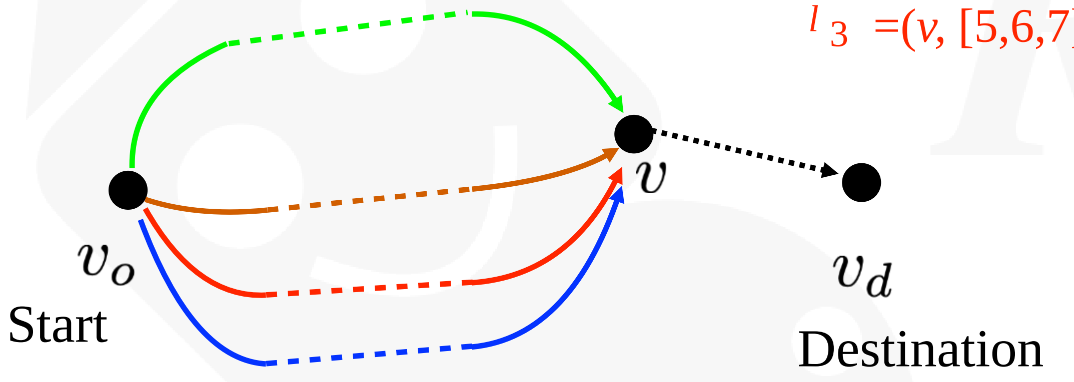
[1] Pulido, Francisco-Javier, Lawrence Mandow, and José-Luis Pérez-de-la-Cruz. "Dimensionality reduction in multiobjective shortest path search." Computers & Operations Research 64 (2015): 60-70.
[2] Hernández, Carlos, William Yeoh, Jorge A. Baier, Han Zhang, Luis Suazo, Sven Koenig, and Oren Salzman. "Simple and efficient bi-objective search algorithms via fast dominance checks." Artificial Intelligence 314 (2023): 103807.
[3] Ren, Zhongqiang, Richard Zhan, Sivakumar Rathinam, Maxim Likhachev, and Howie Choset. "Enhanced multi-objective A* using balanced binary search trees." In Proceedings of the International Symposium on Combinatorial Search, vol. 15, no. 1, pp. 162-170. 2022.

Fast Dominance Check

g-vector

$\alpha(v) = \{l_1, l_2, l_3\}$

$l_1 = (v, [3,9,6])$
$l_2 = (v, [4,8,7])$
$l_3 = (v, [5,6,7])$

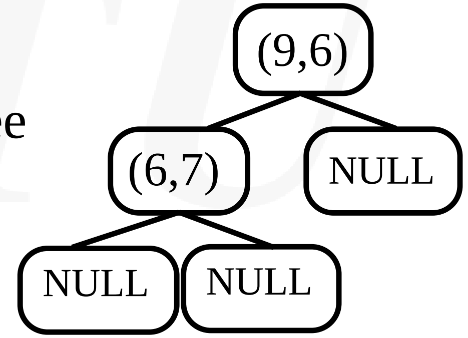


(Existing) 1. By sorting the vectors in lex order, the first component of the vectors can be ignored [1].

Original	{(3,9,6), (4,8,7), (5,6,7)}
Truncated	$\{(9,6),(8,7),(6,7)\}$
Nondom. Trun.	$\{(9,6),(6,7)\}$

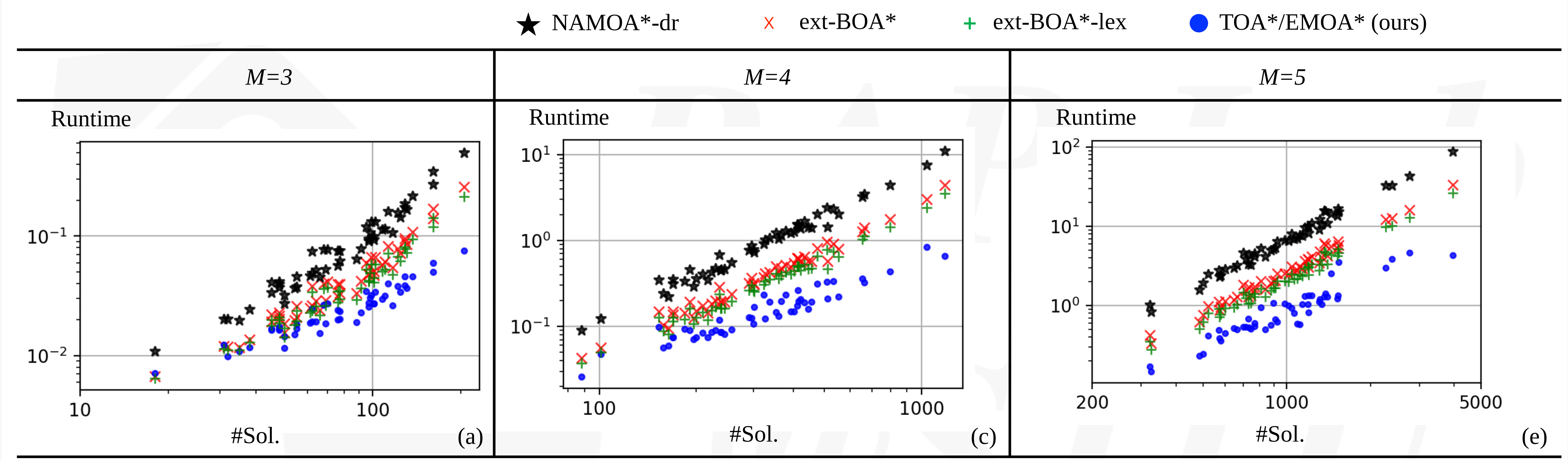
(Existing) 2. When there are only two objectives, all dominance checks are be converted to a scalar comparison in a lazy fashion [2].

Our contribution [3], **incrementally** build a balanced *binary search* tree to further expedite the dominance check for **any number** of objectives.



[1] Pulido, Francisco-Javier, Lawrence Mandow, and José-Luis Pérez-de-la-Cruz. "Dimensionality reduction in multiobjective shortest path search." Computers & Operations Research 64 (2015): 60-70.
[2] Hernández, Carlos, William Yeoh, Jorge A. Baier, Han Zhang, Luis Suazo, Sven Koenig, and Oren Salzman. "Simple and efficient bi-objective search algorithms via fast dominance checks." Artificial Intelligence 314 (2023): 103807.
[3] Ren, Zhongqiang, Richard Zhan, Sivakumar Rathinam, Maxim Likhachev, and Howie Choset. "Enhanced multi-objective A* using balanced binary search trees." In Proceedings of the International Symposium on Combinatorial Search, vol. 15, no. 1, pp. 162-170. 2022.

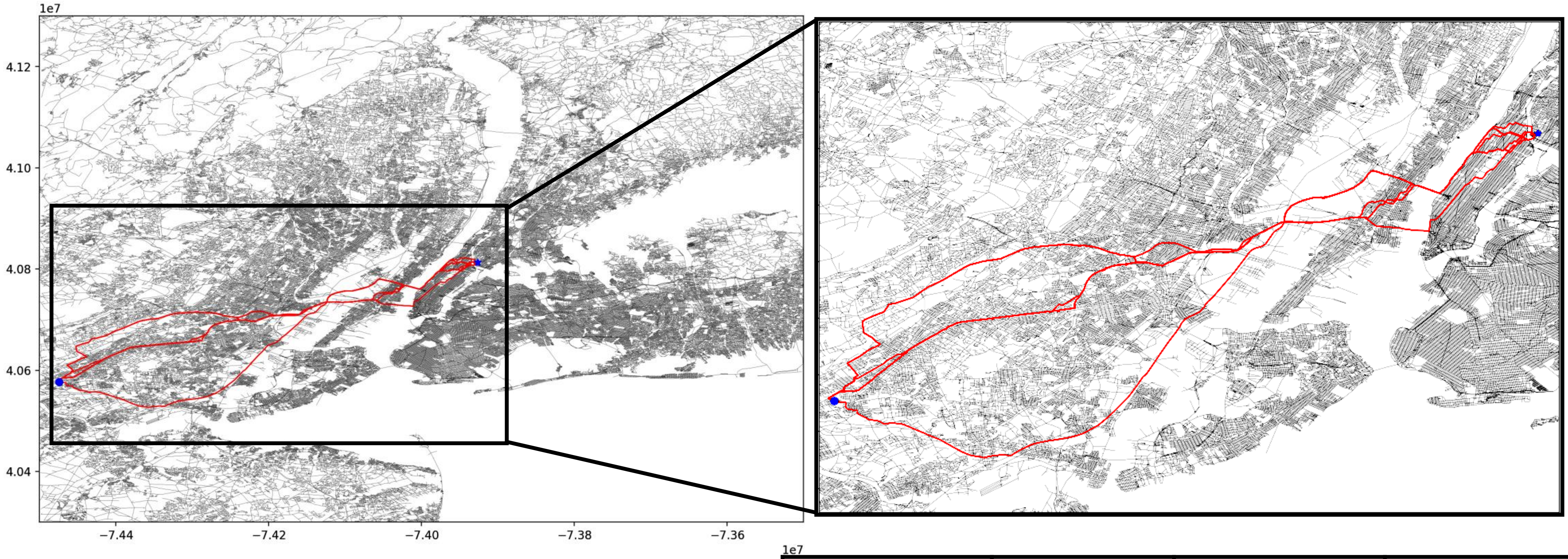
Result



Summary:

1. **Faster** than baselines for up to **an order of magnitude.**
2. More advantageous for **hard instances**.

Result



New York City Map from [1], 264,346 nodes, 733,846 edges.

10 minutes (600 seconds) runtime limit.

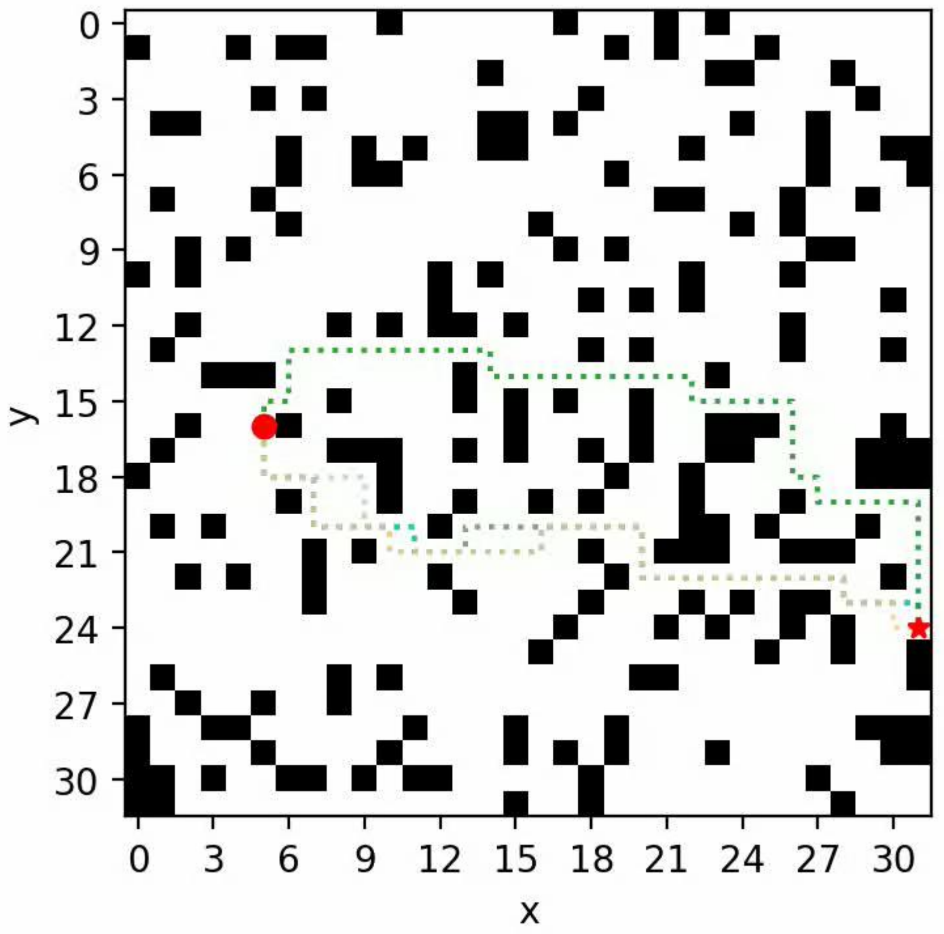
Summary:
- **Double** the success rates.
- Run **faster**.

	Success/All	*Mean Runtime (Sec.)	*Median Runtime (Sec.)
NAMOA*-dr	16/50	25.4	92.9
ext-BOA*	17/50	11.6	40.1
ext-BOA*-lex	17/50	9.7	33.7
Ours	33/50	1.8	5.0

* Mean and median are computed for the 16 instances where all four algorithms succeed. The average and median number of solutions are 389 and 327 respectively.

[1] http://www.diag.uniroma1.it//~challenge9/download.shtml

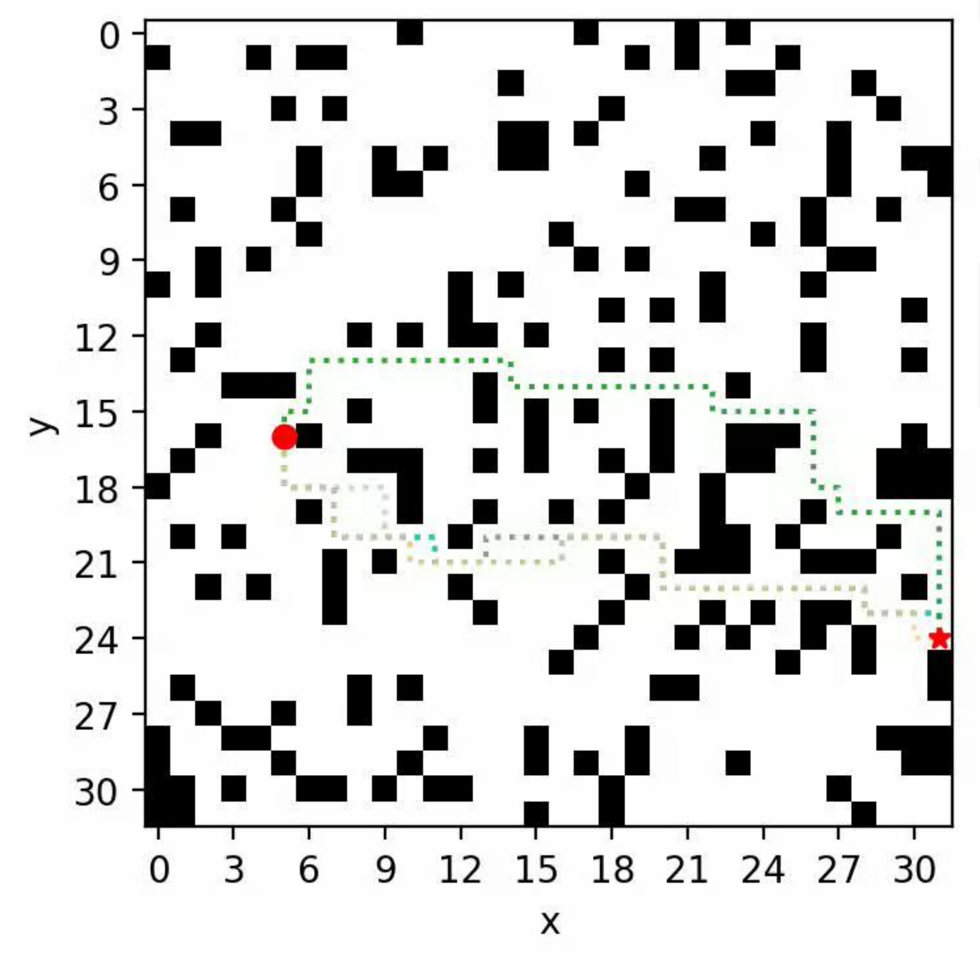
Dynamic Environments



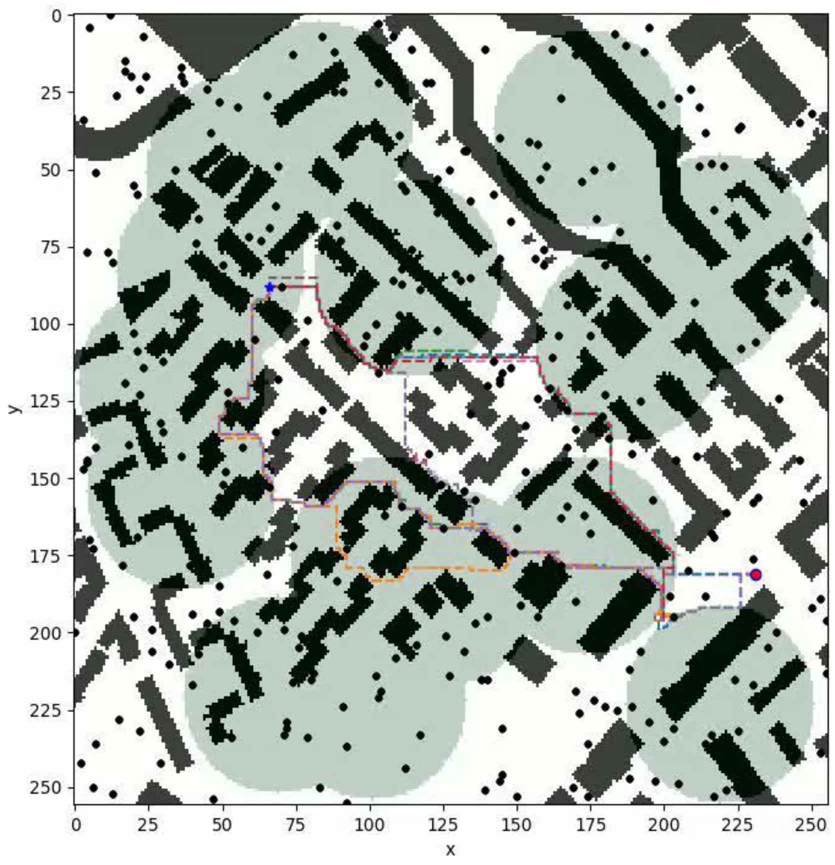
Edge cost may change during path exeution [1].

[1] Ren, Z., Rathinam, S., Likhachev, M. and Choset, H., 2022. Multi-objective path-based D* lite.*IEEE Robotics and Automation Letters*,7(2), pp.3318-3325.

Dynamic Environments



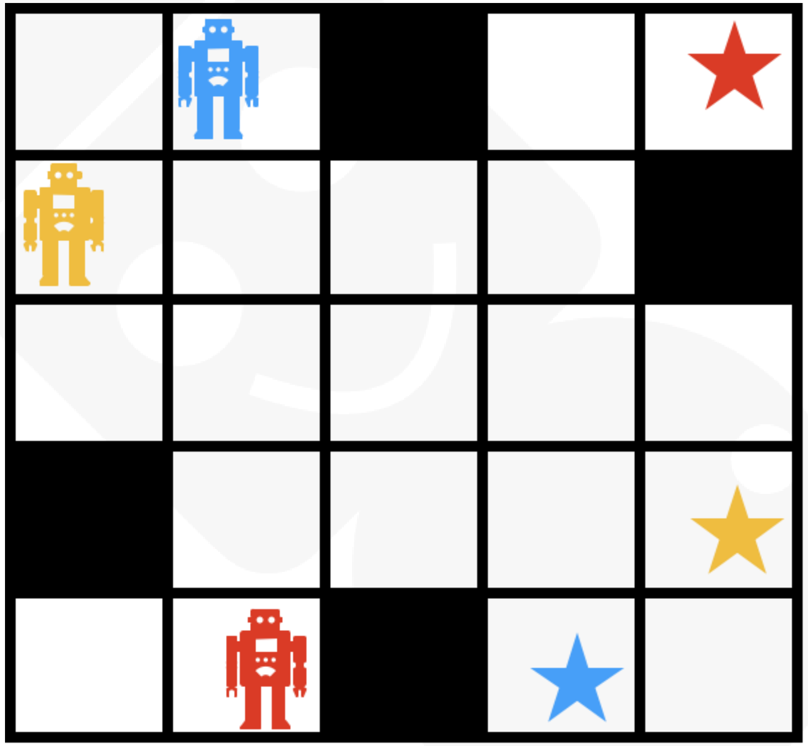
Edge cost may change during path exeution [1].

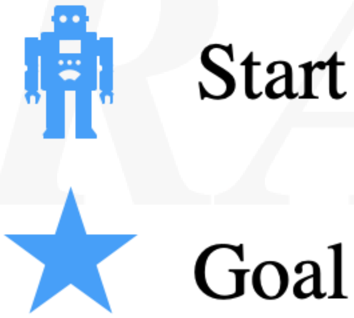


Moving obstacles along known trajectories [2].

[1] Ren, Z., Rathinam, S., Likhachev, M. and Choset, H., 2022. Multi-objective path-based D* lite.*IEEE Robotics and Automation Letters*,*7*(2), pp.3318-3325.
[2] Ren, Z., Rathinam, S., Likhachev, M. and Choset, H., 2022. Multi-objective safe-interval path planning with dynamic obstacles.*IEEE Robotics and Automation Letters*,*7*(3), pp.8154-8161.

Multi-Agent Multi-Objective Path Planning





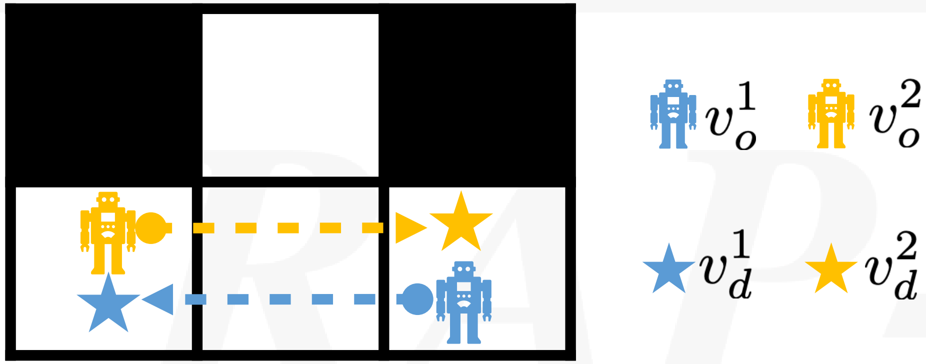
1. Shared workspace G=(V,E)
2. Each action takes a time unit.
3. Avoid agent-agent conflict.
4. Each action incurs a cost vector

$$\vec{c}(e) = (c_1, c_2, \cdots, c_M) \in (R^+)^M$$

5. Minimize the sum of arrival times.

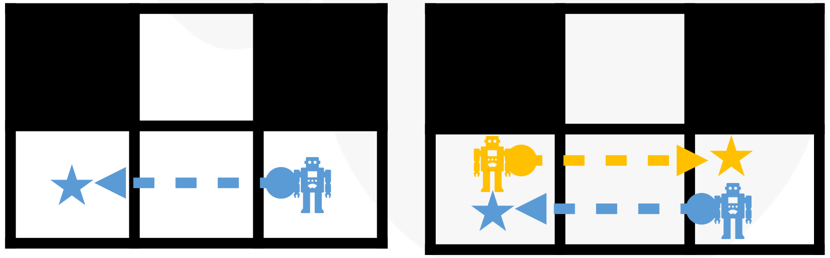
Ren, Z., Rathinam, S. and Choset, H., 2021. Subdimensional expansion for multi-objective multi-agent path finding. *IEEE Robotics and Automation Letters*, *6*(4), pp.7153-7160.
Ren, Z., Rathinam, S. and Choset, H., 2022. A conflict-based search framework for multiobjective multiagent path finding. *IEEE Transactions on Automation Science and Engineering*, *20*(2), pp.1262-1274.

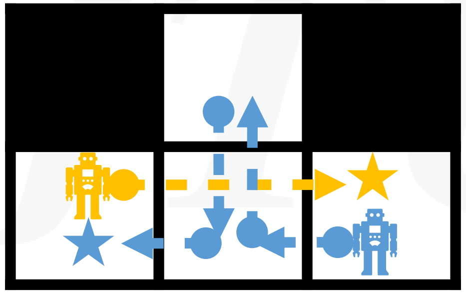
Multi-Agent Path Finding



De-coupled Planning

Coupled Planning

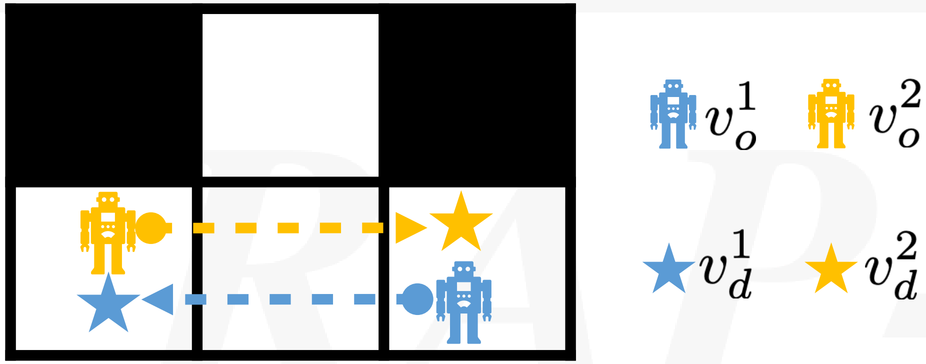




Run fast, incomplete, no solution quality guarantee

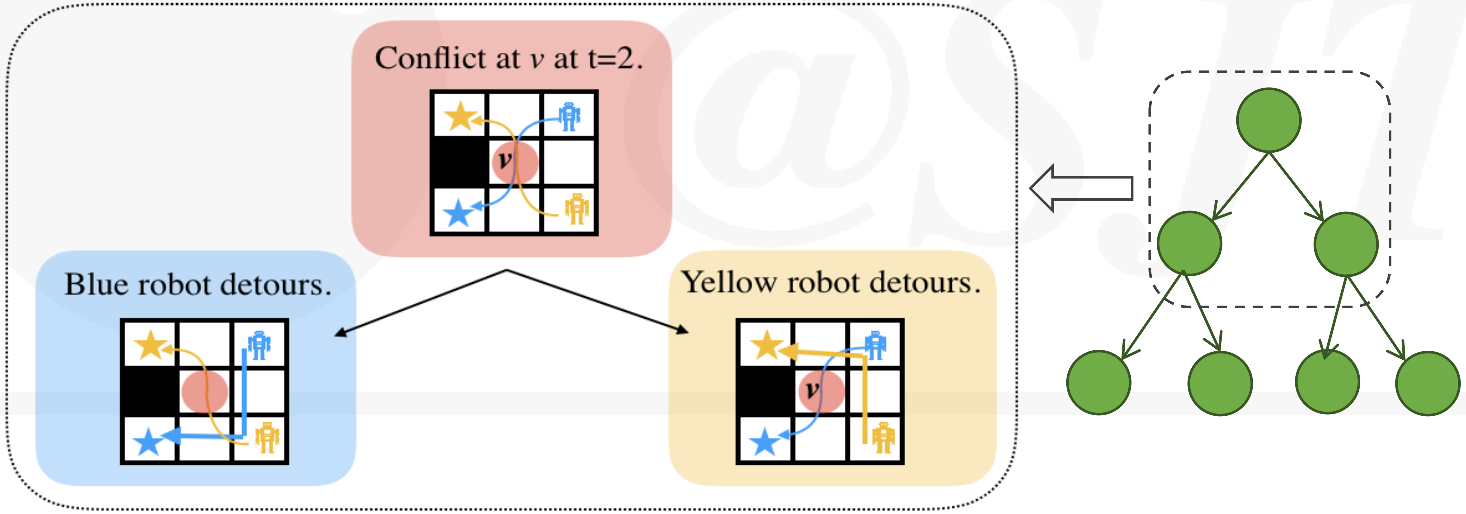
Complete, optimal solution, scales poorly.

Multi-Agent Path Finding



Dynamically Coupled Planning

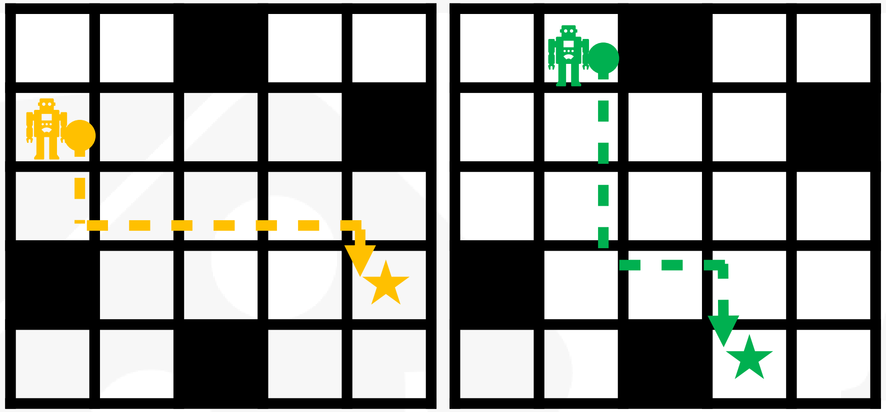
E.g. Conflict-Based Search (CBS) [1]



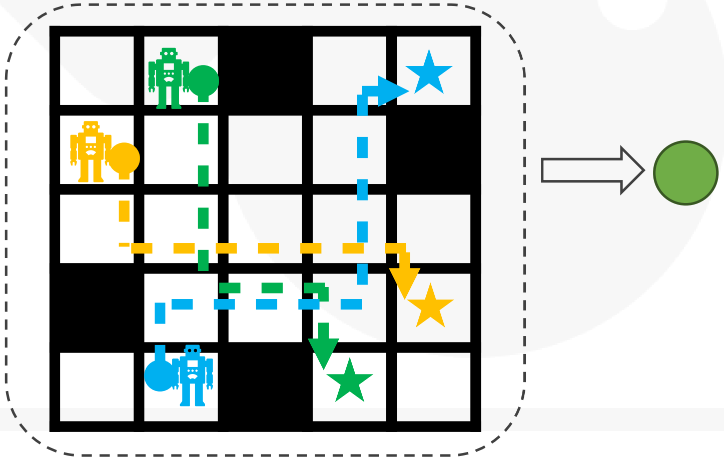
How to handle multiple objectives?

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

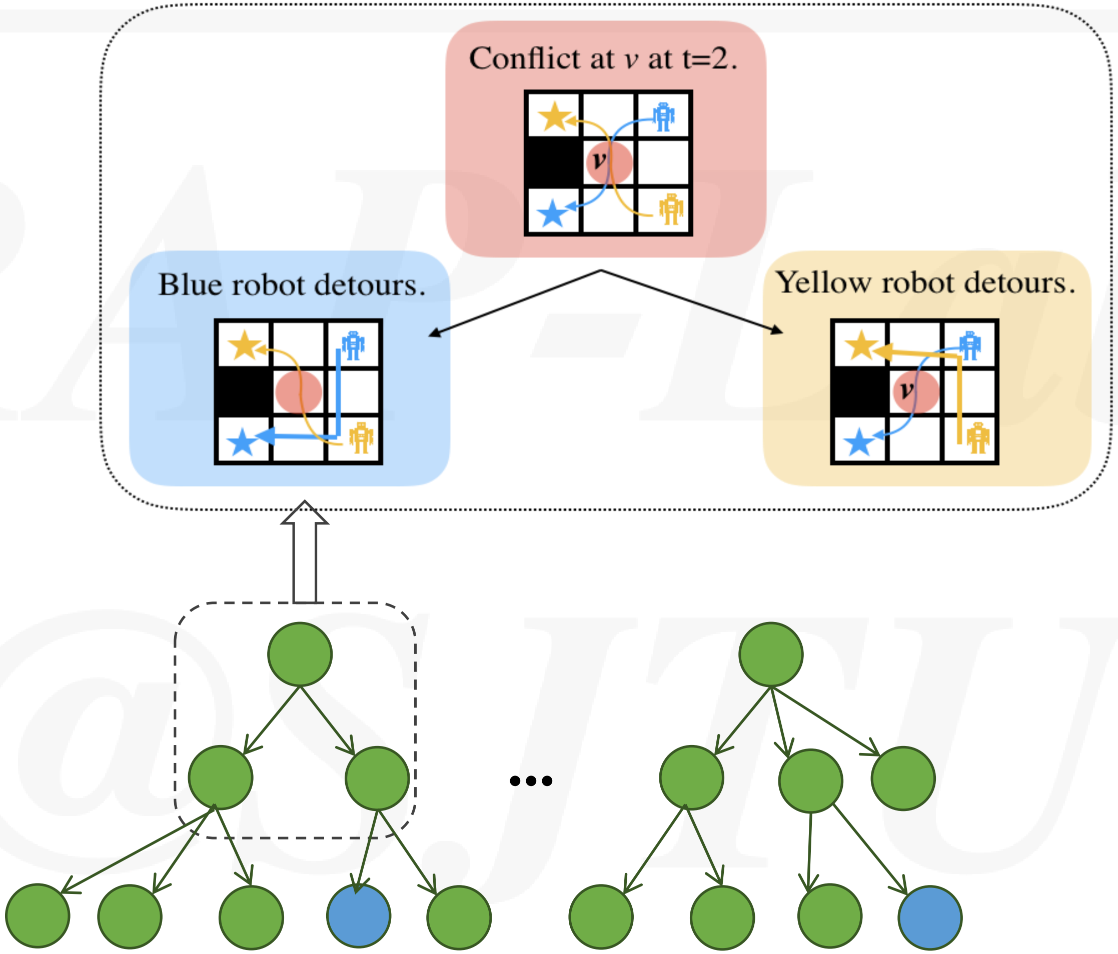
Method: Multi-Objective Conflict-Based Search



1. Ignore conflicts and solve single-agent multi-objective problems for each agent.



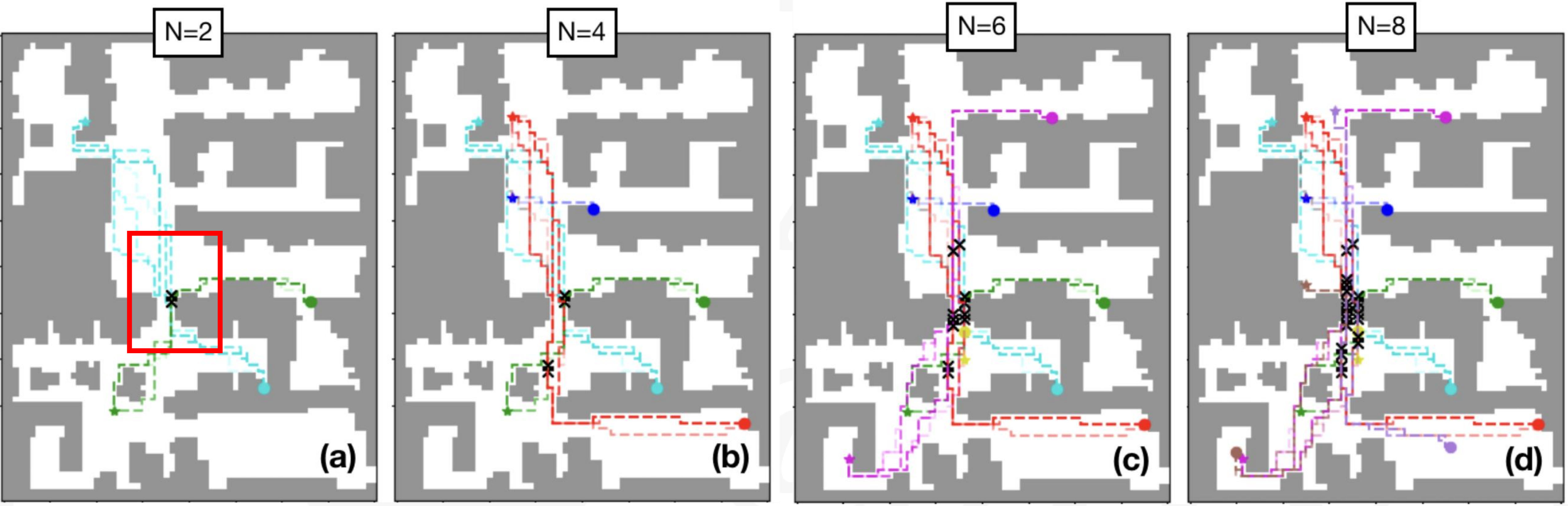
2. Take combination to form joint paths



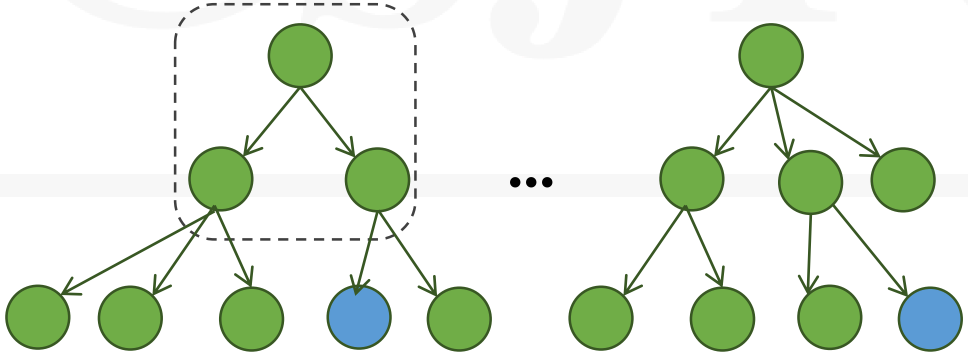
3. Runs CBS-like search to resolve agent-agent conflicts until all Pareto-optimal solutions are found.

Ren, Z., Rathinam, S. and Choset, H., 2022. A conflict-based search framework for multiobjective multiagent path finding. *IEEE Transactions on Automation Science and Engineering*,*20*(2), pp.1262-1274.

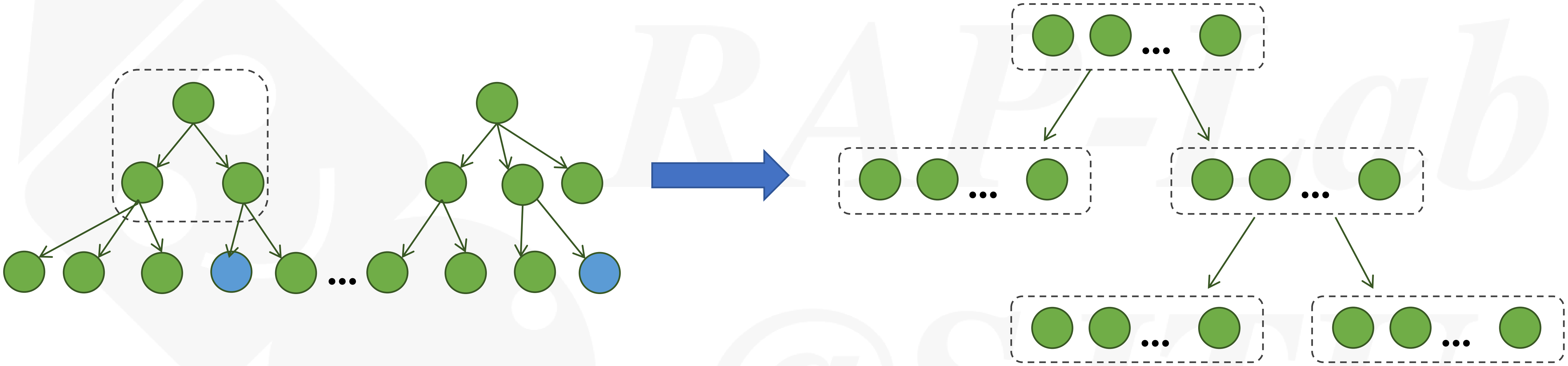
Duplicated Conflict Resolution



The same conflict may occurs
multiple times in different trees



Method: Binary Branching MO-CBS



- Redefine search node, each node contains multiple joint paths
- Resolve the same conflict in multiple joint paths at once.

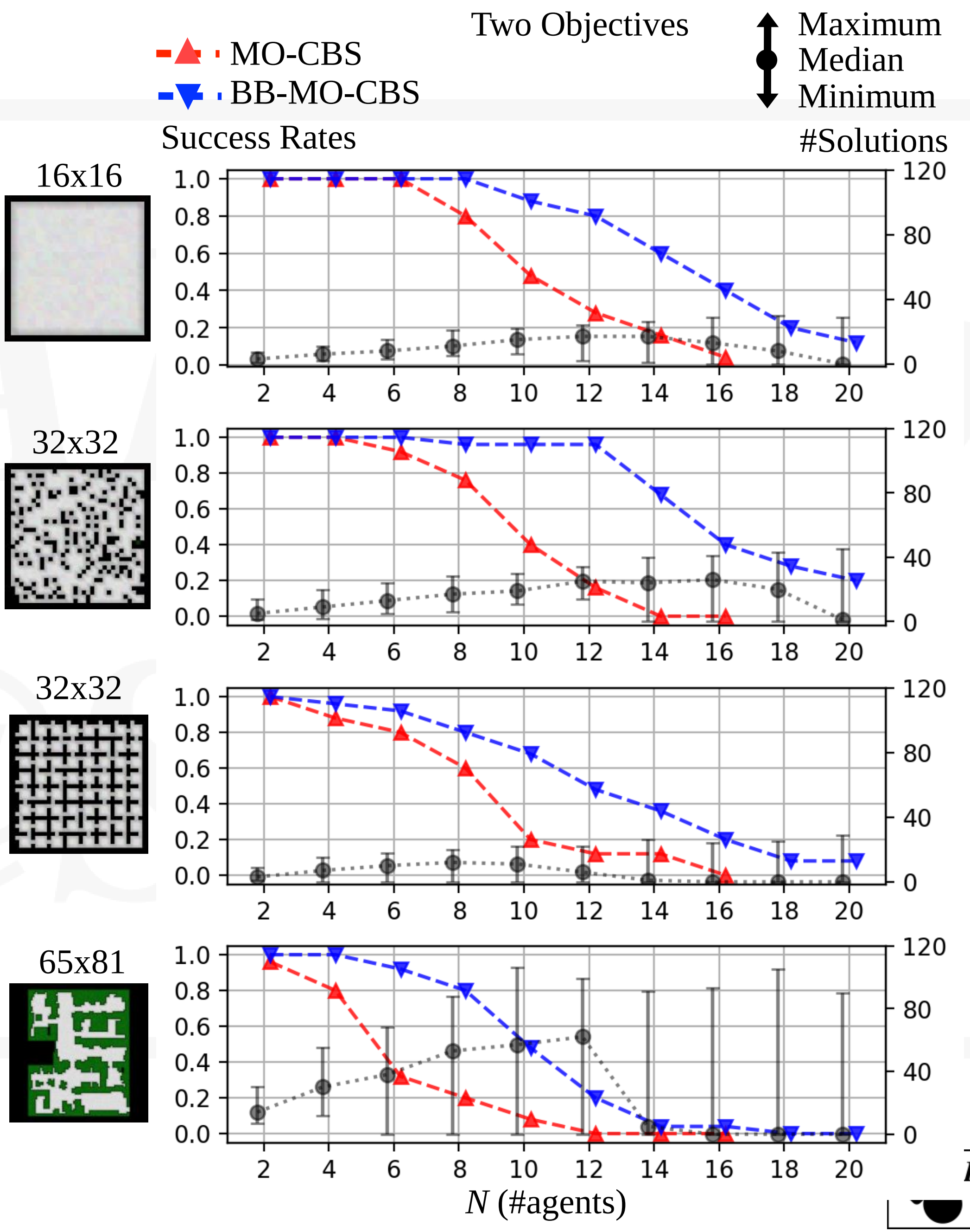
Ren, Z., Li, J., Zhang, H., Koenig, S., Rathinam, S. and Choset, H., 2023, July. Binary branching multi-objective conflict-based search for multi-agent path finding. In*Proceedings of the International Conference on Automated Planning and Scheduling*(Vol. 33, No. 1, pp. 361-369).

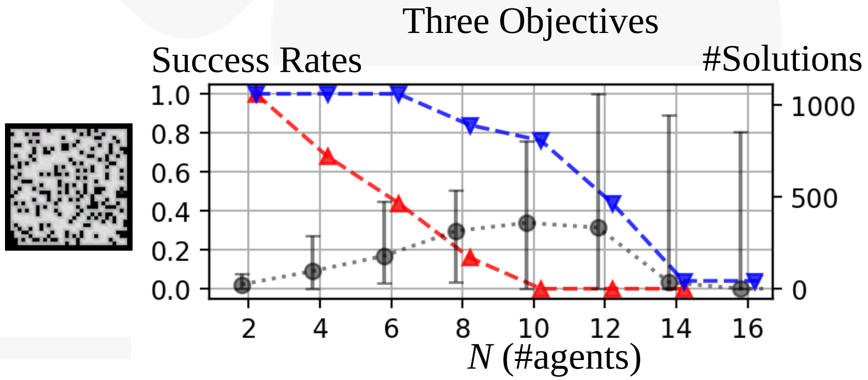
Result

- Different maps with two or three objectives (M=2,3).
- Edge cost vectors randomly sampled from {1,2} for each component.
- 5 min runtime limit for each instance.

Message

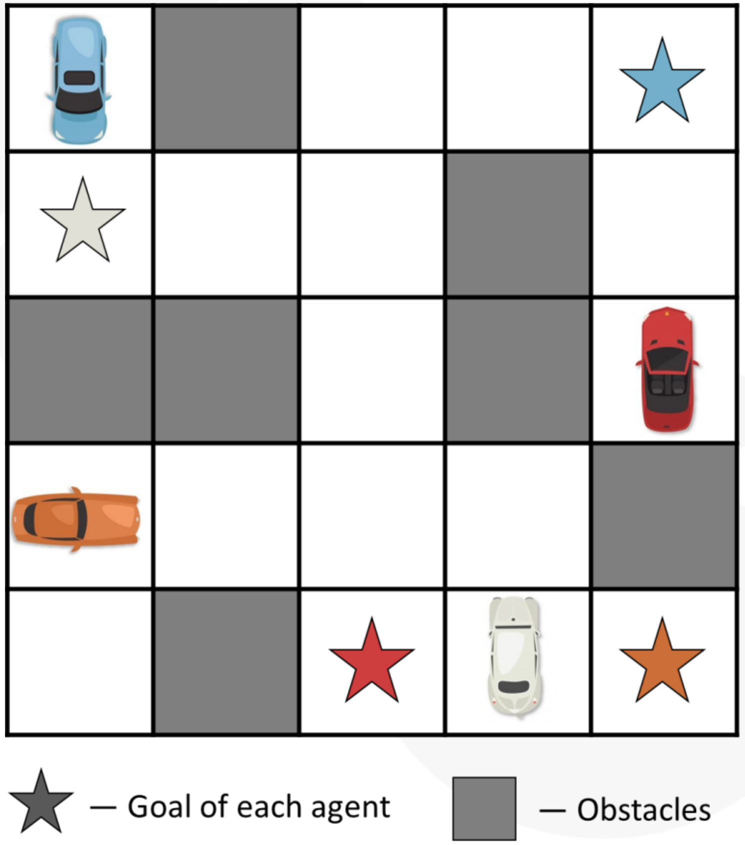
- BB-MO-CBS has higher success rates than MO-CBS in all settings.





Teamwise Cooperative MAPF

Finding conflict-free paths



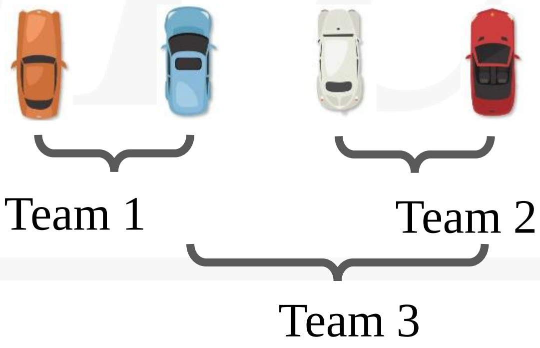
Previous:

Multi-Agent Path Finding

- All agents in one team (fully cooperative)
- **Minimize sum of individual cost (vectors)**

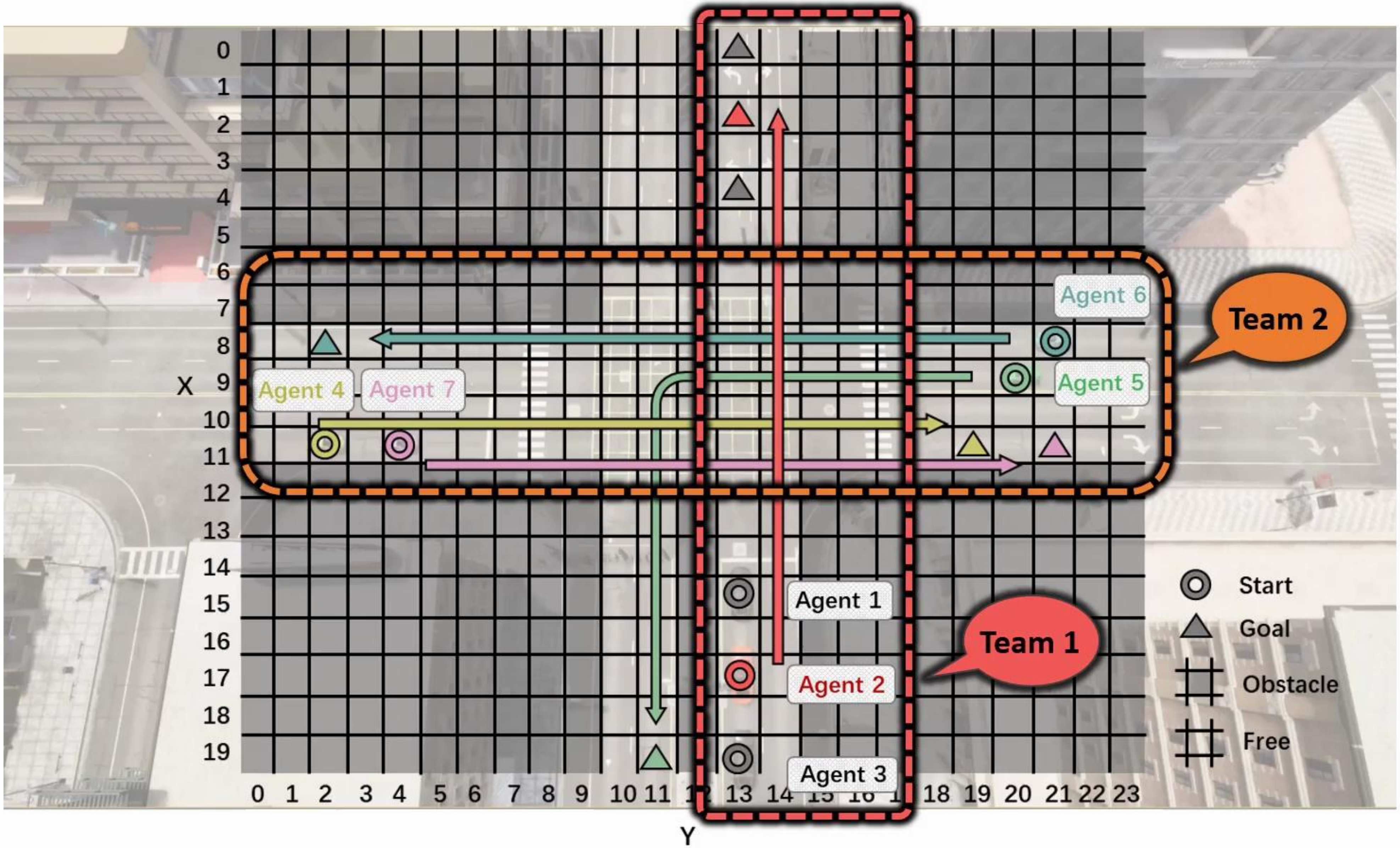
Here:

- Agents are grouped into teams
- **Each team has its own objective**

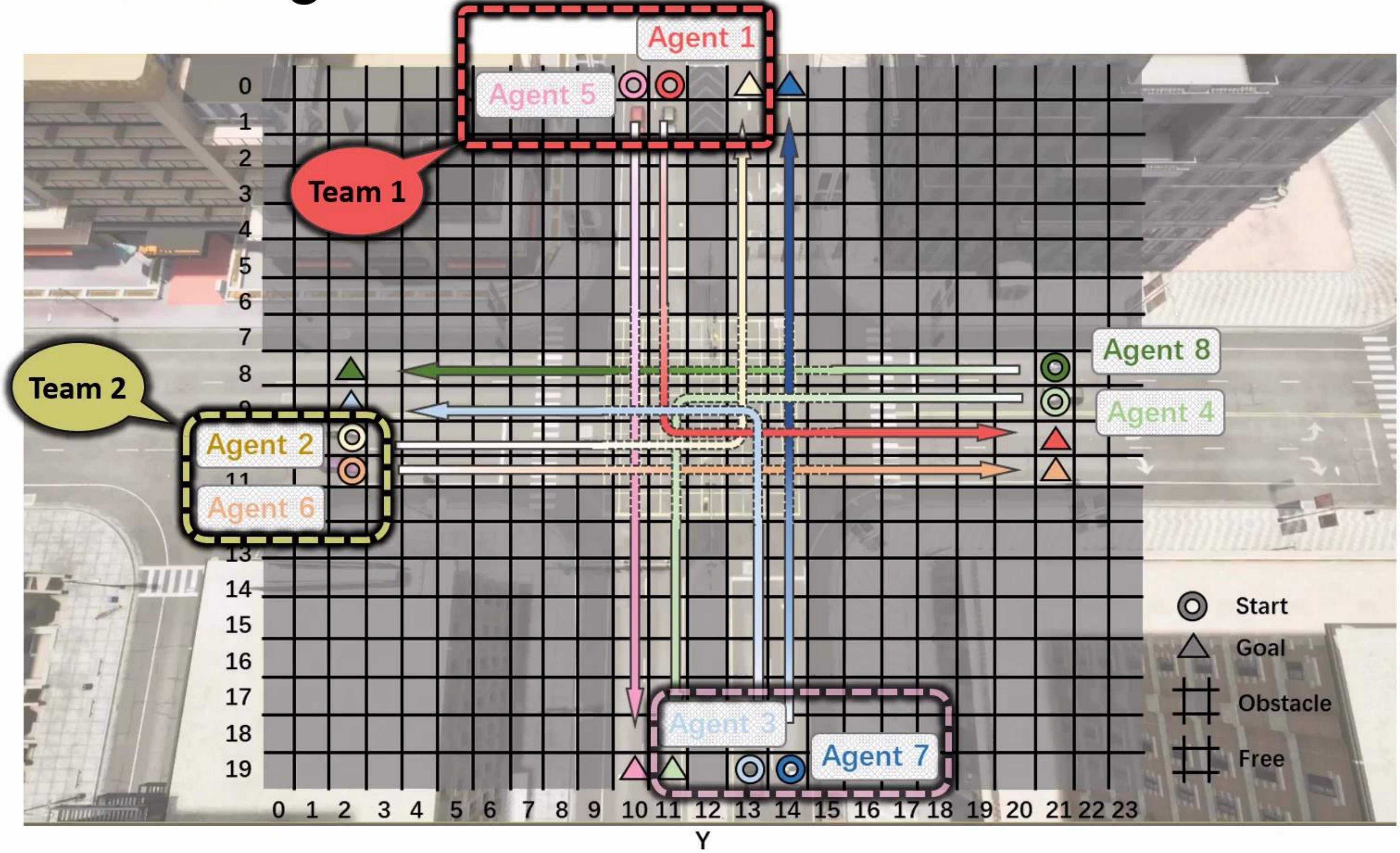


Ren, Z., Zhang, C., Rathinam, S. and Choset, H., 2023, May. Search Algorithms for Multi-Agent Teamwise Cooperative Path Finding. In*2023 IEEE International Conference on Robotics and Automation (ICRA)*(pp. 1407-1413). IEEE.
Ren, Z., Cai, Y., Wang, H., 2024,. Multi-Agent Teamwise Cooperative Path Finding and Traffic Intersection Coordination. IROS Accepted, 2024

Demo 1: **Motorcade**



Demo 3: 8 Agents with 4 Teams

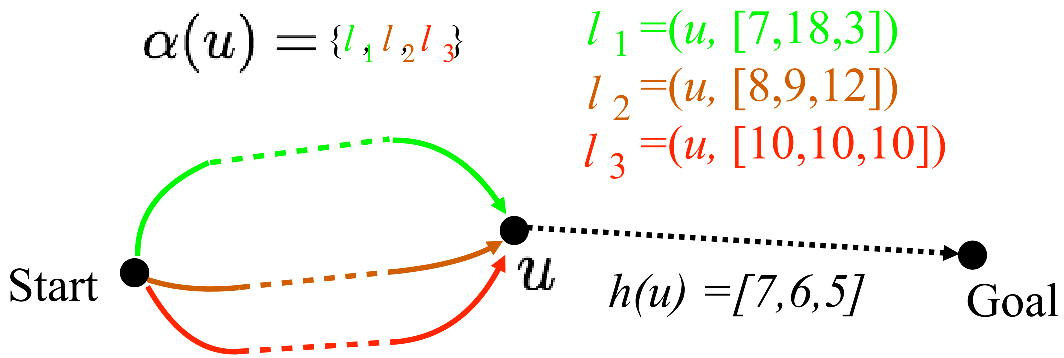


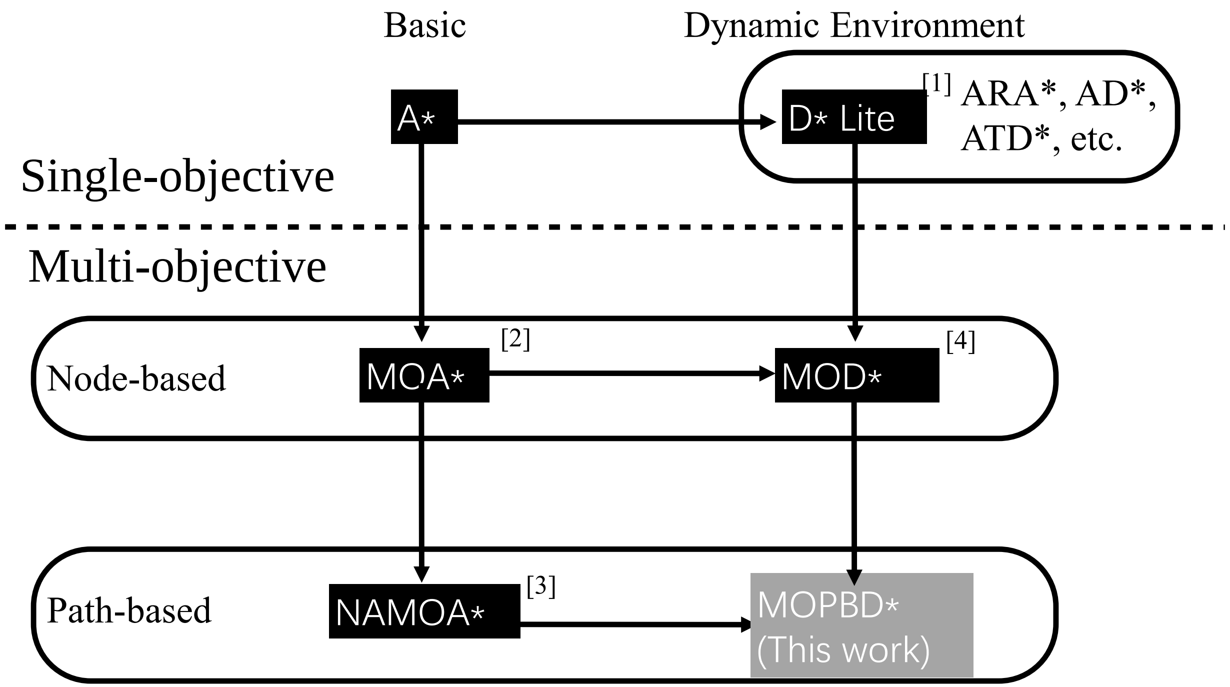
Summary

- (Single-Agent) Multi-Objective Path Planning
  - Fast Dominance Check
  - Dynamic Environments
- Multi-Agent Multi-Objective Path Planning
  - Agent-agent collision avoidance
  - Teamwise cooperativeness
- Discussion and Future Work
  - Robot dynamics and uncertainty
  - Non-additive path costs
  - Bounded sub-optimal methods with scalability

RAP-Lab
@SJTU

Related Work





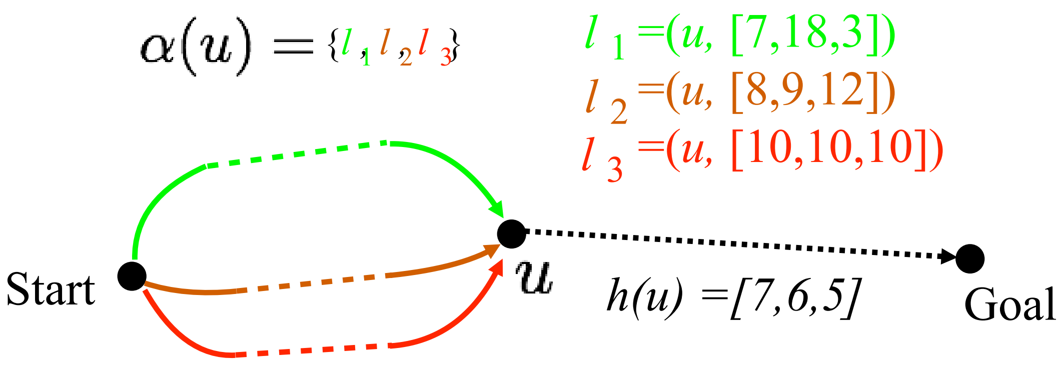
[1] Koenig, Sven, and Maxim Likhachev. "Fast replanning for navigation in unknown terrain." IEEE Transactions on Robotics 21, no. 3 (2005): 354-363.

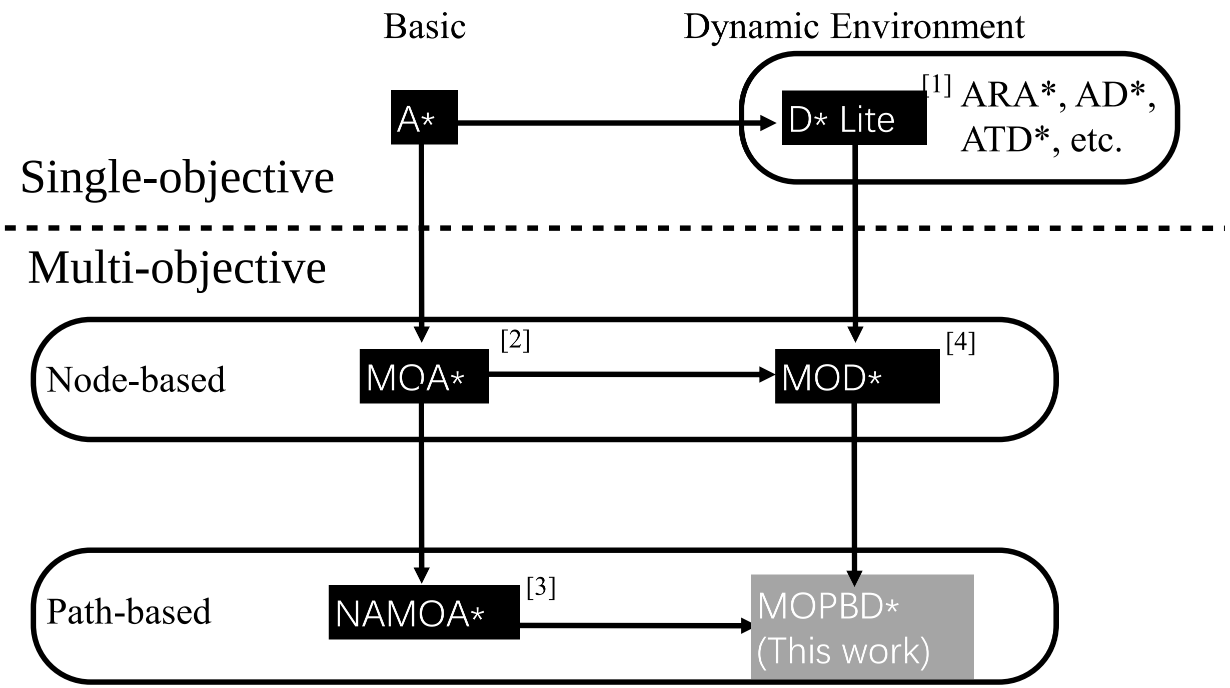
[2] Stewart, Bradley S., and Chelsea C. White III. "Multiobjective a." Journal of the ACM (JACM) 38, no. 4 (1991): 775-814.

[3] Mandow, Lawrence, and José Luis Pérez De La Cruz. "Multiobjective A* search with consistent heuristics." Journal of the ACM (JACM) 57, no. 5 (2008): 1-25.

[4] Oral, Tugcem, and Faruk Polat. "MOD* Lite: an incremental path planning algorithm taking care of multiple objectives." IEEE Transactions on Cybernetics 46, no. 1 (2015): 245-257.

Related Work





Fact: Multiple non-dominated partial solution paths from the start to any other vertices (or say nodes) in the graph.

Node-based Expansion (MOA*, MOD*)

- In each iteration, a node is selected from OPEN and expanded.
- To expand a node, all labels (i.e. partial solution paths) are extended to adjacent nodes.

Path-based Expansion (NAMOA* and its variants)

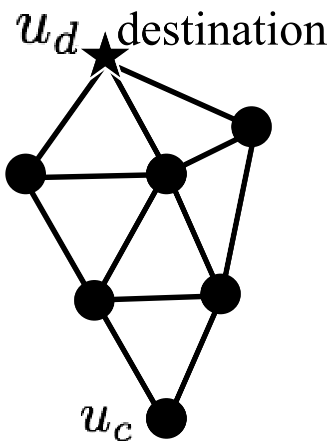
- Labels (i.e. partial solution paths) are stored in OPEN and selected for expansion.
- When a new label is generated at a node, this label (rather than node) is inserted into OPEN.

In conventional A*, the search only needs to store one optimal partial path from the start to any other vertices in the graph. Node-based and Path-based are equivalent.

[1] Koenig, Sven, and Maxim Likhachev. "Fast replanning for navigation in unknown terrain." IEEE Transactions on Robotics 21, no. 3 (2005): 354-363.
[2] Stewart, Bradley S., and Chelsea C. White III. "Multiobjective a." Journal of the ACM (JACM) 38, no. 4 (1991): 775-814.
[3] Mandow, Lawrence, and José Luis Pérez De La Cruz. "Multiobjective A* search with consistent heuristics." Journal of the ACM (JACM) 57, no. 5 (2008): 1-25.
[4] Oral, Tugcem, and Faruk Polat. "MOD* Lite: an incremental path planning algorithm taking care of multiple objectives." IEEE Transactions on Cybernetics 46, no. 1 (2015): 245-257.

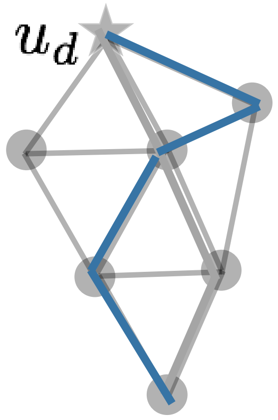
Method

MO-PBD* Algorithm Overview [1]



start (current node)

(a)



(b)

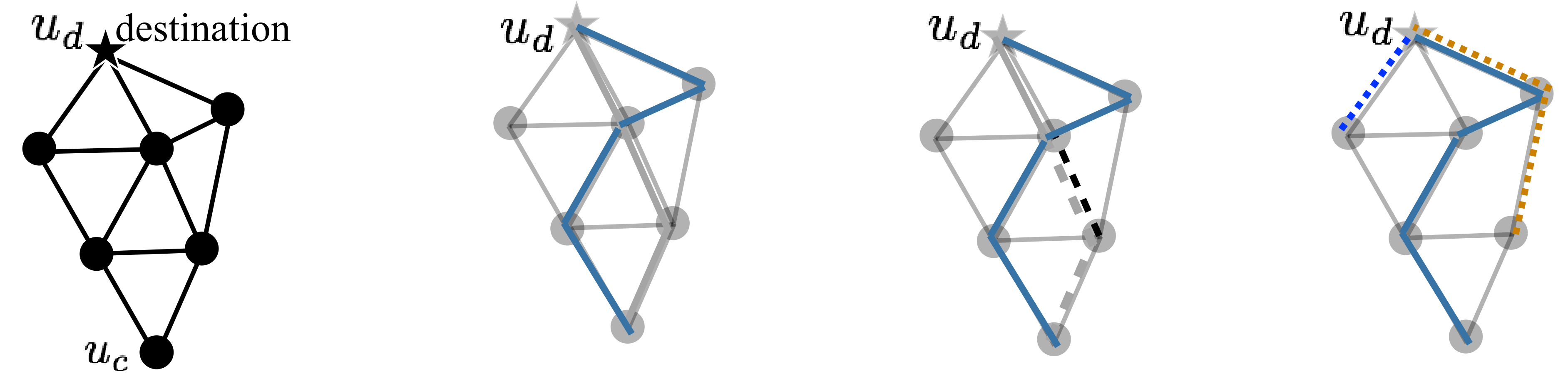
Initially, search in as same way as NAMOA* (path-based) **backwards** from the goal to the start.

Non-dominated paths		Path cost vectors
1st search, (Plot (b))		(10, 3.3, 331) (15, 2.1, 575)

[1] Ren, Zhongqiang, Sivakumar Rathinam, Maxim Likhachev, and Howie Choset. "Multi-Objective Path-Based D* Lite." *IEEE Robotics and Automation Letters* 7, no. 2 (2022): 3318-3325.

Method

MO-PBD* Algorithm Overview [1]



(a)

(b) Initially, search in as same way as NAMOA* (path-based) **backwards** from the goal to the start.

(c) When *cost(e)* changes (either increases or decreases), recursively **find and delete all partial paths** that go through *e*.

(d) Using **a new notion of *consistency*** to find all partial paths that need re-expansion.

Def. Label $l := (v, \vec{g})$

Def. $RHS(v) = \begin{cases} \{0\} & \text{If } v = u_d \\ nondom(G(u) + \vec{c}(u,v)), \forall u \in Adj(v) & \text{Otherwise} \end{cases}$

A set of g-vectors

Def. Label Consistency

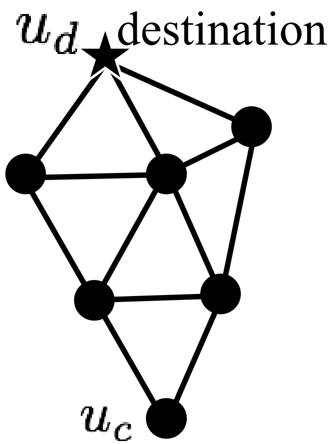
A label l is inconsistent if $\vec{g}(l)$ is in *RHS(v(l))* and $\vec{g}(l)$ is not in *G(v(s))*.

A label l is consistent if $\vec{g}(l)$ is in both *RHS(v(s))* and *G(v(s))*.

[1] Ren, Zhongqiang, Sivakumar Rathinam, Maxim Likhachev, and Howie Choset. "Multi-Objective Path-Based D* Lite." *IEEE Robotics and Automation Letters* 7, no. 2 (2022): 3318-3325.

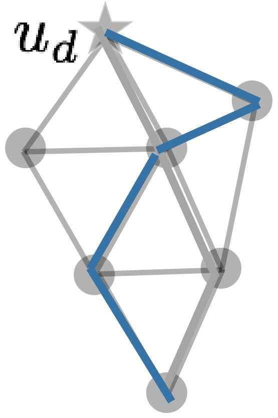
Method

MO-PBD* Algorithm Overview [1]



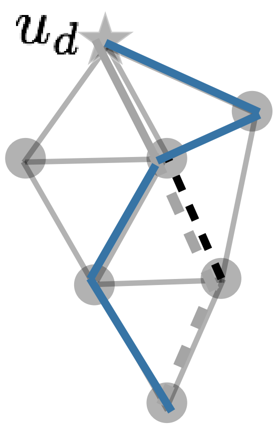
start (current node)

(a)



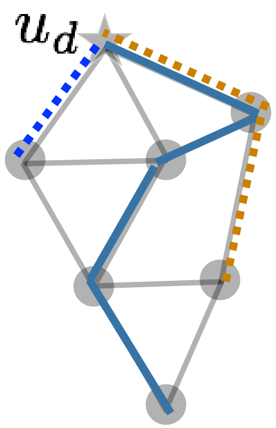
(b)

Initially, search in as same way as NAMOA* (path-based) **backwards** from the goal to the start.



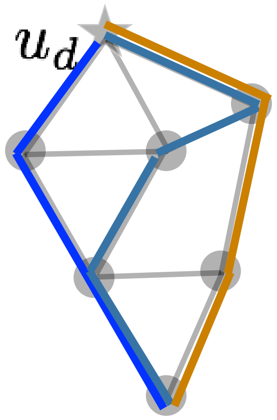
(c)

When *cost(e)* changes (either increases or decreases), recursively **find and delete all partial paths** that go through *e*.



(d)

Using **a new notion of *consistency*** to find all partial paths that need re-expansion.



(e)

Continue the search until all cost-unique Pareto-optimal paths are find.

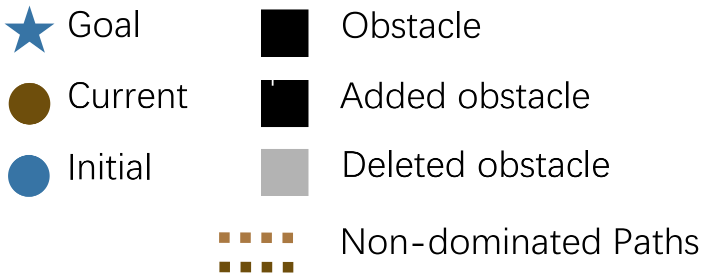
Non-dominated paths		Path cost vectors
1st search, (Plot (b))		(10, 3.3, 331) (15, 2.1, 575)
2nd search, (Plot (e))		(15, 2.1, 575) (12, 3.6, 461) (13, 3.4, 501)

[1] Ren, Zhongqiang, Sivakumar Rathinam, Maxim Likhachev, and Howie Choset. "Multi-Objective Path-Based D* Lite." *IEEE Robotics and Automation Letters* 7, no. 2 (2022): 3318-3325.

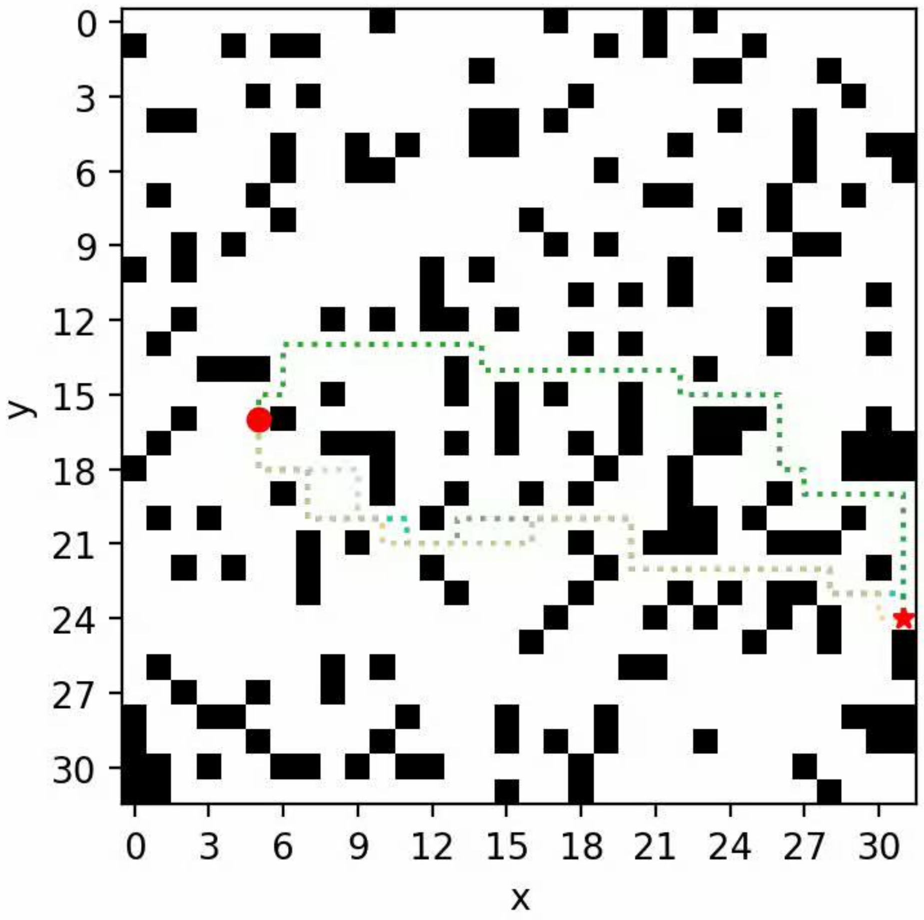
Results

Test Settings

- Test with M=2,3,4 objectives, each component of edge cost vectors is randomly selected from [1,5].
- Iteratively (1) plan, (2) randomly select and execute, (3) add/delete obstacles.
- Run time limit 5 minutes.



Add or delete two obstacles in the 5x5 box area centered on the robot every 7 steps.



Results

Test Settings

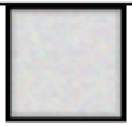
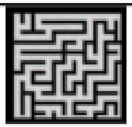
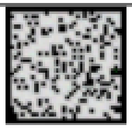
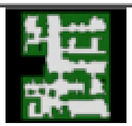
Fixed map (Maze), varying *M*.

M	Planner	Remove Obst.	Add Obst.
2	MOPBD* (ours)	**0.0060 (0.070)**	**0.018 (0.12)**
	NAMOA*	0.042 (0.15)	0.045 (0.19)
3	MOPBD* (ours)	**0.037 (4.6)**	**0.14 (14)**
	NAMOA*	0.099 (4.4)	0.17 (6.2)
4	MOPBD* (ours)	**0.062 (1.44)**	0.24 (15)
	NAMOA*	0.12 (2.13)	**0.17 (5.0)**

Runtime in format Median (Average)

	From scratch	Incremental
Node-based		MOD*
Path-based	NAMOA*	MOPBD* (This work)

Fixed *M*=2 (two objective), varying maps

Grids	Algorithm	Exp.	R.T.	Sol.
 (16x16)	NAMOA*	111.8	**0.03**	3.0
	MOD*	39.1	0.35	3.0
	MOPBD*	**3.9**	0.06	3.0
 (32x32)	NAMOA*	1556.6	0.55	10.5
	MOD*	92.1	3.15	10.5
	MOPBD*	**19.7**	**0.17**	10.5
 (32x32)	NAMOA*	829.5	0.22	4.9
	MOD*	311.0	3.51	4.9
	MOPBD*	**35.0**	**0.12**	4.9
 (65x81)	NAMOA*	5923.3	2.85	16.3
	MOD*	208.4	12.6	12.3
	MOPBD*	**28.0**	**2.43**	16.3

Average over all instances
(*)Timeout in some instances

- MOPBD* outperforms MOD* in all scenario;
- For run time, MOPBD*, in general, outperforms NAMOA* (search from scratch) on average;
- For number of expansion (path-based), MOPBD* outperforms NAMOA*;