Real-World Multi Agent Systems
Lecture 6
Multi Agent Path Planning

Akin Delibasi

Lecturer in Distributed Systems

Objectives and Reading

- What is MAPP?
- Learn the algorithms for MAPP
  - Priority approach
  - Conflict Based Approach
- Drawbacks of CBS
- Meta Agent CBS

Artificial Intelligence 219 (2015) 40–66

Contents lists available at ScienceDirect

Artificial Intelligence

www.elsevier.com/locate/artint

Conflict-based search for optimal multi-agent pathfinding

Guni Sharon [a,*], Roni Stern [a], Ariel Felner [a], Nathan R. Sturtevant [b]

[a] Information Systems Engineering, Ben Gurion University, Be'er Sheba, 85104, Israel
[b] Department of Computer Science, University of Denver, Denver, CO, USA

ARTICLE INFO

Article history:
Received 16 September 2013
Received in revised form 23 November 2014
Accepted 25 November 2014
Available online 2 December 2014

Keywords:
Heuristic search
Multi-agent
Pathfinding

ABSTRACT

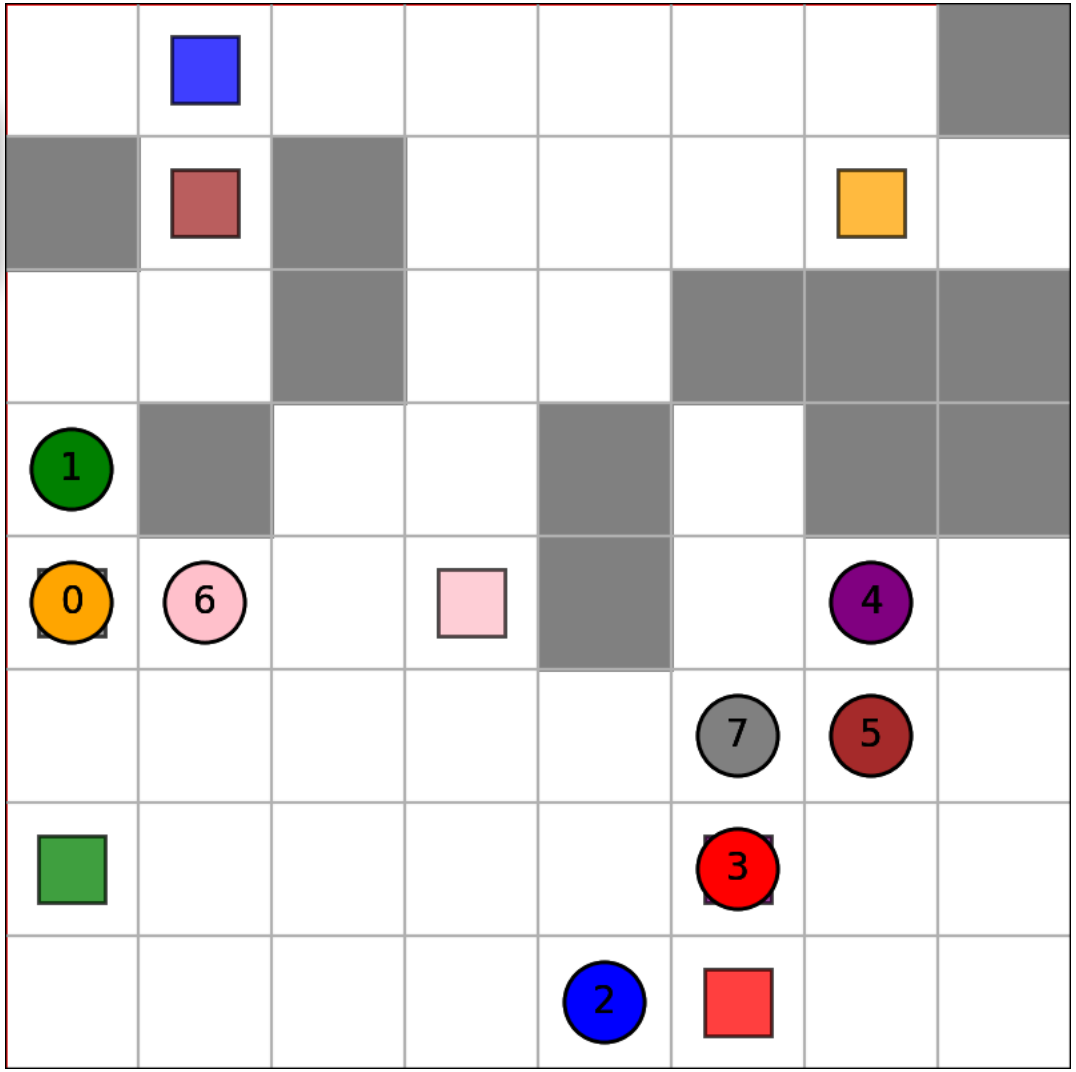
In the *multi-agent pathfinding* problem (MAPF) we are given a set of agents each with respective start and goal positions. The task is to find paths for all agents while avoiding collisions. Most previous work on solving this problem optimally has treated the individual agents as a single 'joint agent' and then applied single-agent search variants of the A* algorithm.

In this paper we present the Conflict Based Search (CBS) a new optimal multi-agent pathfinding algorithm. CBS is a two-level algorithm that does not convert the problem into the single 'joint agent' model. At the high level, a search is performed on a *Conflict Tree* (CT) which is a tree based on conflicts between individual agents. Each node in the CT represents a set of constraints on the motion of the agents. At the low level, fast single-agent searches are performed to satisfy the constraints imposed by the high level CT node. In many cases this two-level formulation enables CBS to examine fewer states than A* while still maintaining optimality. We analyze CBS and show its benefits and drawbacks.

Additionally we present the *Meta-Agent CBS* (MA-CBS) algorithm. MA-CBS is a generalization of CBS. Unlike basic CBS, MA-CBS is not restricted to single-agent searches at the low level. Instead, MA-CBS allows agents to be merged into small groups of joint agents. This mitigates some of the drawbacks of basic CBS and further improves performance. In fact, MA-CBS is a framework that can be built on top of any optimal and complete MAPF solver in order to enhance its performance. Experimental results on various problems show a speedup of up to an order of magnitude over previous approaches.

What is Multi-Agent Path Planning?

MAPP optimizes the paths of multiple agents to reach their goals while avoiding collisions.



Objective

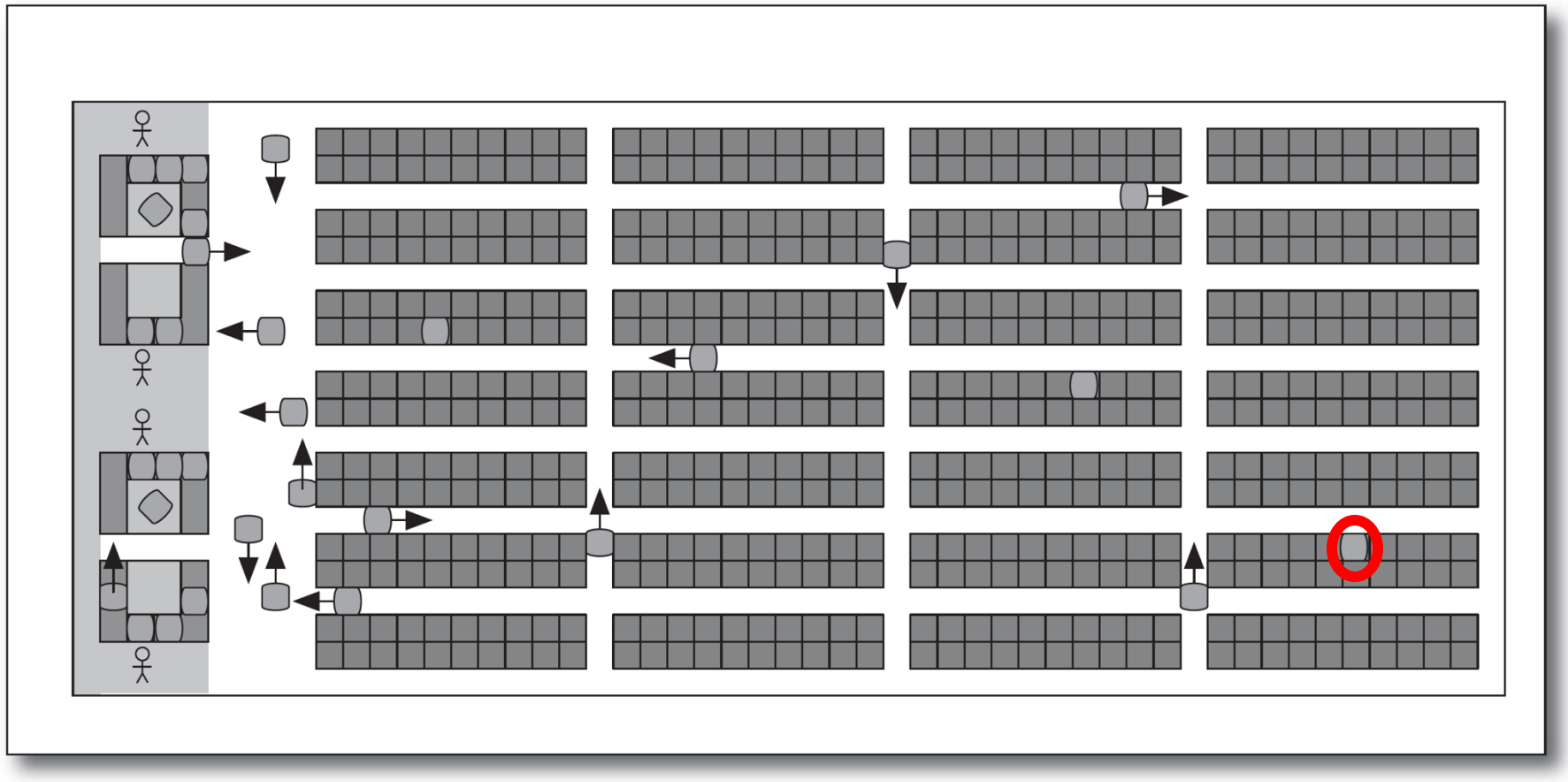
- Makespan: Task-completion time
- Flowtime: The sum of travel times

Industrial applications of MAPP[1]

- E-commerce and Retail
- Airport Surface Operation
- Transportation
- Video Games

1. Thanks to Sven Koening and his team for giving opportunity to use their lecture materials

Warehouse application



Wurman, Peter R., Raffaello D'Andrea, and Mick Mountz. "Coordinating hundreds of cooperative, autonomous vehicles in warehouses." *AI magazine* 29.1 (2008): 9-9.

Path Planning Methods

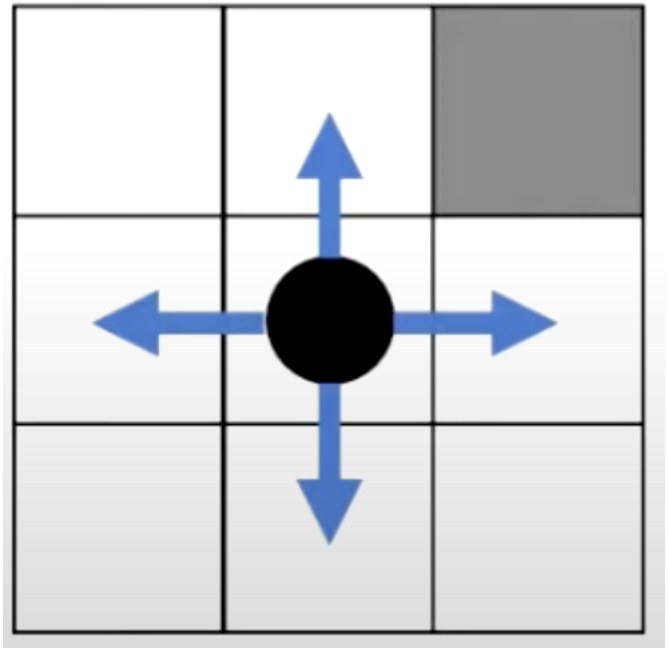
Continuous

- Sampling-Based
  - RRT
  - PRM
- Potential Fields
- Optimal Control Methods
  - MPC
  - Dynamic Programming
- Convex Optimisation

Discrete

- A and variants *
  - ECB-A*
  - ID-A*
- Prioritized Planning
- Conflict-Based Search (CBS)
- Swarm Algorithms
  - ACO
  - PSO
- MARL

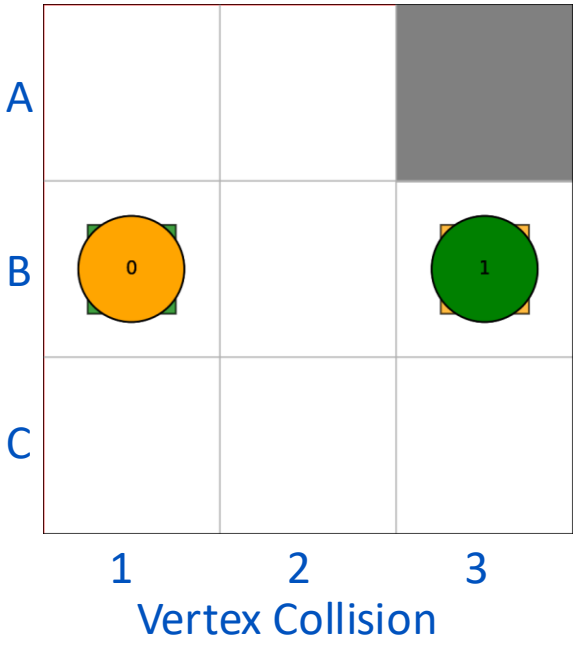
Discrete MAPP Assumptions



Simplifying assumptions

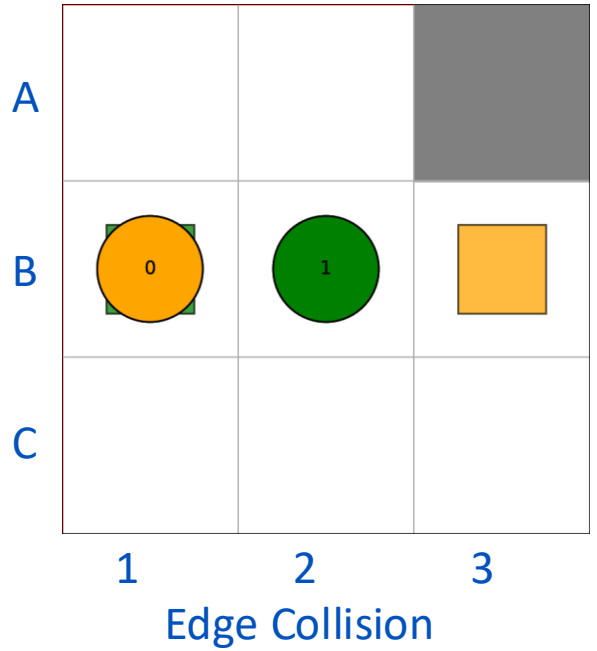
- Point agents
- No kinematic constraints
- Discretised environment

Collisions



Vertex Collision

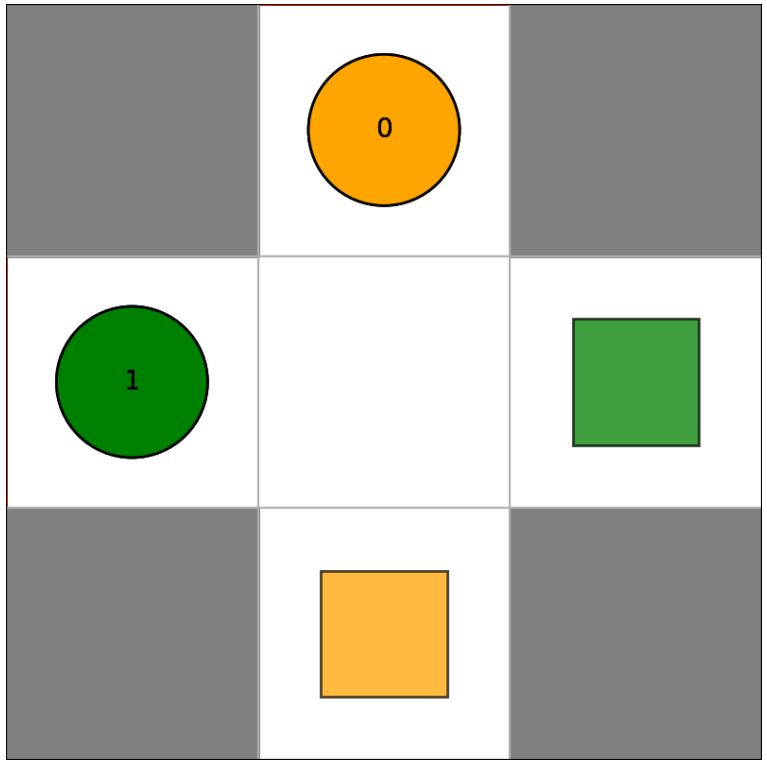
- Agent 0 moves B1 -> B2
- Agent 1 moves B3 -> B2

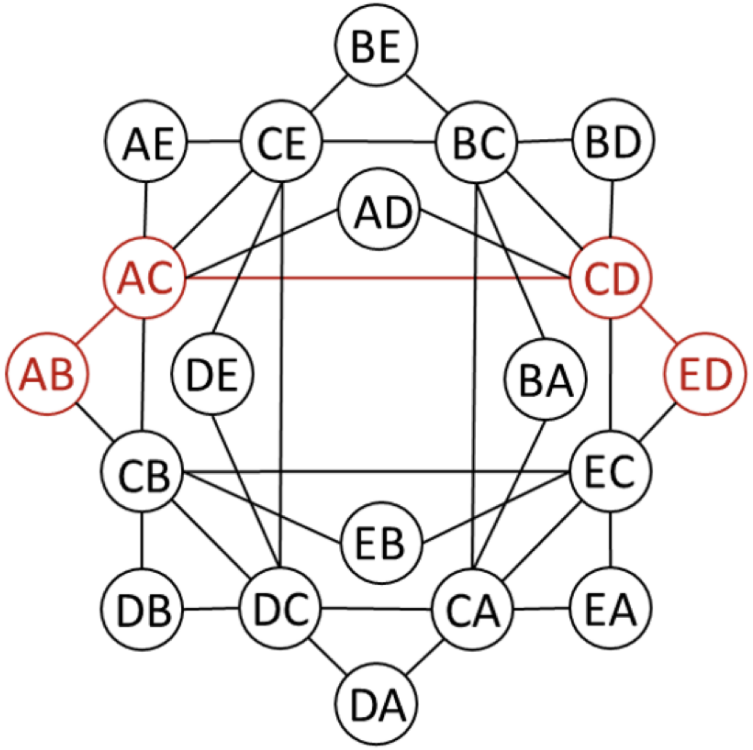


Edge Collision

- Agent 0 moves B1 -> B2
- Agent 1 moves B2 -> B1

Planning in Joint Location Space



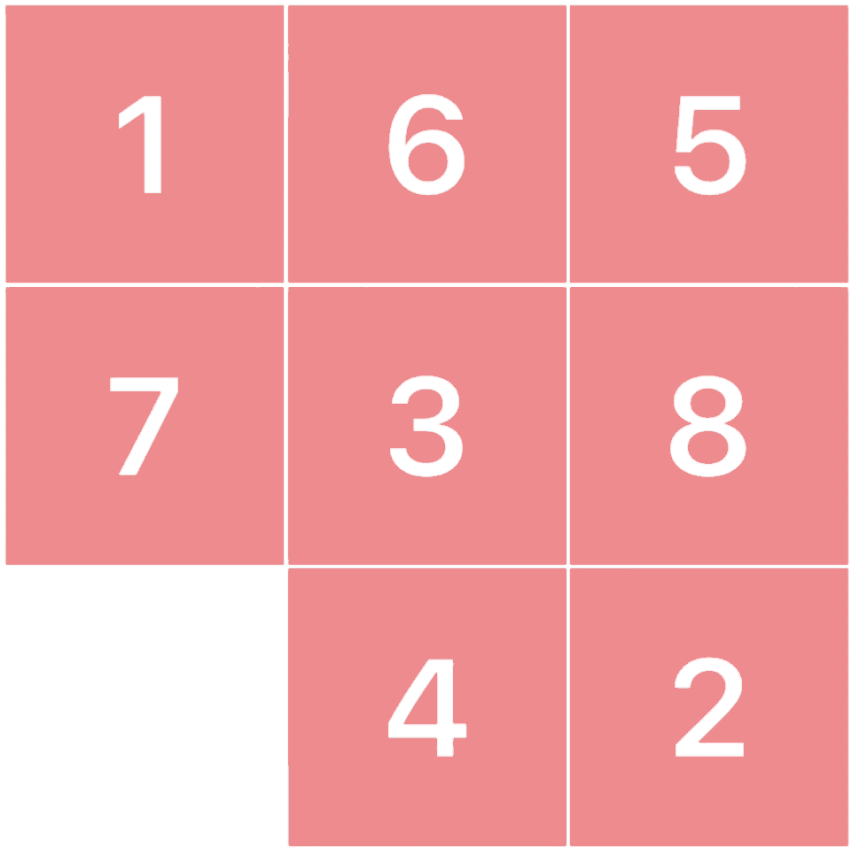


Simple

Slow

The number of vertices of the graph grows exponentially with the number of agents

Optimal MAPP algorithms



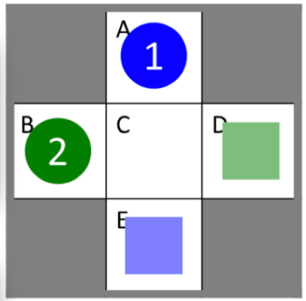
- Optimal MAPP algorithms
  - Theorem [Yu and LaValle][1] : MAPF is NP-hard to solve optimally for makespan or flowtime minimization
- Bounded-Suboptimal MAPP algorithms
  - Theorem [Ma, Tovey, Sharon, Kumar and Koenig][2] : MAPF is NP-hard to approximate within any factor less than 4/3 for makespan minimization on graphs in general

1) Yu, Jingjin, and Steven M. LaValle. "Multi-agent path planning and network flow." *Algorithmic Foundations of Robotics X: Proceedings of the Tenth Workshop on the Algorithmic Foundations of Robotics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.
2) Ma, Hang, TK Satish Kumar, and Sven Koenig. "Multi-agent path finding with delay probabilities." *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 31. No. 1. 2017.

Priority Based Search[2]

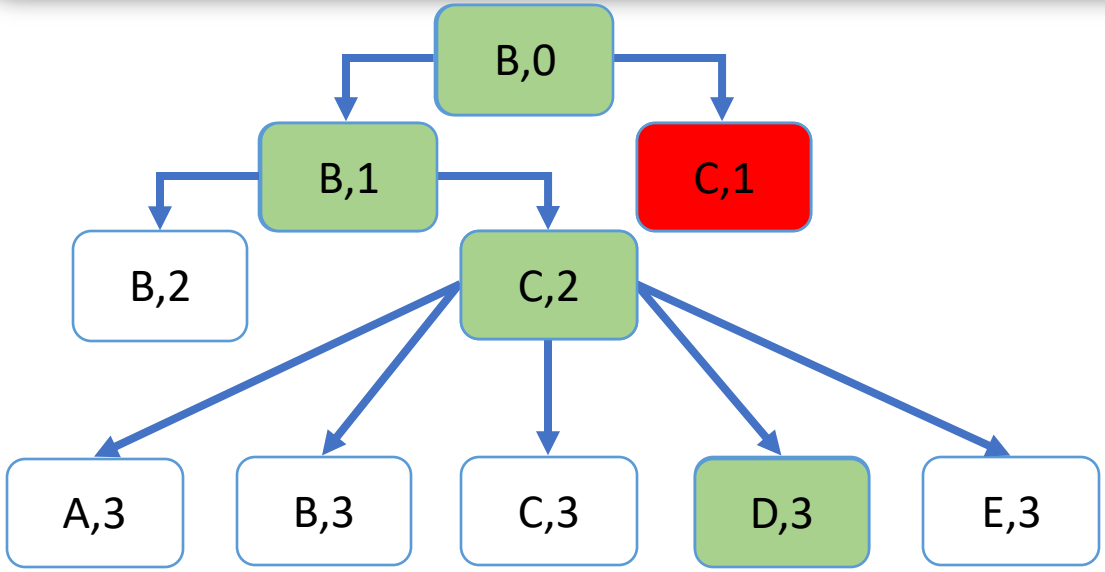
- Order the agents.
- Find paths for the agents, in order of decreasing priority.
  - Does not collide with
    - The environment
    - The paths of all higher-priority agents

- Assume that agent 1 is assigned a higher priority than agent 2
- Find the shortest path [A, C, E] for agent 1



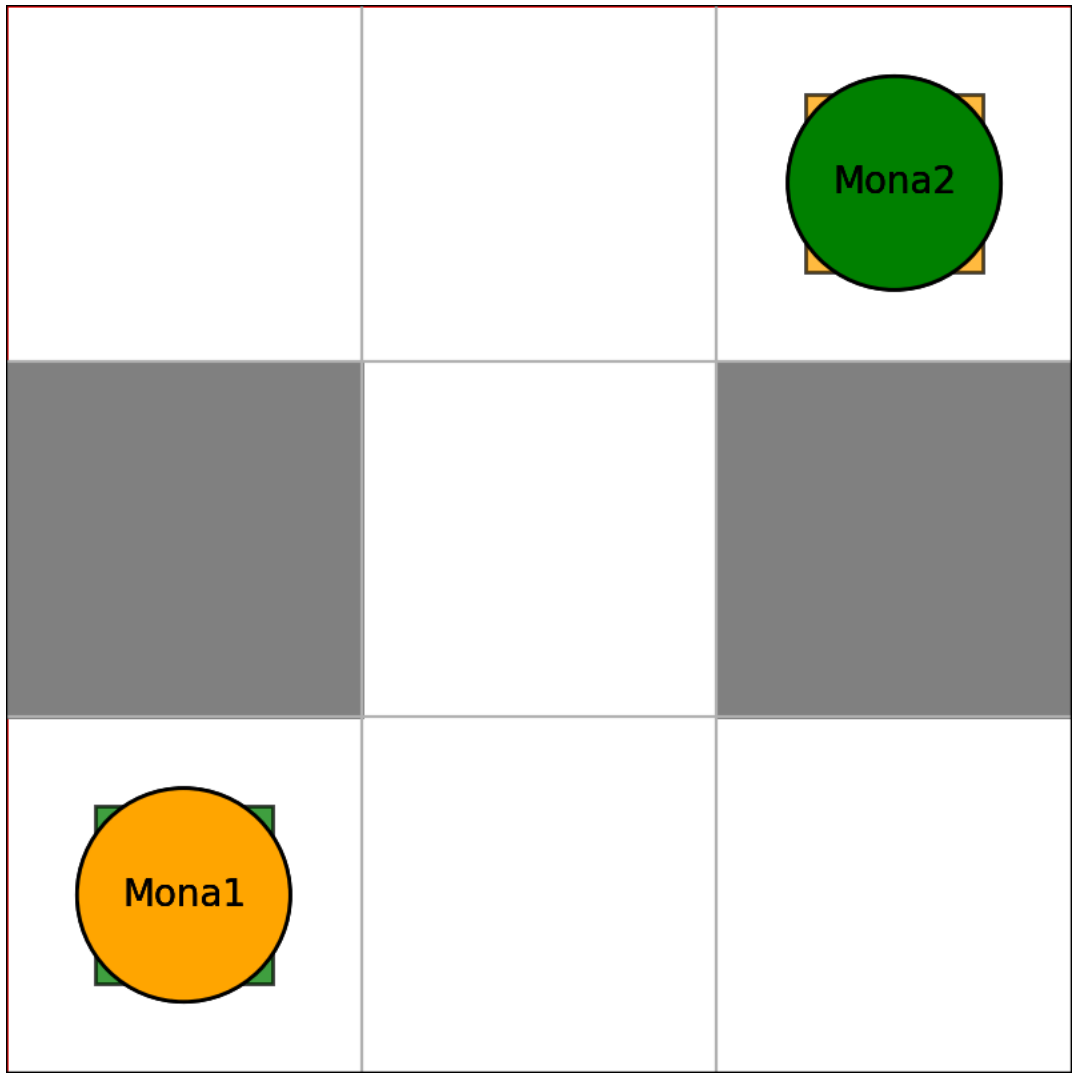
Fast

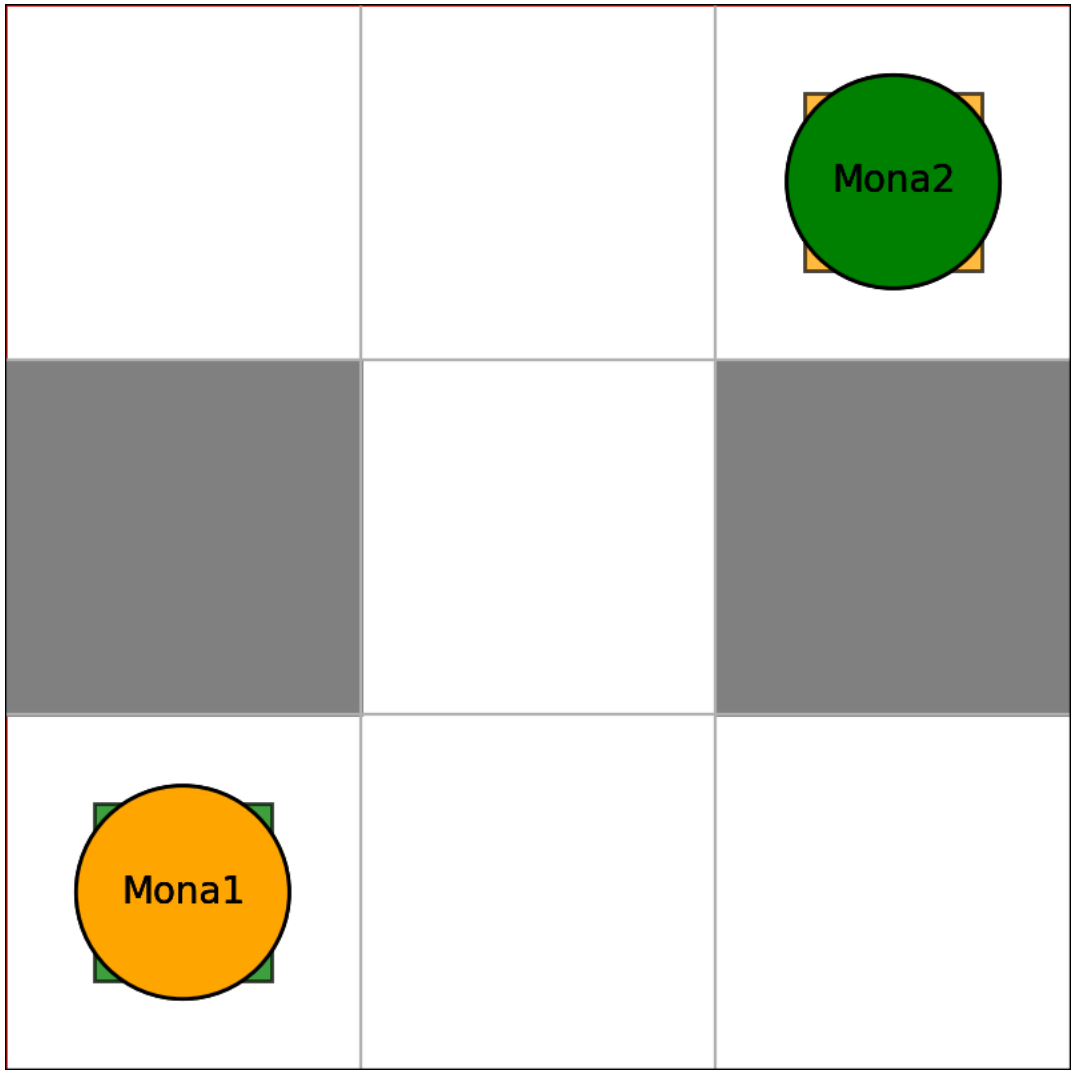
Sub-optimal
incomplete



2. M. Erdmann and T. Lozano-Pérez. On multiple moving objects. Algorithmica, 2:477–521, 1987

Drawbacks of PBS





Conflict-Based Search (CBS)[3]

A) Low-Level

- Objective: Finds the optimal route for each individual agent
- Purpose: Focuses on the specific path planning for each agent independently.
- Output: Provides the optimal path for each agent, considering its own constraints and the environment.

A) High-Level

- Objective: Checks all individual paths and ensures there are no conflicts.
- Purpose: Coordinates and resolves conflicts that might arise between the planned paths of different agents.
- Output: A conflict-free set of paths for all agents.

3. Sharon, Guni, et al. "Conflict-based search for optimal multi-agent pathfinding." *Artificial Intelligence* 219 (2015): 40-66.

Conflict-Based Search (CBS)[3]

Initialization:

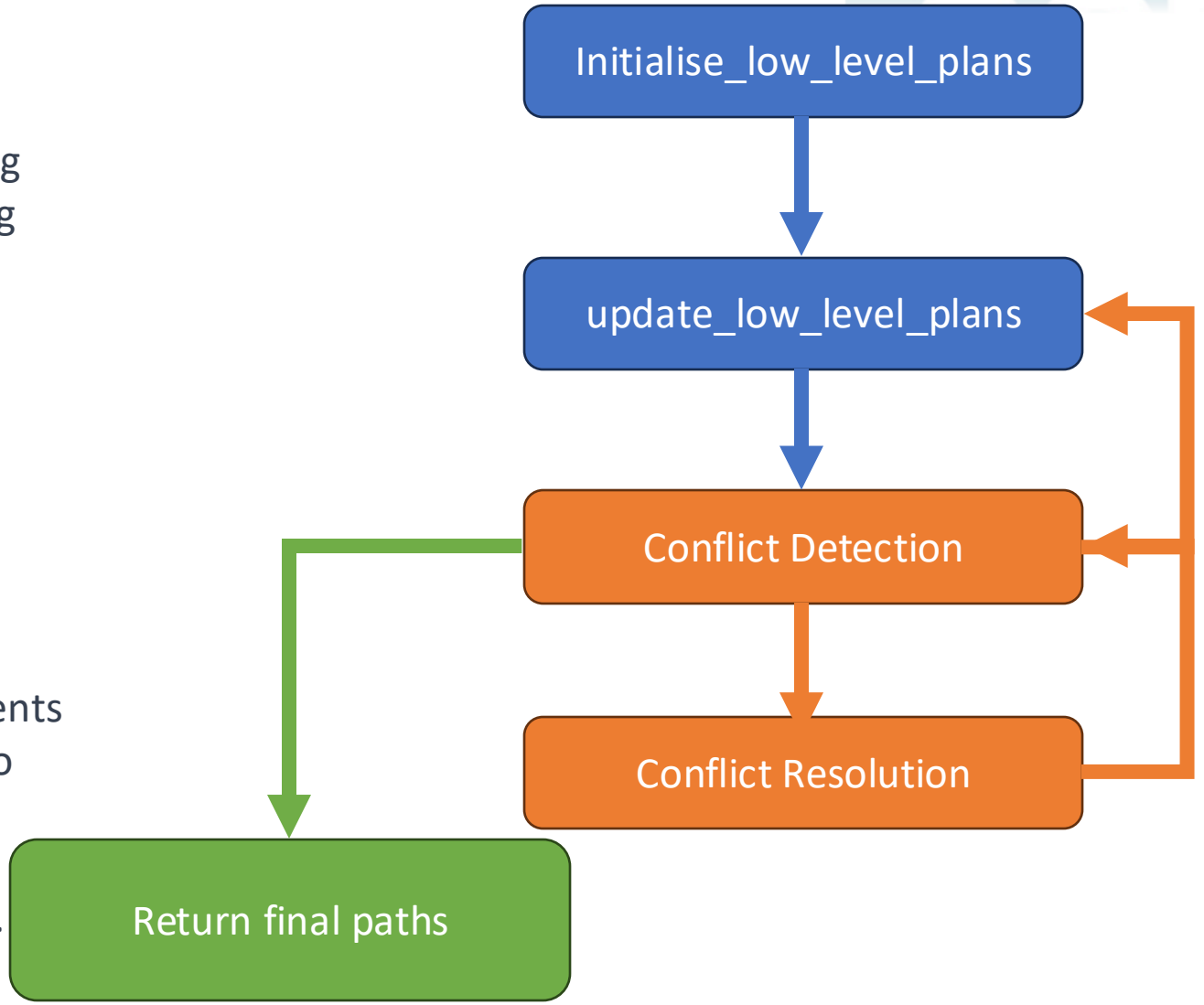
- The low_level_planner is responsible for generating the optimal path for each agent using a pathfinding algorithm. (e.g. A*, Dijkstra's)

Conflict Detection:

- It enters a loop that continues until there are no conflicts.
- The conflict_resolution function is responsible for detecting and resolving conflicts.

Conflict Resolution:

- Conflict resolution involves identifying conflicting agents and paths, determining priority, and adjusting paths to resolve conflicts.
- Prioritization can be based on factors like agent priorities, deadlines, or other domain-specific criteria.

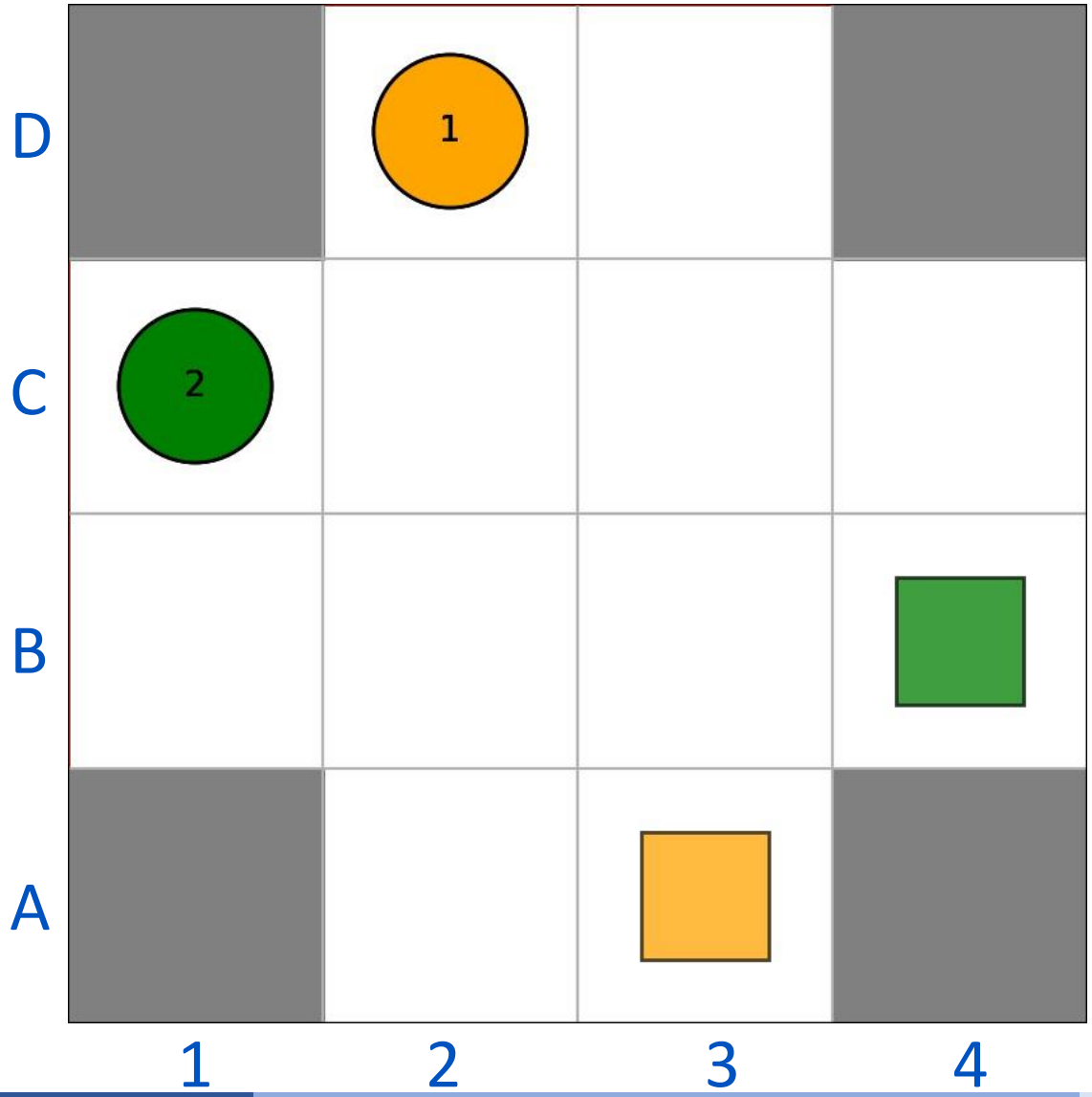


3. Sharon, Guni, et al. "Conflict-based search for optimal multi-agent pathfinding." *Artificial Intelligence* 219 (2015): 40-66.

Pseudo Code:

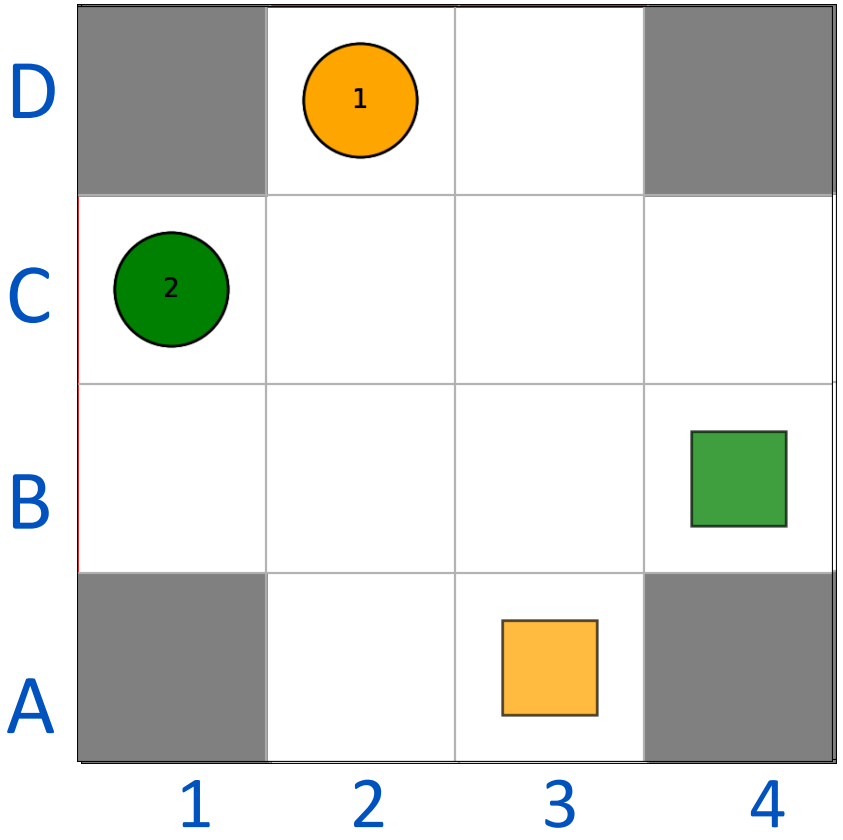
```
function ConflictBasedSearch():
        initialise low-level plans for each agent
                while conflict exists:
                        conflict_resolution()
        return final paths
function conflict_resolution():
        detect conflicts in the low-level plans
        for each conflict:
                prioritize and resolve conflict
                update low-level plans
function prioritise_and_resolve_conflict(conflict):
        identify conflicting agents and their paths
        determine priority based on some criteria
        adjust paths to resolve conflict
function initialise_low_level_plans():
        for each agent:
                plan optimal path using low-level planner

function low_level_planner(agent):
        // Use any pathfinding algorithm (e.g., A*, Dijkstra's) to find the optimal path return optimal path for the given agent
```

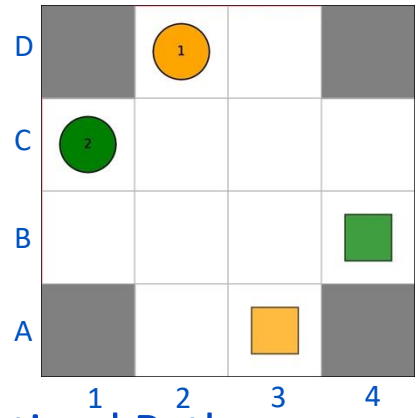
Example CBS



Example CBS

- Use the low-level search to find the optimal path for each agent





Optimal Paths

Agent1

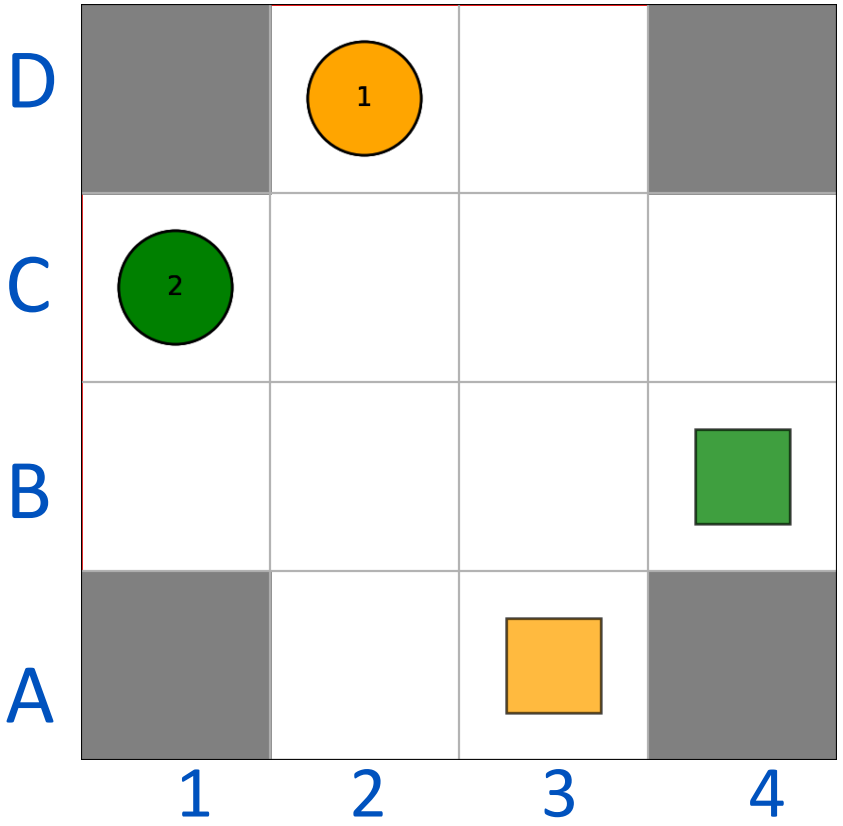
- D2,D3,C3,B3,A3
- D2,C2,C3,B3,A3
- D2,C2,B2,B3,A3
- D2,C2,B2,A2,A3

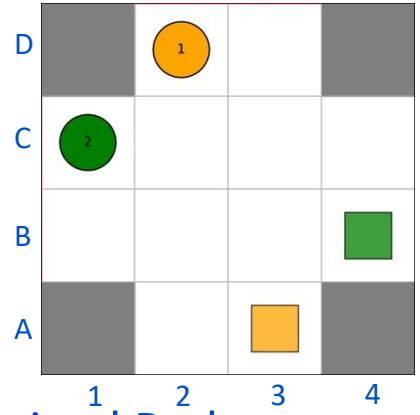
Agent2

- C1,B1,B2,B3,B4
- C1,C2,B2,B3,B4
- C1,C2,C3,B3,B4
- C1,C2,C3,C4,B4

Example CBS

Con:{} | Cost:8
1:D2,C2,B2,A2,A3
2:C1,B1,B2,B3,B4





Optimal Paths

Agent1

- D2,D3,C3,B3,A3
- D2,C2,C3,B3,A3
- D2,C2,B2,B3,A3
- D2,C2,B2,A2,A3

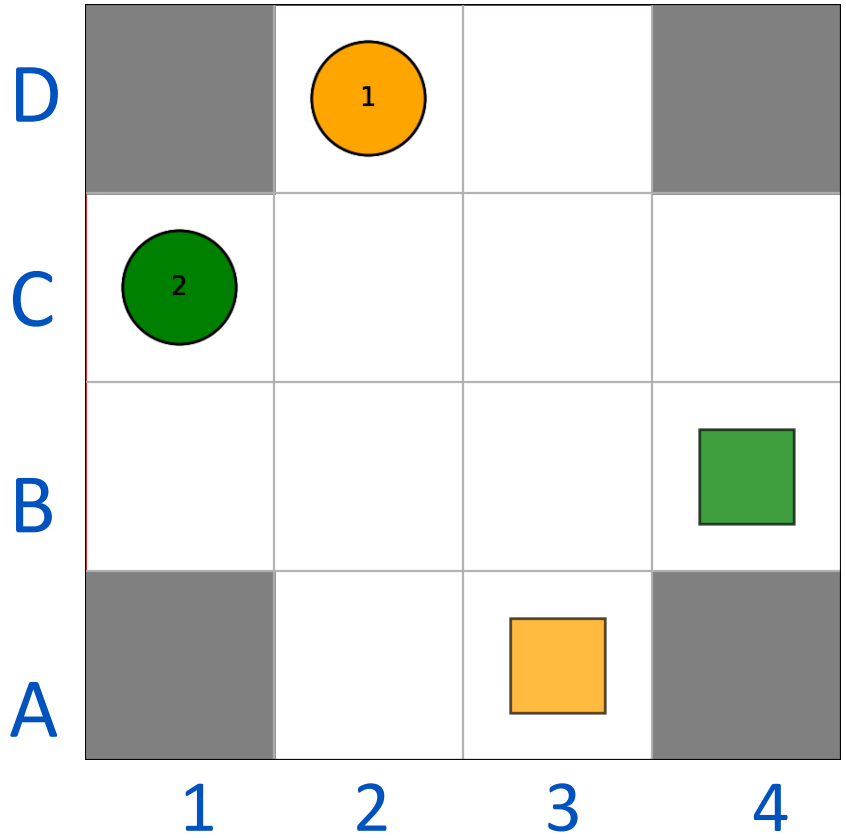
Agent2

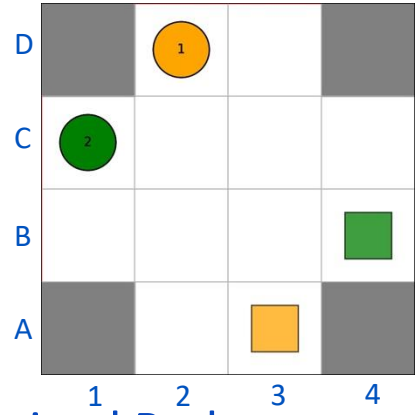
- C1,B1,B2,B3,B4
- C1,C2,B2,B3,B4
- C1,C2,C3,B3,B4
- C1,C2,C3,C4,B4

Example CBS

Con:{} | Cost:8
1:D2,C2,B2,A2,A3
2:C1,B1,B2,B3,B4

Con:{1,B2,2} | Cost:8
1:D2,D3,C3,B3,A3
2:C1,B1,B2,B3,B4





Optimal Paths

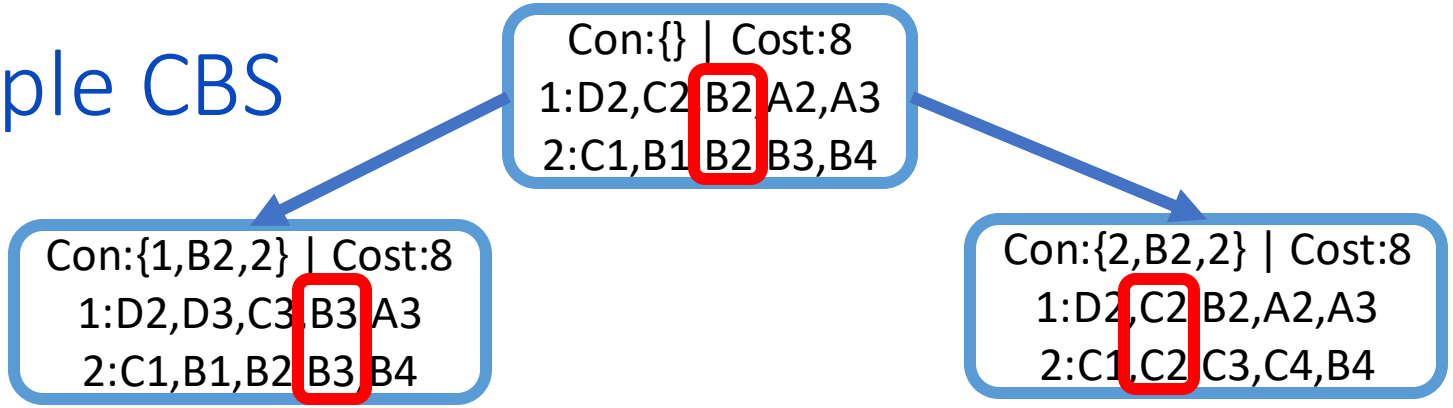
Agent1

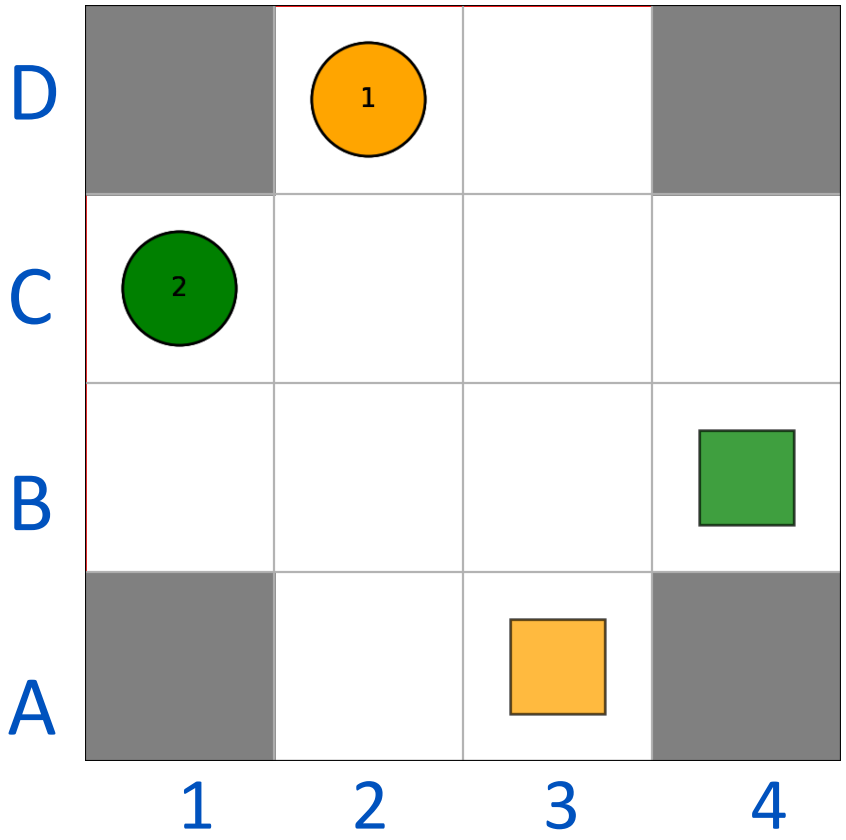
- D2,D3,C3,B3,A3
- D2,C2,C3,B3,A3
- D2,C2,B2,B3,A3
- D2,C2,B2,A2,A3

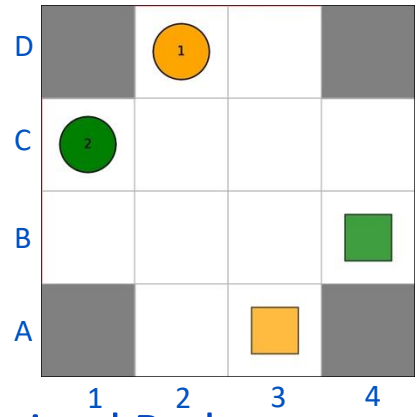
Agent2

- C1,B1,B2,B3,B4
- C1,C2,B2,B3,B4
- C1,C2,C3,B3,B4
- C1,C2,C3,C4,B4

Example CBS







Optimal Paths

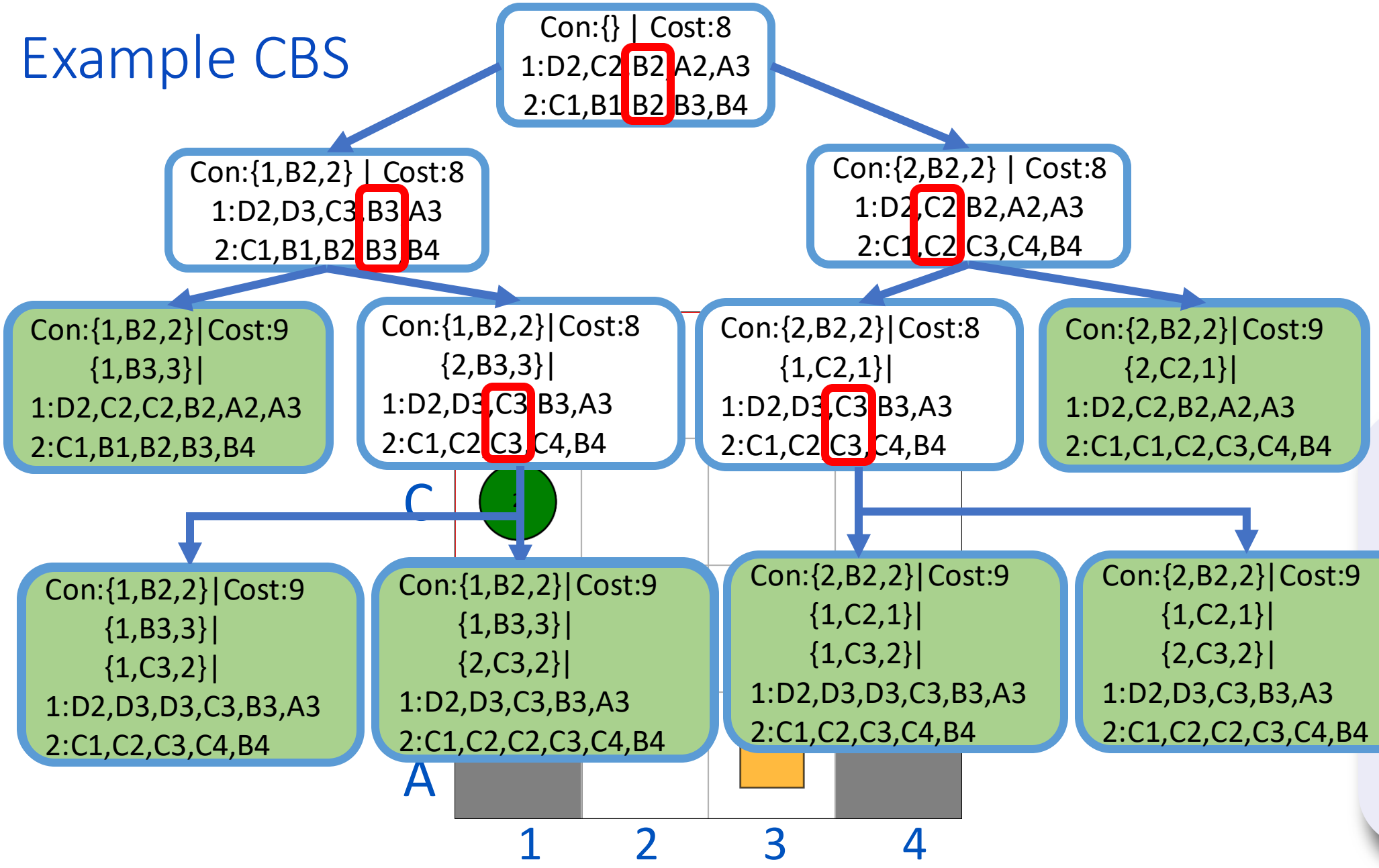
Agent1

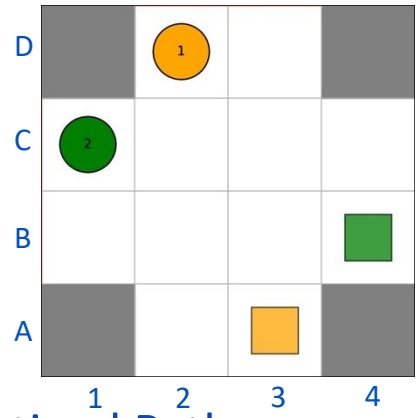
- D2,D3,C3,B3,A3
- D2,C2,C3,B3,A3
- D2,C2,B2,B3,A3
- D2,C2,B2,A2,A3

Agent2

- C1,B1,B2,B3,B4
- C1,C2,B2,B3,B4
- C1,C2,C3,B3,B4
- C1,C2,C3,C4,B4

Example CBS





Optimal Paths

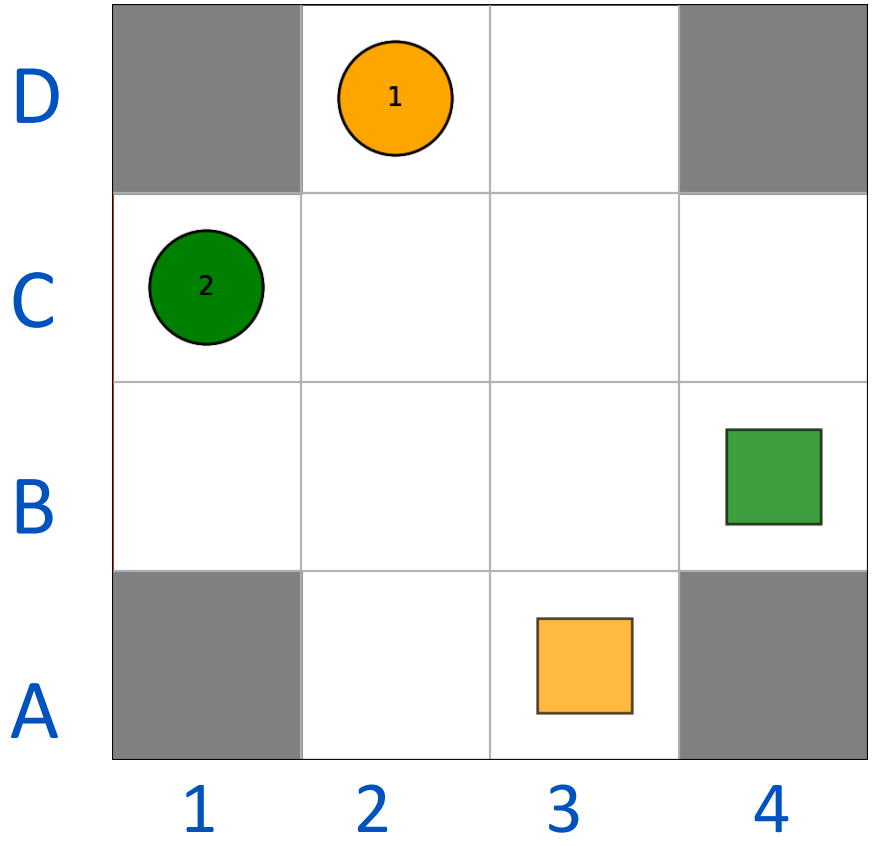
Agent1

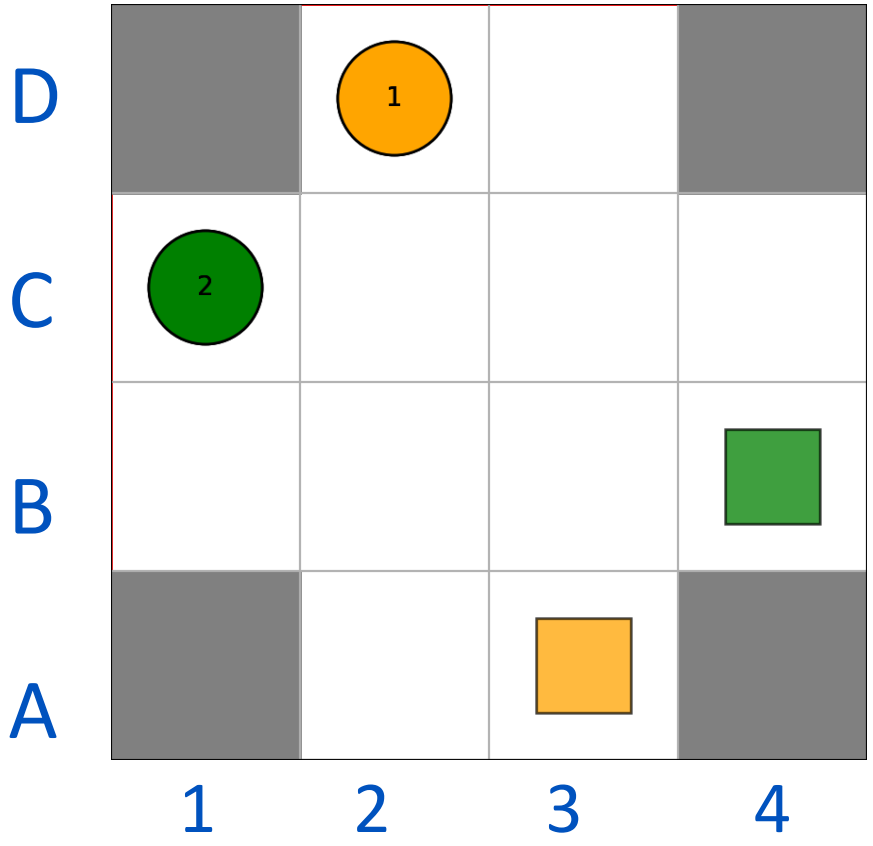
- D2,D3,C3,B3,A3
- D2,C2,C3,B3,A3
- D2,C2,B2,B3,A3
- D2,C2,B2,A2,A3

Agent2

- C1,B1,B2,B3,B4
- C1,C2,B2,B3,B4
- C1,C2,C3,B3,B4
- C1,C2,C3,C4,B4

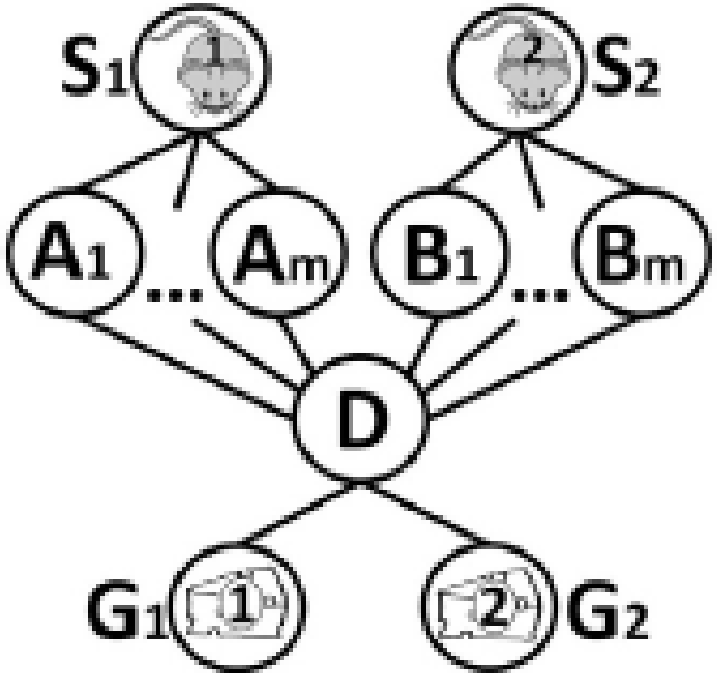
Solutions



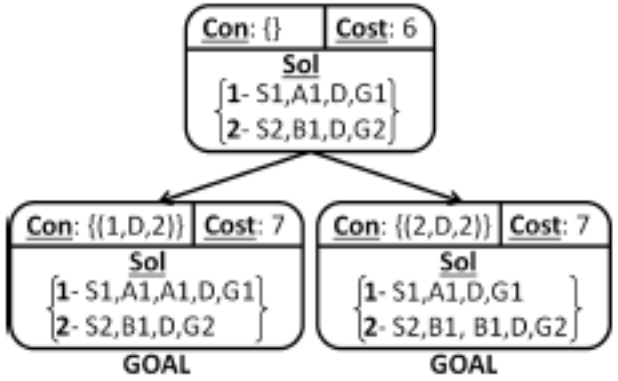


A* vs CBS (bottlenecks)

- In many cases two-level formulation enables CBS to examine fewer states than A* while still maintaining optimality.

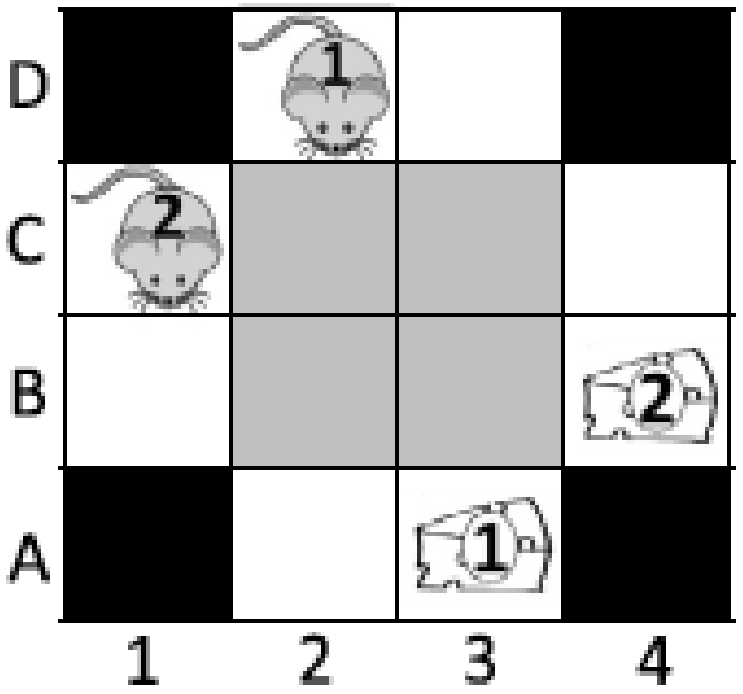


- Each mouse must plan a full path to its respective piece of cheese
- CBS expands fewer nodes than A*



3. Sharon, Guni, et al. "Conflict-based search for optimal multi-agent pathfinding." *Artificial Intelligence* 219 (2015): 40-66.

A* vs CBS (open space)



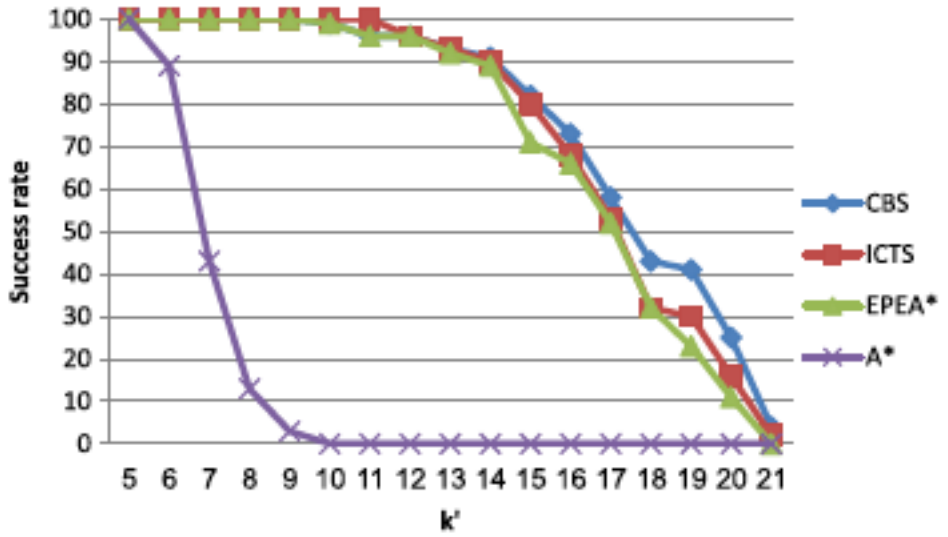
- A* can easily sorted out the conflicted zone by expanding 5 nodes with f=8 and 3 nodes with f=9
- CBS's CT has 5 non-goal nodes with cost 8, and 6 goal nodes with cost 9. In total CBS will expand 54 low-level nodes.

3. Sharon, Guni, et al. "Conflict-based search for optimal multi-agent pathfinding." *Artificial Intelligence* 219 (2015): 40-66.

Empirical Analysis

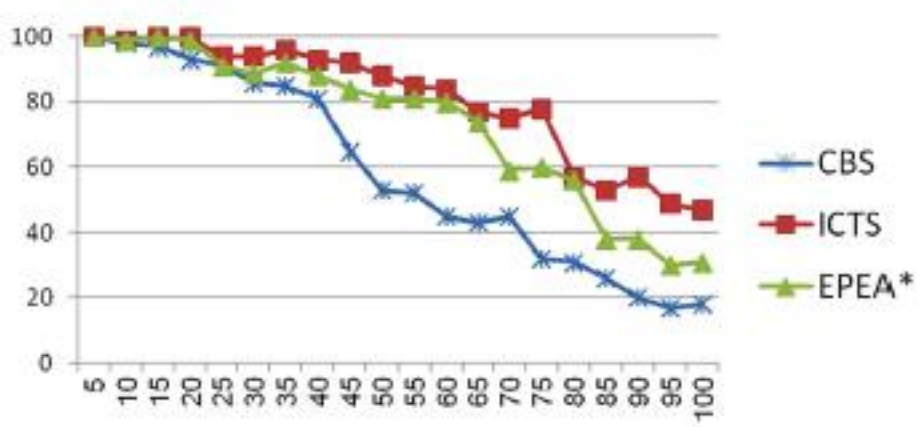
k'	nodes			Run-time (ms)		
	A*	CBS(hl)	CBS(ll)	A*	CBS	p-val
3	640	10	490	8	7	0.01
4	3965	24	1048	207	14	0.02
5	21,851	51	2385	3950	32	0.01
6	92,321	45	1354	37,398	20	0.09
7	NA	117	3994	NA	60	0.00
8	NA	266	8644	NA	148	0.00
9	NA	1362	45,585	NA	879	0.01
10	NA	3225	111,571	NA	2429	0.02
11	NA	8789	321,704	NA	7712	0.44
12	NA	12,980	451,770	NA	**12,363**	0.36
13	NA	15,803	552,939	NA	**16,481**	0.50
14	NA	21,068	736,278	NA	24,441	0.14
15	NA	24,871	826,725	NA	**30,509**	0.45
16	NA	24,602	822,771	NA	**34,230**	0.32
17	NA	17,775	562,575	NA	**25,653**	0.22

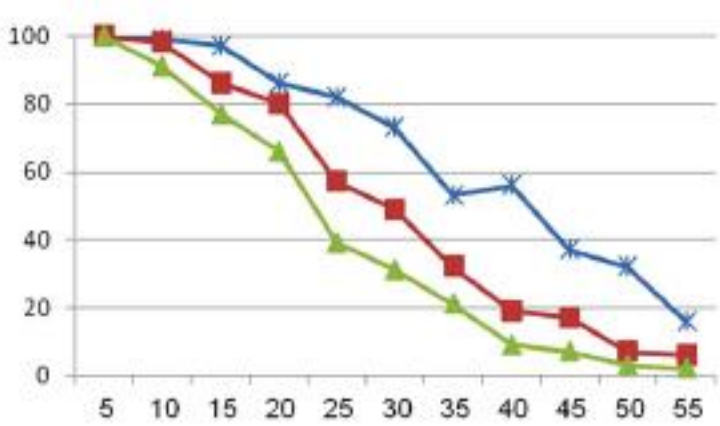
- 8x8 4-connected open grid
- Number of agents are 3 to 21
- Time limit is 5 minutes

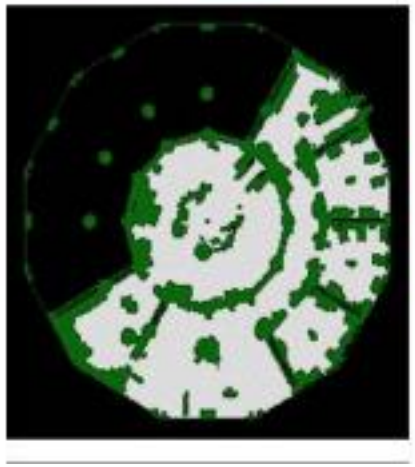


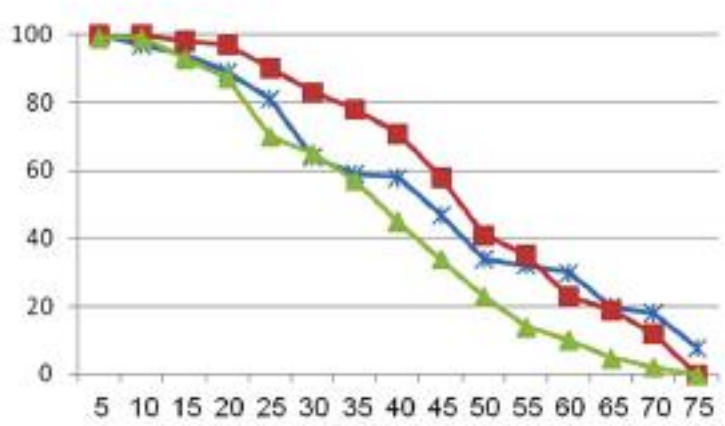
3. Sharon, Guni, et al. "Conflict-based search for optimal multi-agent pathfinding." *Artificial Intelligence* 219 (2015): 40-66.

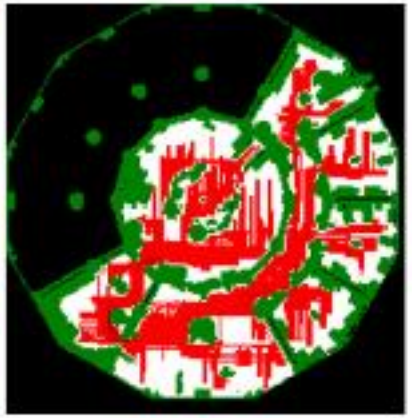
Dragon Age Maps











Meta-Agent Conflict Based Search (MA-CBS)

- Understanding the relation between map topologies and MAPP algorithms performance
- MA-CBS is dynamically adapting algorithms.
- MA-CBS is automatically identifying sets of strongly coupled agents and merging them into a meta-agent.
- This meta-agent is treated as a single agent.
- The low-lever solver of MA-CBS must be a MAPP solver, e.g. A*, M*, EPEA*
- MA-CBS is in fact a framework that can be used on top of another MAPF solver

Dragon Age Maps with Meta Agent Techniques