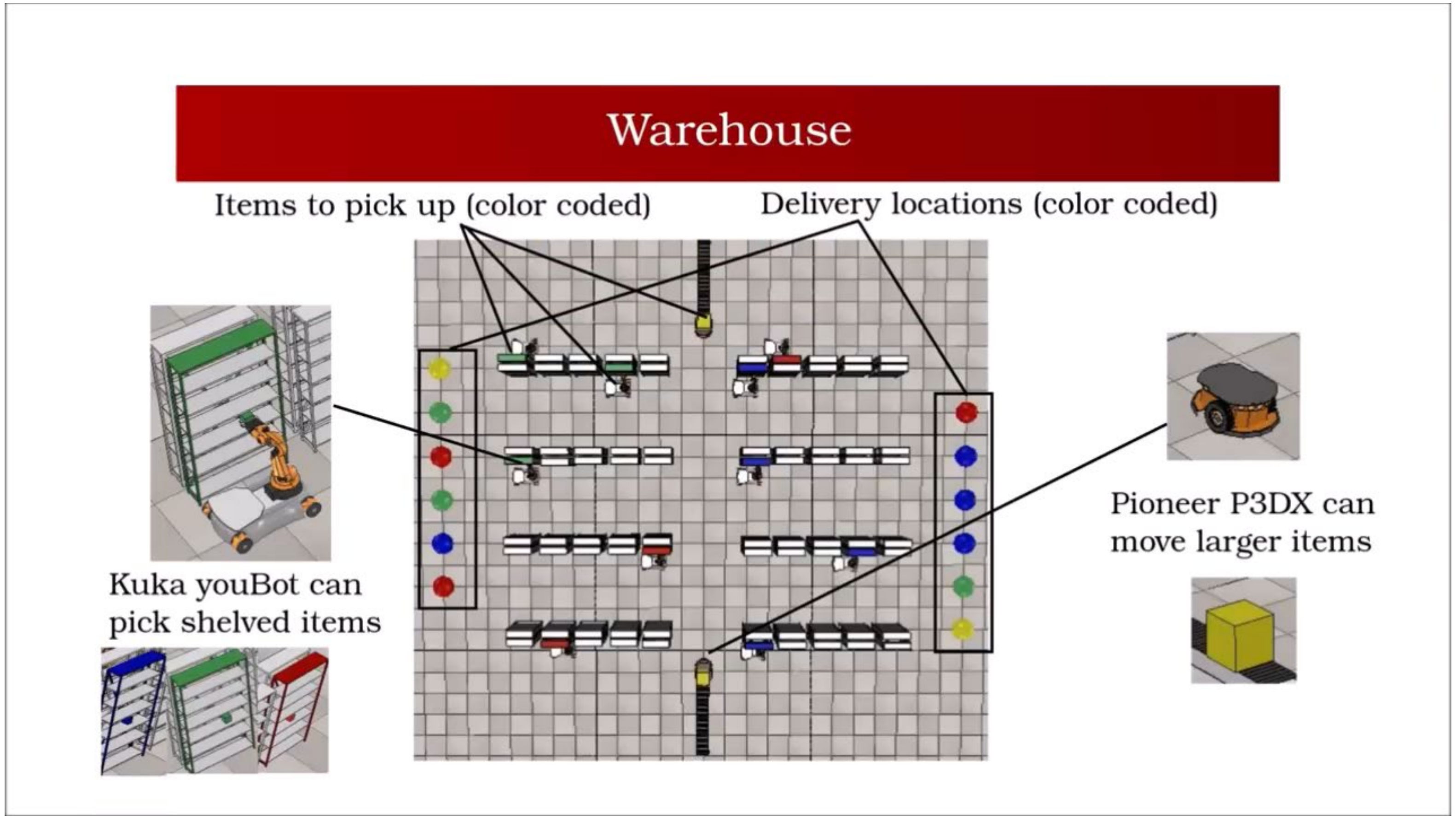
Intelligent Task and Path Planning for Teams of Agents

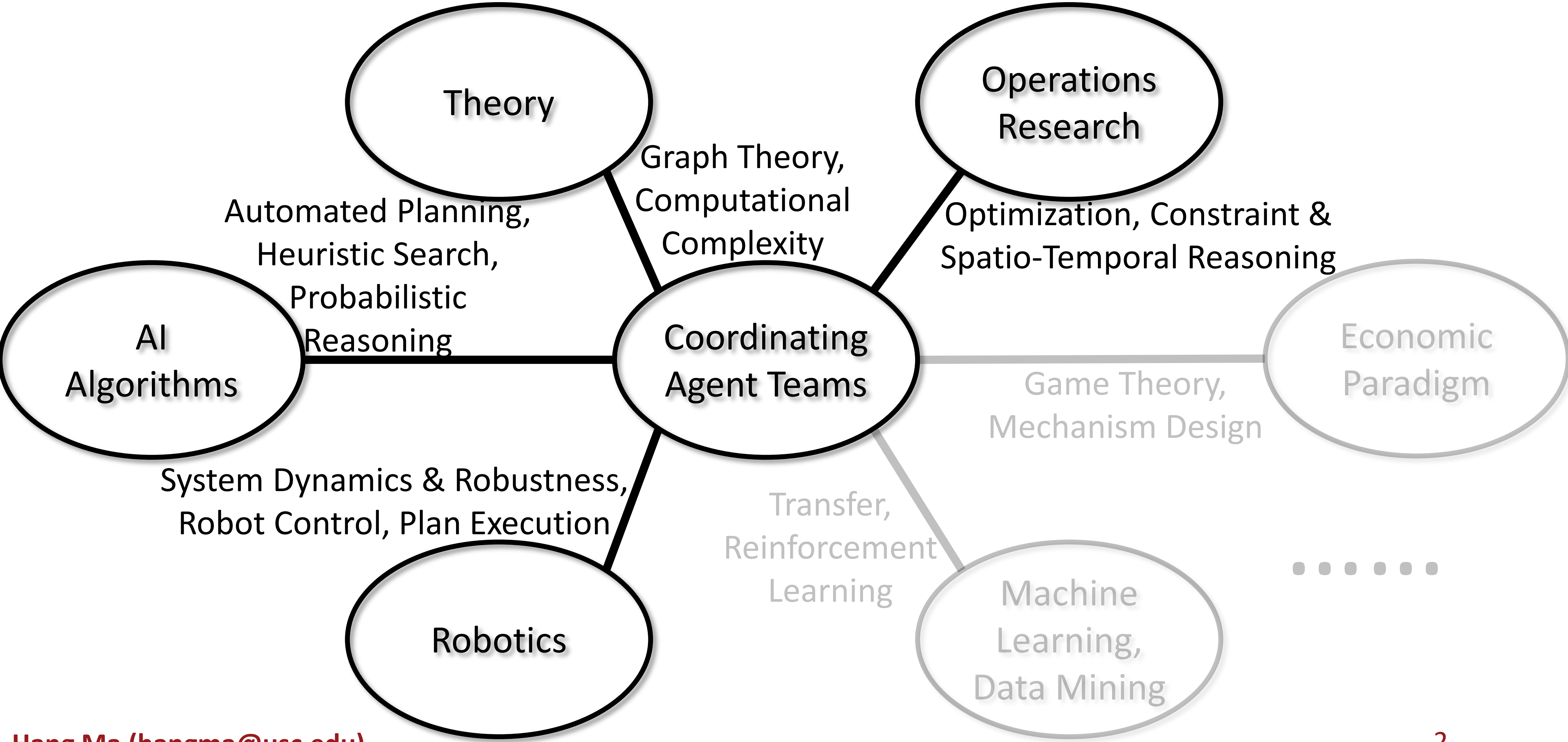
Hang Ma

University of Southern California



USC Viterbi School of Engineering

Research Directions



Story: Towing Vehicles

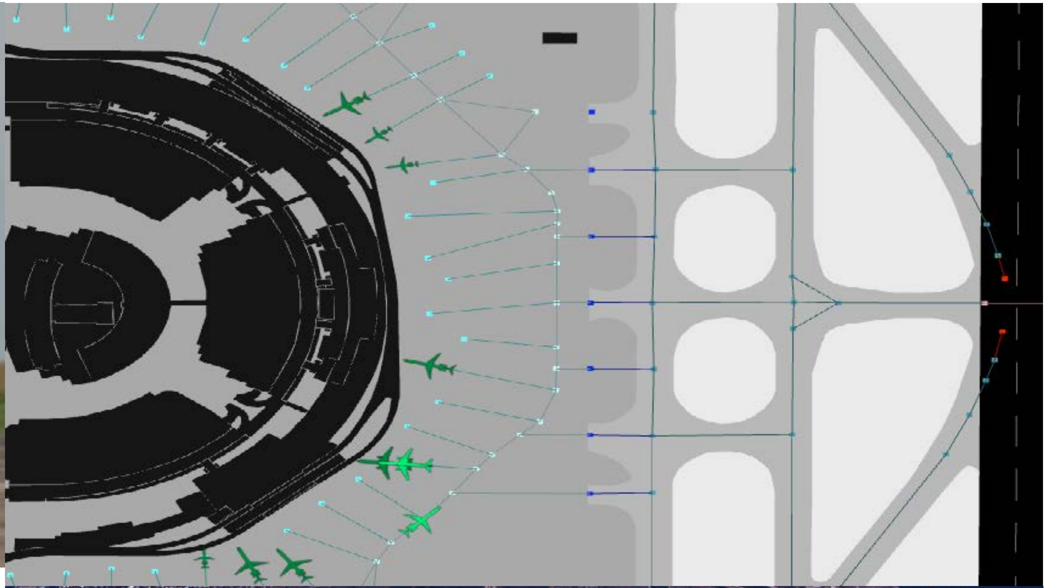
- China Southern 328 (Airbus 380), Oct 2, 2018, LAX

Story: Towing Vehicles

- China Southern 328 (Airbus 380), Nov 11, 2016, LAX

Teams of Intelligent Agents are the Future!

Autonomous Towing Vehicles



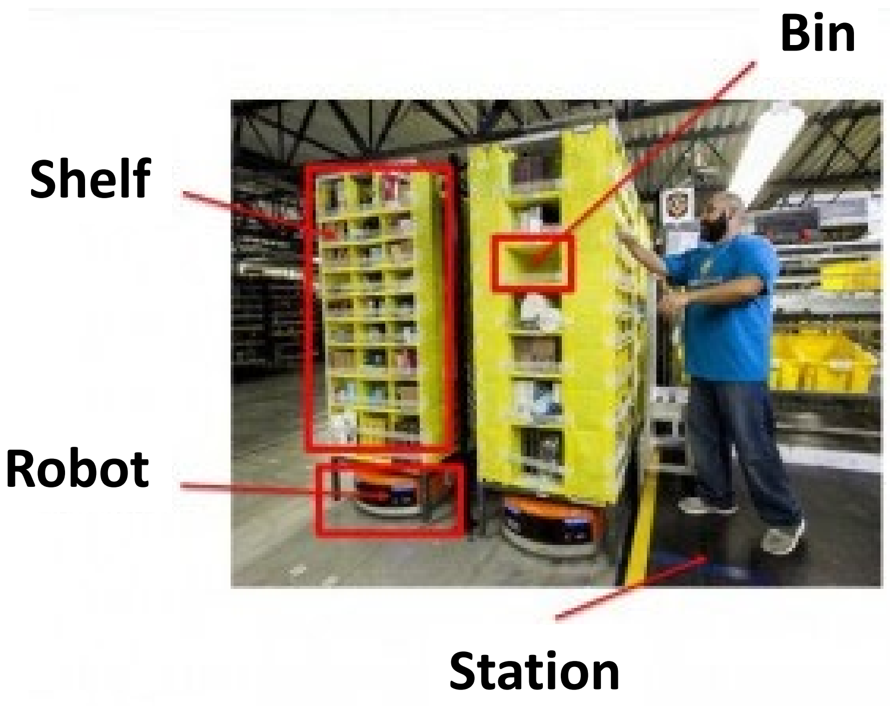
- Reduce pollution
- Reduce energy consumption
- Reduce human workload
- Reduce traffic congestion
- Reduce airport size

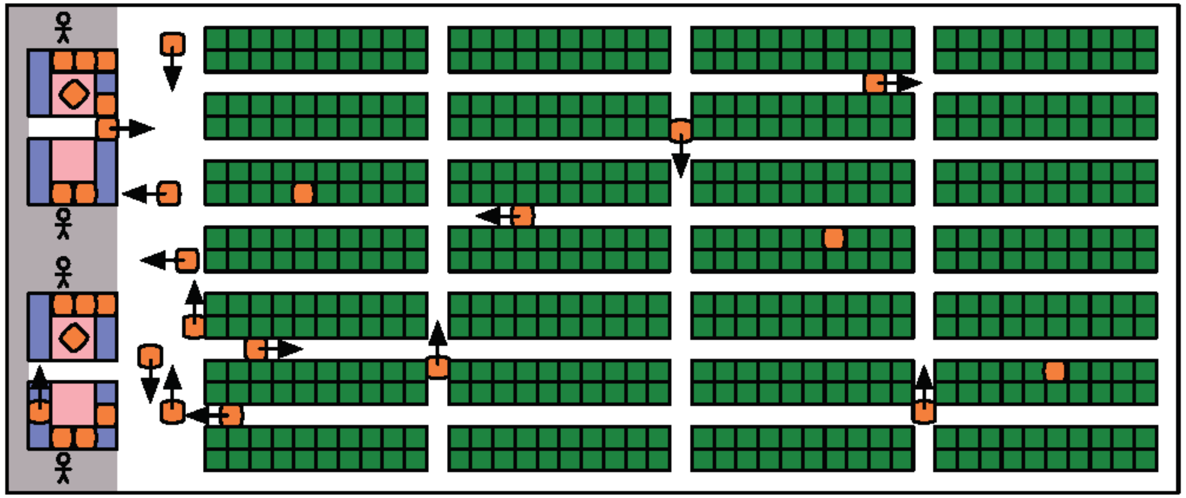
Autonomous engines-off taxiing (joint work with NASA Ames [AAAI PlanHS-16])

R. Morris, C. Pasareanu, K. Luckow, W. Malik, H. Ma, T. K. S. Kumar, and S. Koenig. “Planning, Scheduling and Monitoring for Airport Surface Operations”. AAAI PlanHS. 2016.

Teams of Intelligent Agents are the Future!

Autonomous Warehouse Robots





Goods-to-Person Fulfillment

One worker without robots:

- 28,000 steps, 1,500 items/day

One worker with robots:

- 2,500 steps, 3,000 items/day

Source: Amazon Robotics

Key Research Question

How to design intelligent algorithms that make **fast** and **good** decisions to coordinate these teams of agents?

Fast decisions

- Efficiency
  - Fast computation time

Good decisions

- Effectiveness
  - High throughput
  - Low operating costs
- Robustness
  - No collisions between agents
  - No deadlocks in the long term

Contribution

Increasing System Robustness

Plan Execution with System Dynamics

- **Framework [IEEE IntSys-17]**
- **Kinematic constraints [ICAPS-16] [IROS-16]**
- Uncertainty [AAAI-17]
- Video game environments [AIIDE-17]
- Partial observability [AAAI-15]

Robust Long-Term Task & Path Planning

- **Formalization & algorithms [AAMAS-17]**
- **Kinematic constraints [AAAI-19b]**
- **Offline scheduling [AAMAS-19a]**

Path Planning

- **Theoretical results [AAAI-16]**
- Optimal algorithms [ICAPS-18, 19] [AAAI-19d]
- Suboptimal algorithms [AAAI-19a] [AAMAS-19b]
- Anytime algorithms [IJCAI-18b]
- Overviews [IJCAI WOMPF-16] [AI Matters-17]

Joint Task & Path Planning

- **Optimal task and path planning [AAMAS-16]**
- Tasks with temporal constraints [AAMAS-18] [IJCAI-18a]
- Large-sized agents [AAAI-19c]

Increasing Model Expressivity

Content

Path Planning

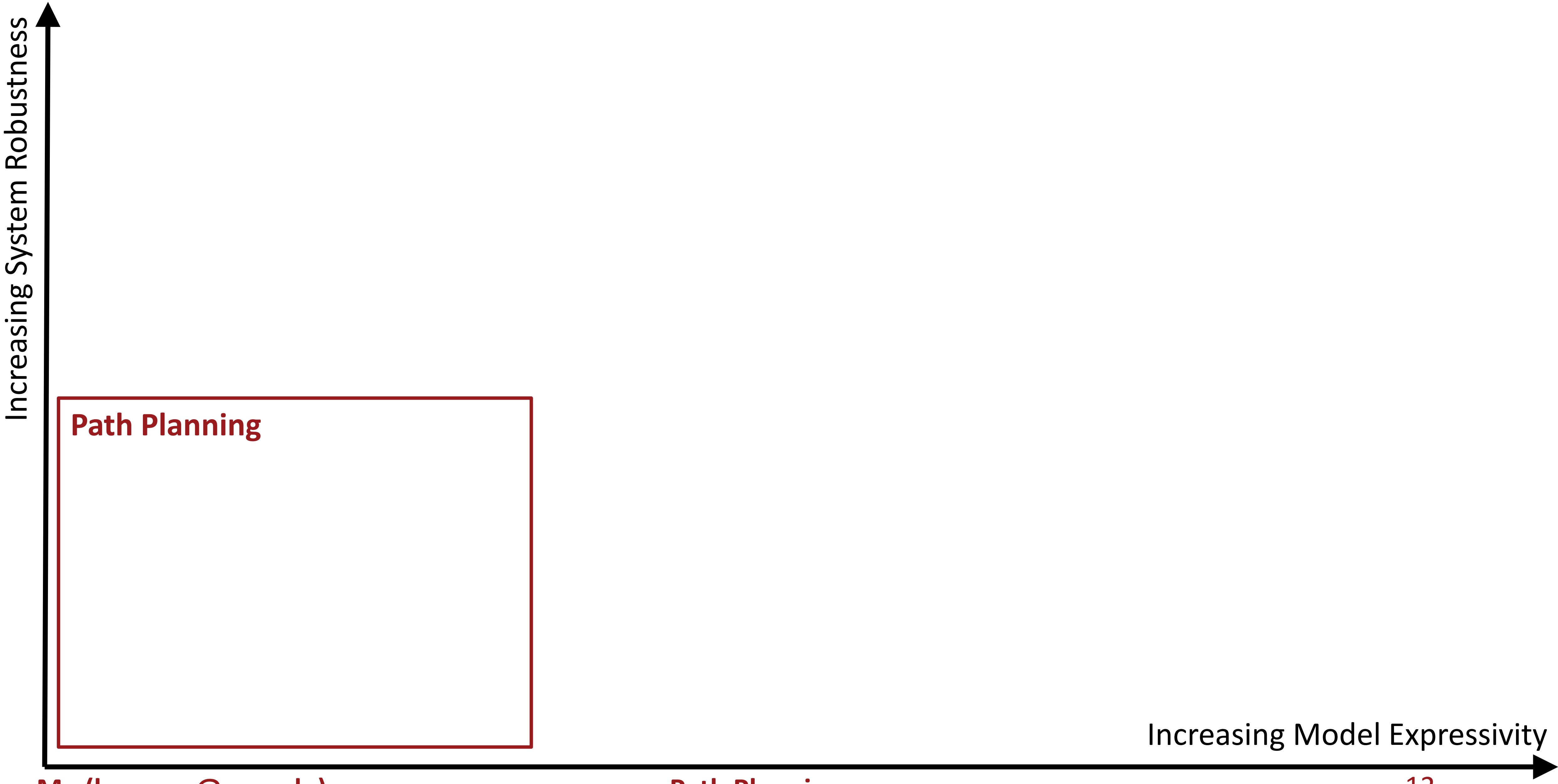
Joint Task & Path Planning

Plan Execution with System Dynamics

Robust Long-Term Task & Path Planning

Future Directions

Contribution



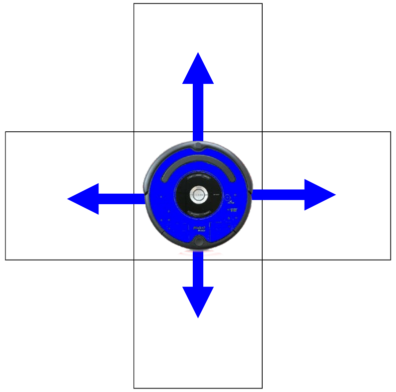
Multi-Agent Path Finding (MAPF)

Problem: Find **collision-free** paths that minimize the **makespan**

Makespan: Earliest time step when all agents have arrived at their goal vertices

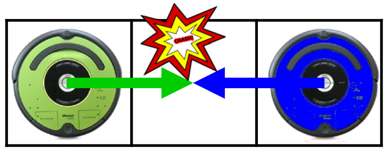
Each time step:

- Agent waits or moves to a neighboring vertex

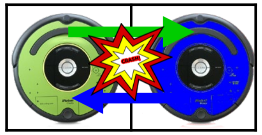


Collisions:

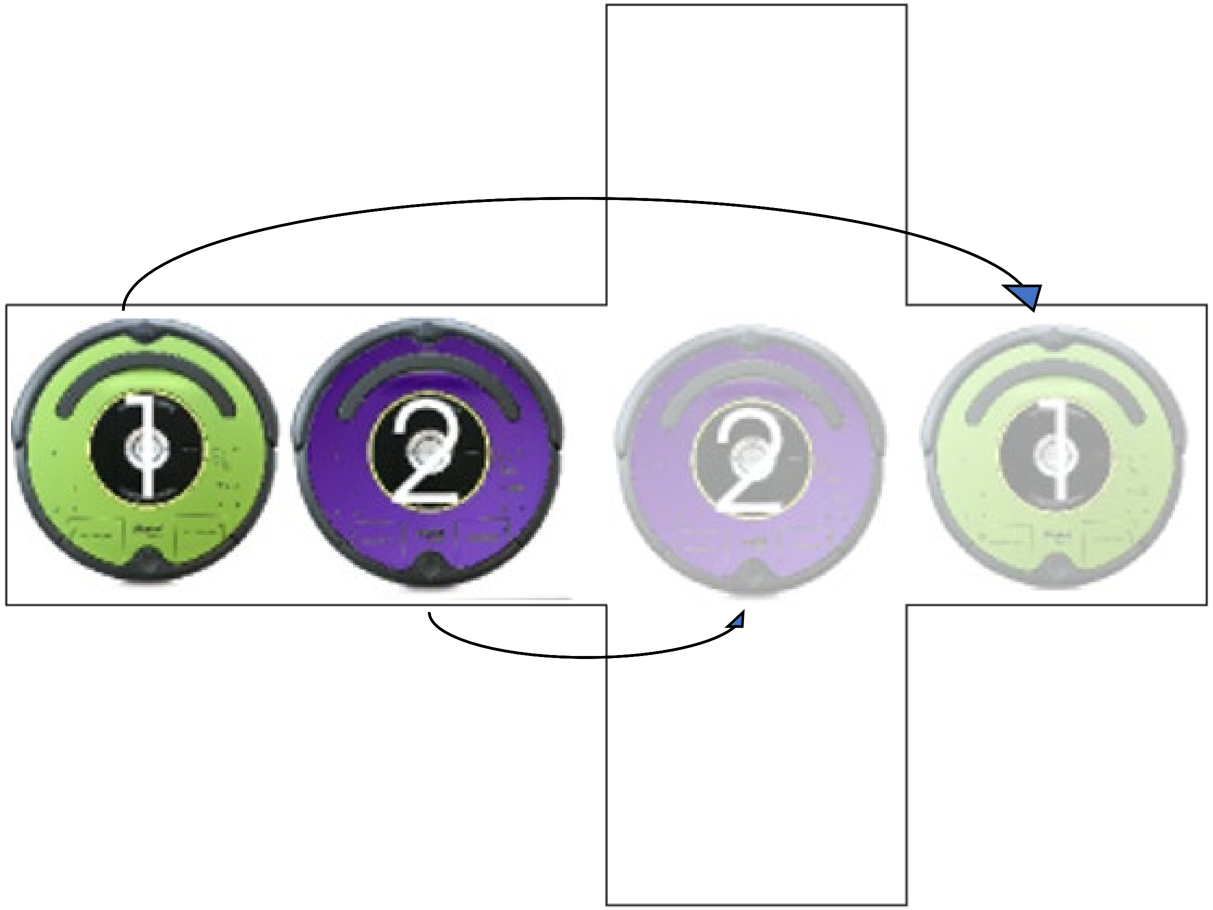
- “Vertex collisions” --- Not allowed



- “Edge collisions” --- Not allowed



MAPF: Example



4-neighbor grid

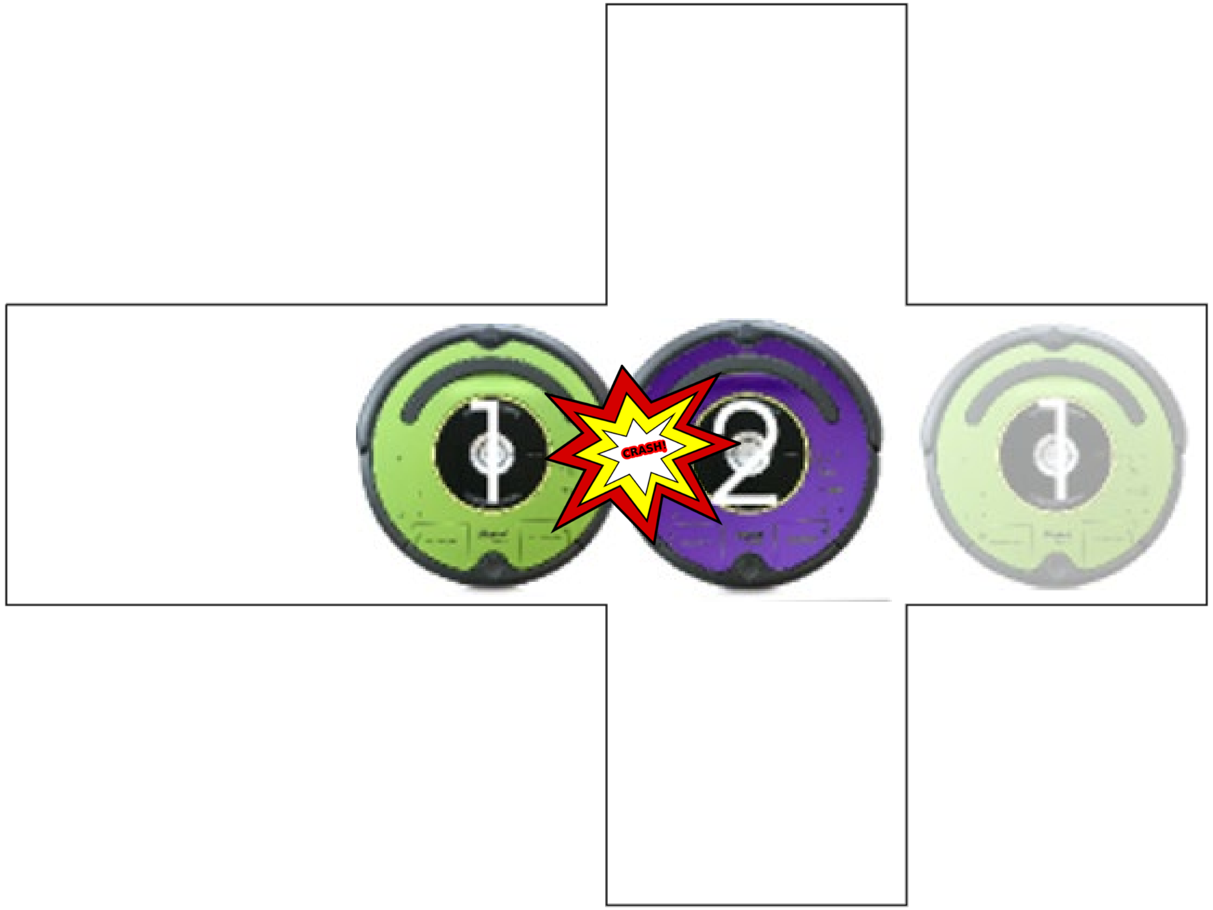
MAPF: Example: Time Step 0

4-neighbor grid

MAPF: Example: Time Step 1

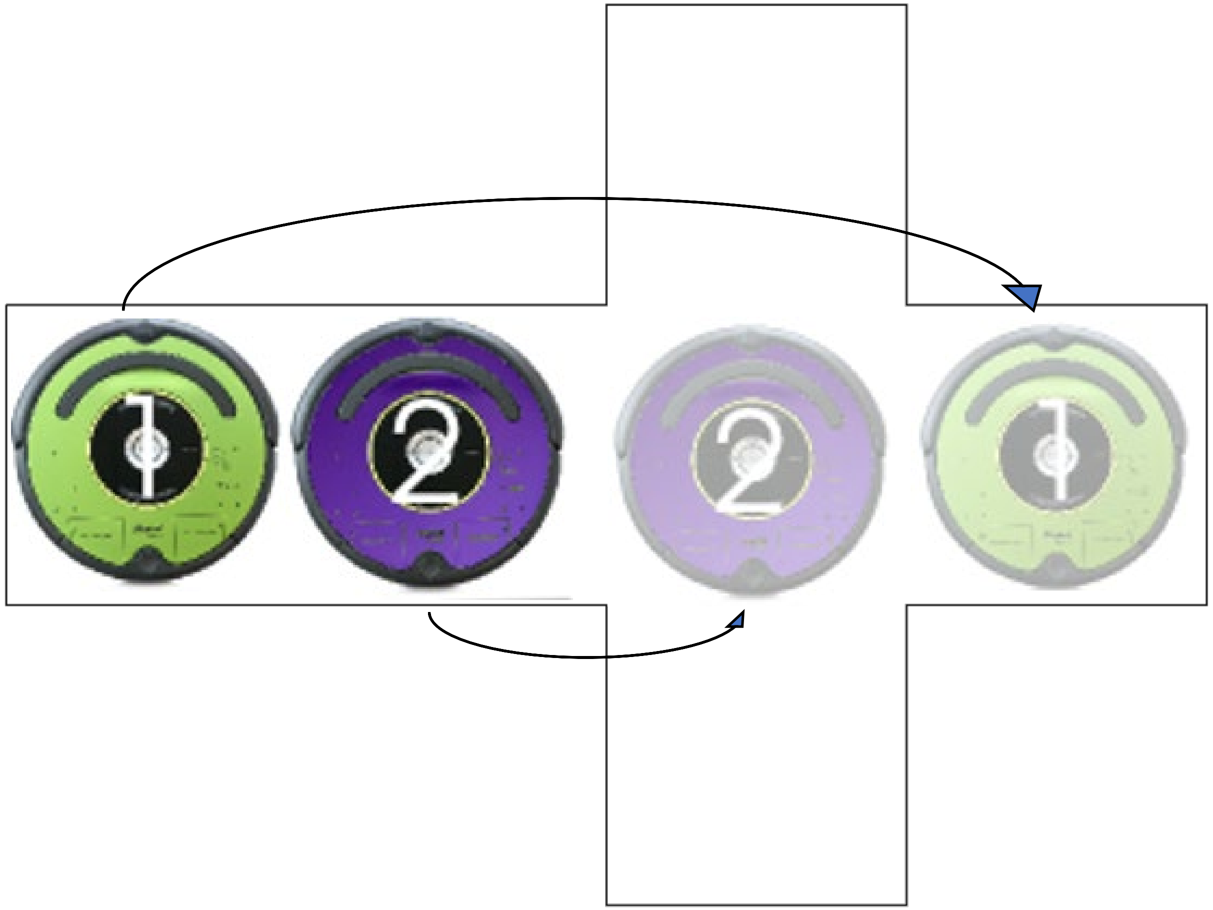
4-neighbor grid

MAPF: Example: No Path



4-neighbor grid

MAPF: Example



4-neighbor grid

MAPF: Example: Time Step 0

4-neighbor grid

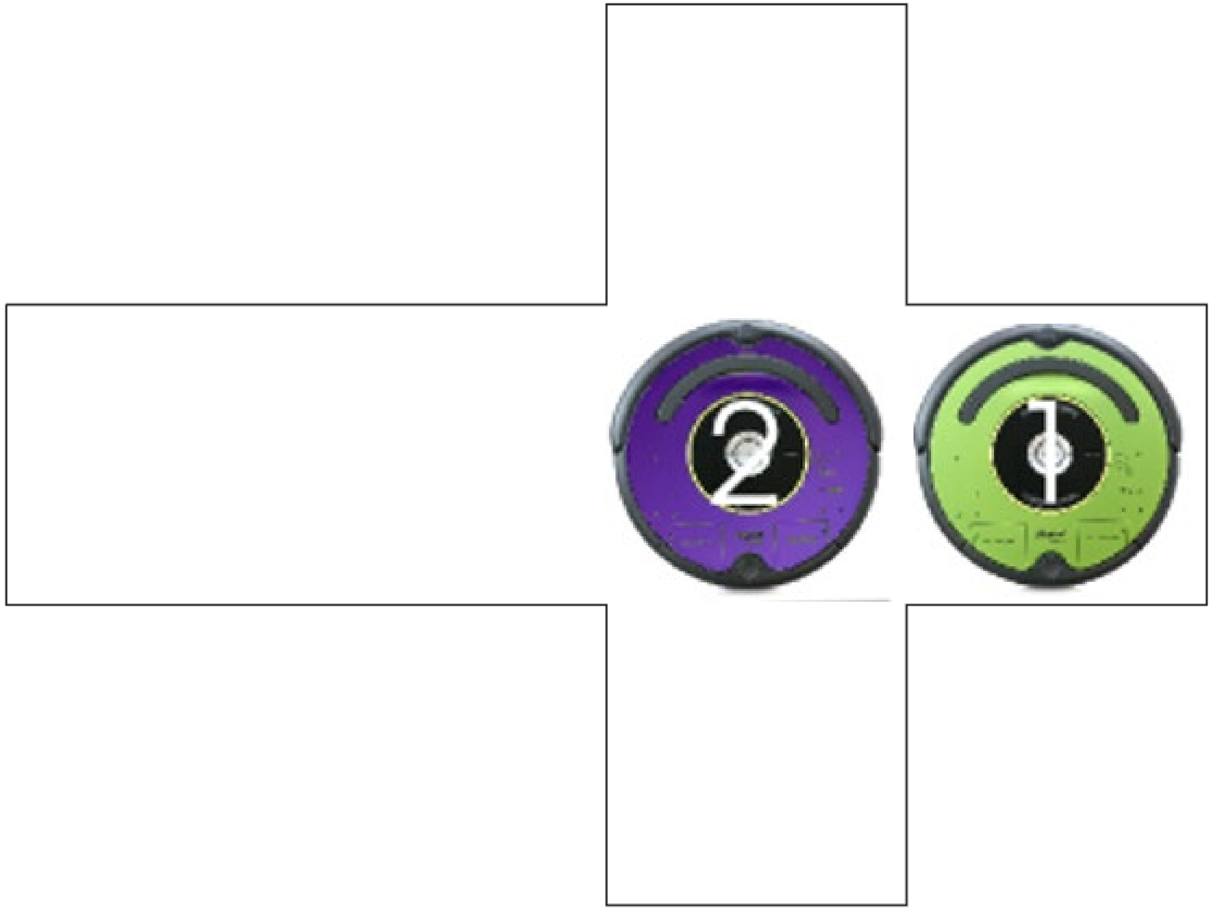
MAPF: Example: Time Step 1

4-neighbor grid

MAPF: Example: Time Step 2

4-neighbor grid

MAPF: Example: Time Step 3



4-neighbor grid

MAPF: Hardness of Approximation [AAAI-16]

Theorem:

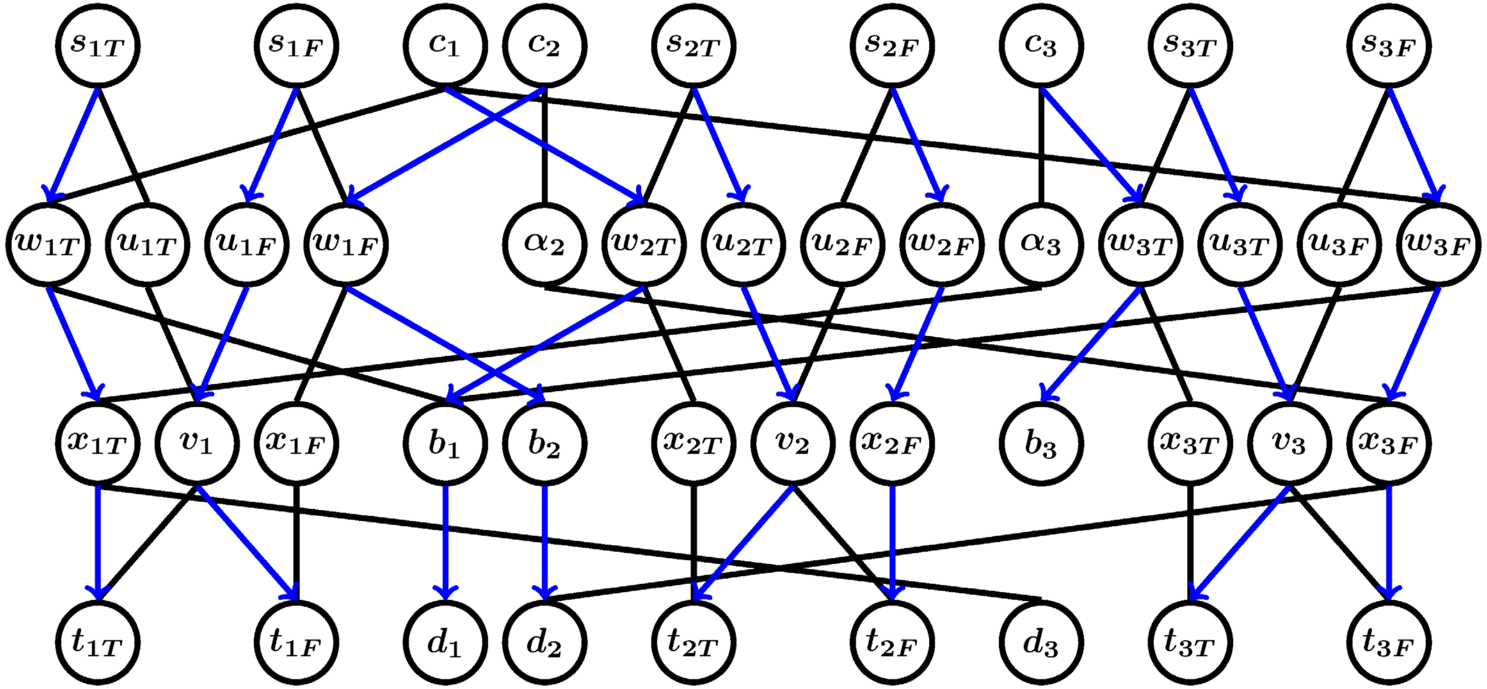
MAPF is NP-hard to approximate within any factor less than 4/3 for makespan minimization

H. Ma, C. Tovey, G. Sharon, T. K. S. Kumar, and S. Koenig. "Multi-Agent Path Finding with Payload Transfers and the Package-Exchange Robot-Routing Problem". AAAI. 2016.

MAPF: Proof

Reduction from (≤3, =3)-SAT (NP-complete)

Example: $(X_1 \vee X_2 \vee \overline{X_3}) \wedge (\overline{X_1} \vee X_2 \vee \overline{X_3}) \wedge (X_1 \vee \overline{X_2} \vee X_3)$

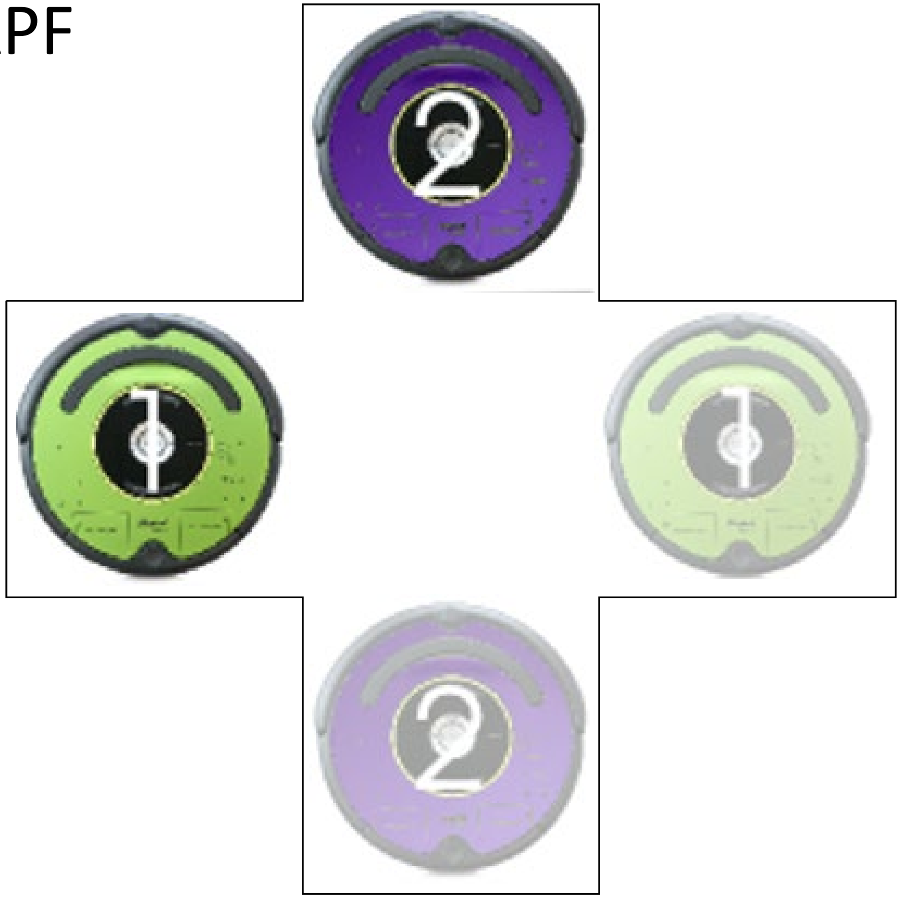


$X_1 \equiv false$ $\quad$ $X_2 \equiv true$ $\quad$ $X_3 \equiv true$

Background: Conflict-Based Search (CBS)

CBS [Sharon et al. 2015]:

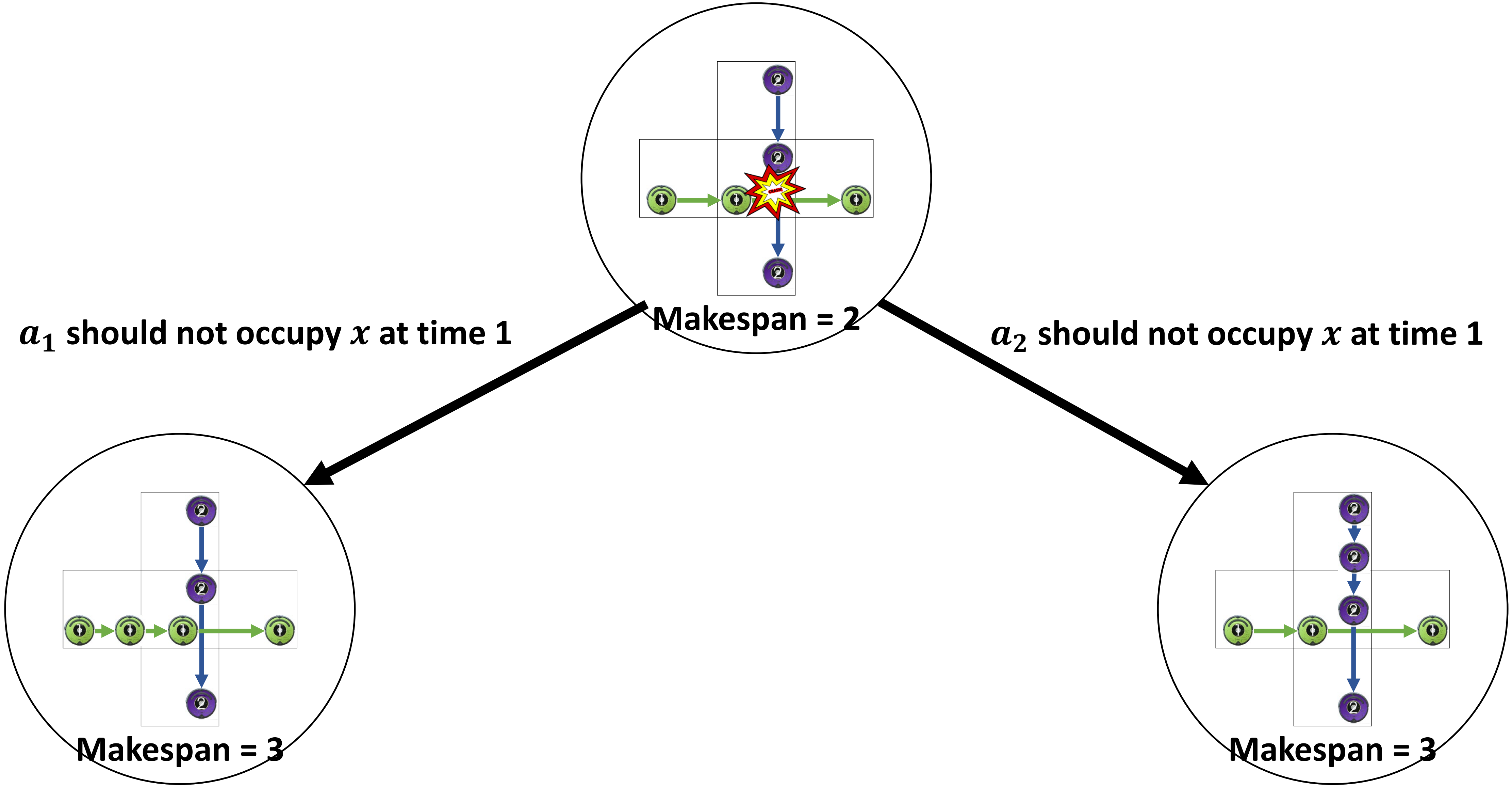
- Complete and optimal for MAPF



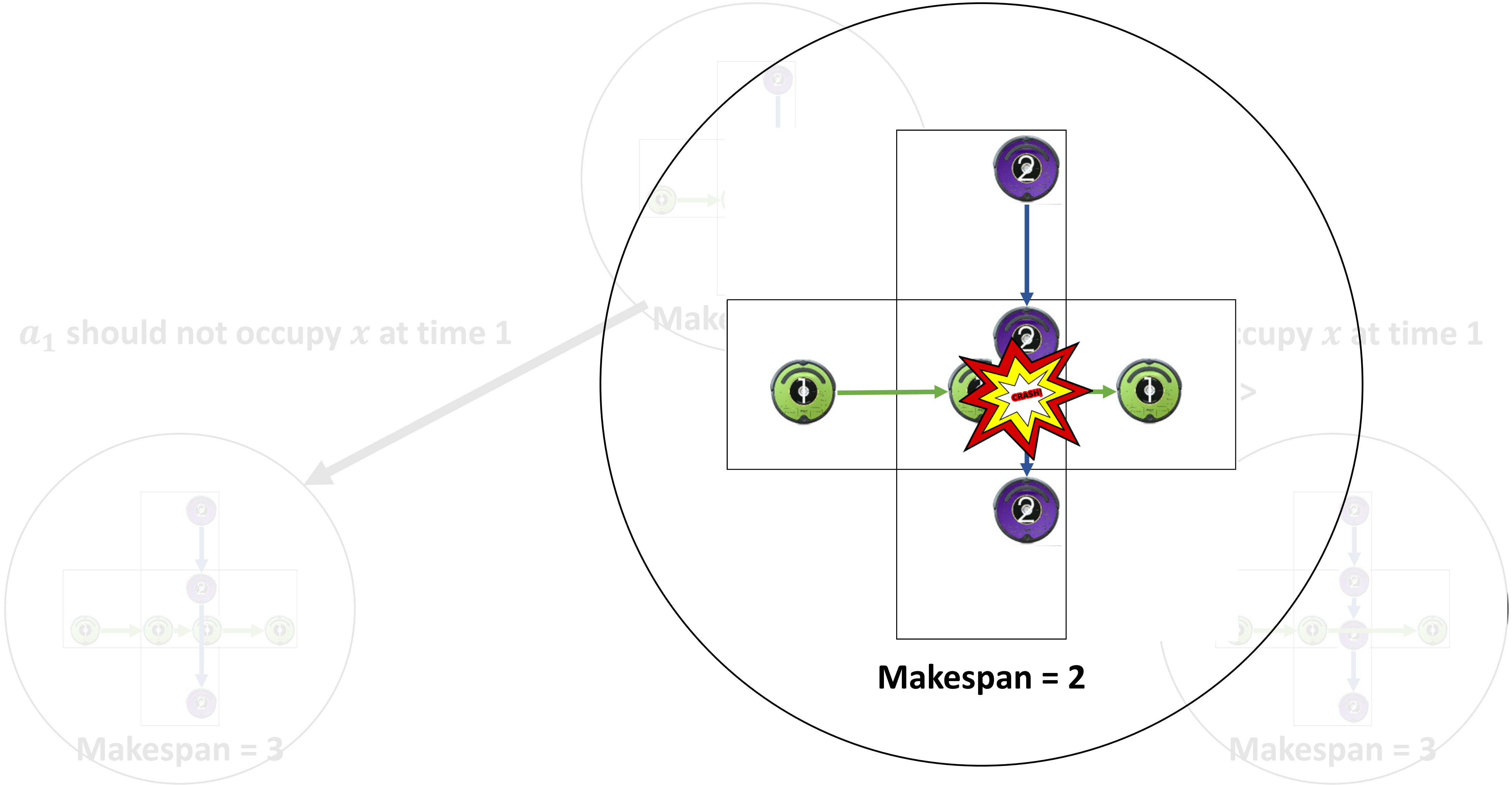
4-neighbor grid

G. Sharon, R. Stern, A. Felner, N. R. Sturtevant. "Conflict-based search for optimal multi-agent pathfinding." Artificial Intelligence 219. pp. 40-66. 2015

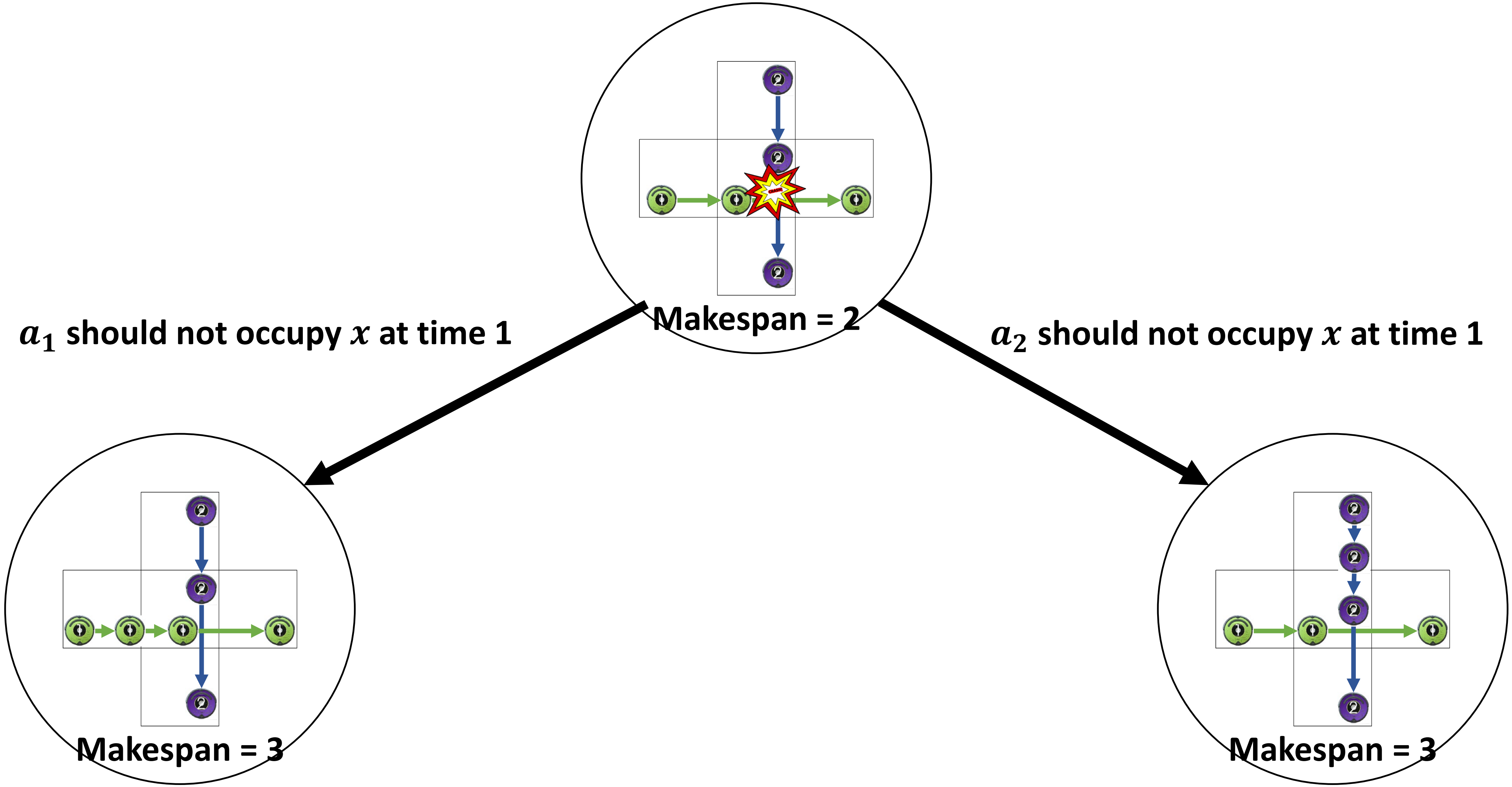
Background: Conflict-Based Search (CBS)



Background: Conflict-Based Search (CBS)



Background: Conflict-Based Search (CBS)

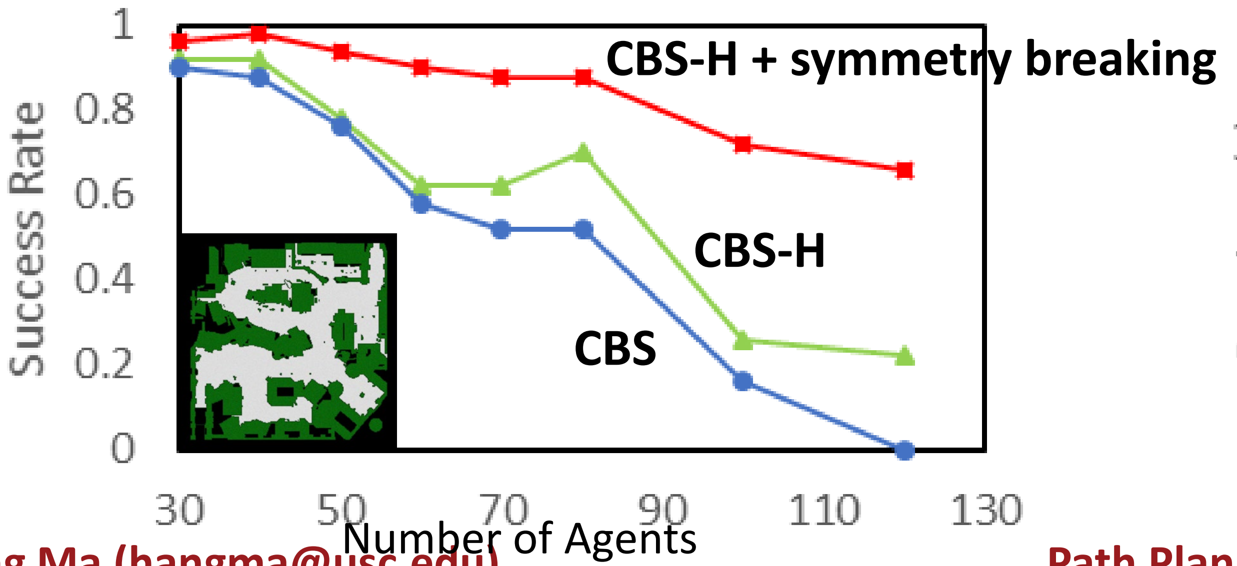


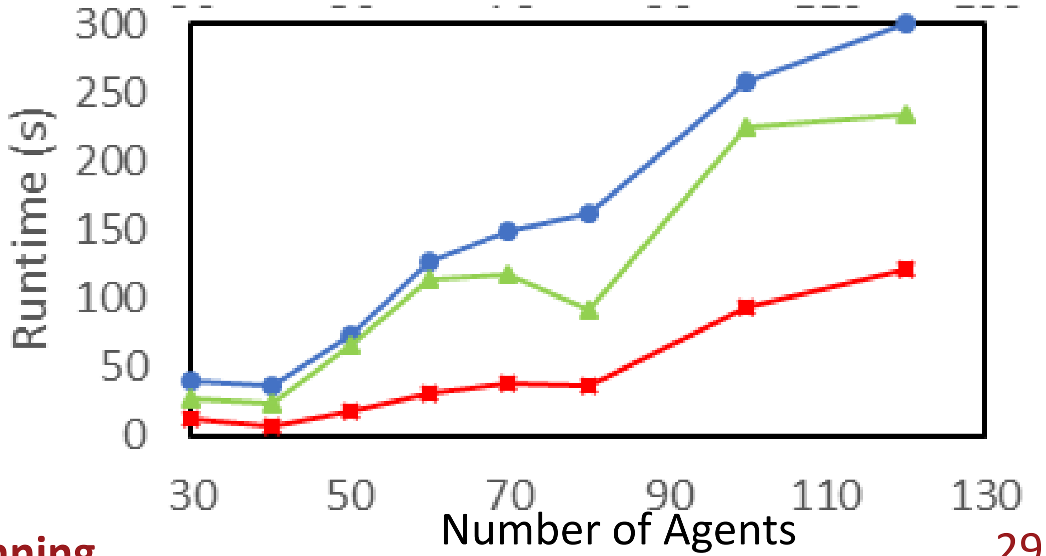
Scaling Up MAPF Algorithms

CBS-based optimal algorithms:

- **CBS-H (heuristic)** [ICAPS-18]: Admissible heuristic guidance for high level
  - Up to 5 times faster than CBS
- **CBS-H+ symmetry breaking** [AAAI-19d]: Add multiple constraints at a time
  - Up to 3 orders of magnitude faster than CBS on 2D grids
- **CBS-H + disjoint node splitting** [ICAPS-19]: Better way of expanding nodes
  - Up to 2 orders of magnitude faster than CBS

Experiments: 5-minute time limits





Scaling Up MAPF Algorithms

Anytime bounded-suboptimal algorithms:

- Anytime FOCAL search [IJCAI-18b]
  - Complete and bounded-suboptimal

Suboptimal algorithms:

- **Prioritized planning + greedy depth-first search version of CBS** [AAAI-19a]
  - Complete only for a subset of instances
  - Almost always return close-to-optimal solutions empirically
  - Average runtime on a 481x530 grid: **0.14 seconds for 100 agents**, **35.18 seconds for 600 agents**
- Constraint reasoning + rapid random restart [AAMAS-19]

Content

Path Planning

Joint Task & Path Planning

Plan Execution with System Dynamics

Robust Long-Term Task & Path Planning

Future Directions

Contribution

Increasing System Robustness

Path Planning
- **Theoretical results [AAAI-16]**
- Optimal algorithms [ICAPS-18, 19] [AAAI-19d]
- Suboptimal algorithms [AAAI-19a] [AAMAS-19b]
- Anytime algorithms [IJCAI-18b]
- Overviews [IJCAI WOMPF-16] [AI Matters-17]

Joint Task & Path Planning

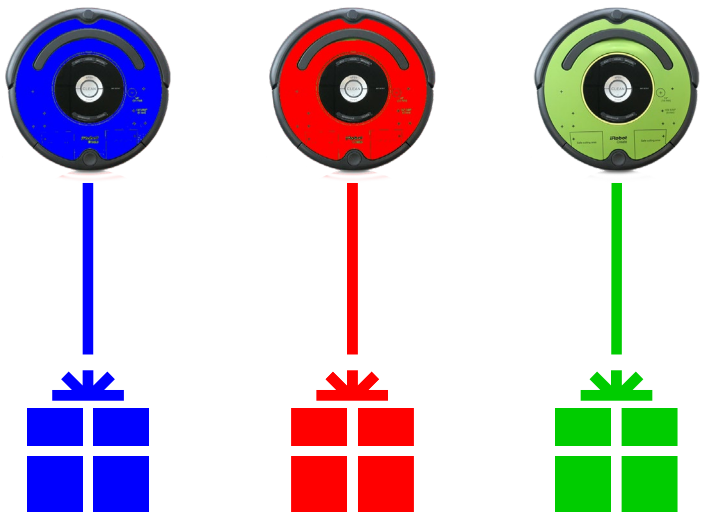
Increasing Model Expressivity

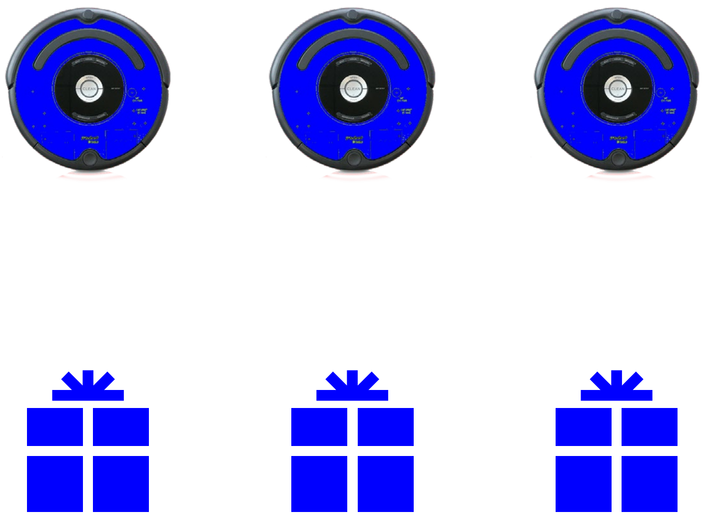
Challenges

"Agents need to decide where to go next"

- The target (task location) for each agent is not given
  - Agents need to be assigned targets
  - Synergy among agents with the same capability could be exploited
  - …
- Searching over all assignments of targets to agents to find optimal solutions results in a large search space (not efficient)
- Assigning targets and planning paths separately is not optimal (not effective)

Anonymous MAPF





Non-anonymous MAPF

NP-hard for makespan minimization

Reducible to integer multi-commodity flow

Solved with CBS

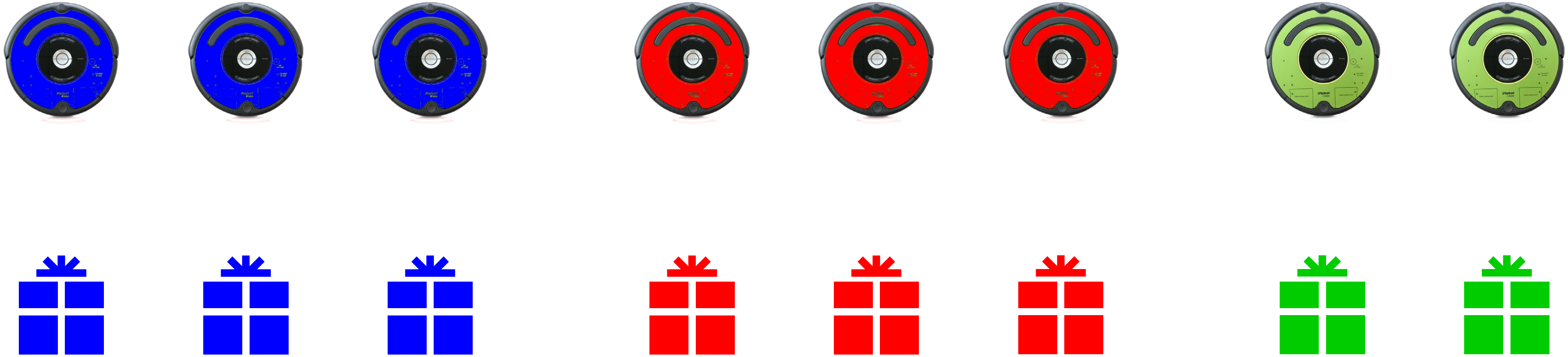
Anonymous MAPF

Polynomial-time solvable for makespan minimization

Reducible to integer single-commodity flow

Solved with max-flow algorithm

Target Assignment and Path Finding [AAMAS-16]

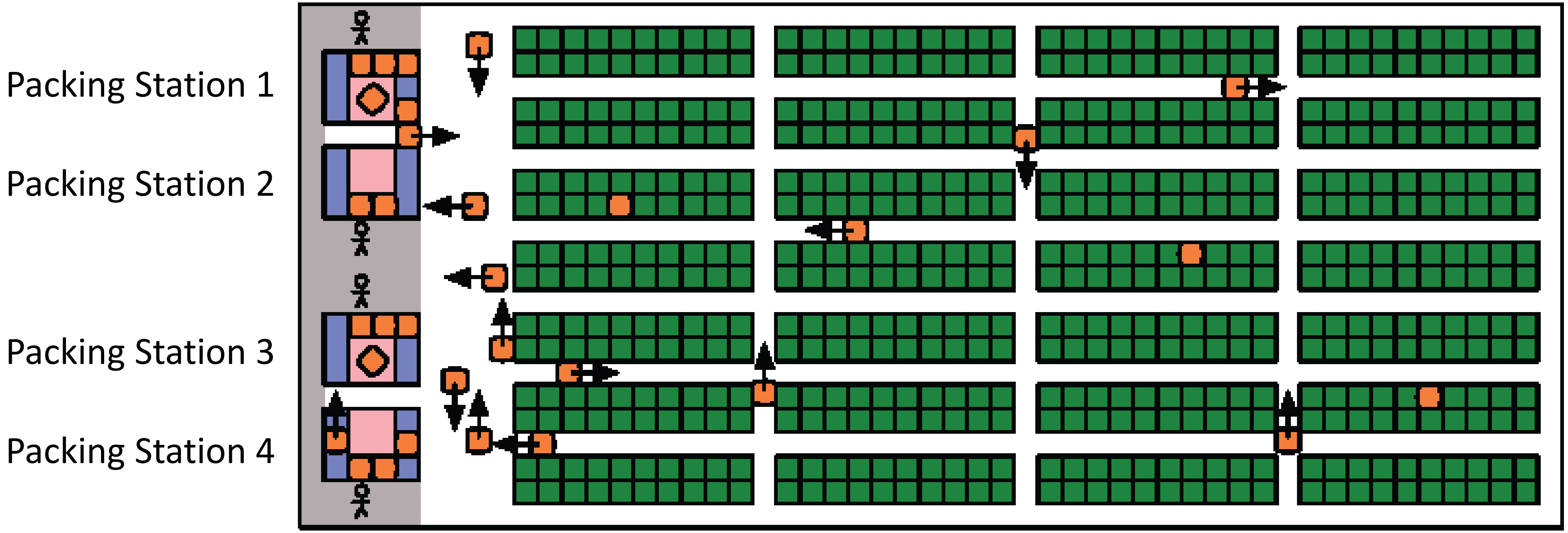


Mix of non-anonymous and anonymous MAPF

Target Assignment and Path Finding (**TAPF**)

H. Ma and S. Koenig. "Optimal Target Assignment and Path Finding for Teams of Agents". AAMAS. 2016.

TAPF: Multiple Teams



Team 0: Agents that move to the storage locations of shelves

Team 1: Agents that move shelves to Packing Station 1

Team 2: Agents that move shelves to Packing Station 2

Team 3: Agents that move shelves to Packing Station 3

TAPF: Hardness of Approximation [AAAI-16]

Theorem:

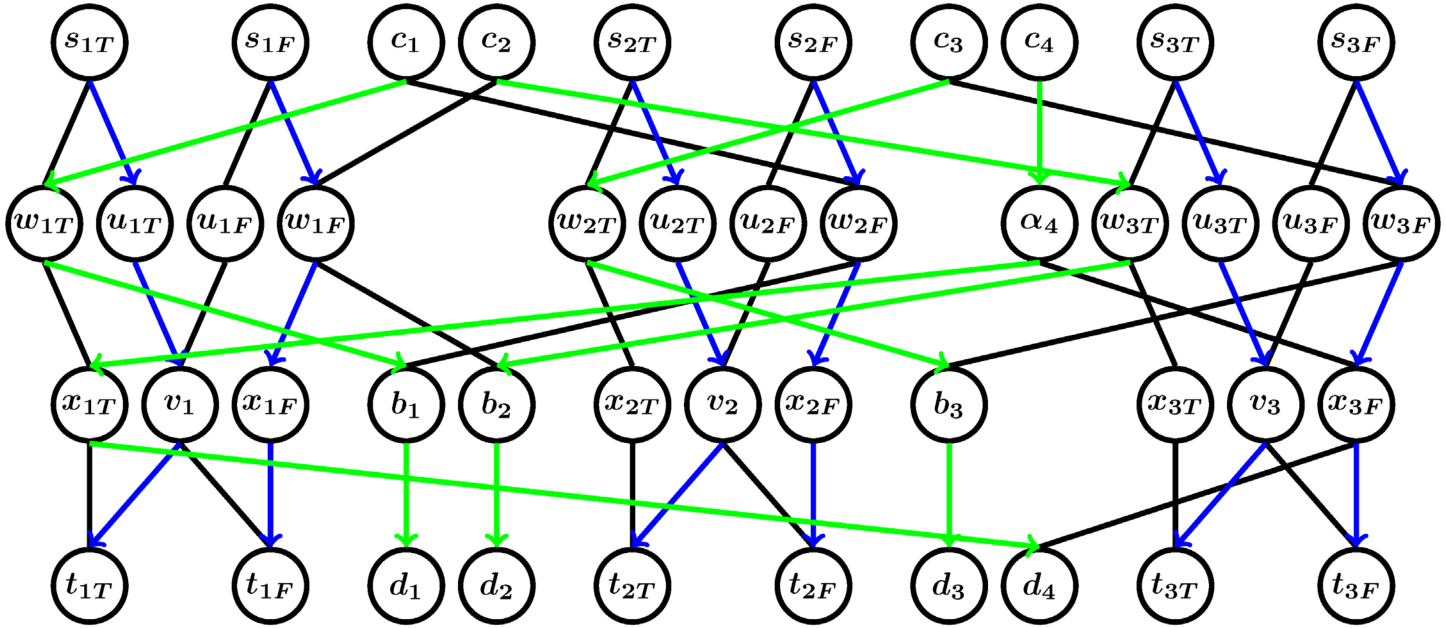
TAPF (with more than one team) is NP-hard to approximate within any factor less than 4/3 for makespan minimization

H. Ma, C. Tovey, G. Sharon, T. K. S. Kumar, and S. Koenig. "Multi-Agent Path Finding with Payload Transfers and the Package-Exchange Robot-Routing Problem". AAAI. 2016.

TAPF: Proof

Reduction from $2/\overline{2}/3$-SAT (NP-complete)

Example: $(X_1 \vee \overline{X_2}) \wedge (\overline{X_1} \vee X_3) \wedge (X_2 \vee \overline{X_3}) \wedge (X_1 \vee X_2 \vee \overline{X_3})$



$X_1 \equiv true$ $\quad$ $X_2 \equiv true$ $\quad$ $X_3 \equiv true$

Conflict-Based Min-Cost Flow (CBM)

How to solve TAPF optimally? Ideas from:

Conflict-Based Search for MAPF (NP-hard)

+

Max-flow algorithm for anonymous MAPF (P)

=

Conflict-Based Min-Cost Flow for TAPF (NP-hard)

CBM: Example

4-neighbor grid

CBM: Example

CBS for MAPF:

- Find path for a single agent (A*)
- Look for **collisions** in paths

If ∃**collision**: $<a_1, a_2, x, t>$

Option 1: constraint $<a_1, x, t>$

Option 2: constraint $<a_2, x, t>$

CBM for TAPF:

Team = Meta-agent

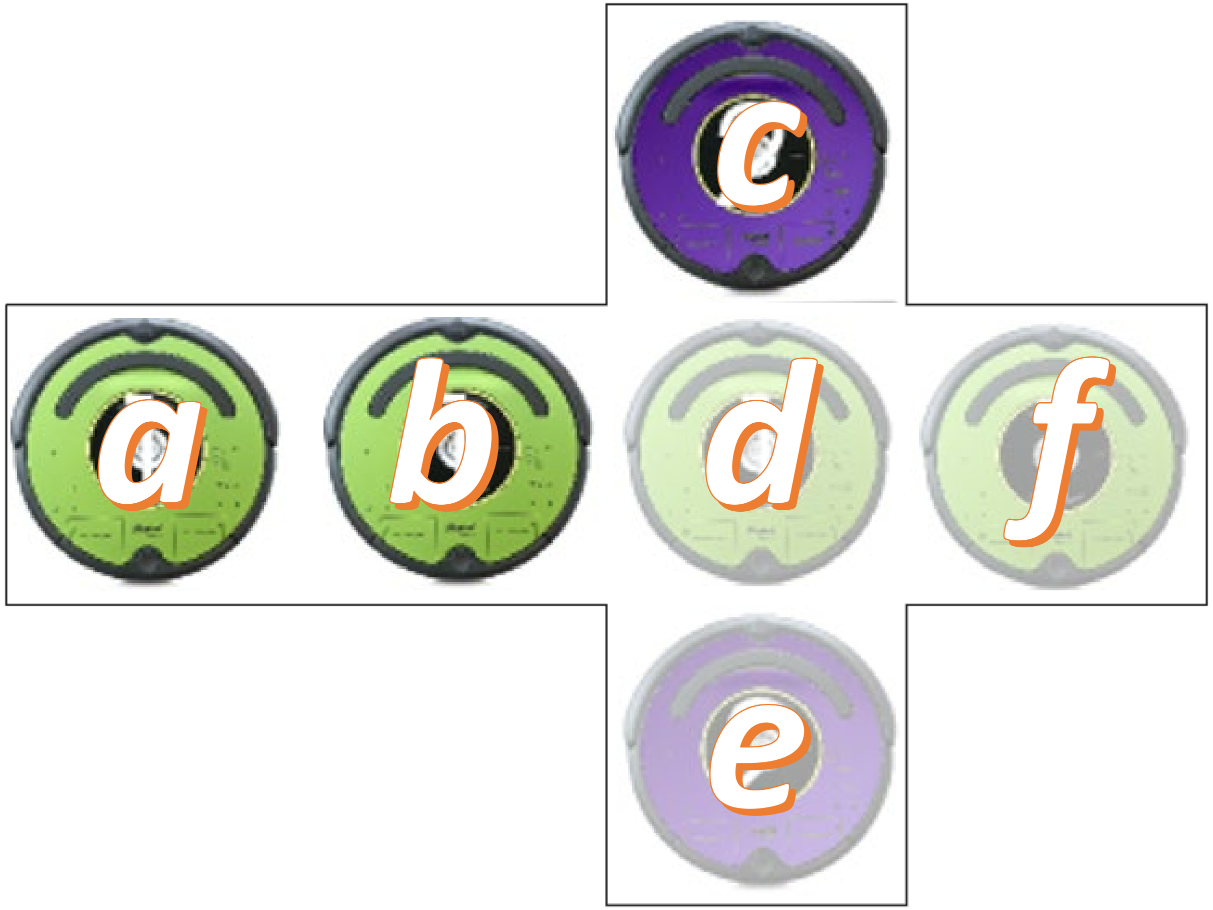
- Find paths for a single team (**maxflow**)
- Look for **collisions** in paths

If ∃**collision**: <*team1, team2, x, t*>

Option 1: constraint <*team1, x, t*>

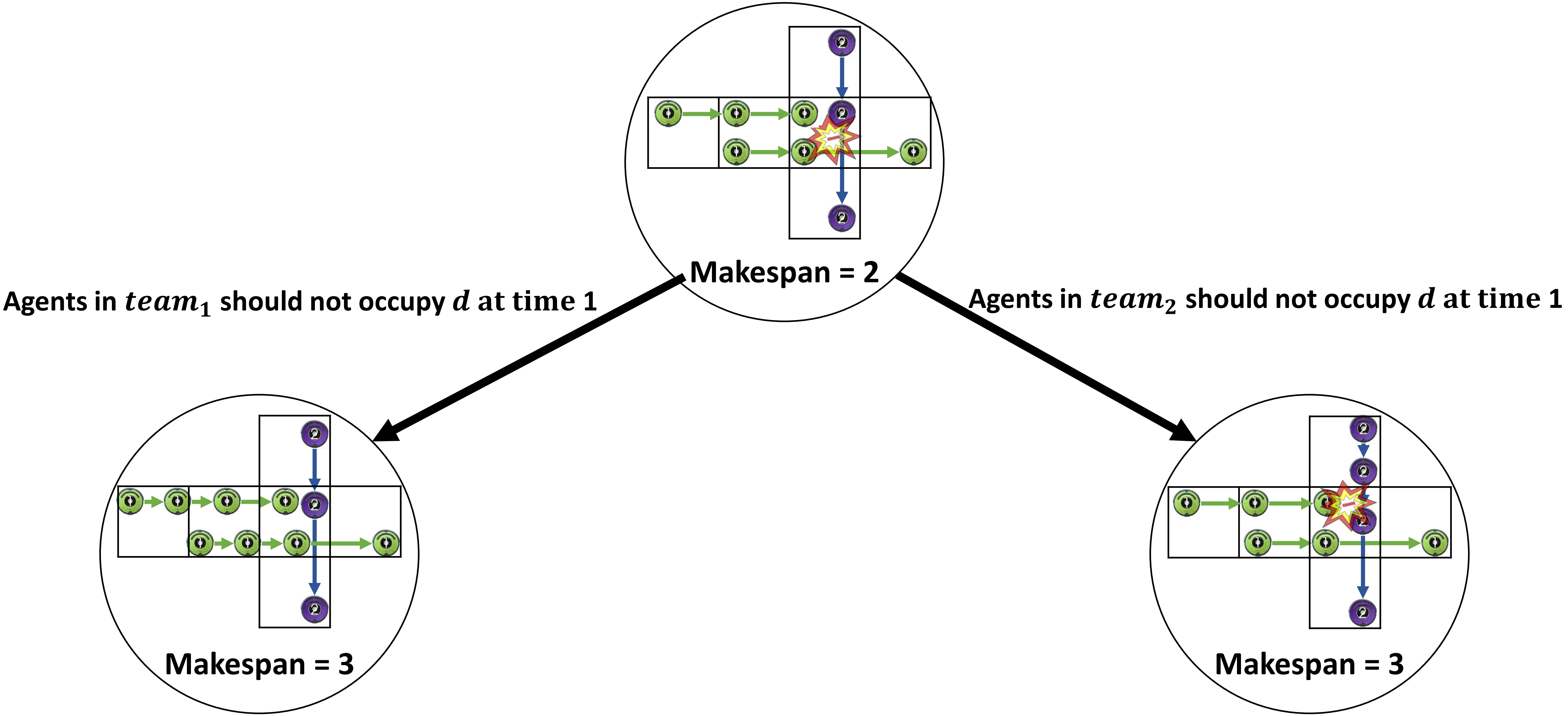
Option 2: constraint <*team1, x, t*>

CBM: Example



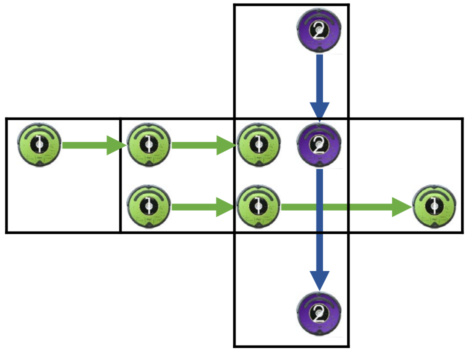
4-neighbor grid

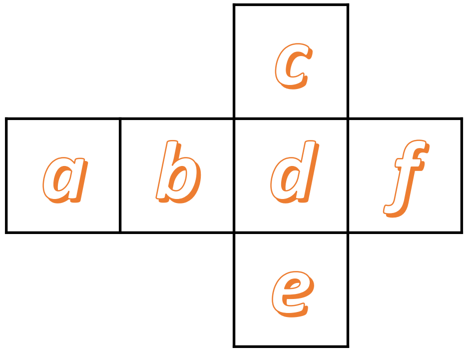
CBM: Example



Makespan = 2
Agents in $team_1$ should not occupy d at time 1
Agents in $team_2$ should not occupy d at time 1
Makespan = 3
Makespan = 3

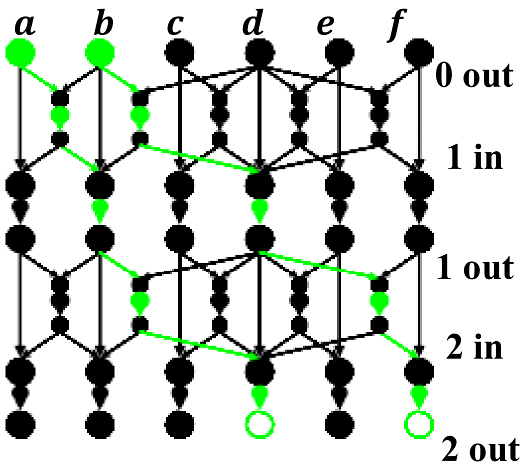
CBM: Find Paths for Single Teams

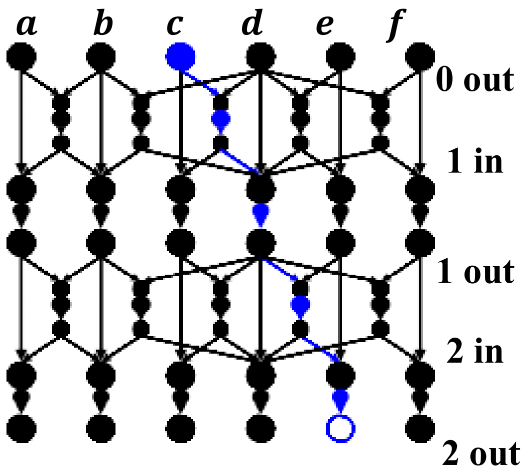




$team_1$
$\langle a, b, d \rangle$
$\langle b, d, f \rangle$

$team_2$
$\langle c, d, e \rangle$



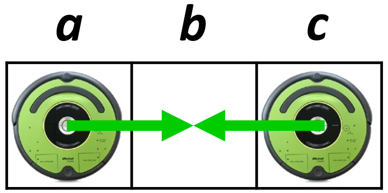


$team_1$ $team_2$

CBM: Gadgets

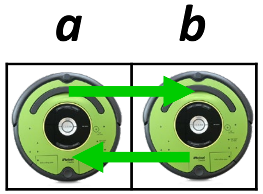
- "Vertex collisions"

--- Not allowed

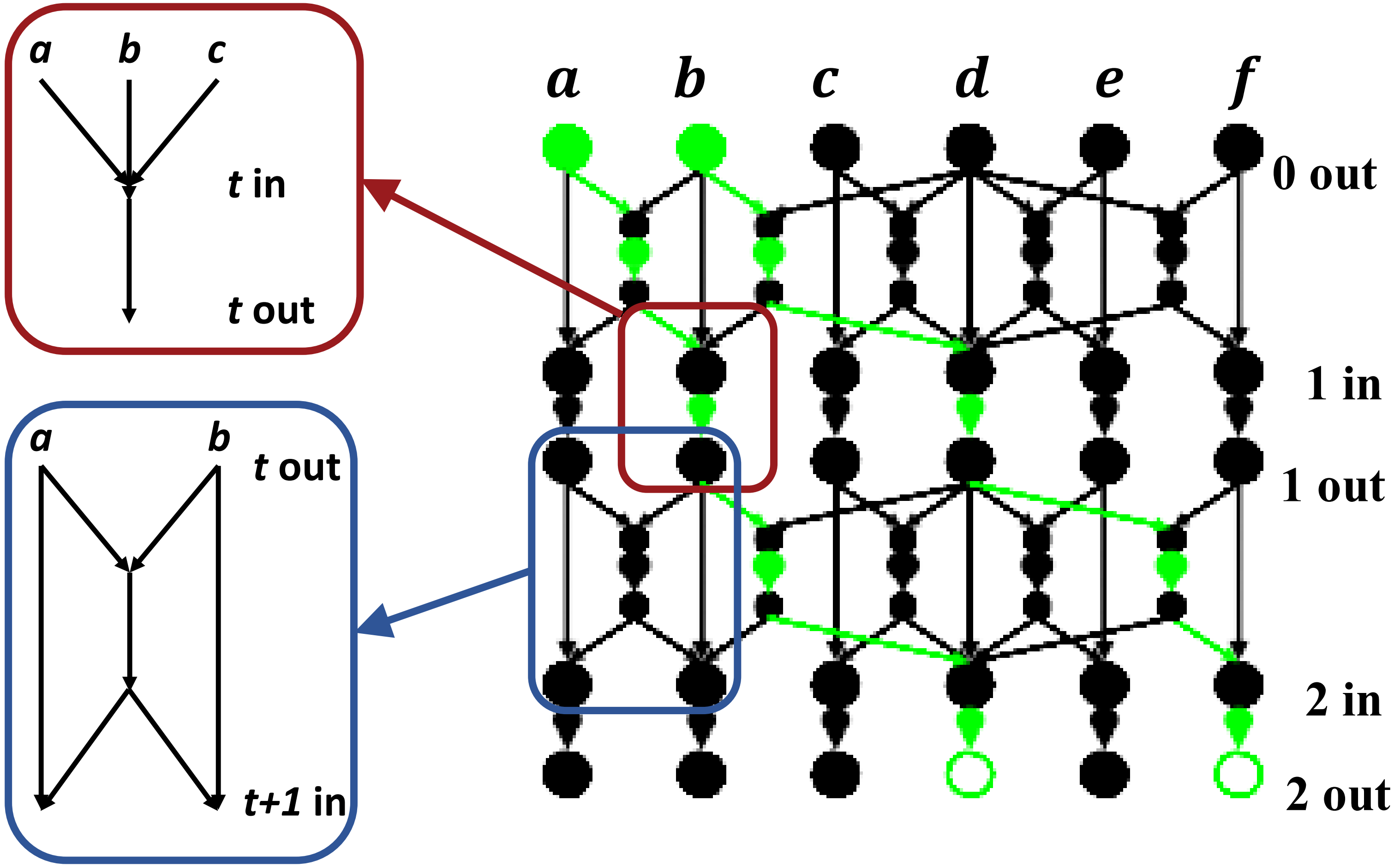


- "Edge collisions"

--- Not allowed

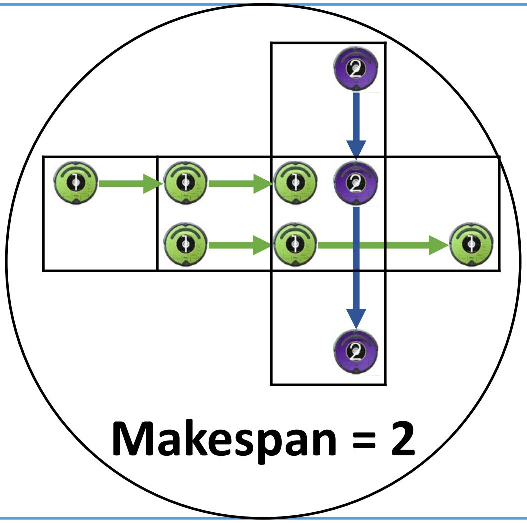


All edges have capacity one



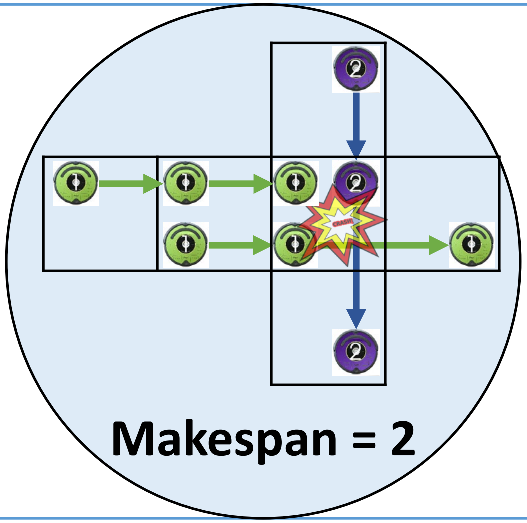
CBM: Store Paths and Makespan

Unexpanded Nodes



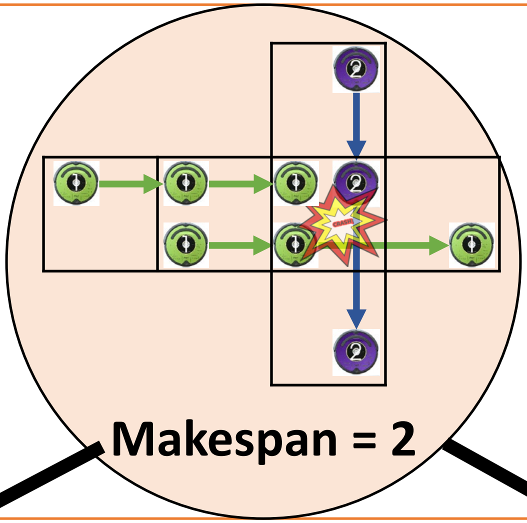
CBM: Pop Node and Look for Collisions

Unexpanded Nodes



CBM: Two Options

Expanded Nodes



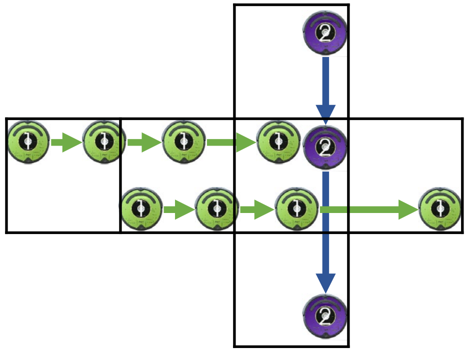
Agents in $team_1$ should not occupy d at time 1

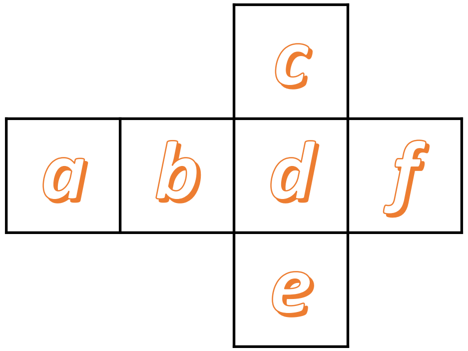
Agents in $team_2$ should not occupy d at time 1

CBM: Option 1: Find New Paths for $team_1$

Constraints

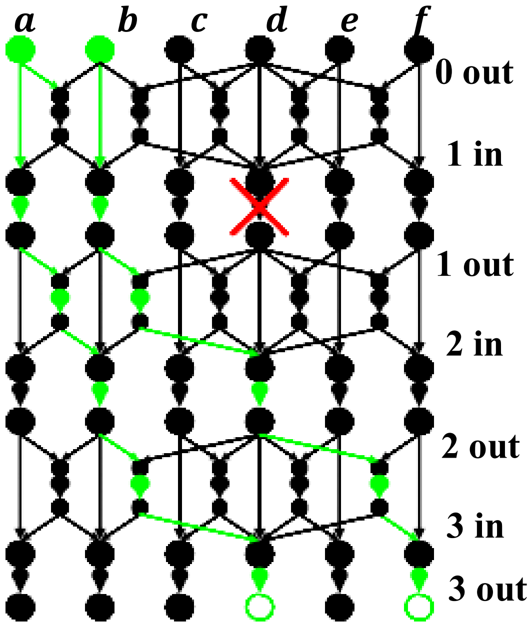
Agents in $team_1$ should not occupy d at time 1



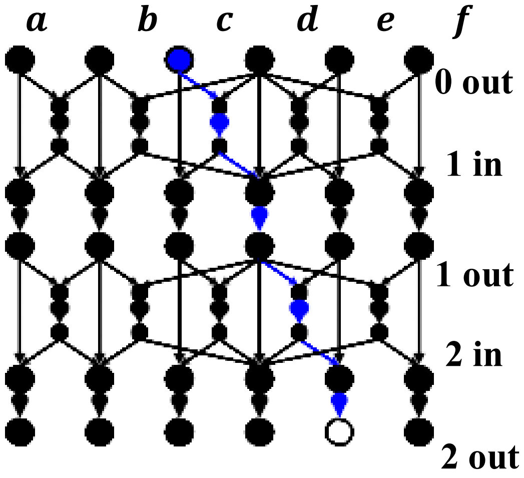


$team_1$
$\langle a, a, b, d \rangle$
$\langle b, b, d, f \rangle$

$team_2$
$\langle c, d, e \rangle$

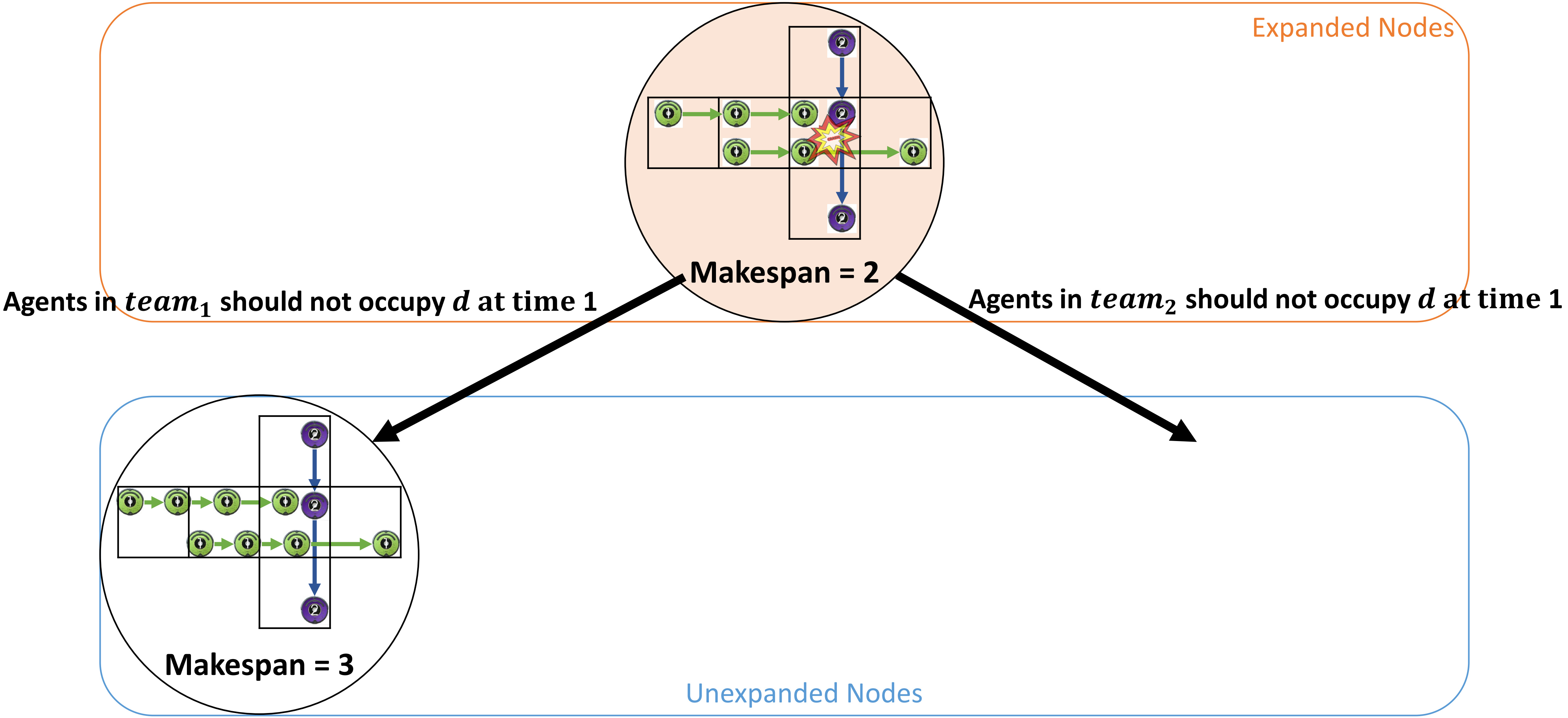


$team_1$



$team_2$

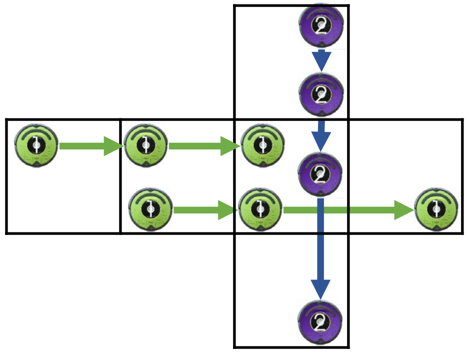
CBM: Store Paths and Makespan

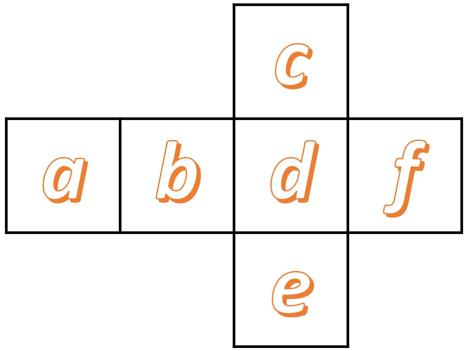


CBM: Option 2: Find New Paths for $team_2$

Constraints

Agents in $team_2$ should not occupy d at time 1





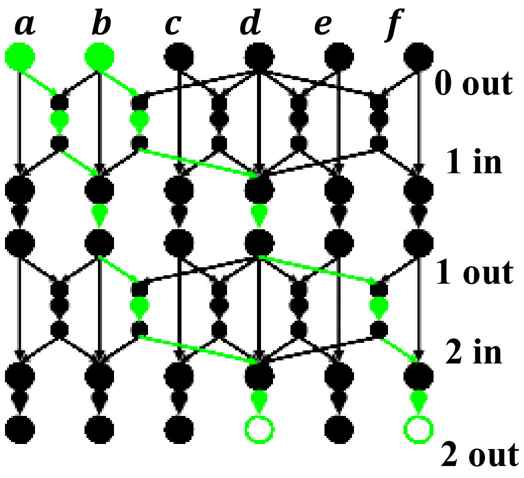
$team_1$

$\langle a, b, d \rangle$

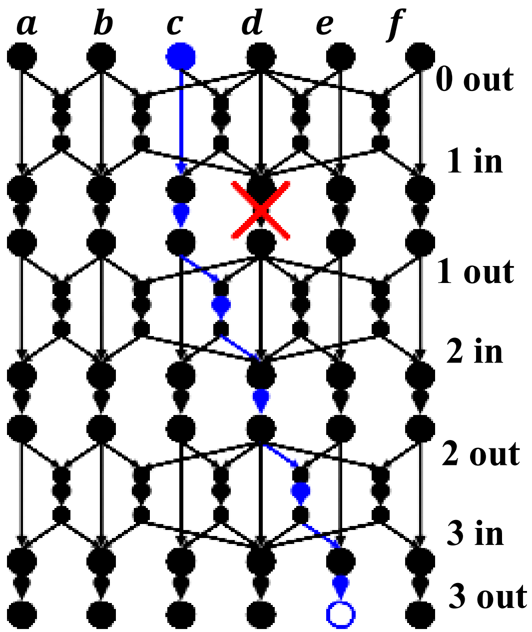
$\langle b, d, f \rangle$

$team_2$

$\langle c, c, d, e \rangle$

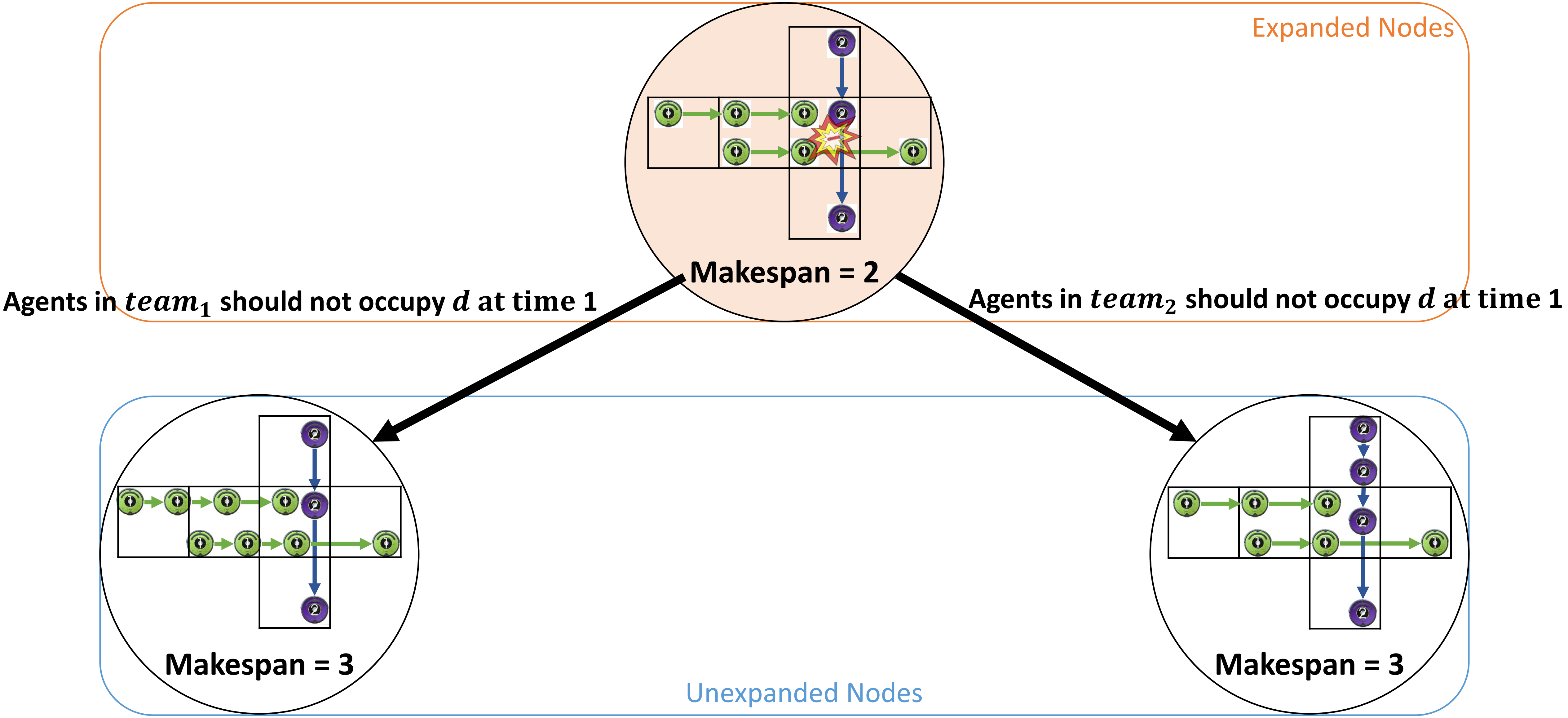


$team_1$

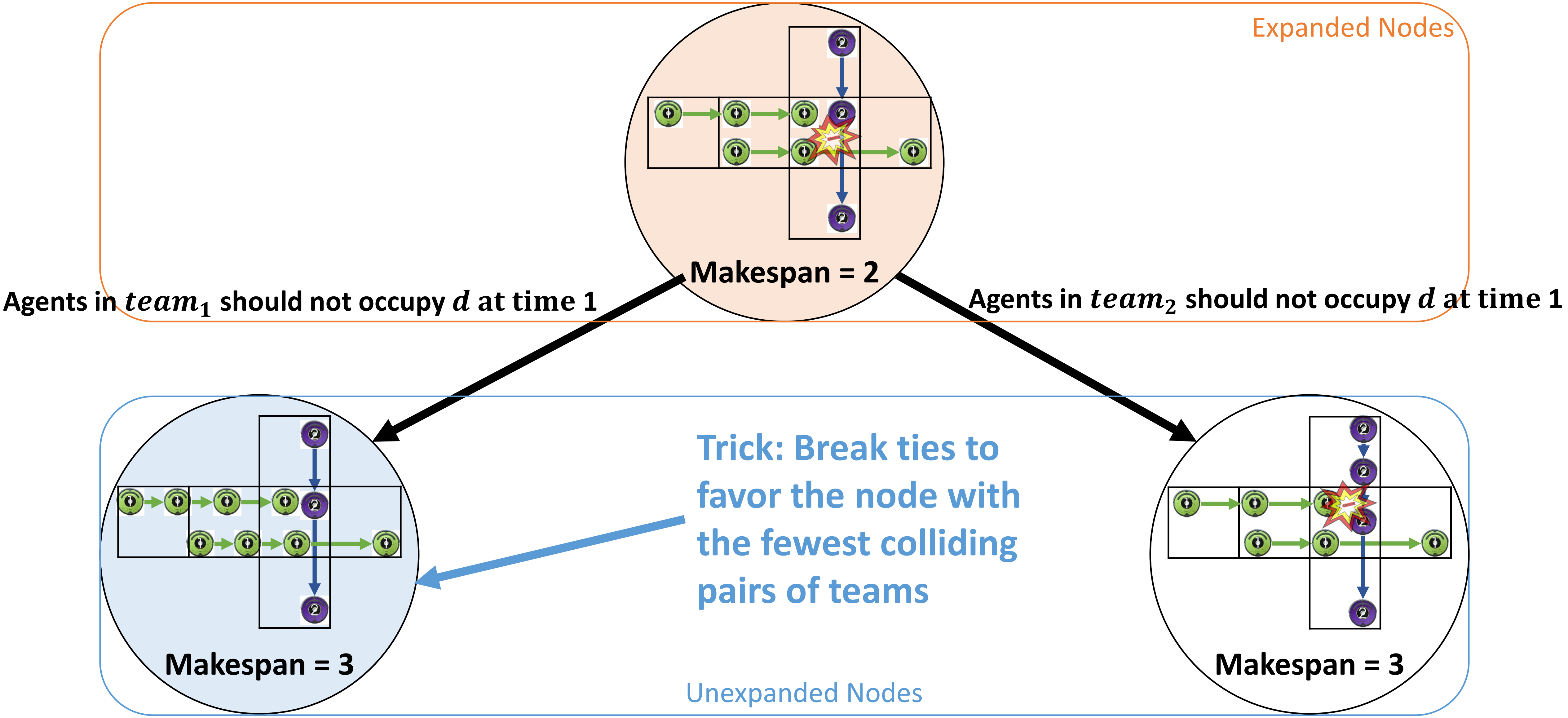


$team_2$

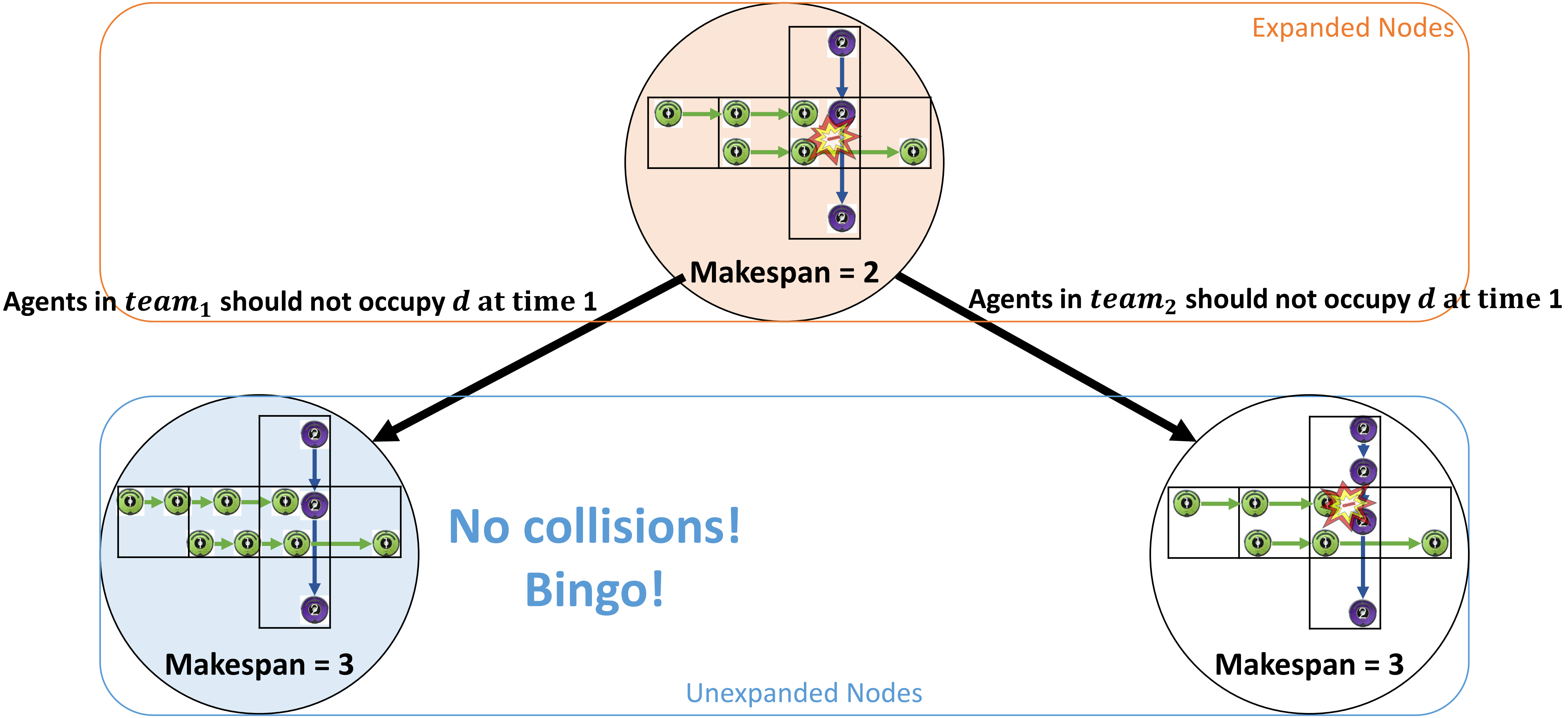
CBM: Store Paths and Makespan



CBM: Pop Node and Look for Collisions



CBM: Return Paths



Summary and Properties of CBM

Key ideas:

- Break down to the NP-hard sub-problem for different teams and the P-solvable sub-problems for agents in every team
- Use CBS for the NP-hard sub-problem
- Use a **min-cost max-flow algorithm** for the P-solvable sub-problems

Polynomial time – to choose paths that result in few collisions with agents from other teams

Robustness:

- No collisions
- Complete for all TAPF instances

Effectiveness:

- Optimal

Experiment: CBM vs ILP

Setup:

- 30× 30 grids, each with 10% randomly blocked cells, 5-minute time limits
- 10 to 50 agents, 5 agents per team

Results:

- CBM exploits more of the problem structure than a generic solver

agents	runtime (seconds)		success	
	CBM	ILP	CBM	ILP
10	**0.34**	18.24	**100%**	**100%**
20	**0.78**	62.85	**100%**	94%
30	**1.71**	108.75	**100%**	66%
40	**2.95**	152.98	**100%**	14%
50	**5.32**	161.95	**100%**	4%

Experiment: Scalability: Simulated Warehouse

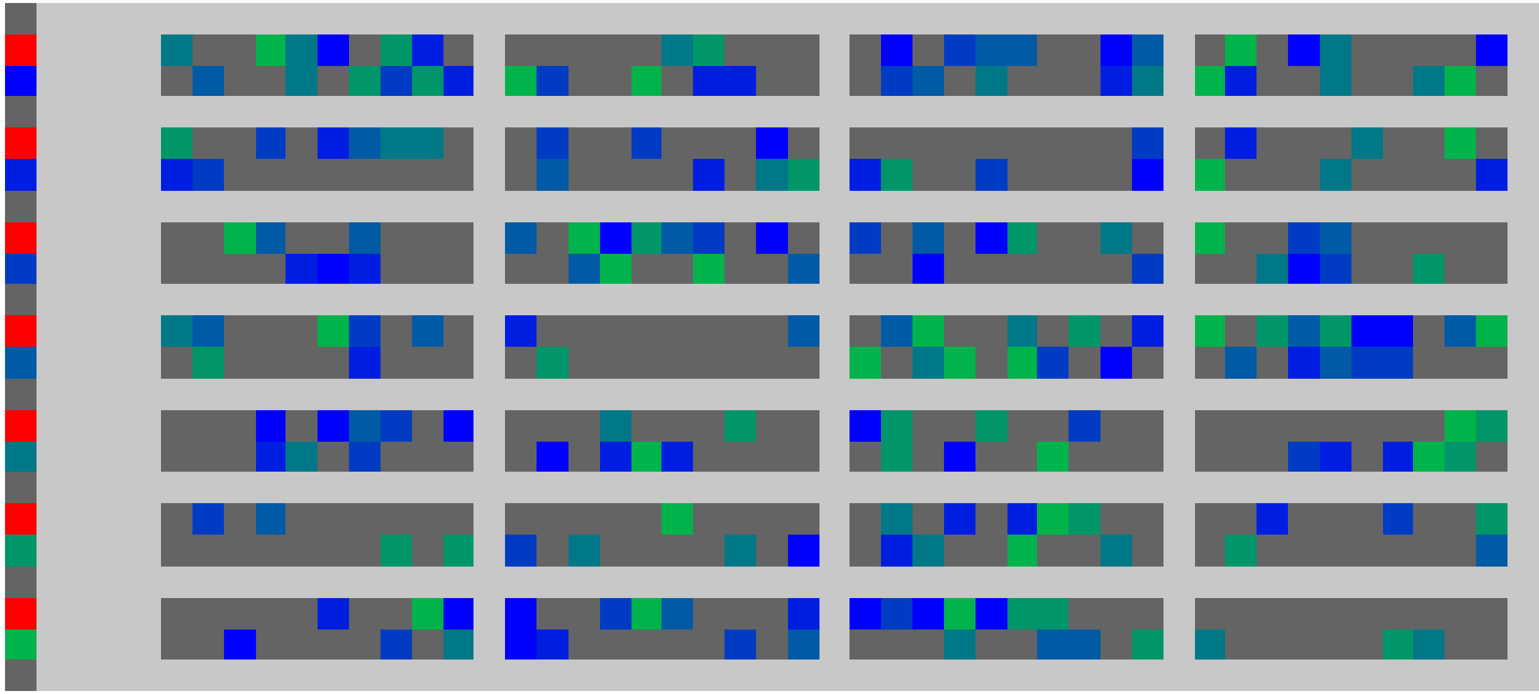
Setup:

- 49× 22 grid, 7 packing stations with incoming and outgoing queues
- **420 agents,** 7 “incoming” teams of 30 agents, one “outgoing” team of 210 agents

Results:

- Runtime ≈ 1 minute (averaged over 50 random instances)

Packing Station 1
Packing Station 2
Packing Station 3
Packing Station 4
Packing Station 5
Packing Station 6
Packing Station 7



Content

Path Planning

Joint Task & Path Planning

Plan Execution with System Dynamics

Robust Long-Term Task & Path Planning

Future Directions

Contribution

Increasing System Robustness

Plan Execution with System Dynamics

Path Planning

- **Theoretical results [AAAI-16]**
- Optimal algorithms [ICAPS-18, 19] [AAAI-19d]
- Suboptimal algorithms [AAAI-19a] [AAMAS-19b]
- Anytime algorithms [IJCAI-18b]
- Overviews [IJCAI WOMPF-16] [AI Matters-17]

Joint Task & Path Planning

- **Optimal task and path planning [AAMAS-16]**
- Tasks with temporal constraints [AAMAS-18] [IJCAI-18a]
- Large-sized agents [AAAI-19c]

Increasing Model Expressivity

Challenges

"Agents should be able to execute the plan"

- 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 (require robustness)
- Replanning whenever plan execution deviates from the plan is intractable since it is NP-hard to find good plans (require efficiency)

MAPF-POST: Kinematic Constraints [ICAPS-16]

MAPF-POST: Makes use of a simple temporal network (STN) to post-process the output of a MAPF/TAPF solver

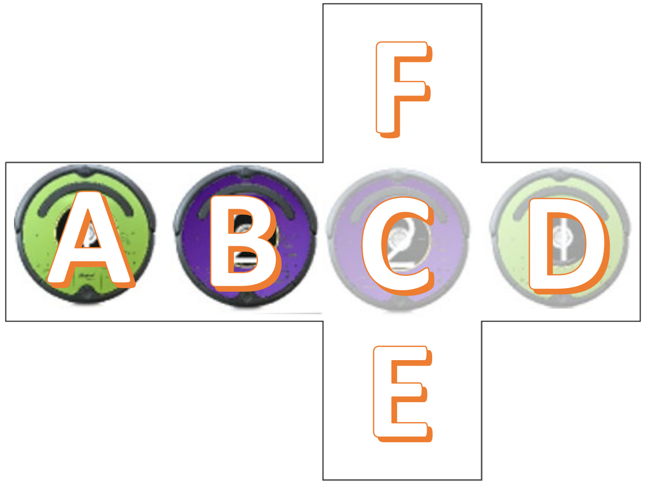
- Take into account edge lengths
- Take into account velocity limits (for both robots and edges)
- Guarantee a safety distance among robots
- Avoid replanning in many cases

Solved by linear programming or shortest path algorithms

Polynomial time

W. Hönig, T. K. S. Kumar, L. Cohen, H. Ma, H. Xu, N. Ayanian, and S. Koenig. "Multi-Agent Path Finding with Kinematic Constraints". ICAPS. 2016.

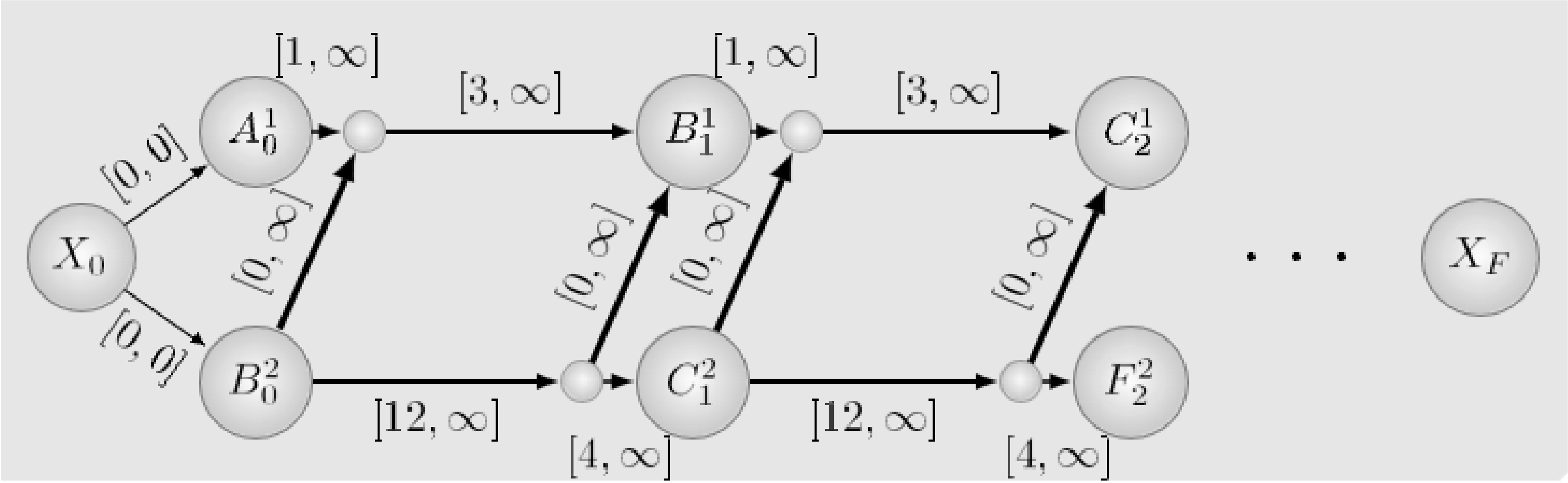
MAPF-POST: Example



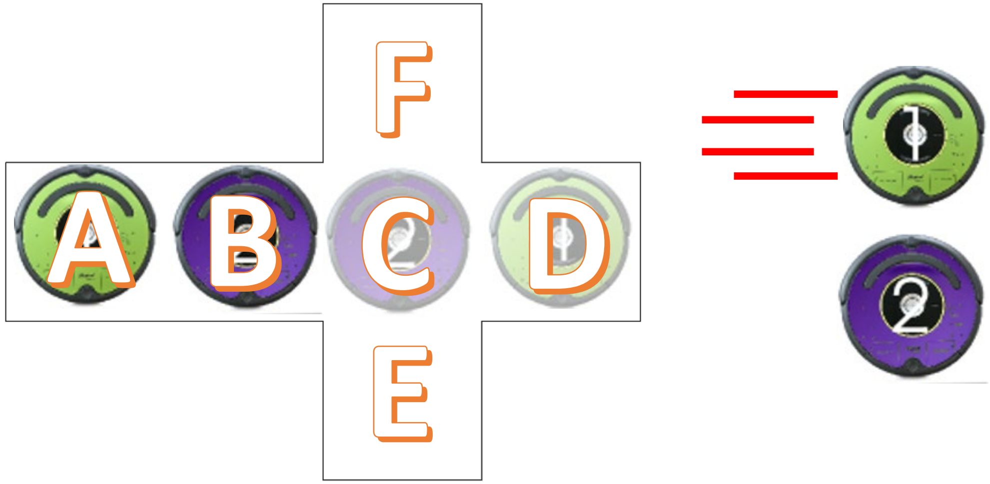
Agent 1 $A \rightarrow B \rightarrow C \rightarrow D$ (fast)

Agent 2 $B \rightarrow C \rightarrow F \rightarrow C$ (slow)

node = event that an agent arrives at a location



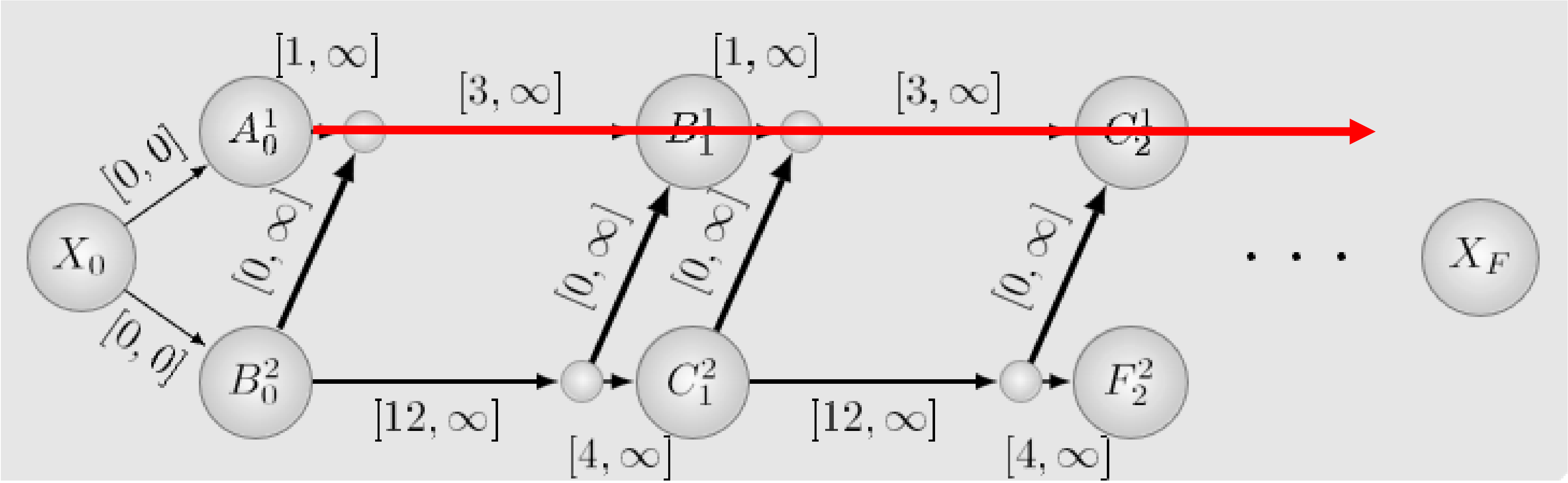
MAPF-POST: Example



Agent 1 $A \rightarrow B \rightarrow C \rightarrow D$ (fast)

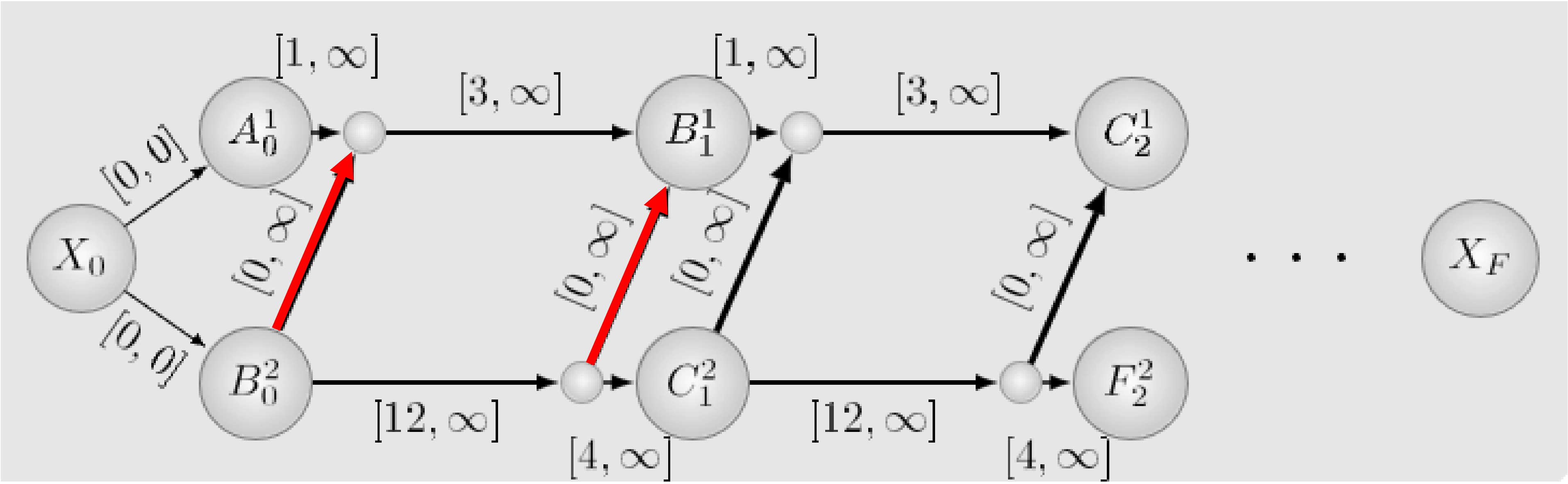
Agent 2 $B \rightarrow C \rightarrow F \rightarrow C$ (slow)

Type 1 edge = order in which the same agent arrives at locations

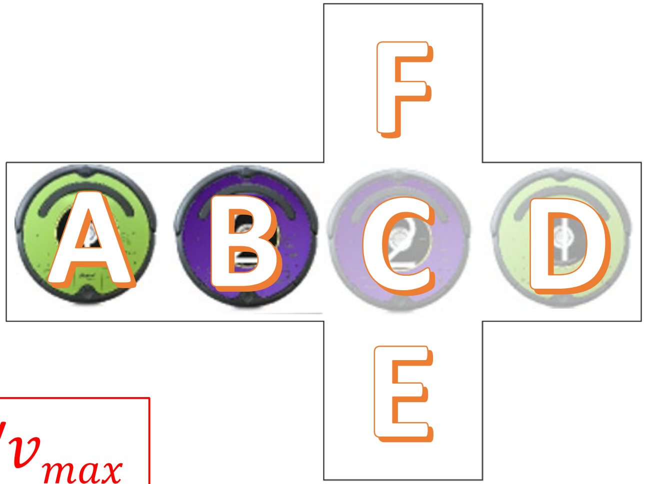


MAPF-POST: Example

Type 2 edge = order in which two different agents arrive at the same location



MAPF-POST: Example

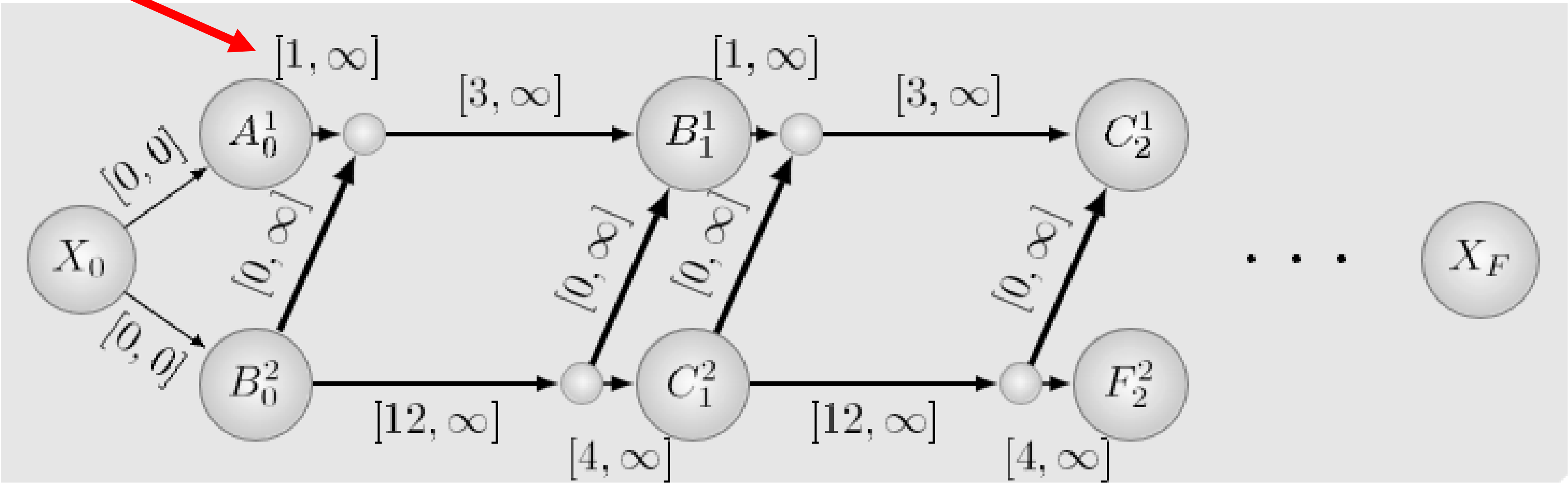


Agent 1 A → B → C → D (fast)

Agent 2 B → C → F → C (slow)

safety_distance/v_{max}

time bounds = kinematic constraints



Plan Generation and Execution Framework [IEEE IntSys-17]

H. Ma, W. Hönig, L. Cohen, T. Uras, H. Xu, T. K. S. Kumar, N. Ayanian, and S. Koenig. "Overview: A Hierarchical Framework for Plan Generation and Execution in Multi-Robot Systems". IEEE Intelligent Systems 32(6). 2017.

Plan Generation and Execution Framework

Main loop

- Run CBS/CBM to find a MAPF/TAPF plan (slow)

H. Ma, W. Hönig, L. Cohen, T. Uras, H. Xu, T. K. S. Kumar, N. Ayanian, and S. Koenig. "Overview: A Hierarchical Framework for Plan Generation and Execution in Multi-Robot Systems". IEEE Intelligent Systems 32(6). 2017.

Plan Generation and Execution Framework

Main loop

- Run CBS/CBM to find a MAPF/TAPF plan (slow)
- Construct a simple temporal network for the plan

H. Ma, W. Hönig, L. Cohen, T. Uras, H. Xu, T. K. S. Kumar, N. Ayanian, and S. Koenig. "Overview: A Hierarchical Framework for Plan Generation and Execution in Multi-Robot Systems". IEEE Intelligent Systems 32(6). 2017.

Plan Generation and Execution Framework

Main loop

- Run CBS/CBM to find a MAPF/TAPF plan (slow)
- Construct a simple temporal network for the plan
- Determine the earliest arrival times in the nodes

H. Ma, W. Hönig, L. Cohen, T. Uras, H. Xu, T. K. S. Kumar, N. Ayanian, and S. Koenig. "Overview: A Hierarchical Framework for Plan Generation and Execution in Multi-Robot Systems". IEEE Intelligent Systems 32(6). 2017.

Plan Generation and Execution Framework

Main loop

- Run CBS/CBM to find a MAPF/TAPF plan (slow)
- Construct a simple temporal network for the plan
- Determine the earliest arrival times in the nodes

- Calculate speeds for the robots from the earliest arrival times

H. Ma, W. Hönig, L. Cohen, T. Uras, H. Xu, T. K. S. Kumar, N. Ayanian, and S. Koenig. "Overview: A Hierarchical Framework for Plan Generation and Execution in Multi-Robot Systems". IEEE Intelligent Systems 32(6). 2017.

Plan Generation and Execution Framework

Main loop

- Run CBS/CBM to find a MAPF/TAPF plan (slow)
- Construct a simple temporal network for the plan
- Determine the earliest arrival times in the nodes

- Calculate speeds for the robots from the earliest arrival times
- Move robots along their paths in the plan with these speeds

H. Ma, W. Hönig, L. Cohen, T. Uras, H. Xu, T. K. S. Kumar, N. Ayanian, and S. Koenig. "Overview: A Hierarchical Framework for Plan Generation and Execution in Multi-Robot Systems". IEEE Intelligent Systems 32(6). 2017.

Plan Generation and Execution Framework

Main loop

- Run CBS/CBM to find a MAPF/TAPF plan (slow)
- Construct a simple temporal network for the plan
- Determine the earliest arrival times in the nodes
- Calculate speeds for the robots from the earliest arrival times
- Move robots along their paths in the plan with these speeds
- If plan execution deviates from the plan, then

H. Ma, W. Hönig, L. Cohen, T. Uras, H. Xu, T. K. S. Kumar, N. Ayanian, and S. Koenig. "Overview: A Hierarchical Framework for Plan Generation and Execution in Multi-Robot Systems". IEEE Intelligent Systems 32(6). 2017.

Plan Generation and Execution Framework

Main loop

- Run CBS/CBM to find a MAPF/TAPF plan (slow)
- Construct a simple temporal network for the plan
- Determine the earliest arrival times in the nodes

- Calculate speeds for the robots from the earliest arrival times
- Move robots along their paths in the plan with these speeds
- If plan execution deviates from the plan, then

H. Ma, W. Hönig, L. Cohen, T. Uras, H. Xu, T. K. S. Kumar, N. Ayanian, and S. Koenig. "Overview: A Hierarchical Framework for Plan Generation and Execution in Multi-Robot Systems". IEEE Intelligent Systems 32(6). 2017.

Plan Generation and Execution Framework

Main loop

- Run CBS/CBM to find a MAPF/TAPF plan (slow)
- Construct a simple temporal network for the plan
- Determine the earliest arrival times in the nodes
- Calculate speeds for the robots from the earliest arrival times
- Move robots along their paths in the plan with these speeds
- If plan execution deviates from the plan, then

H. Ma, W. Hönig, L. Cohen, T. Uras, H. Xu, T. K. S. Kumar, N. Ayanian, and S. Koenig. "Overview: A Hierarchical Framework for Plan Generation and Execution in Multi-Robot Systems". IEEE Intelligent Systems 32(6). 2017.

Plan Generation and Execution Framework

Main loop

- Run CBS/CBM to find a MAPF/TAPF plan (slow)
- Construct a simple temporal network for the plan
- Determine the earliest arrival times in the nodes
- If they do not exist, then
- Calculate speeds for the robots from the earliest arrival times
- Move robots along their paths in the plan with these speeds
- If plan execution deviates from the plan, then

H. Ma, W. Hönig, L. Cohen, T. Uras, H. Xu, T. K. S. Kumar, N. Ayanian, and S. Koenig. "Overview: A Hierarchical Framework for Plan Generation and Execution in Multi-Robot Systems". IEEE Intelligent Systems 32(6). 2017.

TAPF: Demo on Real Robots [IROS-16]

Setup:

- 4x3 grid, 1m^2 cells
- 8 robots, 2 teams, 4 robots per team

TAPF solver: CBM

Post-processing procedure: MAPF-POST

Architecture: ROS with decentralized execution

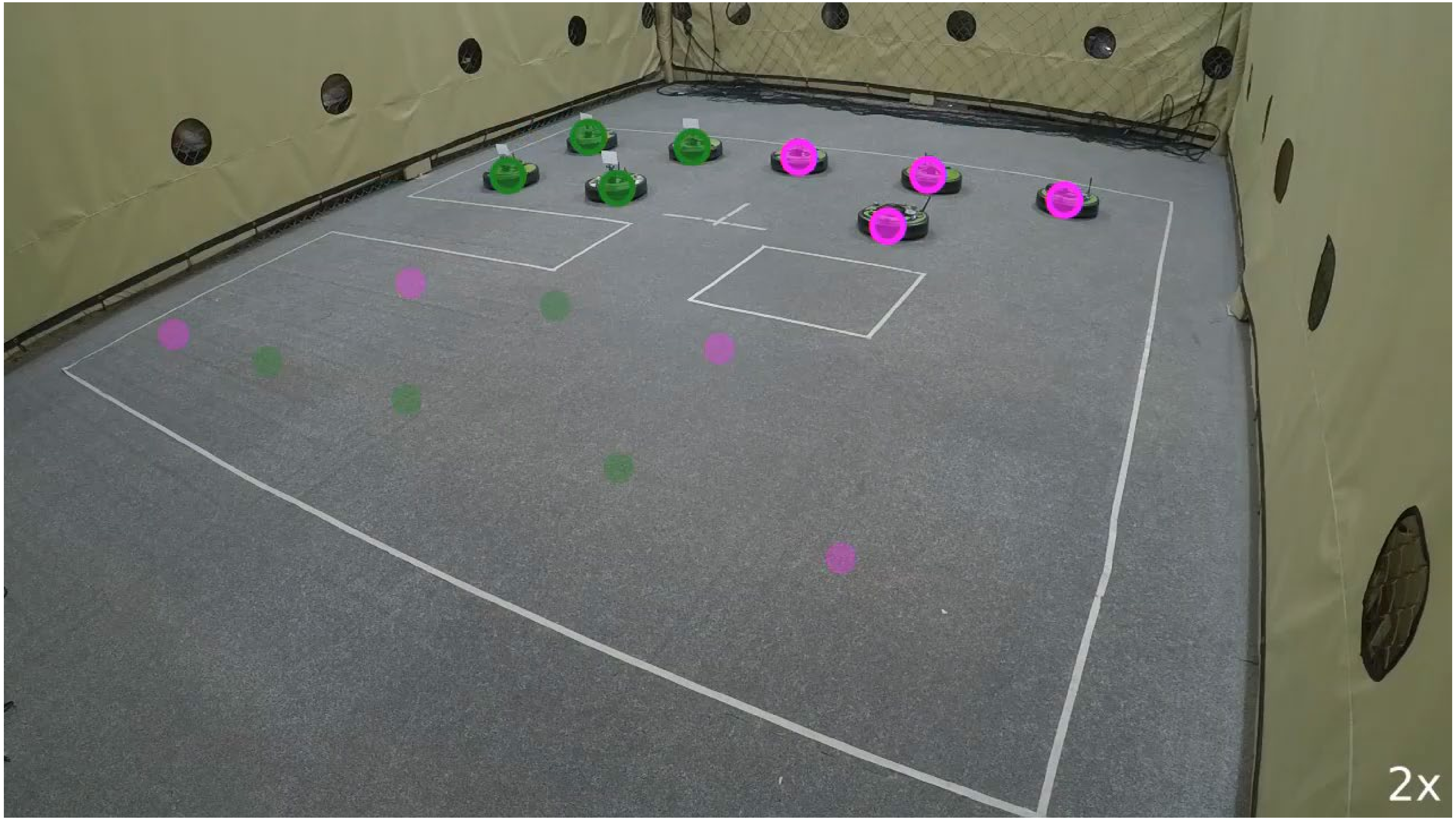
- Robot controller with state [x,y,Θ] (attempts to meet deadline)
- PID controller (corrects for heading error and drift)

Robots: iRobot Create2 robots

Test environment: VICON MX Motion Capture System

W. Hönig, T. K. S. Kumar, H. Ma, N. Ayanian, and S. Koenig. "Formation Change for Robot Groups in Occluded Environments". IROS. 2016.

Demo on Real Robots



2x

Content

Path Planning

Joint Task & Path Planning

Plan Execution with System Dynamics

Robust Long-Term Task & Path Planning

Future Directions

Contribution

Increasing System Robustness

Plan Execution with System Dynamics

- **Framework [IEEE IntSys-17]**
- **Kinematic constraints [ICAPS-16] [IROS-16]**
- Uncertainty [AAAI-17]
- Video game environments [AIIDE-17]
- Partial observability [AAAI-15]

Robust Long-Term Task & Path Planning

Path Planning

- **Theoretical results [AAAI-16]**
- Optimal algorithms [ICAPS-18, 19] [AAAI-19d]
- Suboptimal algorithms [AAAI-19a] [AAMAS-19b]
- Anytime algorithms [IJCAI-18b]
- Overviews [IJCAI WOMPF-16] [AI Matters-17]

Joint Task & Path Planning

- **Optimal task and path planning [AAMAS-16]**
- Tasks with temporal constraints [AAMAS-18] [IJCAI-18a]
- Large-sized agents [AAAI-19c]

Increasing Model Expressivity

Challenges

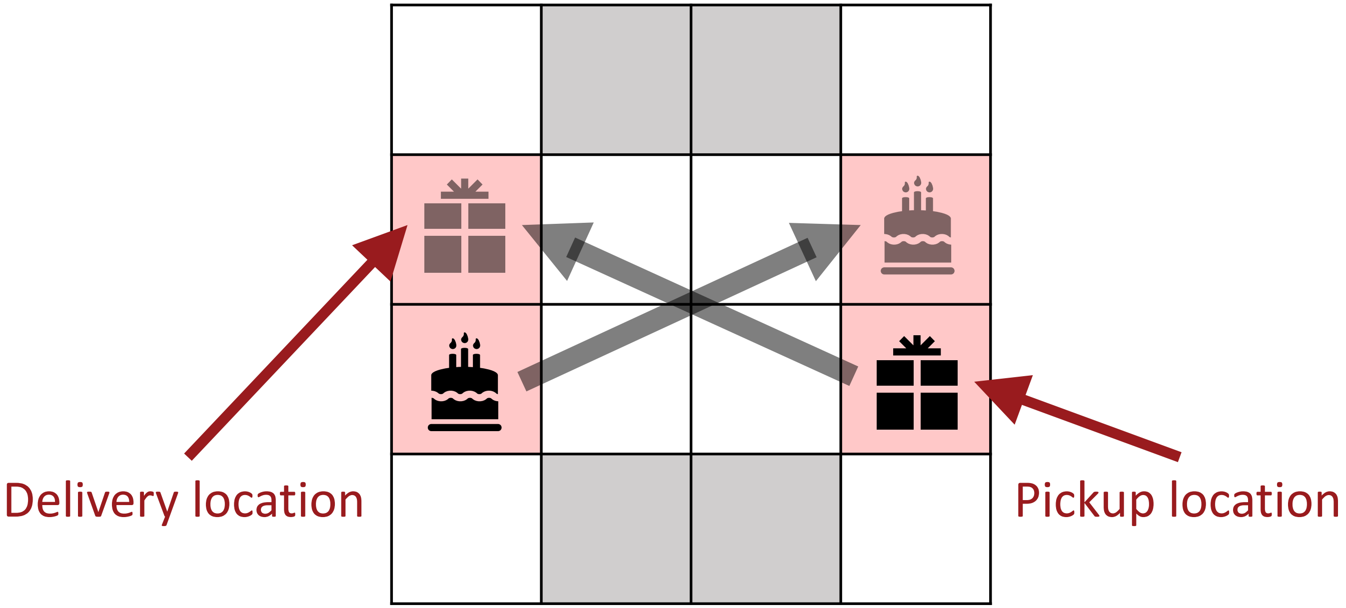
"Agents need to be retasked"

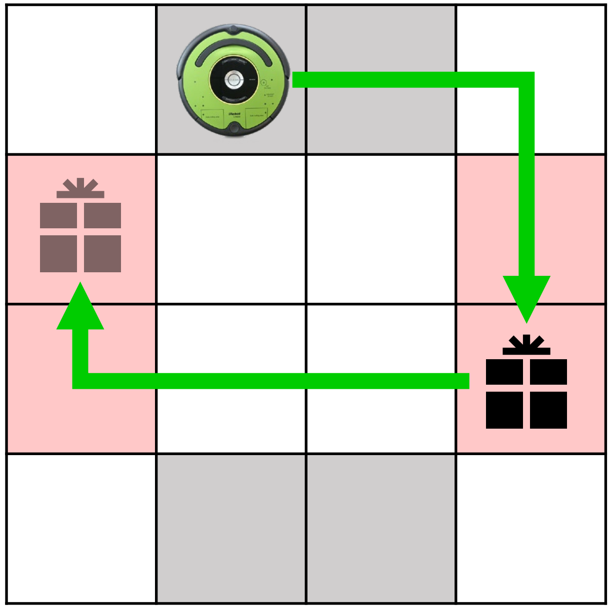
- Agents need to be assigned new tasks after finishing their current ones
  - Agents need to decide which task to execute next
  - Agents can block each other from executing new tasks
  - …
- Assigning tasks and planning paths for the agents iteratively can lead to deadlocks (require robustness)
- Tasks can appear at any time and decisions should be made during execution in real-time (require efficiency)

Multi-Agent Pickup and Delivery [AAMAS-17]

Multi-Agent Pickup and Delivery (MAPD) is a **"lifelong"** generalization of the **"one-shot"** task- and path- planning problems:

- A task can enter the system at any time.
- Agents have to constantly attend to a stream of new tasks.



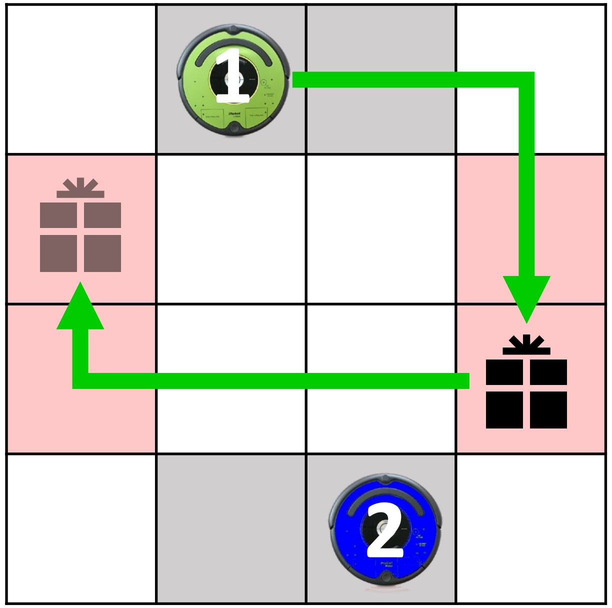


H. Ma, J. Li, T. K. S. Kumar, and S. Koenig. "Lifelong Multi-Agent Path Finding for Online Pickup and Delivery Tasks". AAMAS. 2017.

MAPD: Task Assignment

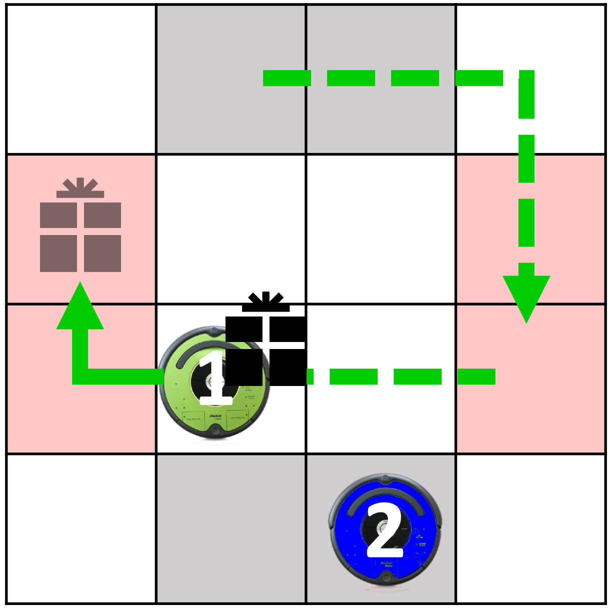
Free agent

- *agent1*, *agent2*
- Can be assigned to any unexecuted task



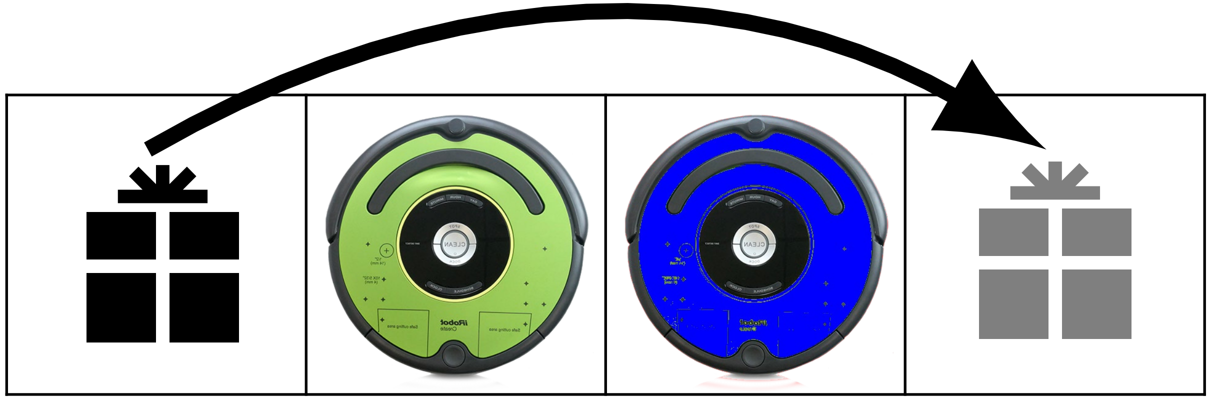
Occupied agent

- *agent1*
- Must finish executing its current task



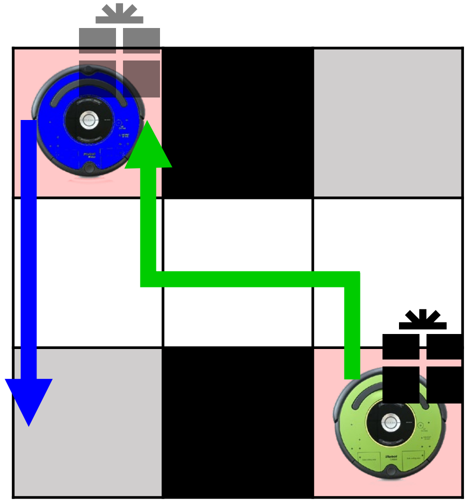
MAPD: Solvability

Not every MAPD instance is solvable



MAPD: Well-Formed Conditions

Task parking locations	All pickup and delivery locations of tasks	Might not be safe to park forever
Non-task parking locations	Designated safe parking locations	Safe to park forever



A MAPD instance is well-formed *iff*

1. # tasks is finite
2. # **non-task parking locations** ≥ # agents
3. For any two parking locations, there exists a path between them that traverses no other parking locations

MAPD Algorithms

1. **Decoupled** algorithms
   - Approach
     - Assign tasks greedily
     - **Simple version:** do not allow task transfers between free agents
     - **Complex version:** allow task transfers from one free agent to another
   - Good scalability
   - Extendable to a fully distributed setting
2. **Centralized** algorithms
   - Approach
     - Use the Hungarian algorithm to assign tasks globally
     - Use CBS in an inner loop to plan paths
   - Good solution quality

Summary and Properties of MAPD Algorithms

Key ideas:

- Assign tasks in an outer loop
- Plan paths in an inner loop
- Use parking locations to buffer

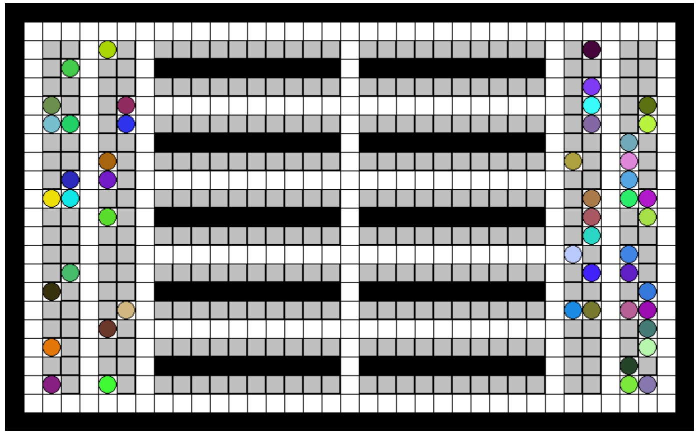
Robustness:

- No collisions
- Long-term robustness: All (finitely many) tasks are executed in a finite time
- Complete for all well-formed instances

Effectiveness:

- Not optimal
- Trade off between better solutions and faster runtime (empirically)

Experiment: Comparison



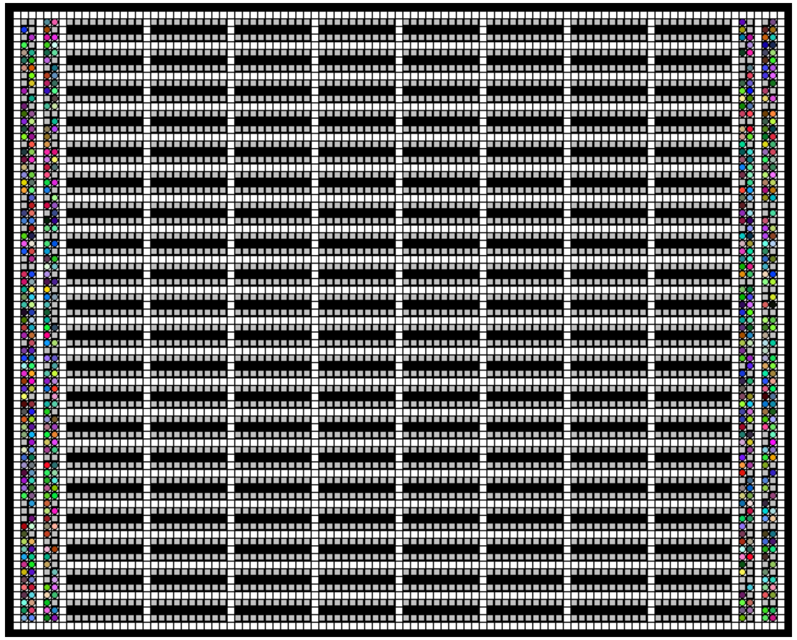
Setup:

- 21 × 35 grid
- 500 tasks, 10 to 50 agents

Results:

- Effectiveness (service time, throughput, makespan)
  - **CENTRALIZED** > DECOUPLED with task transfers > DECOUPLED
- Efficiency (runtime per timestep)
  - **DECOUPLED** ≈ 10 milliseconds
  - DECOUPLED with task transfers ≈ 200 milliseconds
  - CENTRALIZED ≈ 4,000 milliseconds

Experiment: Scalability of DECOUPLED



Setup:

- 81 × 81 grid
- 1000 tasks, 100 to 500 agents

Results:

- Runtime per timestep
  - 100 agents: ≈ 0.09 seconds
  - 500 agents: ≈ 6 seconds

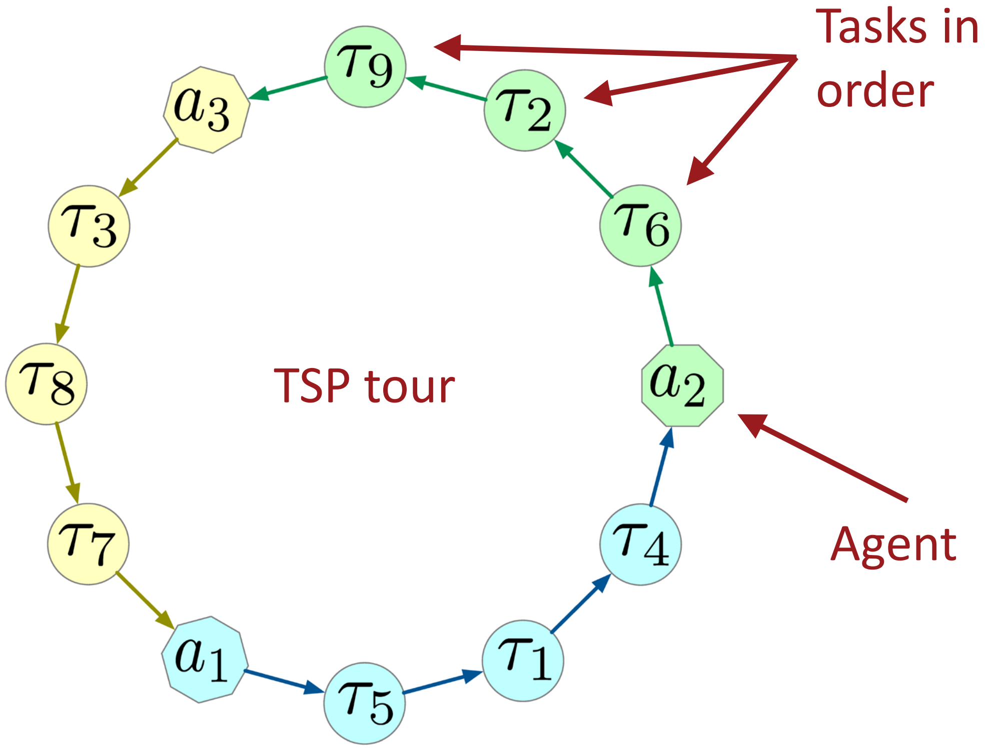
agents	100	200	300	400	500
service time	463.25	330.19	301.97	289.08	284.24
runtime (milliseconds)	90.83	538.22	1,854.44	3,881.11	6,121.06

MAPD: Offline Task Scheduling [AAMAS-19a]

Asymmetric TSP with dynamic edge weights

Meta-heuristic TSP solver

Chronologically ordered task sequences, one for each agent



Improved CENTRALIZED:

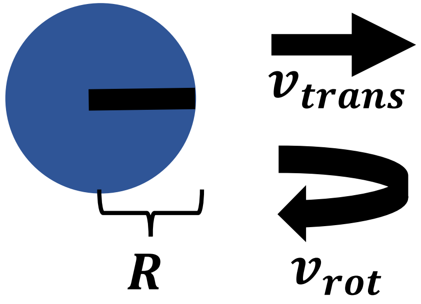
- Tasks execution order scheduling
  - Reduce makespan by up to 46%
- Joint task and path planning for free agents (via min-cost max flow)
  - Up to 10 times faster

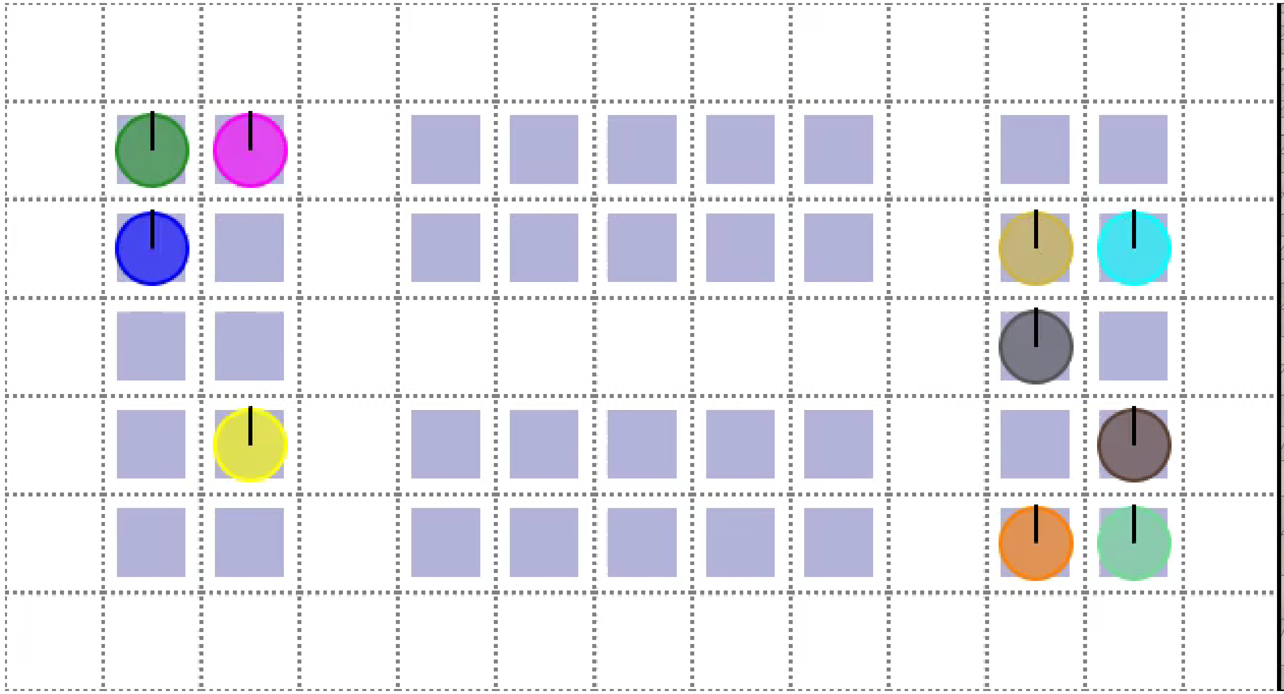
M. Liu, H. Ma, J. Li, and S. Koenig. "Task and Path Planning for Multi-Agent Pickup and Delivery". AAMAS. 2019.

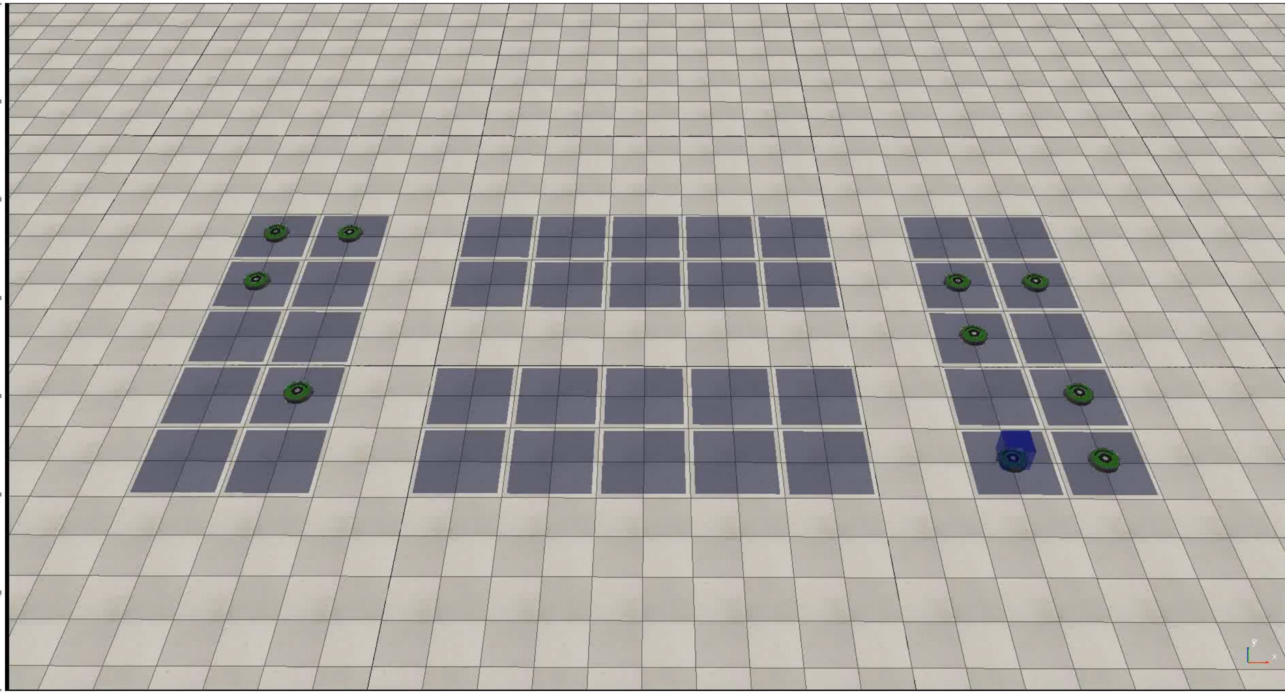
MAPD: Kinematic Constraints [AAAI-19b]

Improved DECOUPLED: Novel data structure for storing paths

- Allow for efficient operations in an online setting
- Consider kinematic constraints directly during planning







H. Ma, W. Hönig, T. K. S. Kumar, N. Ayanian, and S. Koenig. "Lifelong Path Planning with Kinematic Constraints for Multi-Agent Pickup and Delivery". AAAI. 2019.

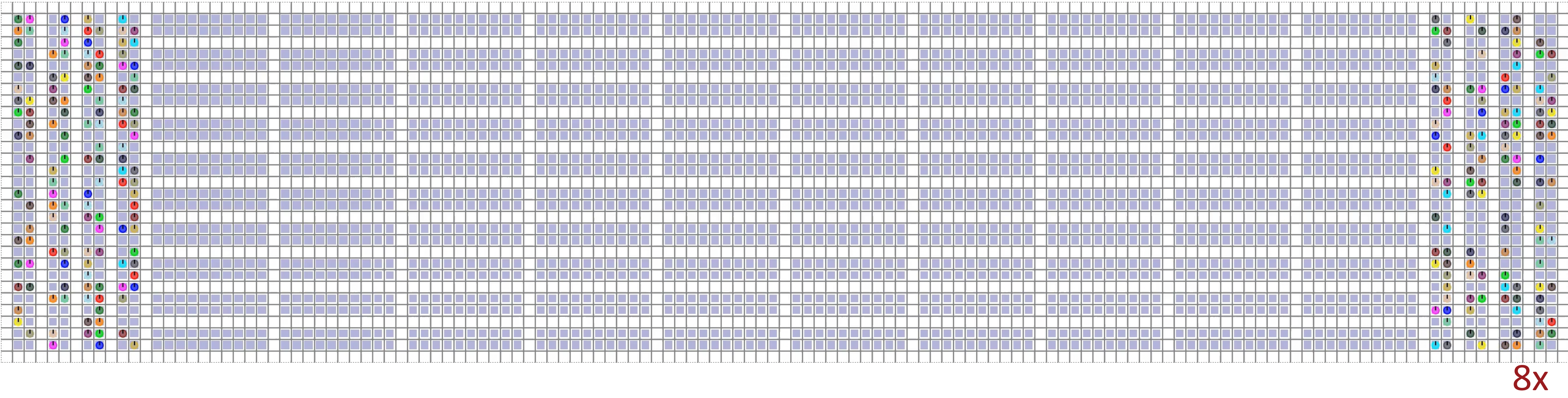
MAPD: Kinematic Constraints

Setup:

- 2000 tasks, 250 agents

Results:

- Makespan ≈ 30 minutes, total runtime < 10 seconds
- More efficient and effective than discrete MAPD algorithms + MAPF-POST



Contribution

Increasing System Robustness

Plan Execution with System Dynamics

- **Framework [IEEE IntSys-17]**
- **Kinematic constraints [ICAPS-16] [IROS-16]**
- Uncertainty [AAAI-17]
- Video game environments [AIIDE-17]
- Partial observability [AAAI-15]

Robust Long-Term Task & Path Planning

- **Formalization & algorithms [AAMAS-17]**
- **Kinematic constraints [AAAI-19b]**
- **Offline scheduling [AAMAS-19a]**

Path Planning

- **Theoretical results [AAAI-16]**
- Optimal algorithms [ICAPS-18, 19] [AAAI-19d]
- Suboptimal algorithms [AAAI-19a] [AAMAS-19b]
- Anytime algorithms [IJCAI-18b]
- Overviews [IJCAI WOMPF-16] [AI Matters-17]

Joint Task & Path Planning

- **Optimal task and path planning [AAMAS-16]**
- Tasks with temporal constraints [AAMAS-18] [IJCAI-18a]
- Large-sized agents [AAAI-19c]

Increasing Model Expressivity

Content

Path Planning

Joint Task & Path Planning

Plan Execution with System Dynamics

Robust Long-Term Task & Path Planning

Future Directions

Summary

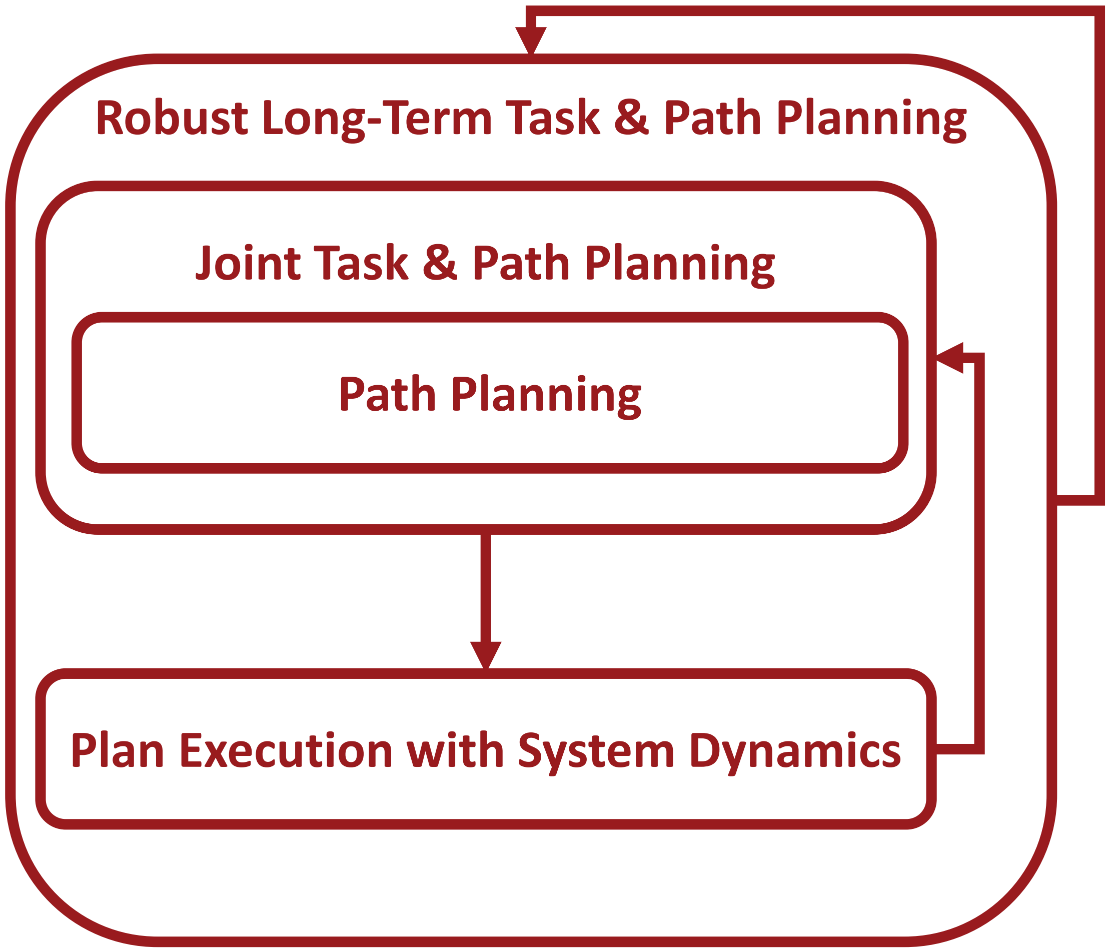
An algorithmic framework for coordinating long-term task- and motion-level operations for teams of agents

"Agents need to be retasked"

"Agents need to decide where to go next"

"Agents should not collide with each other"

"Agents should be able to execute the plan"



Research Questions

- How to design intelligent algorithms that make **fast** and **good** decisions to coordinate teams of agents?
  - Coordination of task- and motion- level operations: A fundamental building block for multi-agent systems

Larger Teams of Agents are the Future!

US Unmanned Aircraft System (UAS) Management System (NASA, Federal Aviation Administration, etc.)

- 1 year delayed = more than $10 billion dollars loss

Last-Mile Delivery: 2kg package in 10km radius

- Drones: 0.1 dollars
- Ground transportation: 2-8 dollars

Expected drone production (U.S.) in 2021:

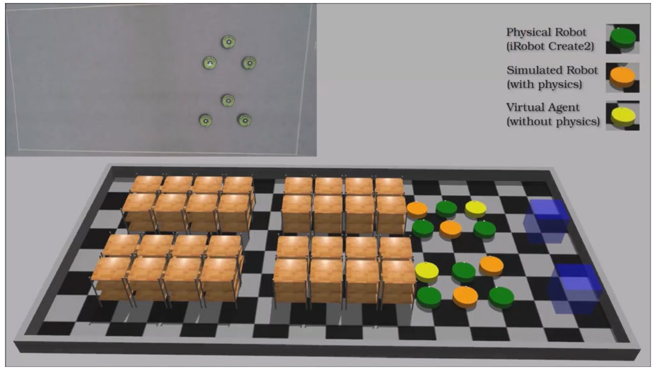
- 29,000,000 consumer drones
- 805,000 commercial drones

Source: Dronelife

Research Questions

- How to design intelligent algorithms that make **fast** and **good** decisions to coordinate teams of agents?
  - Task- and motion- level coordination: A fundamental building block for multi-agent systems
- How to coordinate teams that consist of a **mega-scale** number of agents?

Research Directions & Collaboration



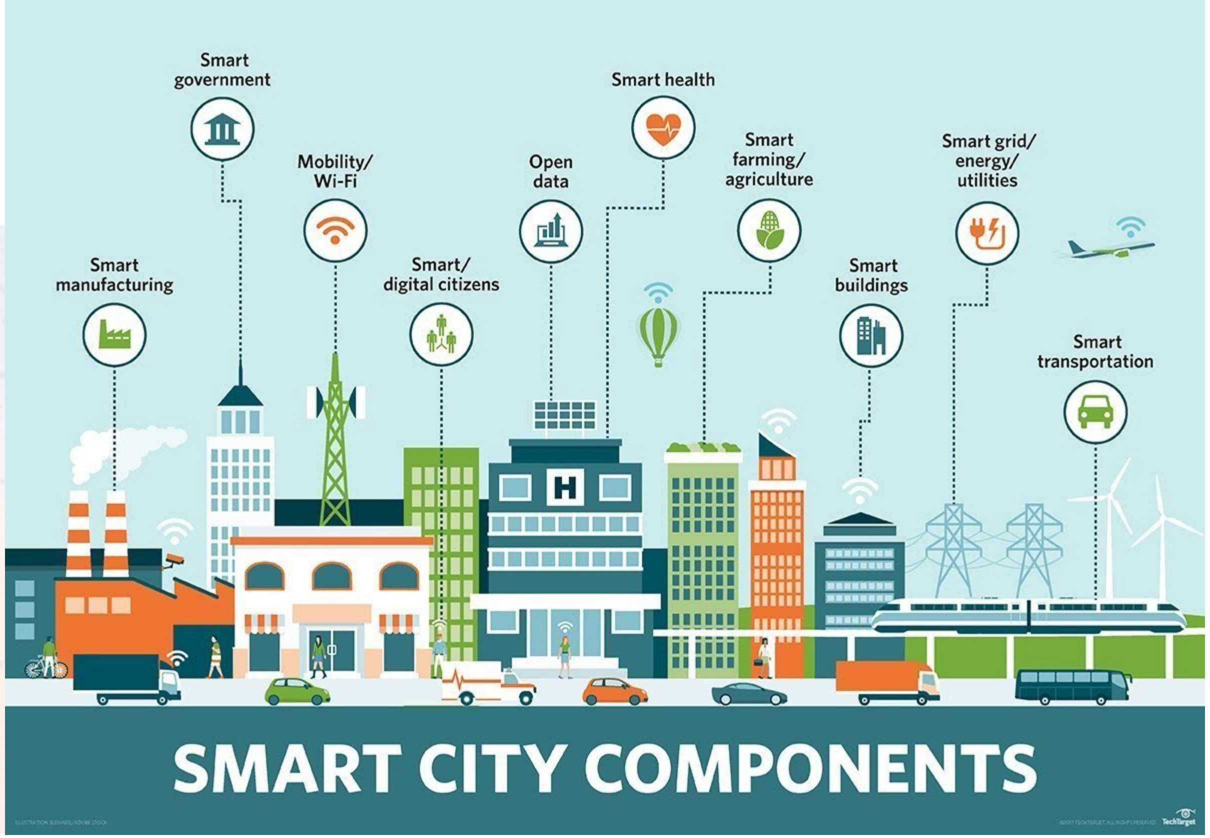
Source: USC ACT Lab

- **Larger** teams:
  - Mining historical data and learning
  - GPU/cloud computing
  - Mixed/virtual reality

Smarter Teams of Agents are the Future!

Smart Cities, Buildings, Homes, etc. (agents in other forms)

- Smart transportation
  - Route planning
  - Parking space assignment
  - Intersection management
  - Ride sharing (package transfers)
  - …
- Smart homes
  - Smart device scheduling
  - Smart grid optimization
  - …
- Other sustainability and security domains
- …



Source: IoT Agenda

Research Questions

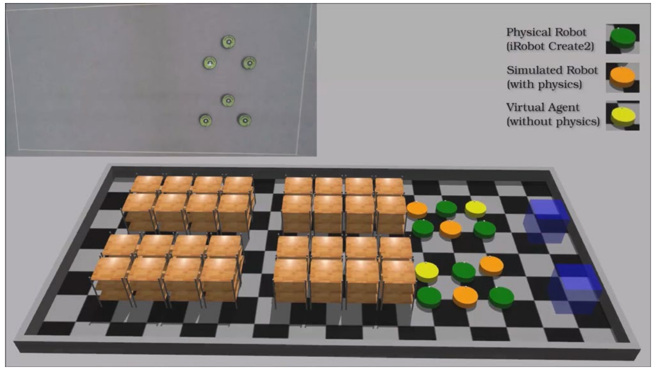
- How to design intelligent algorithms that make **fast** and **good** decisions to coordinate teams of agents?
  - Task- and motion- level coordination: A fundamental building block for multi-agent systems

- How to coordinate teams that consist of a **mega-scale** number of agents?

- How to build and coordinate teams of **smarter** agents that can better collaborate with and assist humans?

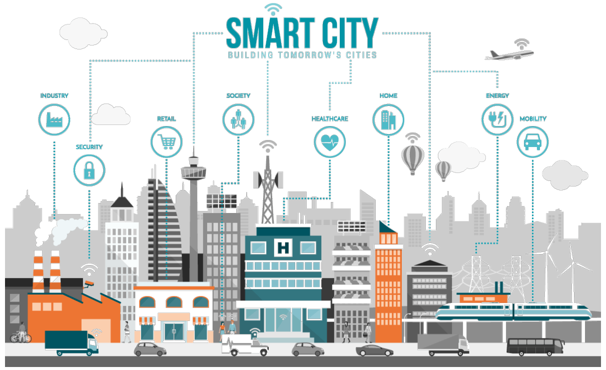
Research Directions & Collaboration

Source: USC ACT Lab



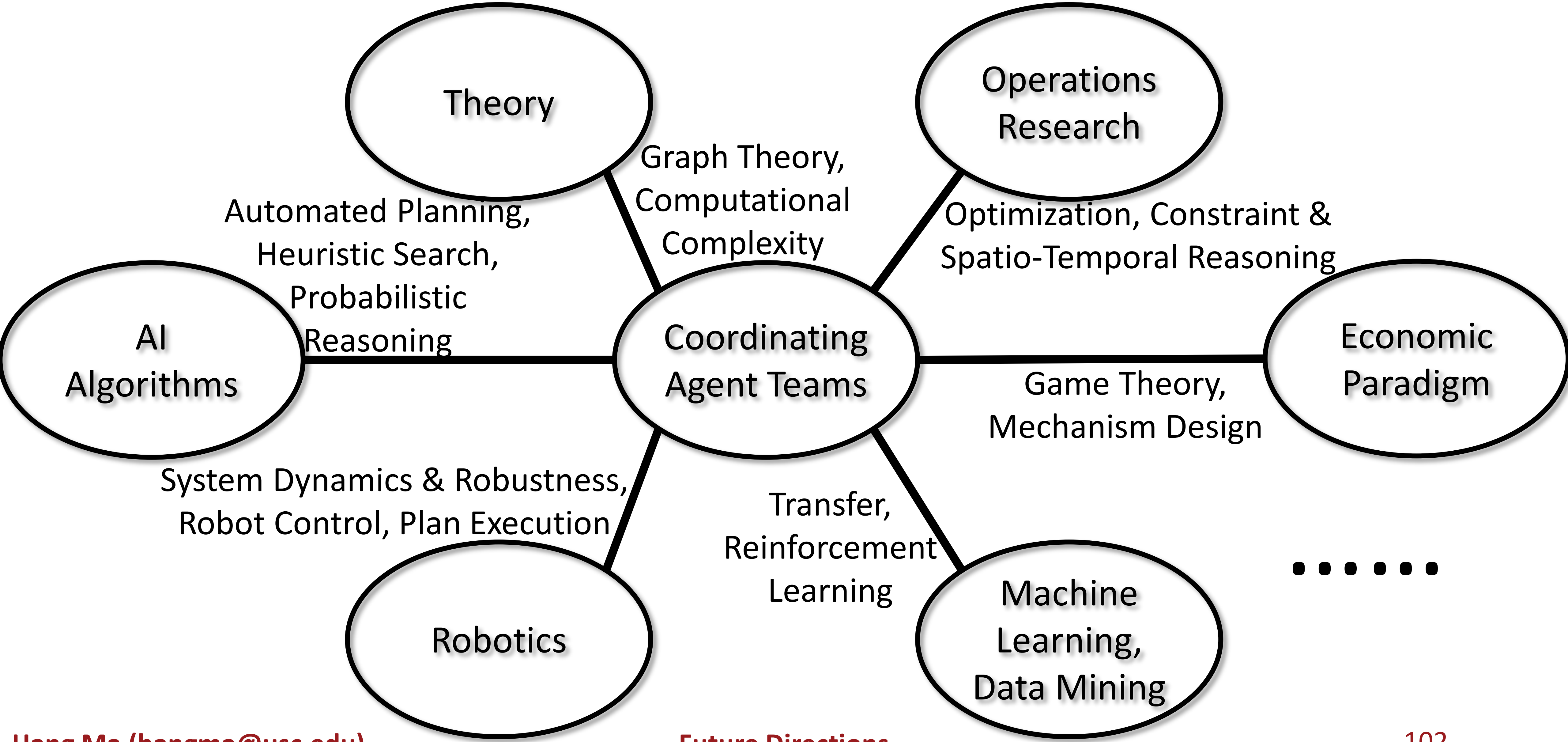
Source: Digital Trends

Source: ARC Web

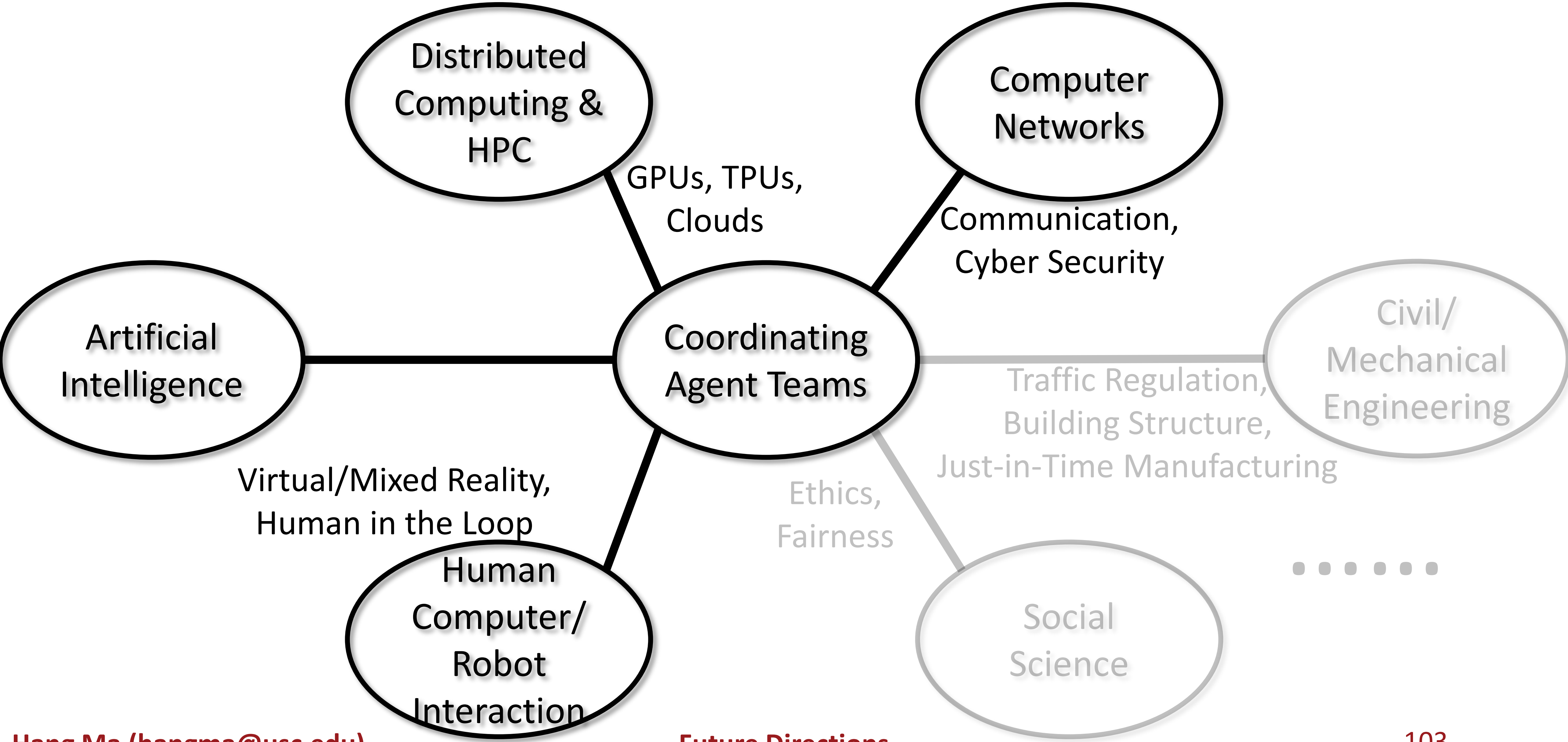


- **Larger** teams:
  - Mining historical data and learning
  - GPU/cloud computing
  - Mixed/virtual reality
- **Smarter** teams:
  - Human-centered design
  - Data-driven techniques
  - Coordinating self-interest agents
  - Smart manufacturing
  - Smart traffic control, smart building construction
  - Ethics and fairness of algorithms

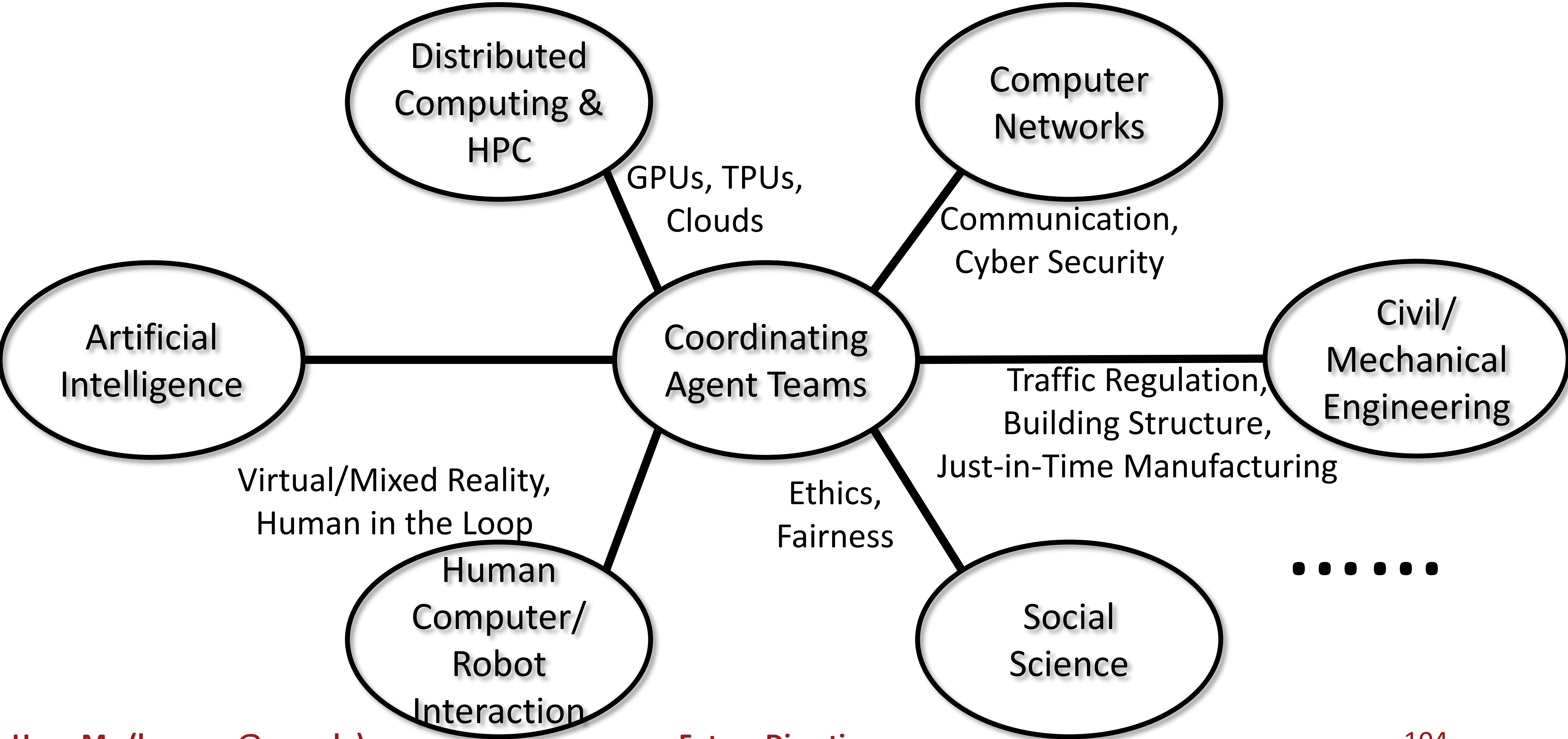
Research Directions



Research Directions



Research Directions

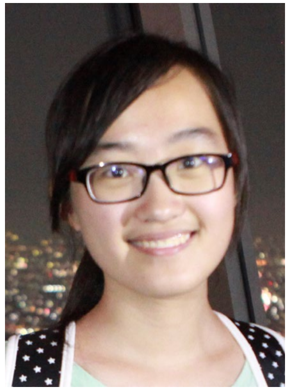


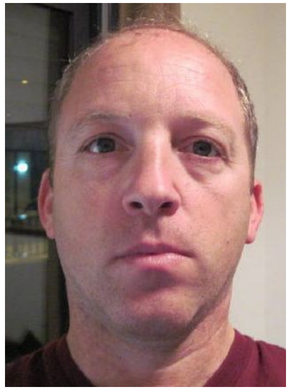
Collaborators: Teams of Intelligent People

AI Team


Sven Koenig


T. K. Satish Kumar


Jiaoyang Li


Ariel Felner


Liron Cohen

and many more…

Robotics Team


Wolfgang Hönig

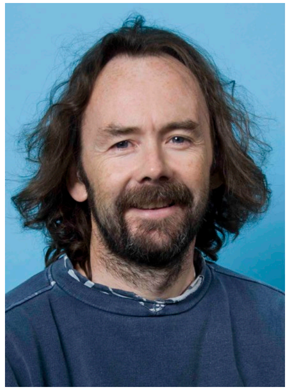

Nora Ayanian

Theory Team


Craig Tovey

OR Team


Daniel Harabor

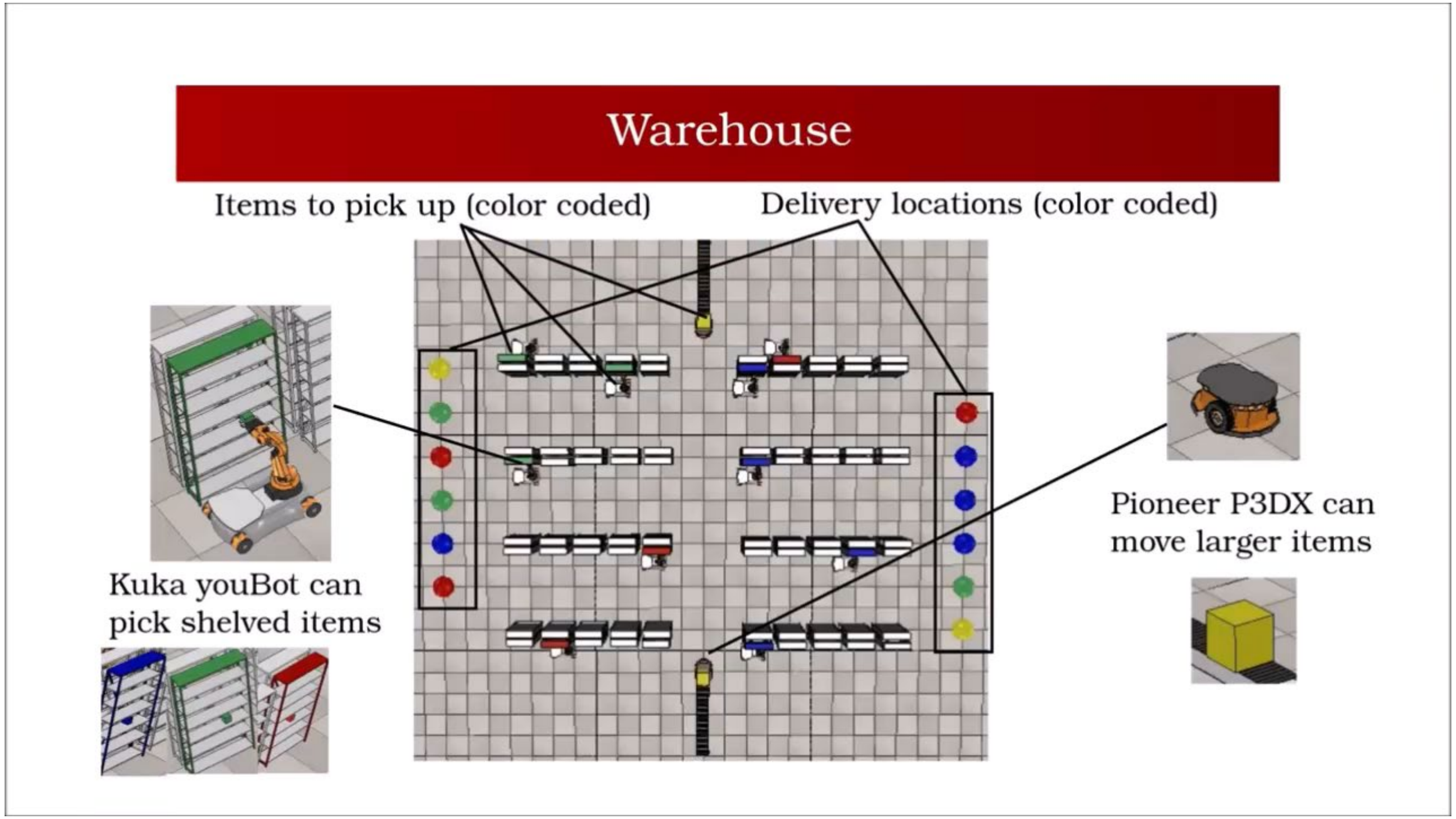

Peter Stuckey

Thank You!

Hang Ma (hangma@usc.edu)

More information:

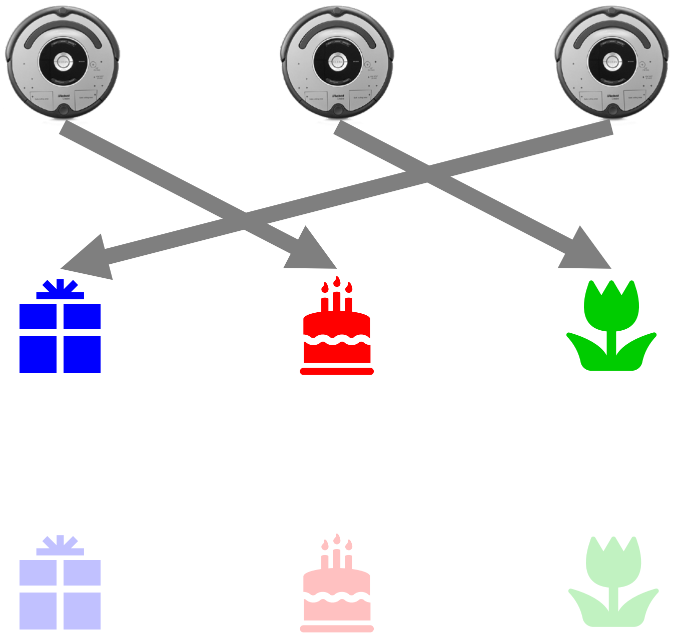
http://www-scf.usc.edu/~hangma/



MAPD: Task Assignment

Free agent

- **Anonymous**
- Can be assigned to any unexecuted task



Occupied agent

- **Non-anonymous**
- Must finish executing its current task