Multi-Agent Path Finding in Configurable Environments

Matteo Bellusci
matteo.bellusci@mail.polimi.it
CSE Track

POLITECNICO MILANO 1863

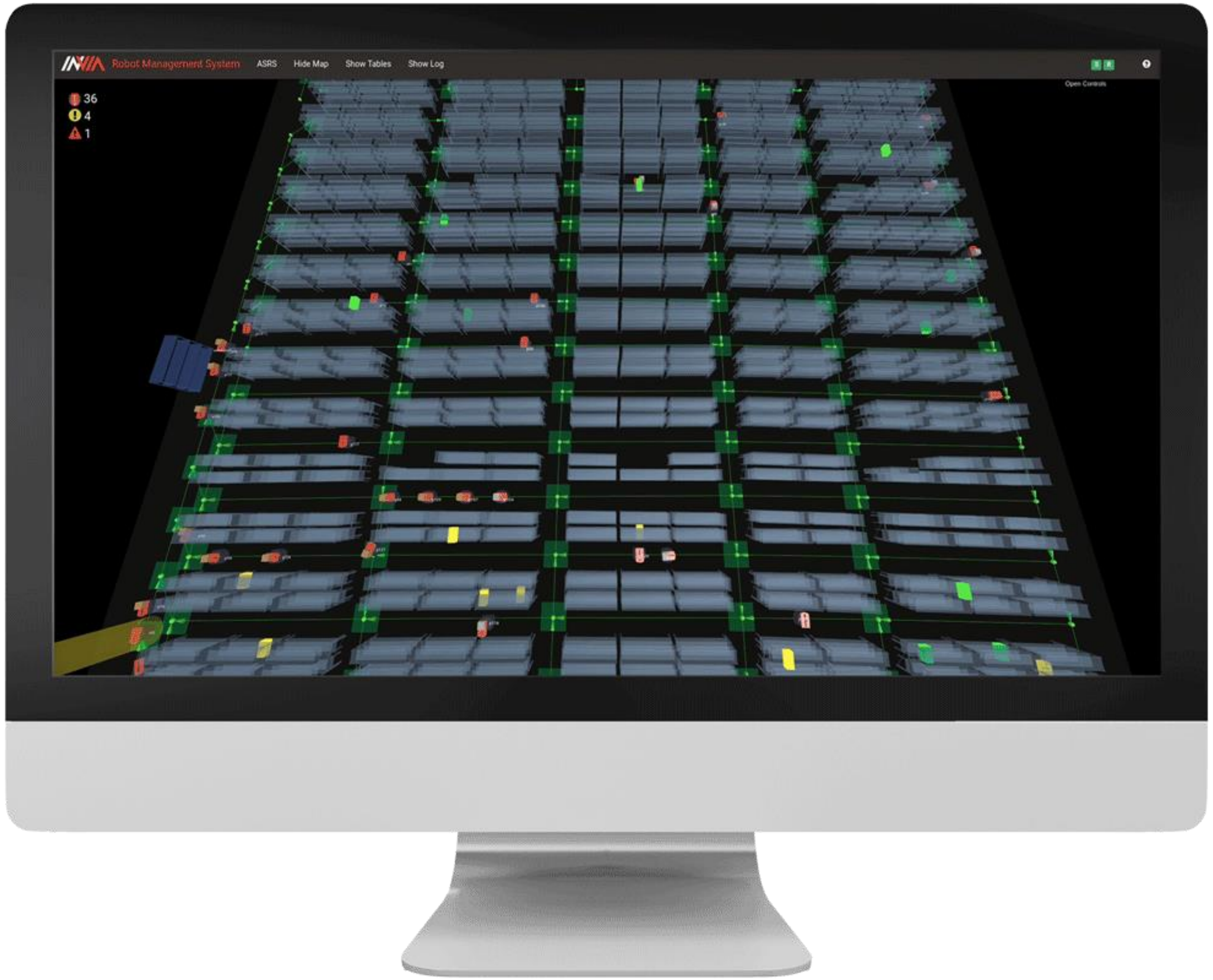
HONOURS PROGRAMME HP-SR in Information Technology

Introduction

- Multi Agent Path Finding (MAPF) is a challenging Artificial Intelligence topic that has many applications

- The task in MAPF problems is to find non-interfering paths for multiple agents, each one with a unique start and goal position in a known environment

Applications



Applications

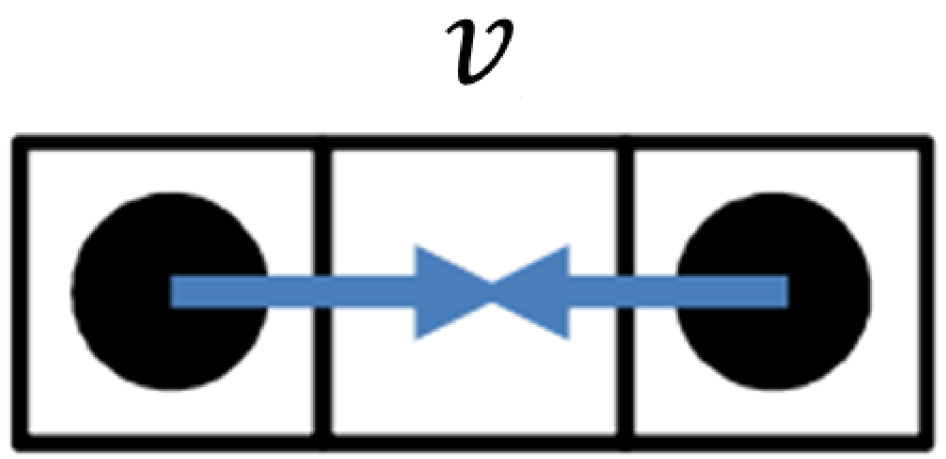
Preliminaries

- In the MAPF problem we are given:
  - A graph $G = (V, E)$
  - A set of k labeled agents $A = \{a_1, \dots, a_k\}$
  - A set of k start positions $s = \{s_1, \dots, s_k\}$, $s_i \in V$
  - A set of k goal positions $g = \{g_1, \dots, g_k\}$, $g_i \in V$
- Goal: Find a set of k non-conflicting paths, one for each agent, that minimize some objective function
  - Makespan: latest arrival time of an agent at its goal location
  - Flowtime (SIC): sum of the arrival times of all the agents at their goal locations

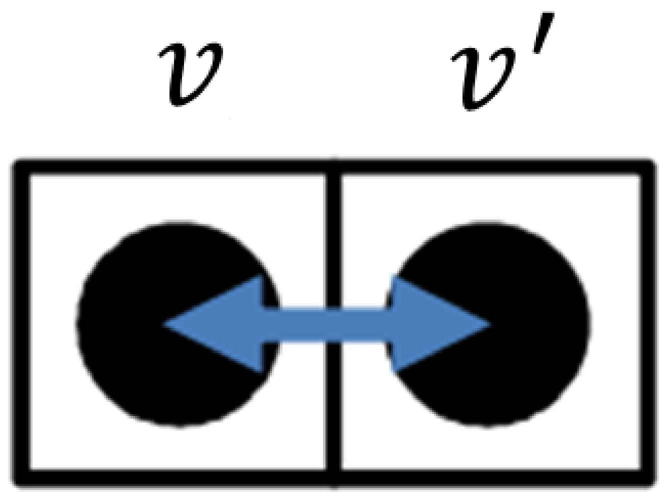
Preliminaries

- Time is discretized into timesteps and, at each timestep, every agent can either change location moving to an adjacent vertex or wait at its current position

- A **vertex conflict** is a tuple (a_1, a_2, v, t) meaning that agents a_1 and a_2 are occupying the same vertex v at timestep t



- An **edge conflict** is a tuple (a_1, a_2, v, v', t) meaning that, from timestep t to $t+1$, agent a_1 is traveling from v to v' while a_2 traveling from v' to v

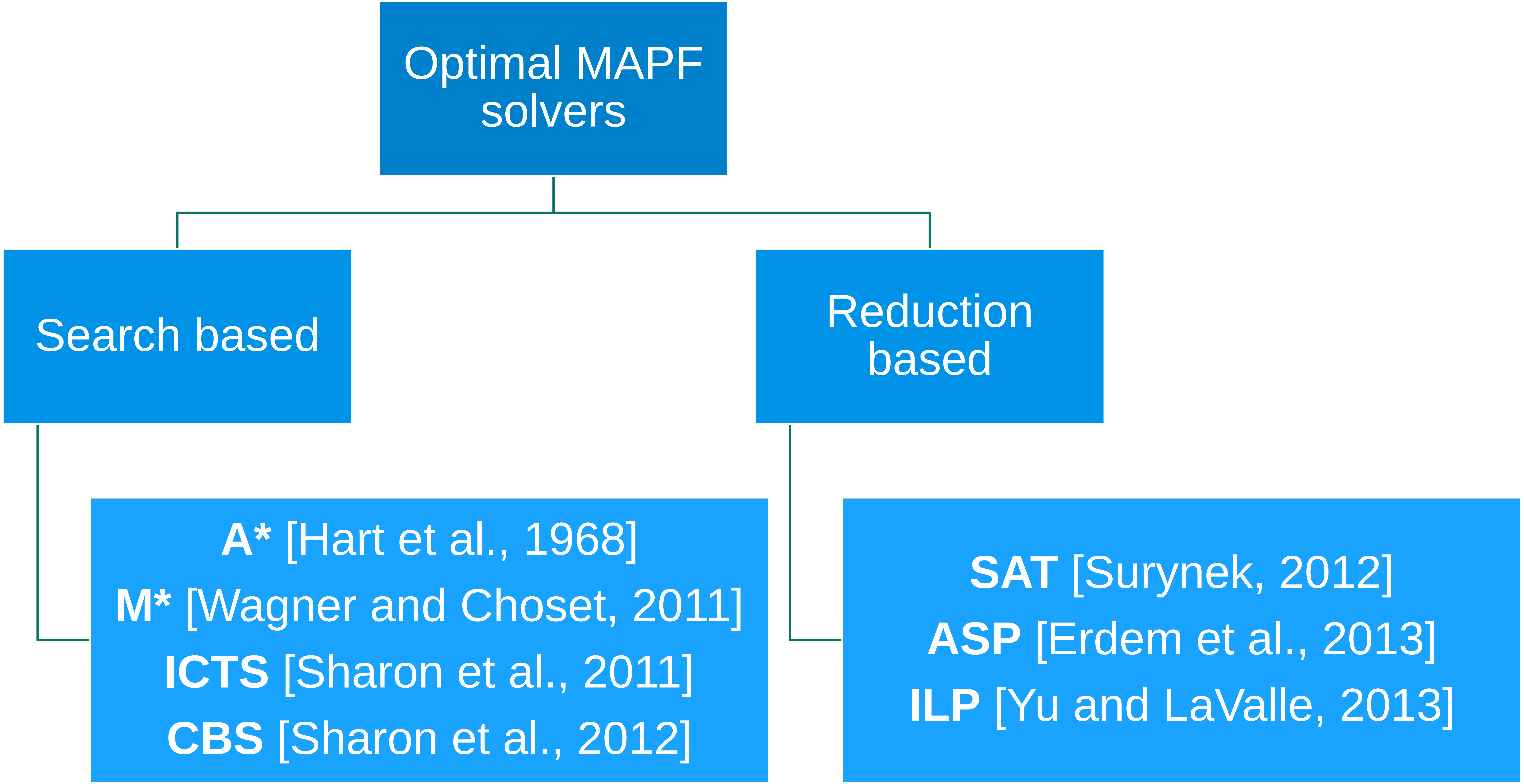


- A **path** π_i for an agent a_i can be modeled as a sequence of vertices $(v_0, \dots, v_m)$ which brings a_i from $v_0 = s_i$ to $v_m = g_i$

- A **plan** is a set of k paths

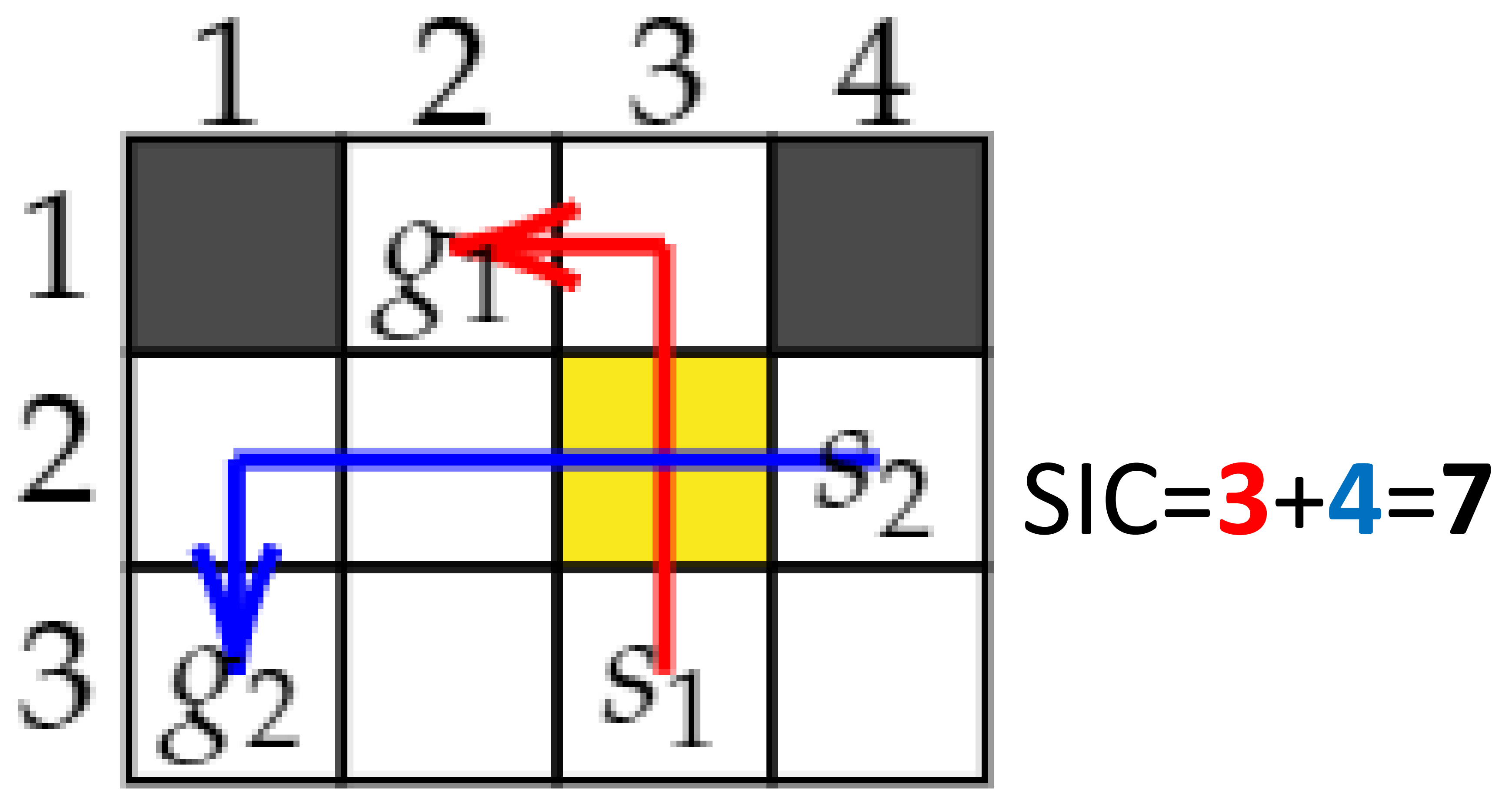
- A **solution** for the MAPF problem is a conflict-free (no collisions) plan

Optimal MAPF solvers

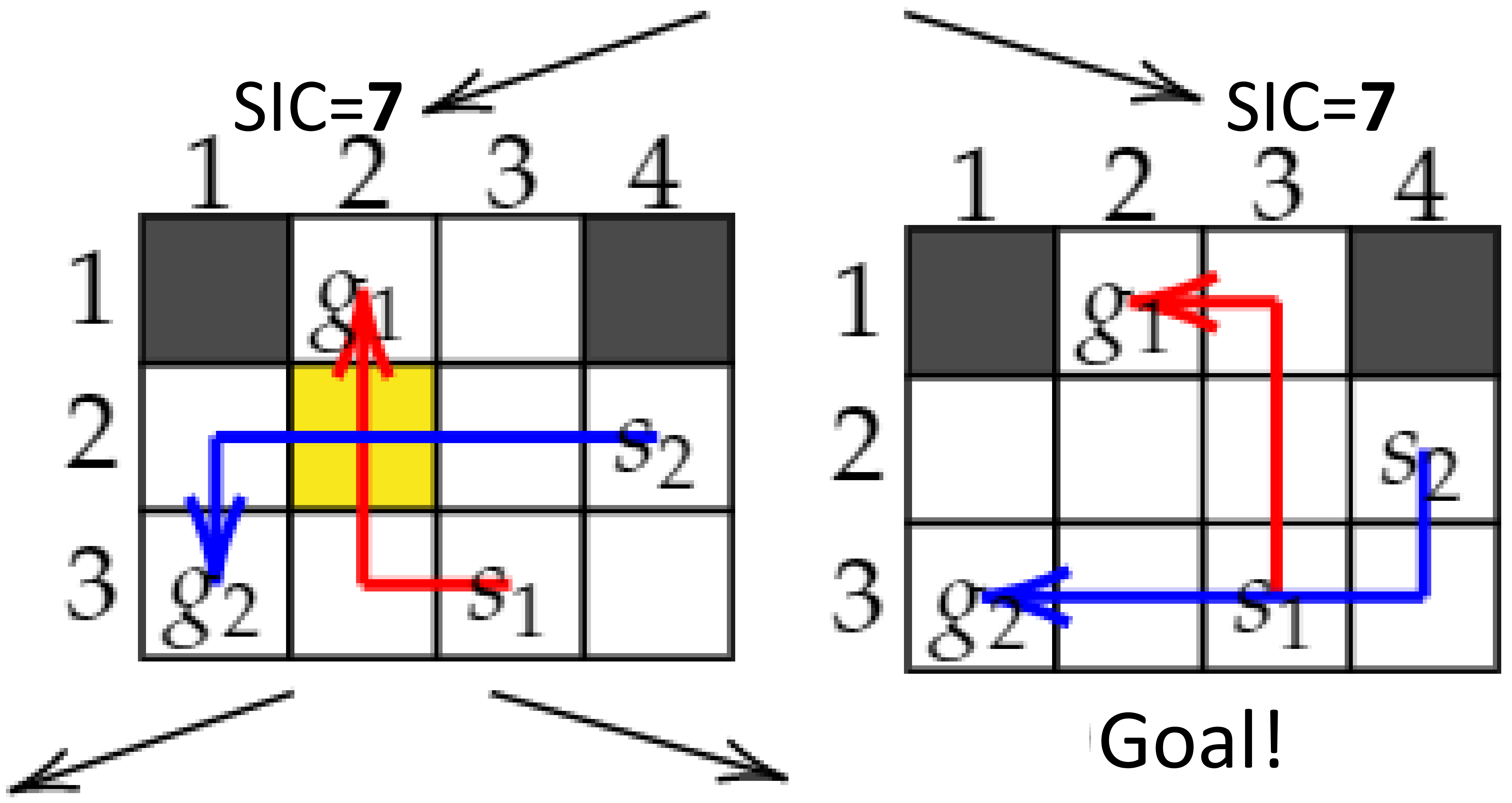


Conflict-Based Search

The red and blue agents collide in the yellow cell (x=3,y=2) at time 1



Add constraint: the red agent is not allowed to be in cell (3,2) at time 1



Add constraint: the blue agent is not allowed to be in cell (3,2) at time 1

Conflict-Based Search

- We have just seen the general idea behind CBS-based solvers
- An improved version of CBS is available with the name of Improved Conflict-Based Search (ICBS) [Boyarski et al., 2015]
  - ICBS-h (or CBSH) [Felner et al., 2018] is the state-of-the-art CBS-based solver

Non-static environments

- In classical MAPF problems, we make strong assumptions concerning the environment

- Real-world environments may have different features and properties

  - Dynamic obstacles

  - Destructible obstacles

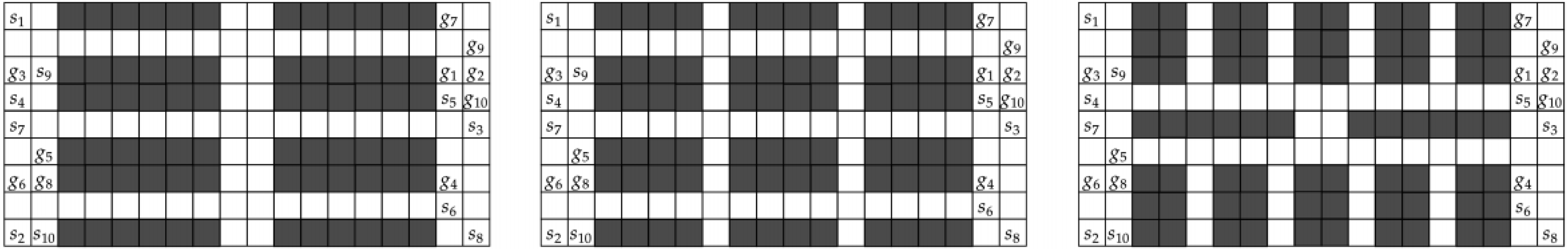
  - Shrinking maps

  - Toll booths

Configurable Environments

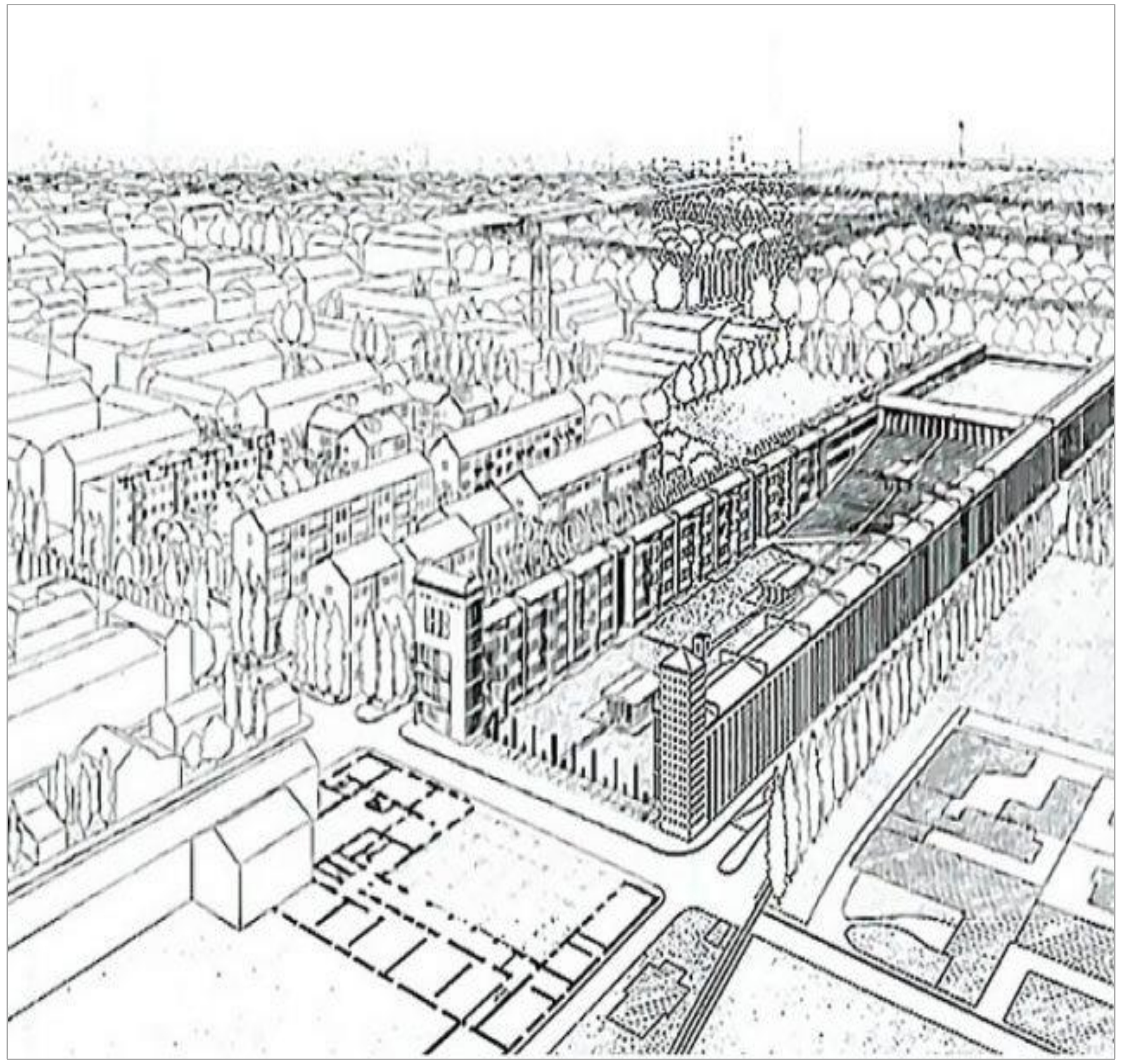
- We introduce MAPF in Configurable environments (C-MAPF), a novel variant of the MAPF problem where the environment is configurable

- An environment is said to be configurable when its structure and topology can be controlled within some given constraints



(a) (b) (c)

MAPF in configurable environments



MAPF in configurable environments

- In the C-MAPF problem we are given:
  - A non-empty **family** $\mathcal{G} = \{G_1, \ldots, G_n\}$ of n graphs $G_j = (V_j, E_j)$
    - Define $V^{\cap} = \bigcap_j V_j$, $E^{\cap} = \bigcap_j E_j$, $V^{\cup} = \bigcup_j V_j$, and $E^{\cup} = \bigcup_j E_j$
  - A set of k labeled agents $A = \{a_1, \ldots, a_k\}$
  - A set of k start positions $s = \{s_1, \ldots, s_k\}$, $s_i \in V^{\cap}$
  - A set of k goal positions $g = \{g_1, \ldots, g_k\}$, $g_i \in V^{\cap}$
- A path π_i for an agent a_i is **applicable** in a graph $G_j \in \mathcal{G}$ if it possible to simulate π_i in G_j
- A **solution** $\langle \pi, G \rangle$ for the C-MAPF problem is a conflict-free plan $\pi = \{\pi_1, \ldots, \pi_k\}$ and a configuration $G \in \mathcal{G}$ such that all the paths in π are applicable in G

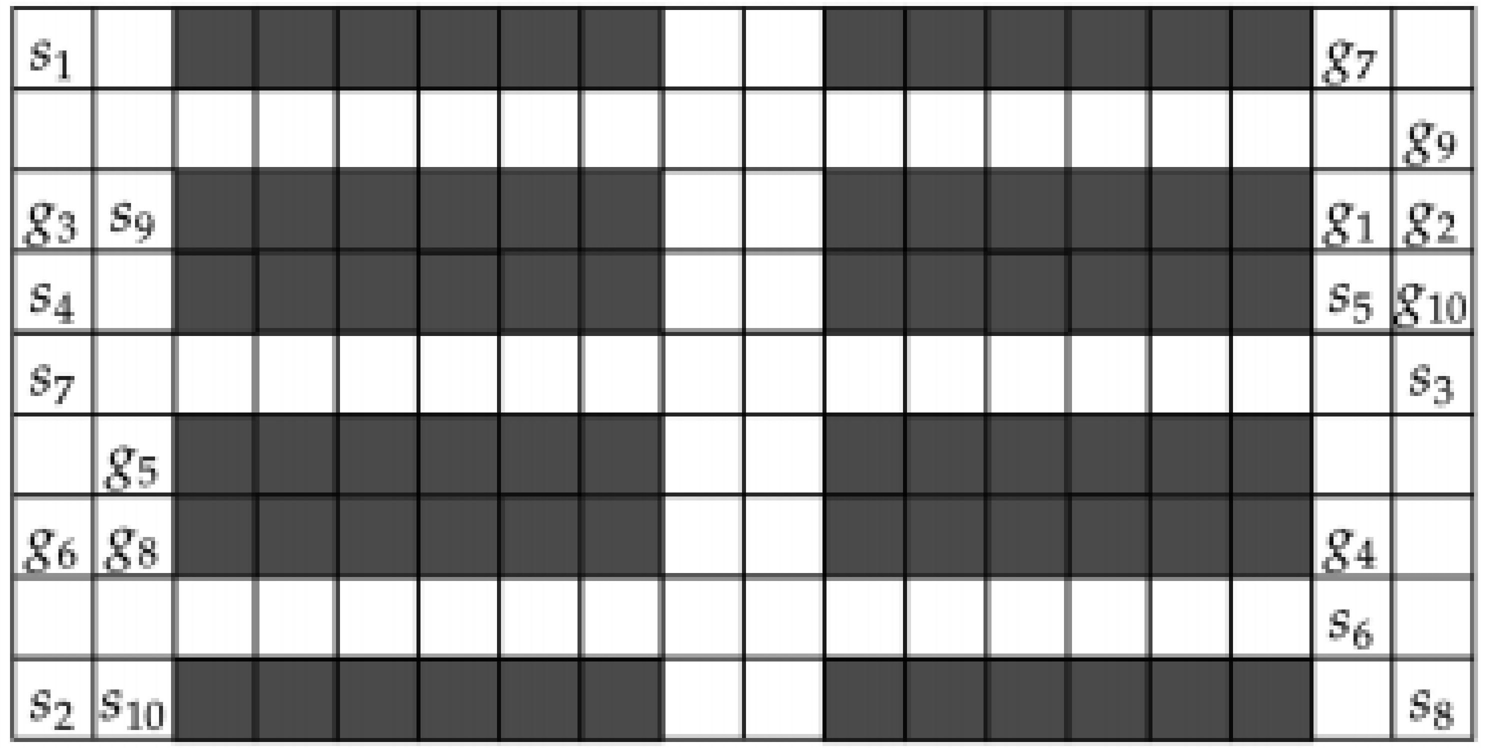
NP-hardness

- The C-MAPF problem is trivially a generalization of the MAPF problem
- Theorem [Yu and LaValle, 2013]: MAPF problems are NP-hard to solve optimally for both makespan and flowtime minimization

Brute-force C-MAPF solver

- The C-MAPF problem can be solved in a brute-force manner by invoking any MAPF solver
- Note that solvability for MAPF problems can be determined in polynomial time [Yu and Rus, 2015]

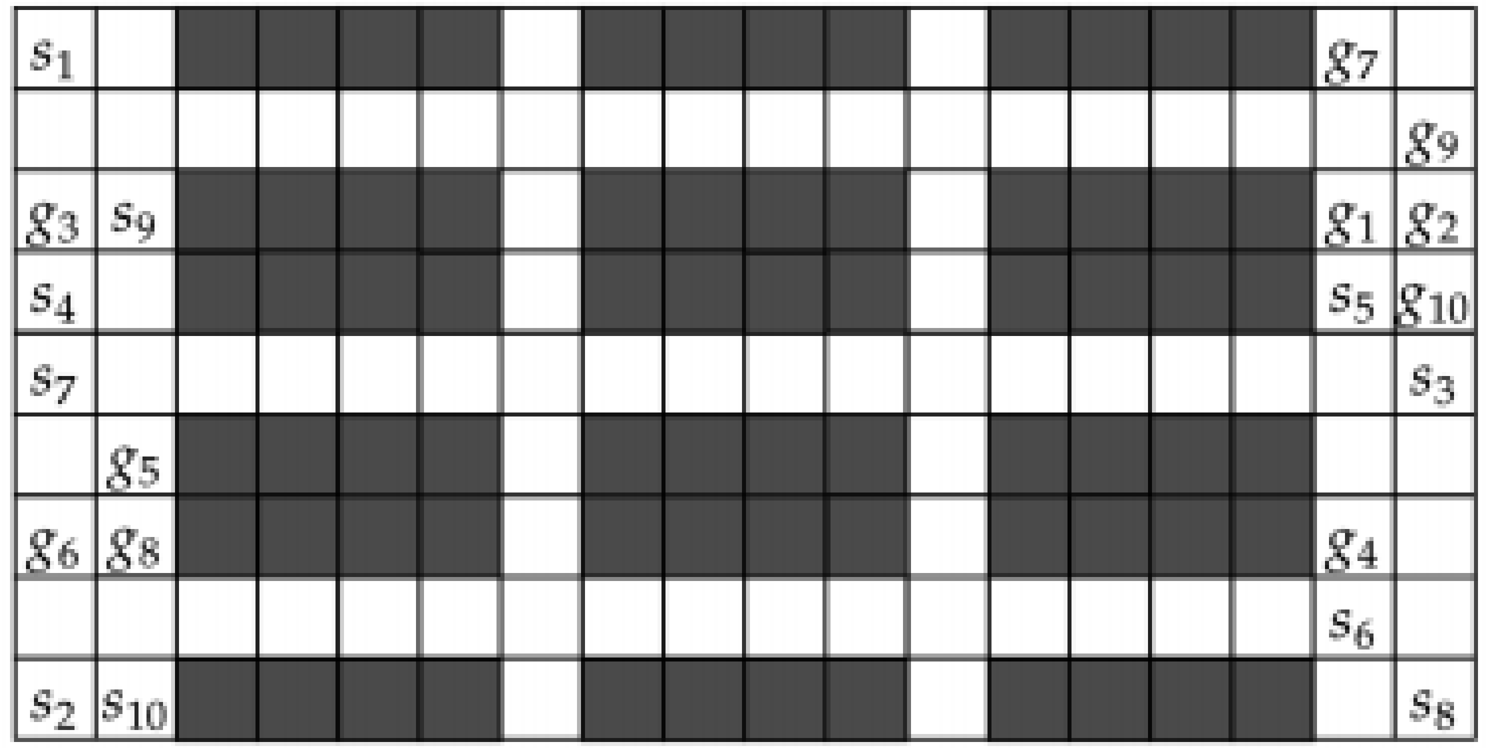
MAPF solver



(a)

Obtain plan π_a

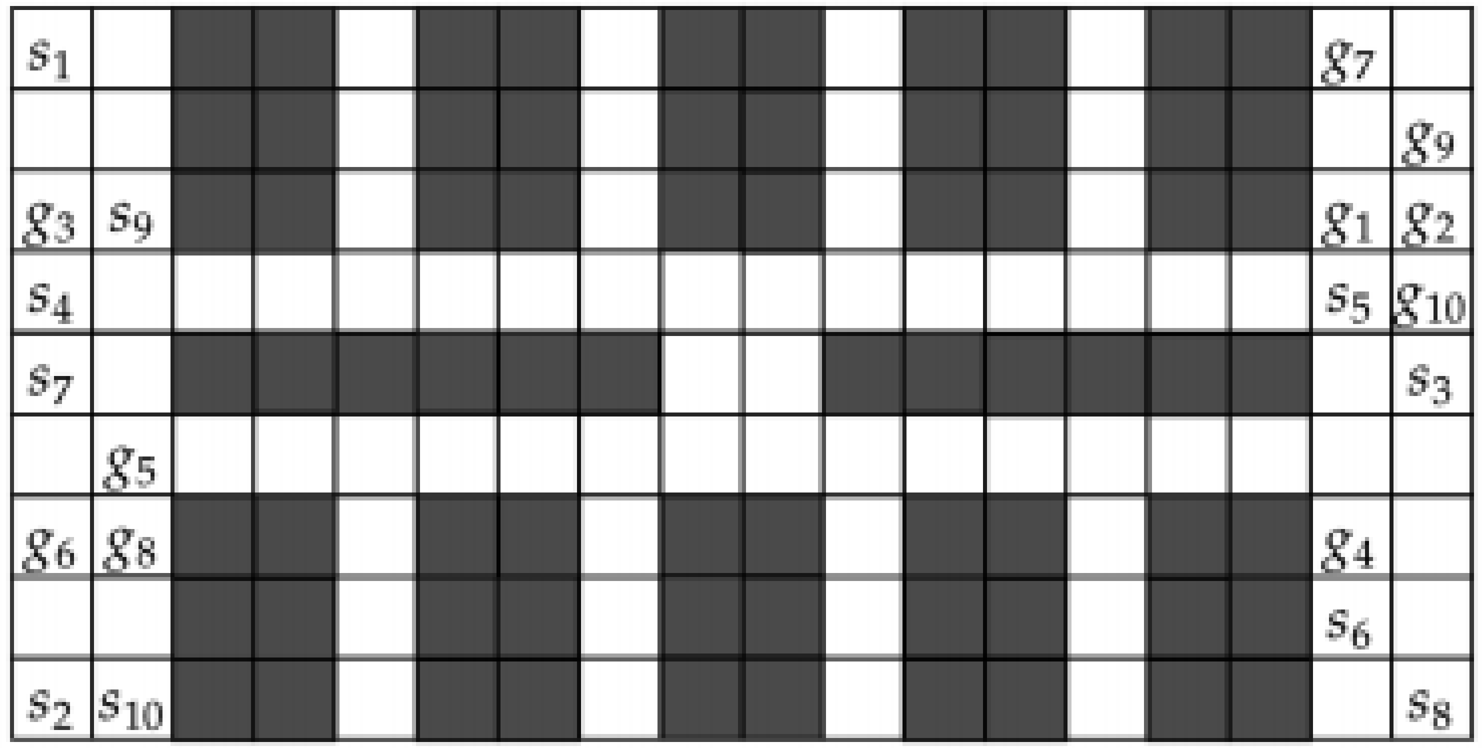
MAPF solver



(b)

Obtain plan π_b

MAPF solver



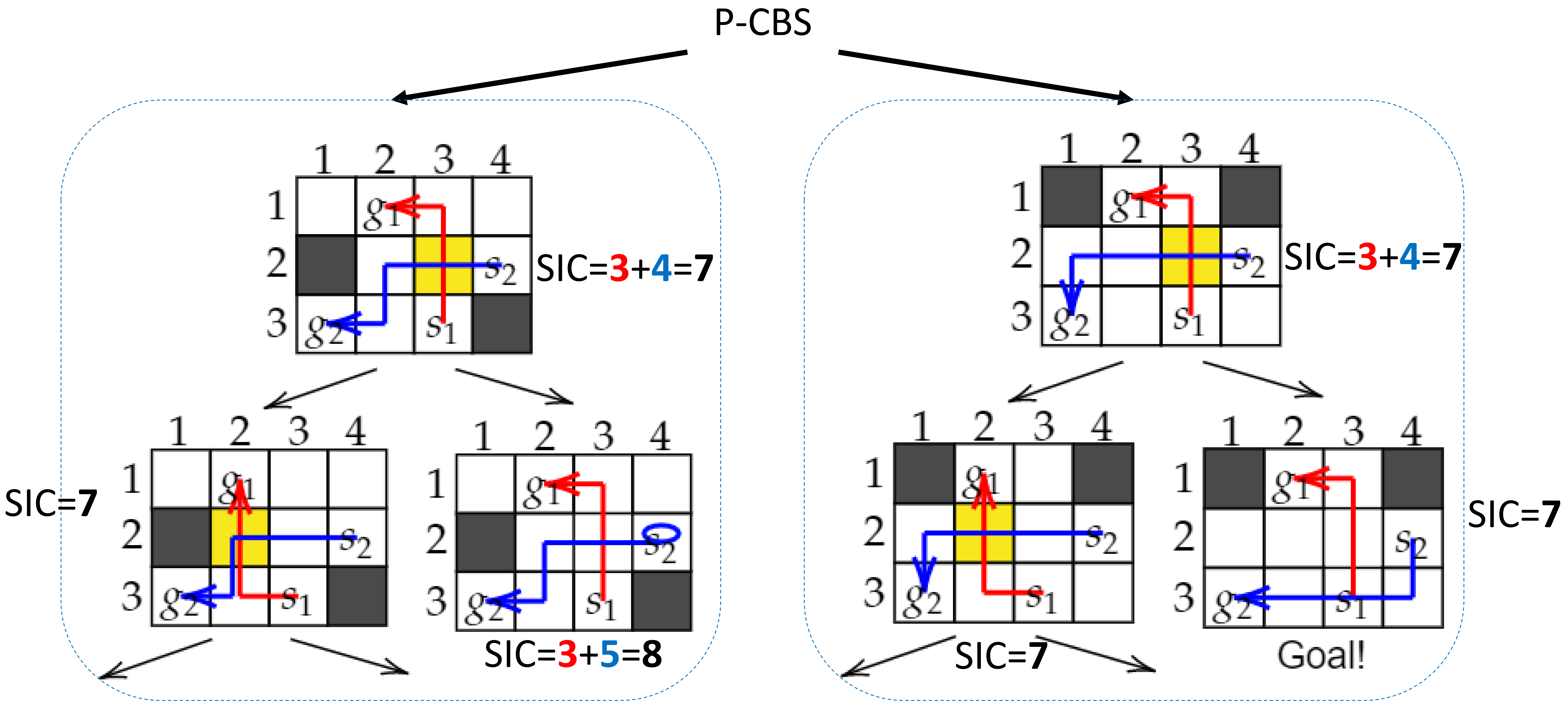
(c)

Obtain plan π_c

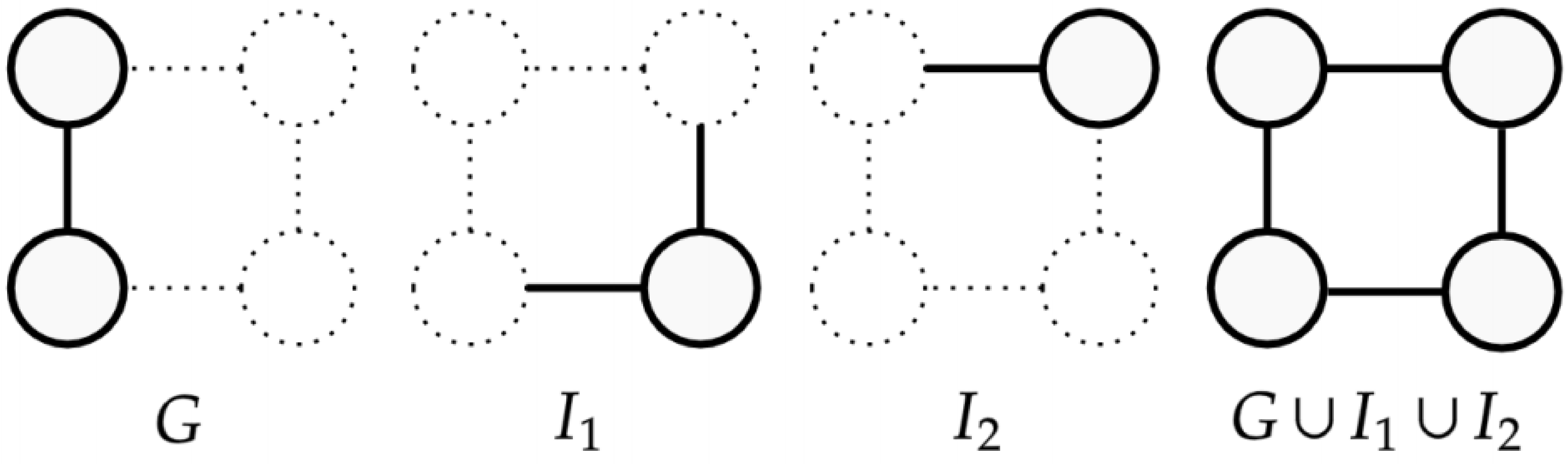
Parallel CBS (P-CBS)

- Nodes store a configuration $G \in \mathcal{G}$, which remains the same for their child nodes
- At the beginning of the search, P-CBS generates a node N_j, equivalent to the root node of plain CBS, for each configuration $G_j \in \mathcal{G}$
- These nodes are all added to OPEN and the search continues as in CBS.
- P-CBS terminates if a solution exists. If P-CBS terminates, it returns the optimal solution according to the flowtime

Parallel CBS (P-CBS)



Graph improvements



- A graph improvement $I = (V', E')$ for a graph $G = (V, E)$ is a structure composed of a set of vertices V' and a set of edges E' such that $V \cap V' = \emptyset$ and $E \cap E' = \emptyset$

- In general, adding a graph improvement $I = (V', E')$ to a graph $G = (V, E)$ does not result in a well-formed graph $G' = G \cup I = (V \cup V', E \cup E')$ (e.g., some edges can be dangling)

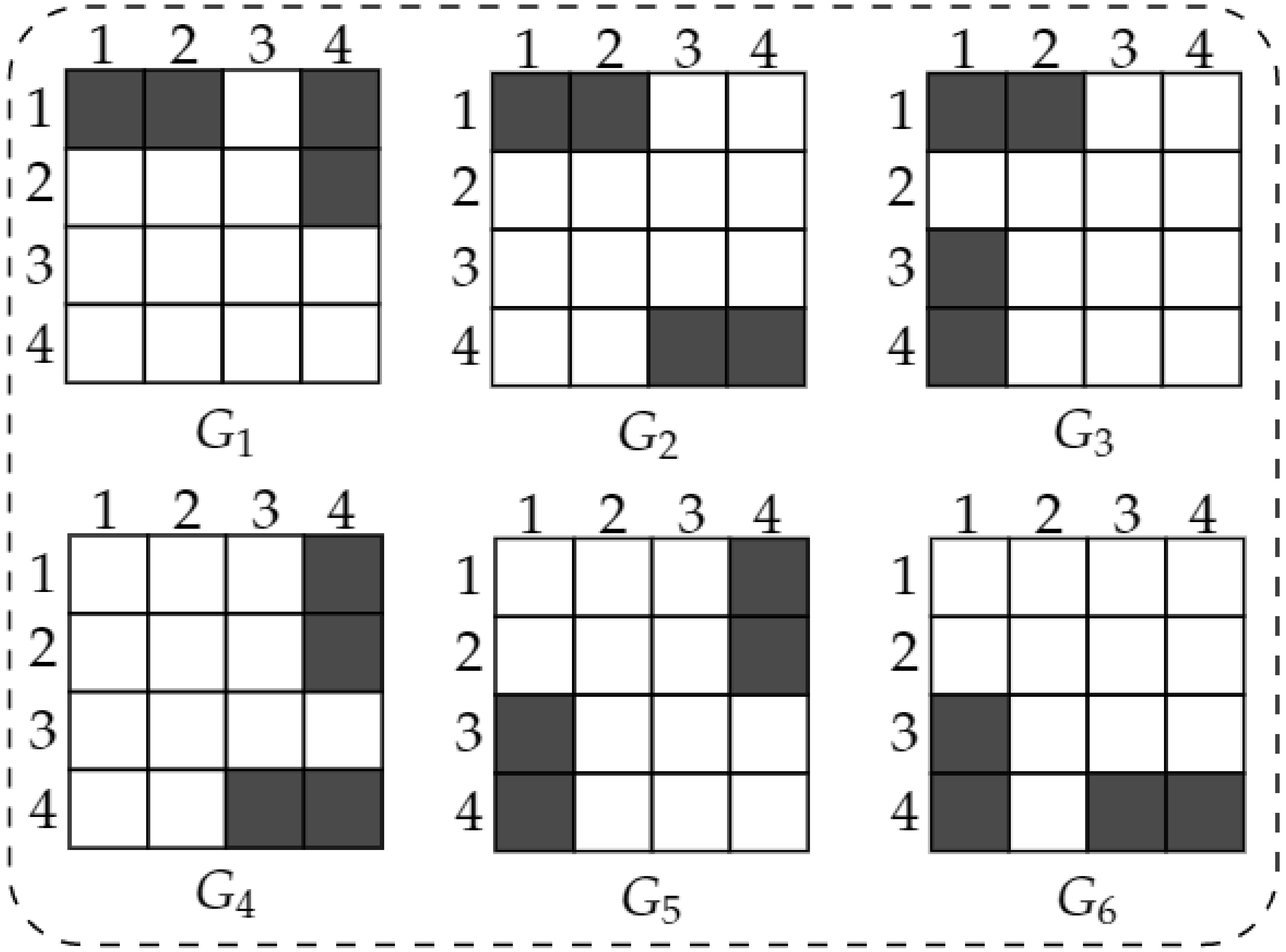
- The idea is to obtain the configurations as an appropriate combination of some graph improvements

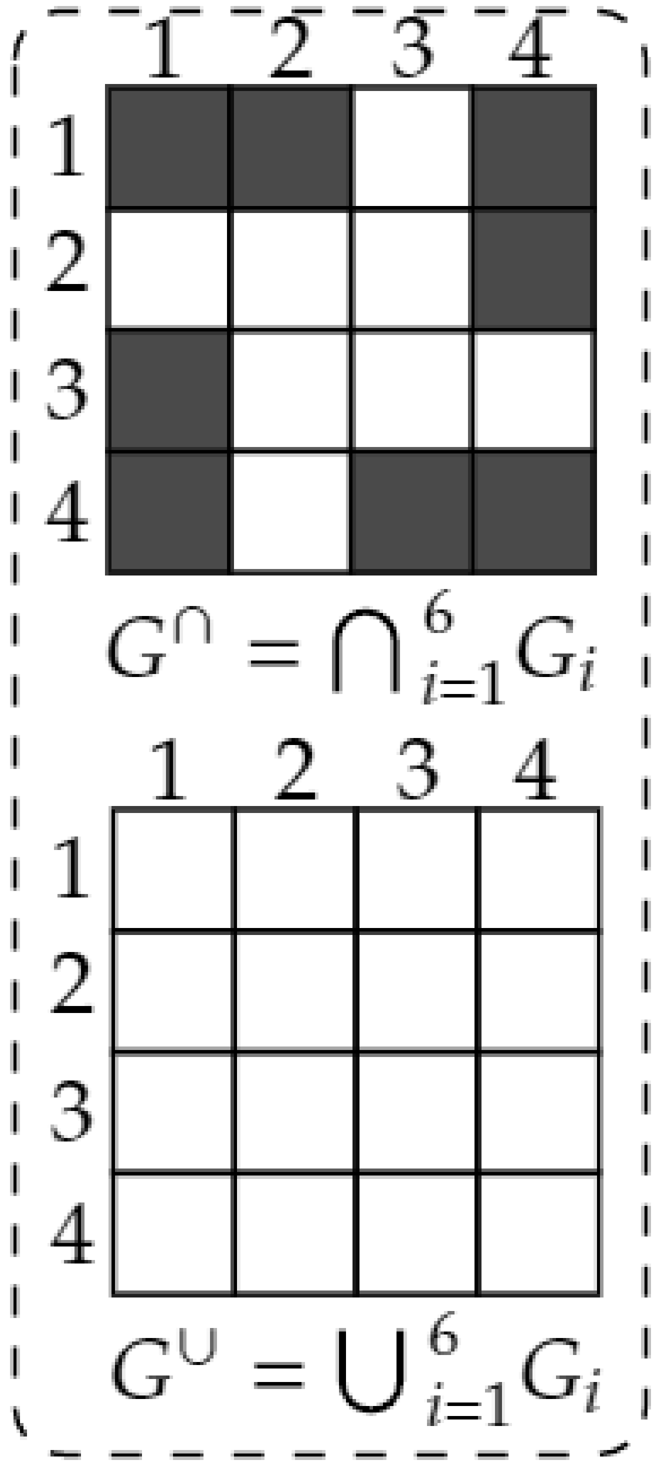
C-MAPF: Operative formulation

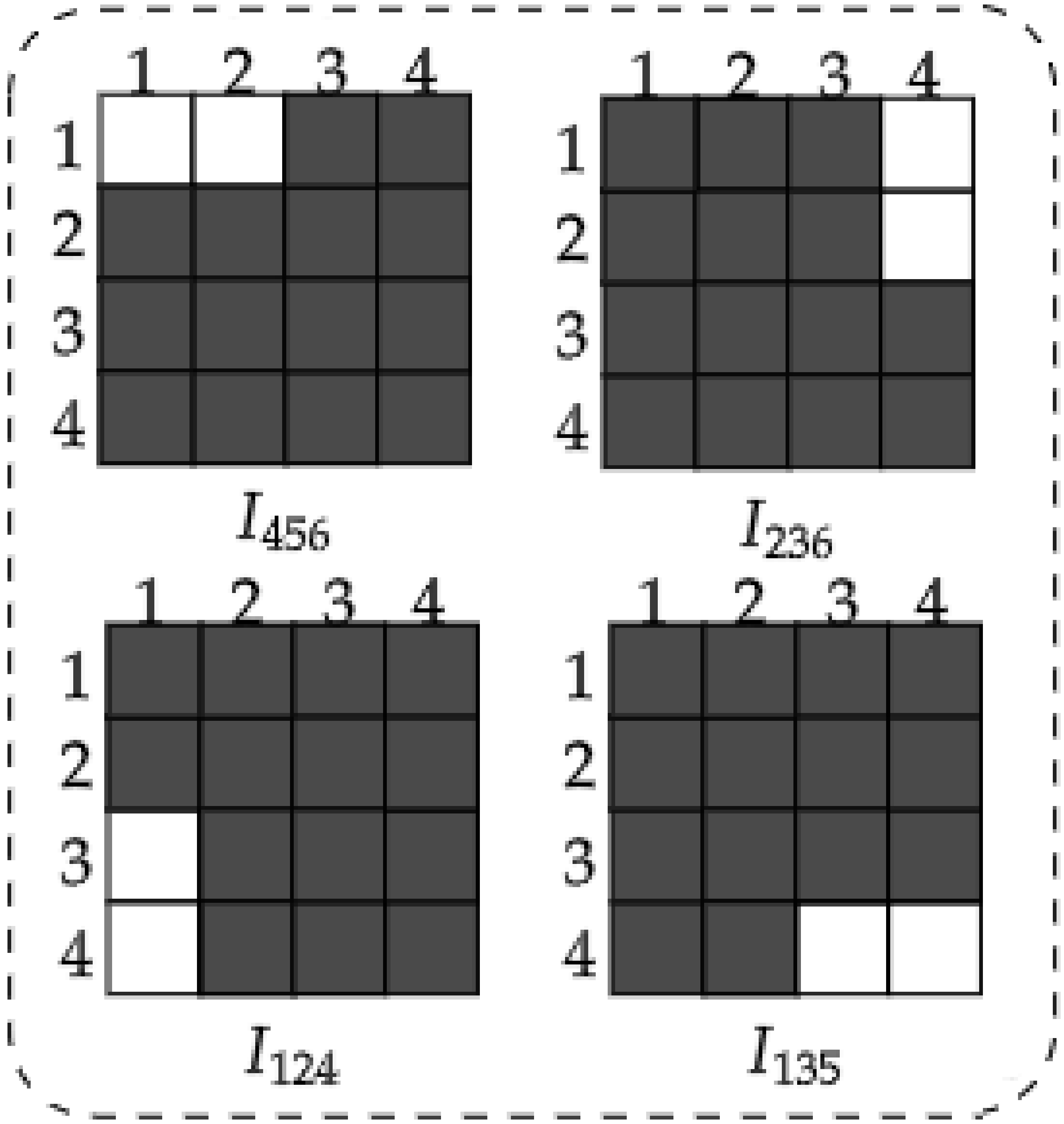
- In the operative formulation of the C-MAPF problem we are given:
  - A graph $G = (V, E)$
  - A set of k labeled agents $A = \{a_1, \ldots, a_k\}$
  - A set of m disjoint graph improvements $\boldsymbol{\mathcal{I}} = \{I_1, \ldots, I_m\}$ for G
  - A validation function $f: \boldsymbol{\mathcal{P}}(\boldsymbol{\mathcal{I}}) \rightarrow \{0,1\}$
  - A set of k start positions $s = \{s_1, \ldots, s_k\}$, $s_i \in V$
  - A set of k goal positions $g = \{g_1, \ldots, g_k\}$, $g_i \in V$
- The validation function tells, by returning 1, which combinations of graph improvements produce configurations in $\boldsymbol{\mathcal{G}}$

C-MAPF: Operative formulation

- It is always possible to translate a C-MAPF problem formulated by listing all the configurations in $\mathcal{G}$ into the operative formulation and vice versa







$$f(\mathcal{J}') = \begin{cases} 1, & |\mathcal{J}'| = 2 \\ 0, & \text{otherwise} \end{cases}$$

Abstract CBS (A-CBS)

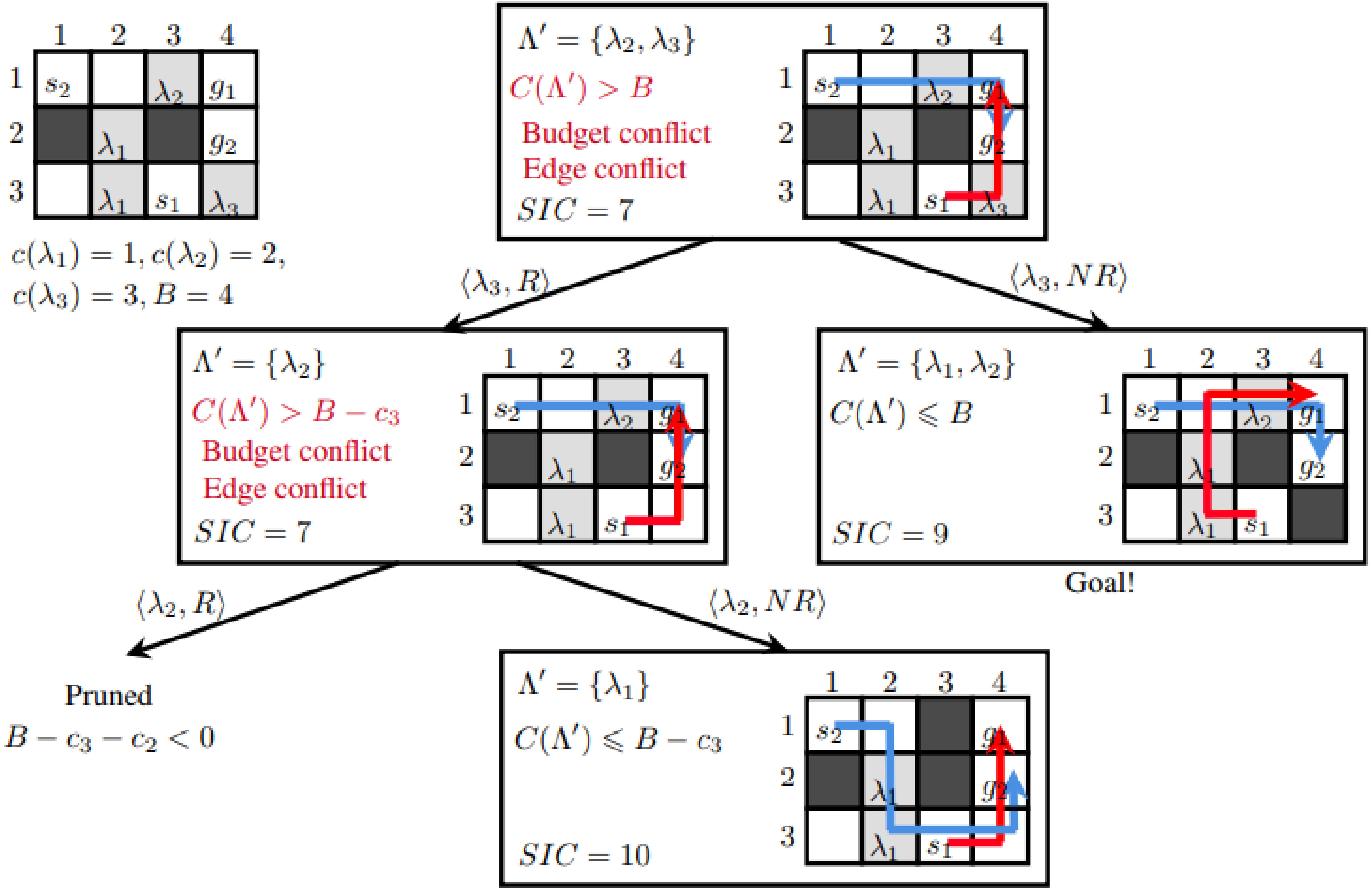
- The idea behind A-CBS is that, unless otherwise imposed by environment constraints, the low-level algorithm considers all the graph improvements as added to G

- A-CBS first checks if there exists a configuration in which the plan of the node is applicable by invoking the **extended validation function** $ef: \mathcal{P}(\mathcal{I}) \rightarrow \{0,1\}$.
  - Given the set of graph improvements $\mathcal{I}' \subseteq \mathcal{I}$ used by the plan in the node, $ef(\mathcal{I}')$ it is defined as:

$$ef(\mathcal{I}') = \begin{cases} 1, & \exists \mathcal{I}'' \subseteq \mathcal{I} - \mathcal{I}' \mid G \cup \left(\bigcup_{I_j \in \mathcal{I}' \cup \mathcal{I}''} I_j \right) \in \mathcal{G} \\ 0, & \text{otherwise} \end{cases}$$

- If the test doesn't fail, A-CBS proceeds following the behavior of CBS, otherwise, A-CBS imposes **environment constraints**, limiting the search space.

Abstract CBS (A-CBS)



Abstract CBS (A-CBS)

- In practically implementing A-CBS, multiple environment constraints can be considered at each step
- A-CBS maintains the nodes that reported an environment conflict on top of the high-level tree
- The number of configurations, n, represents a strict upper bound on the number of environment conflicts encountered during the high-level search of A-CBS
  - This result reassures us in the case in which the C-MAPF problem instance presents many graph improvements with respect to the number of configurations (i.e., when $m \gg n$)

Experimental results

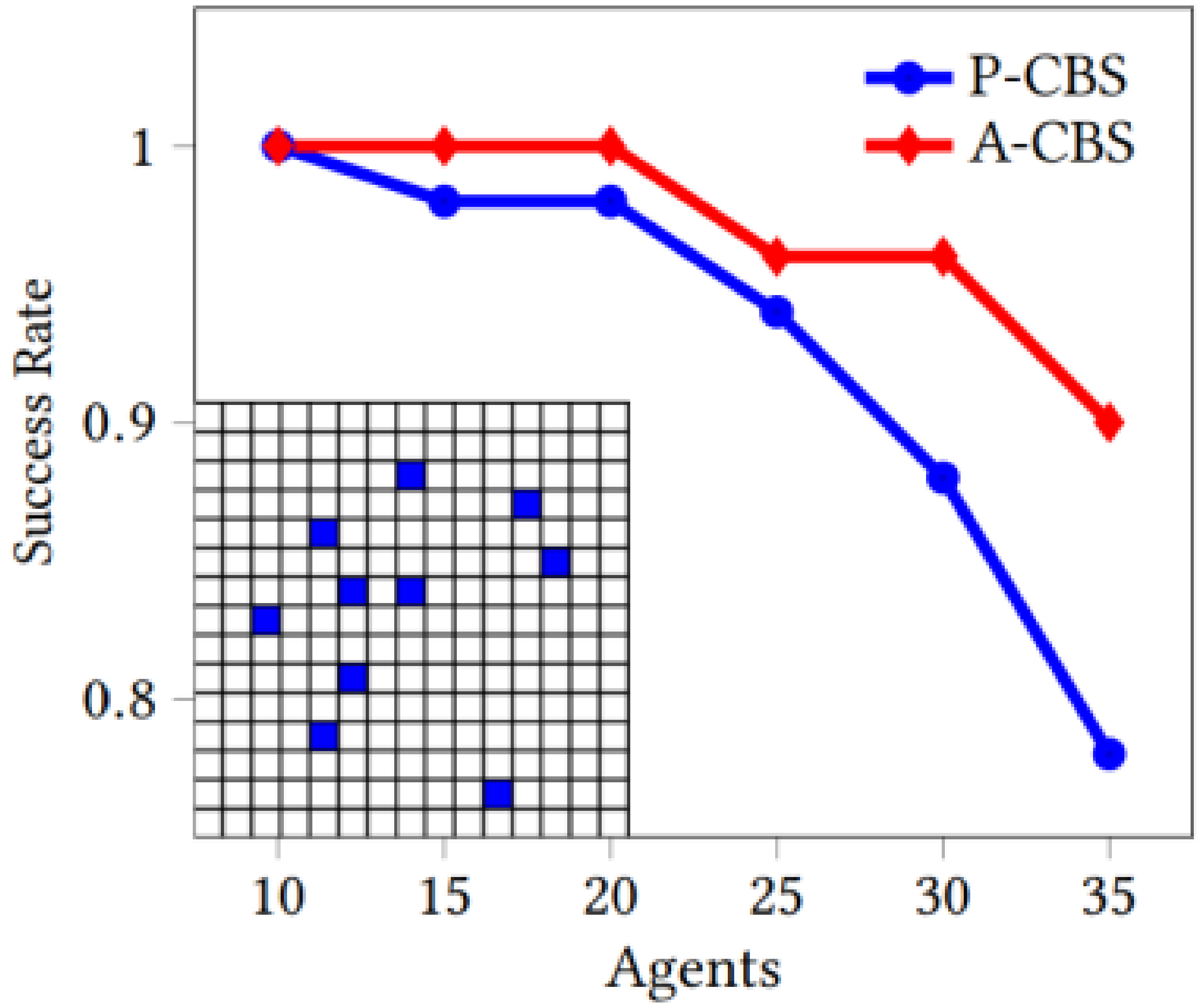
- Our algorithms use the conflict prioritization and the high-level heuristic values of CBSH

- For small grids: timeout of 30 seconds, solver can remove $\beta = 30\%$ of obstacles

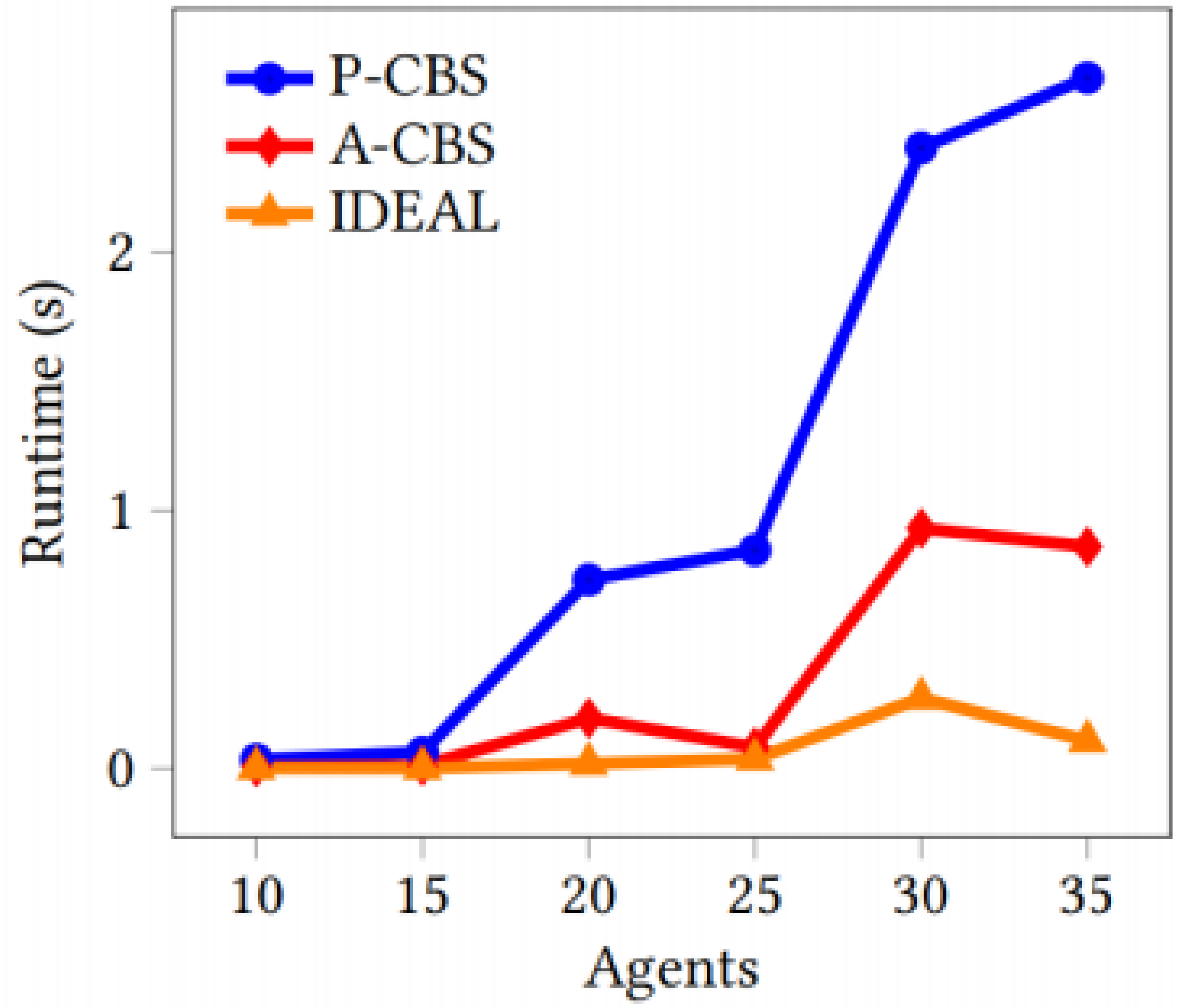
  - Note that, given η the number of obstacles, the number of configuration is: $n = n(\eta) = \binom{\eta}{\lfloor \beta\eta \rfloor} \approx 2^{H(\beta)\eta}$. For $\beta = 30\%$, $n \approx 2^{0.88 \cdot \eta}$ $(n(25) > 10^6)$

- For large maps: timeout of 5 minutes, 25 obstacles with a cost between 1 and 5, budget of 10

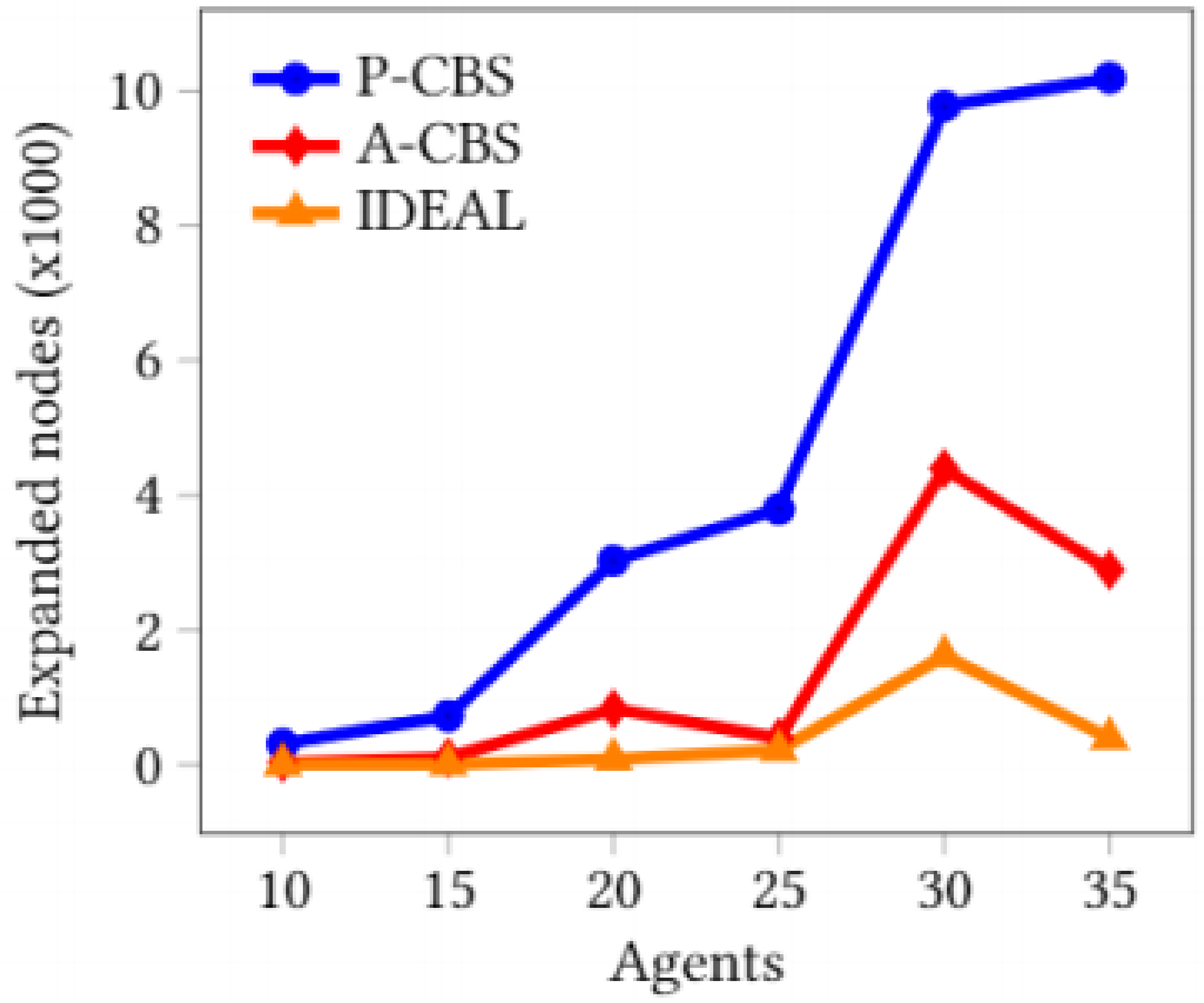
Experimental results



(a) Success rate for varying k



(b) Runtime for varying k

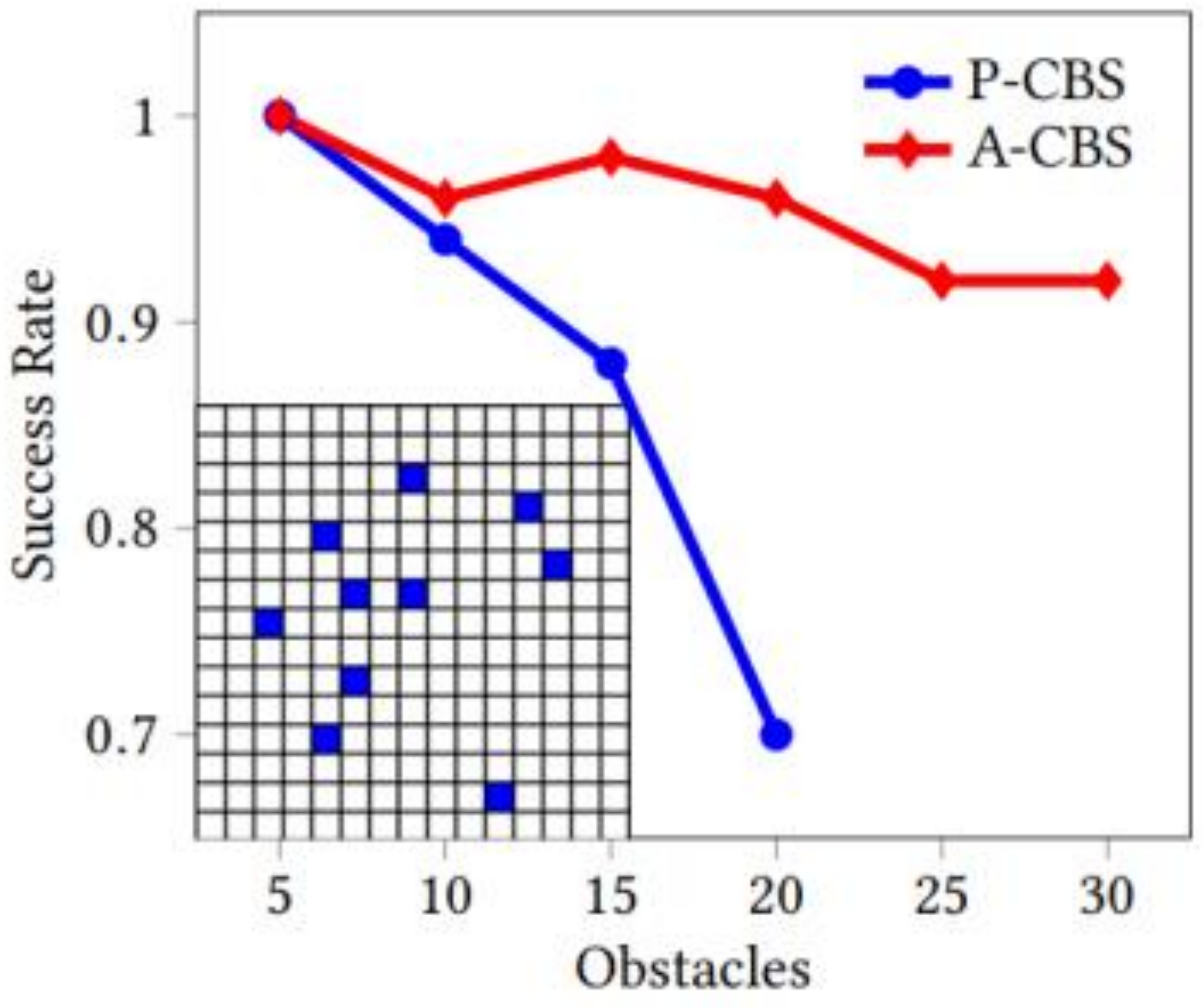


(c) Expanded nodes for varying k

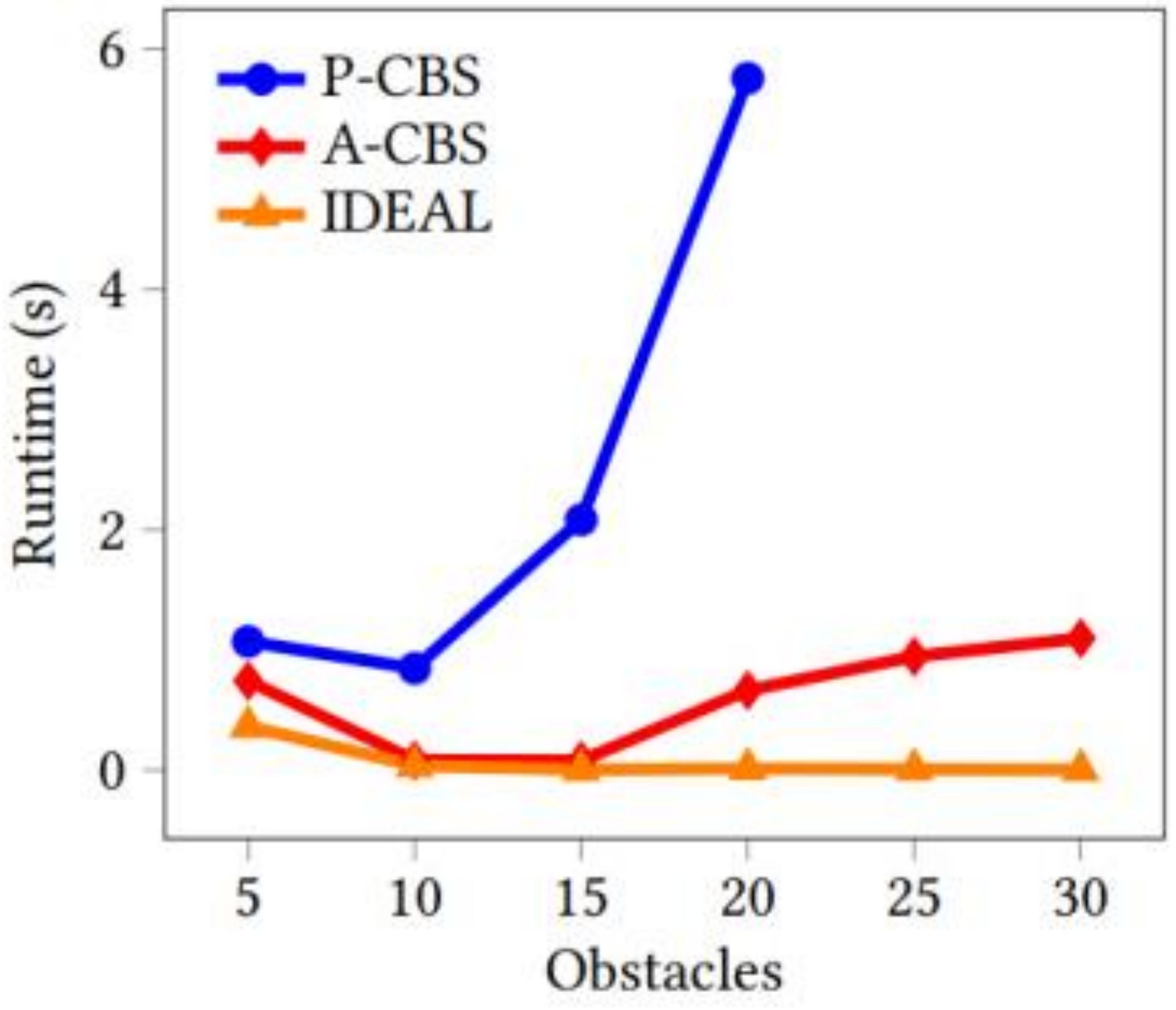
$\eta = 10$

		Runtime (ms)			Expanded CT nodes		
k	#	**P-CBS**	**A-CBS**	**IDEAL**	**P-CBS**	**A-CBS**	**IDEAL**
10	50	34	5	1	310	34	7
15	49	62	10	1	726	107	13
20	49	731	194	18	3030	828	87
25	47	846	81	36	3798	396	220
30	44	2405	930	272	9776	4395	1618
35	39	2676	859	107	10192	2895	396

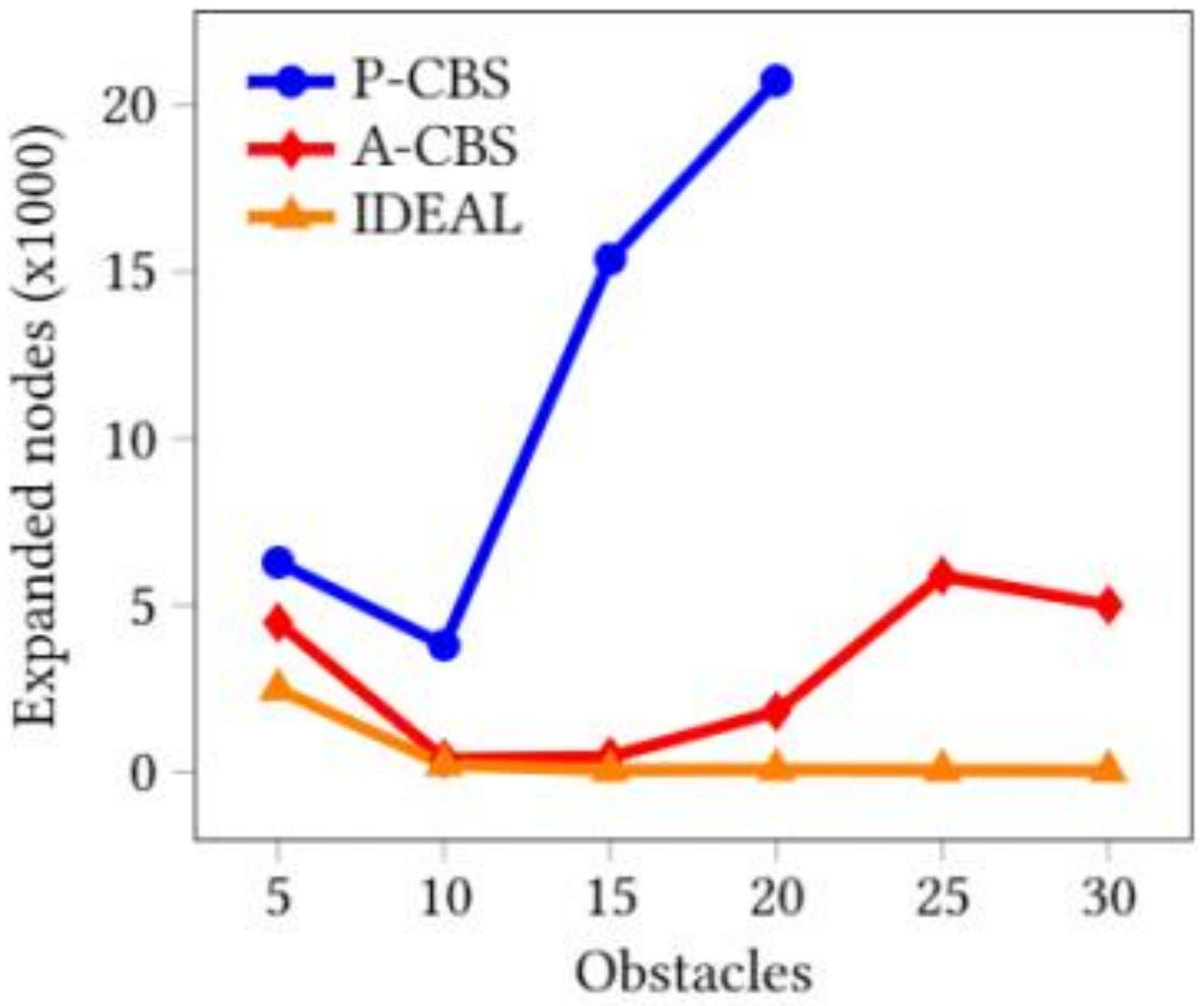
Experimental results



(d) Success rate for varying η



(e) Runtime for varying η

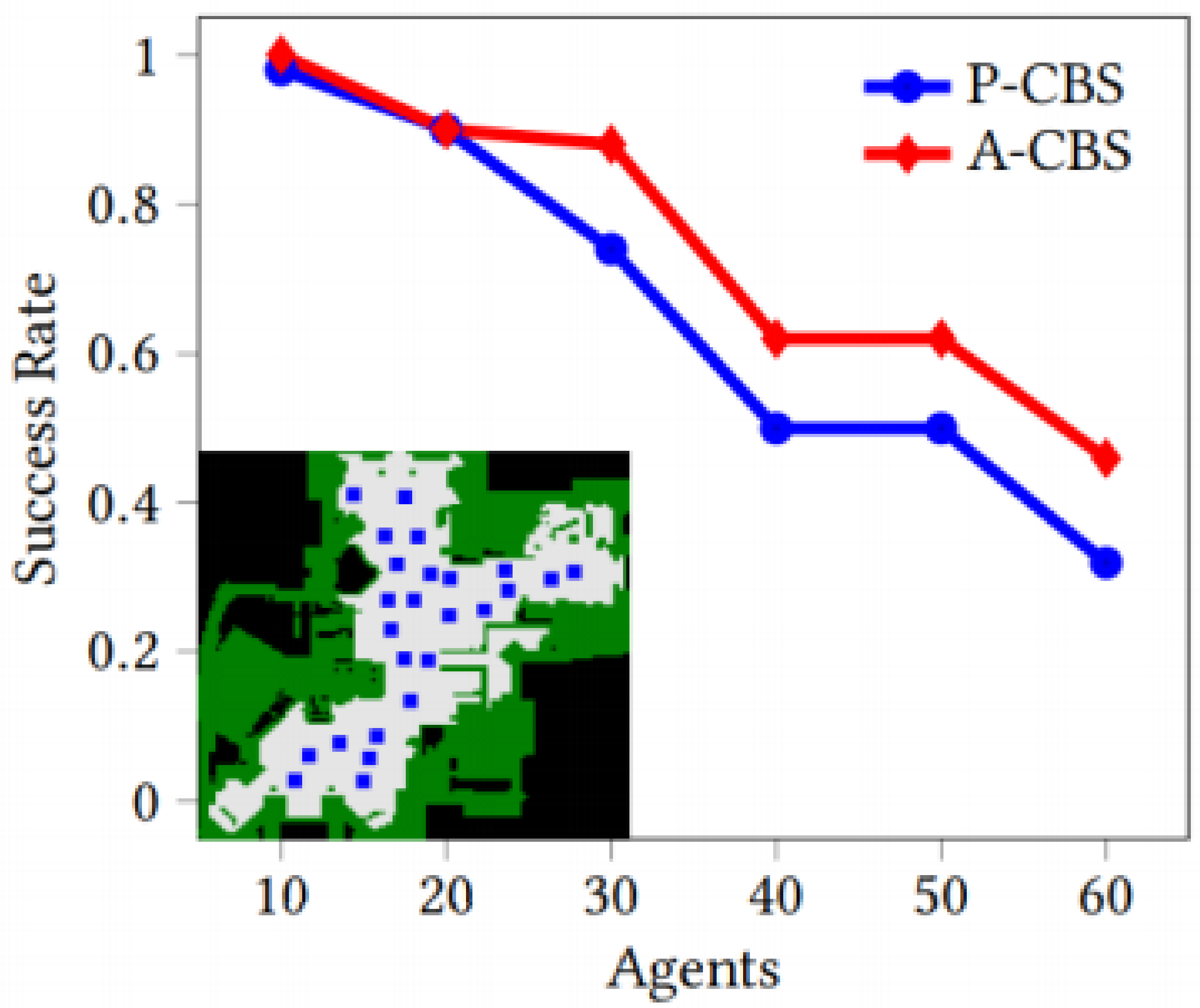


(f) Expanded nodes for varying η

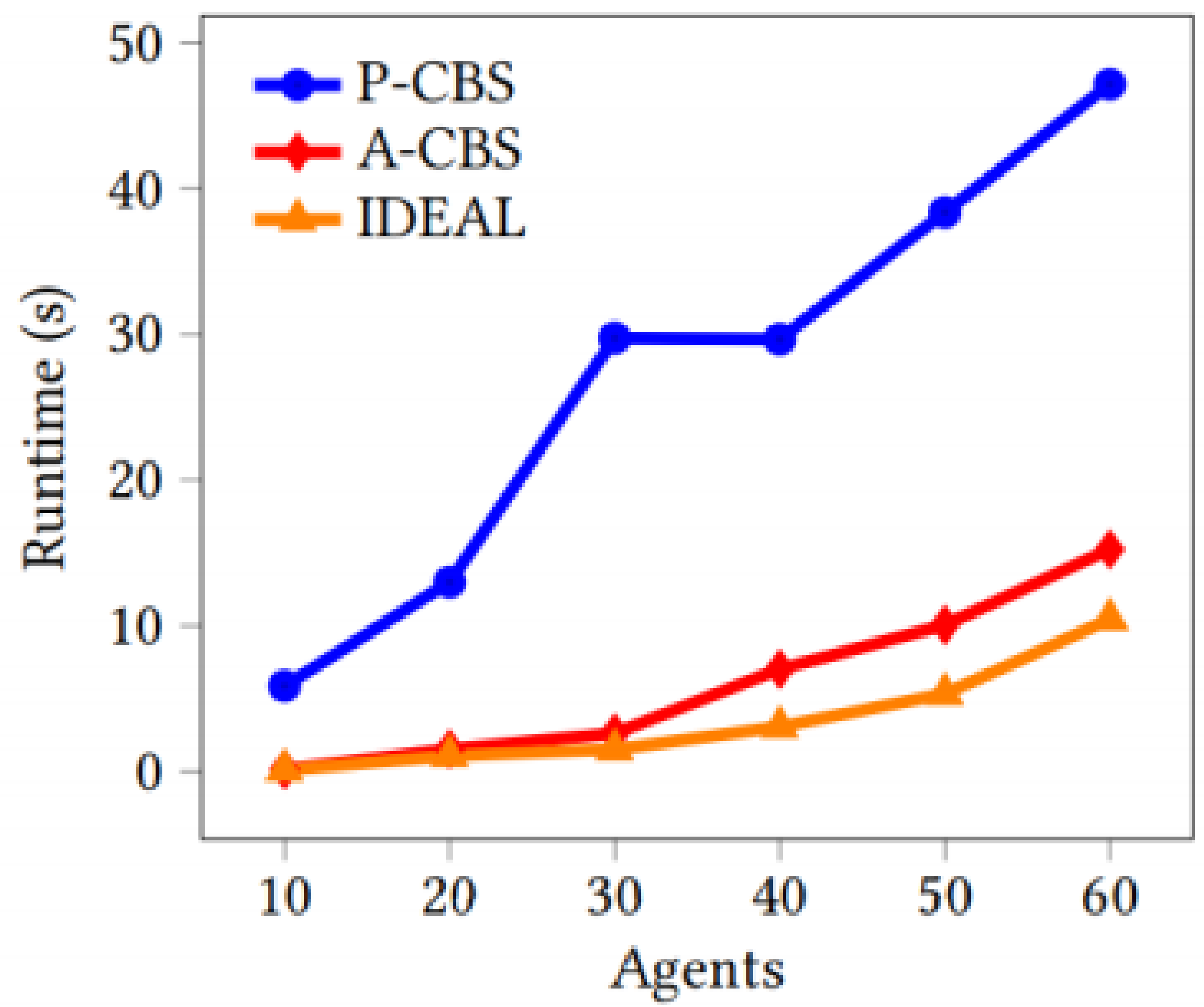
$k = 25$

		Runtime (ms)			**Expanded CT nodes**		
η	#	**P-CBS**	**A-CBS**	**IDEAL**	**P-CBS**	**A-CBS**	**IDEAL**
5	50	1096	742	363	6281	4484	2482
10	47	846	81	36	3798	396	220
15	44	2082	78	3	15381	462	51
20	35	5755	663	12	20730	1814	75
25	46*	NA	944*	7*	NA	5884*	55*
30	46*	NA	1096*	2*	NA	5005*	35*

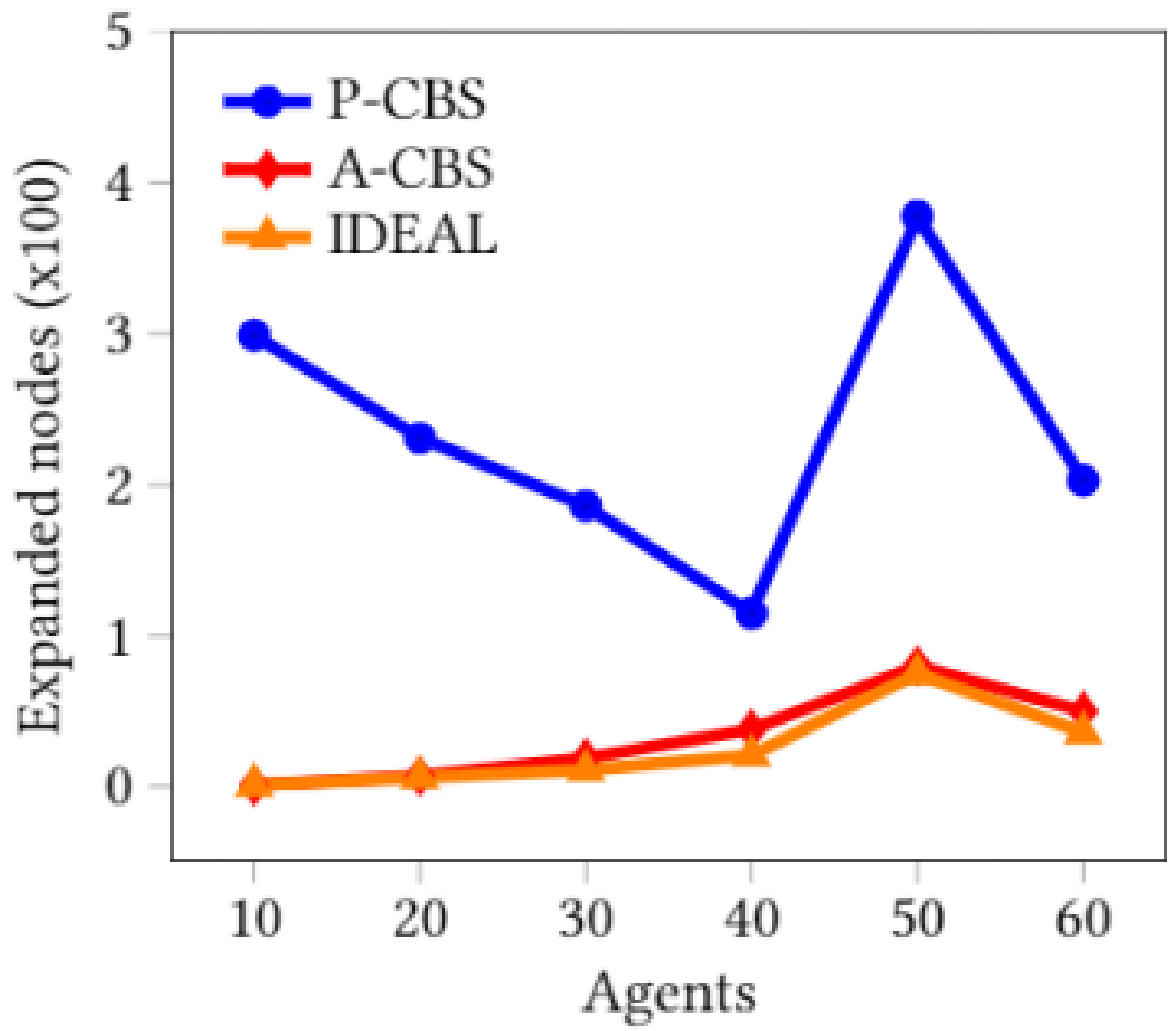
Experimental results



(a) Success rate for varying k



(b) Runtime for varying k



(c) Expanded nodes for varying k

		Runtime (ms)			**Expanded CT nodes**		
k	#	**P-CBS**	**A-CBS**	**IDEAL**	**P-CBS**	**A-CBS**	**IDEAL**
10	49	5874	121	89	299	1	1
20	45	12975	1517	1064	231	7	6
30	37	29742	2611	1503	186	19	11
40	25	29661	7030	3063	115	38	21
50	25	38357	10043	5331	378	80	75
60	16	47144	15240	10845	203	50	36

Presentations

- A version of this work has been submitted:
  - to the Amazon Research Awards (ARA) of 2019, a program that offers awards to faculty members at academic institutions worldwide
  - to the International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS) to be held in May 2020 in Auckland (New Zealand)

research awards AAMAS 2020

Thank you

For more information about MAPF see the *brand new* website http://mapf.info