Research Project Proposal:

Multi-Agent Path Finding with Removable Obstacles

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

POLITECNICO MILANO 1863

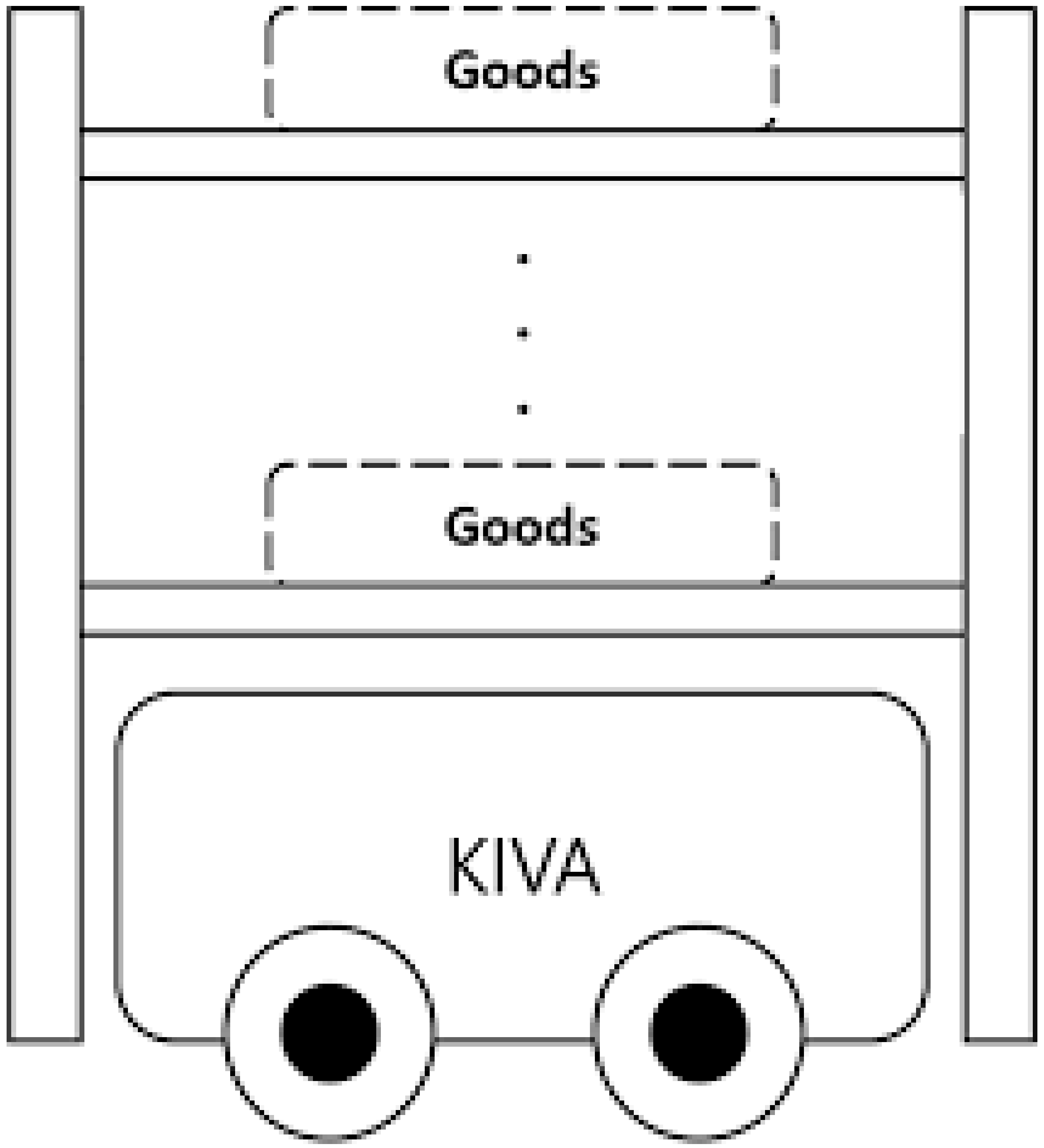
HONOURS PROGRAMME HP-SR in Information Technology

Introduction

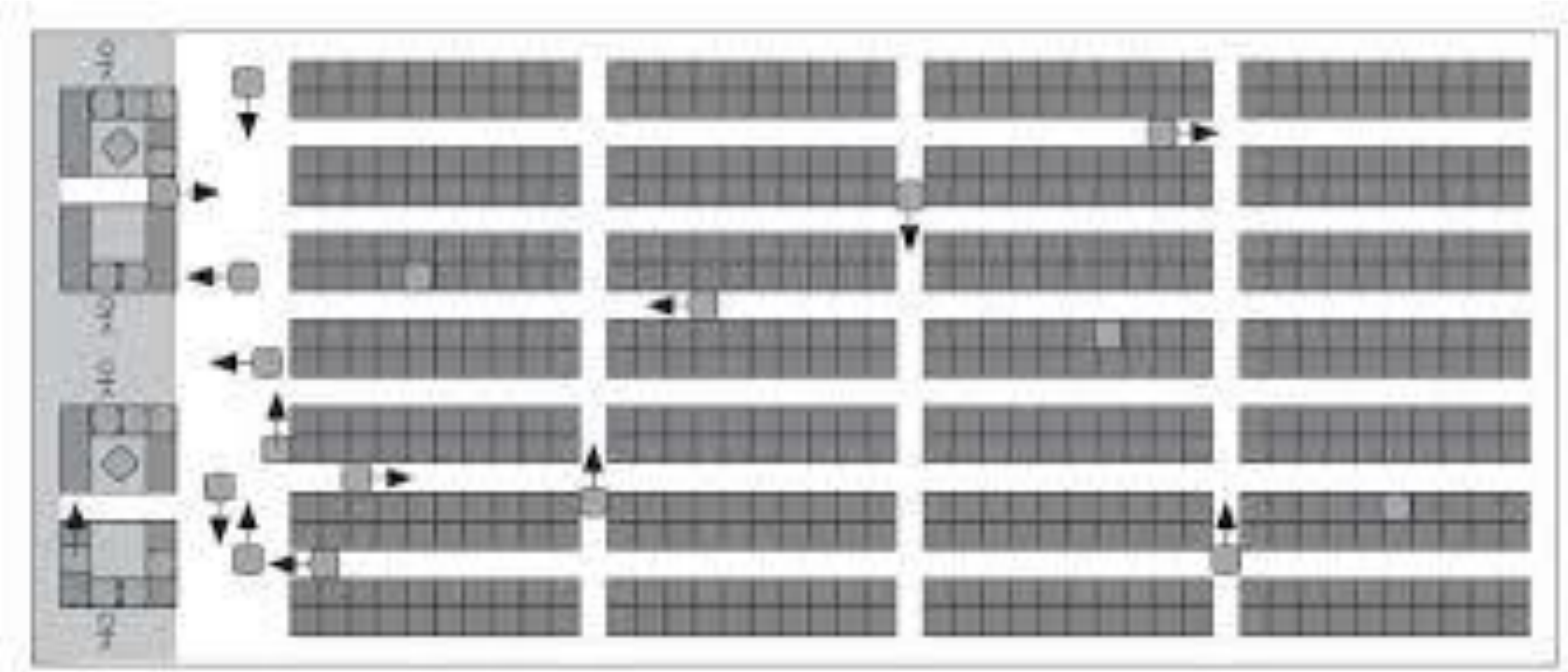
- Multi-Agent Path Finding (MAPF) is a challenging Artificial Intelligence topic that has many applications

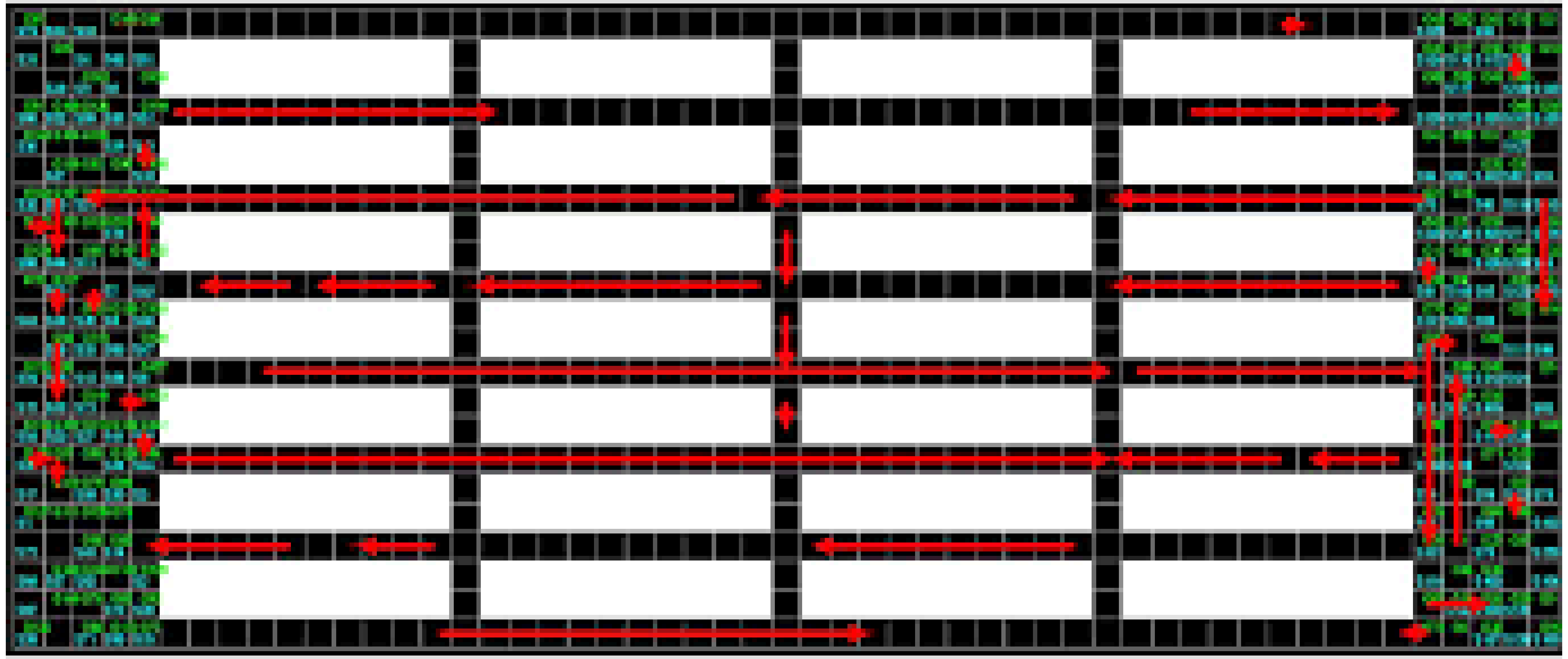
- The task in MAPF problems is to find non-interfering paths for multiple agents, each one with a unique start and goal position in a known environment

Applications



Applications



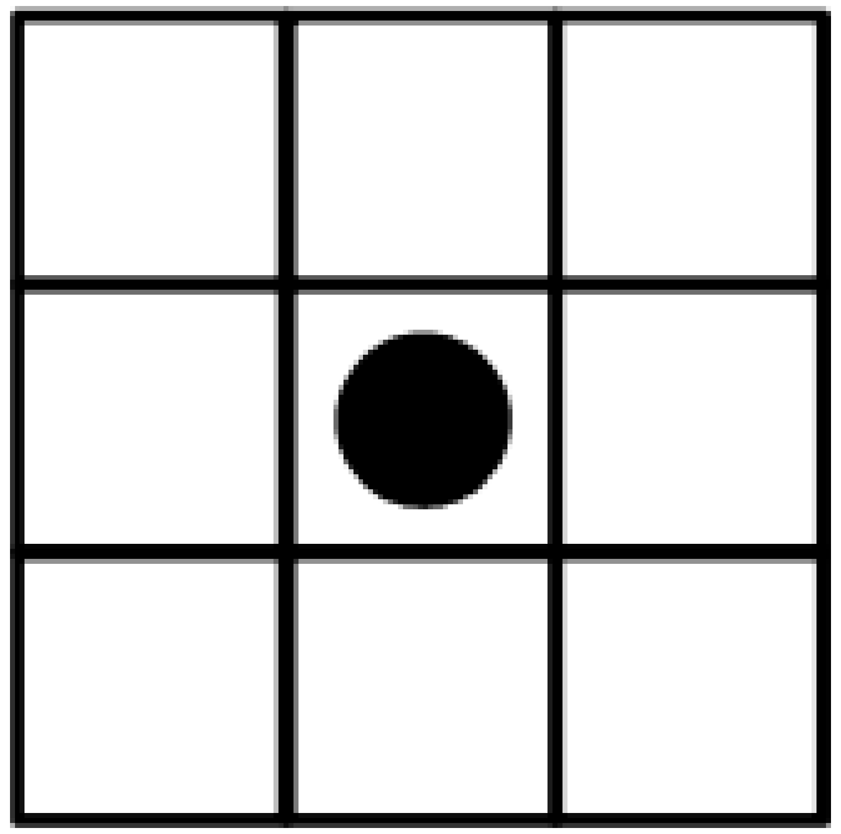


Applications

Preliminaries

- Robot

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



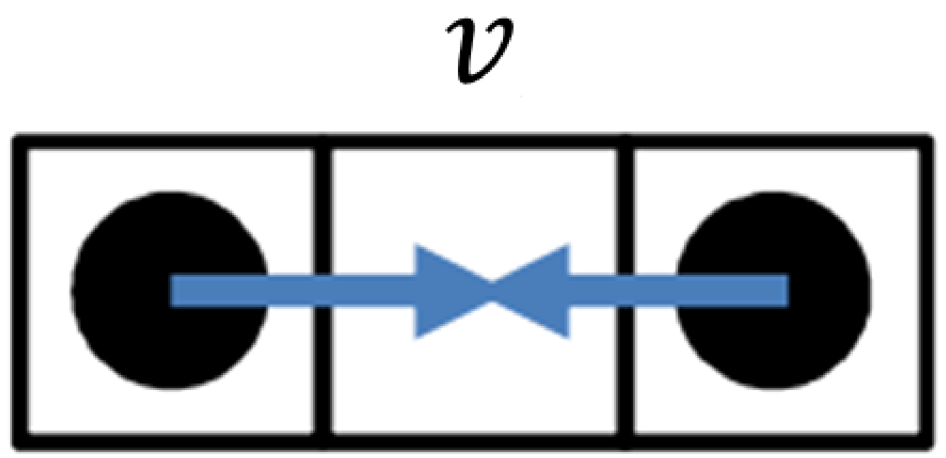
Preliminaries

- In the 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 k start positions $s = \{s_1, \ldots, s_k\}$, $s_i \in V$
  - A set of k goal positions $g = \{g_1, \ldots, g_k\}$, $g_i \in V$
- 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: sum of the arrival times of all the agents at their goal locations

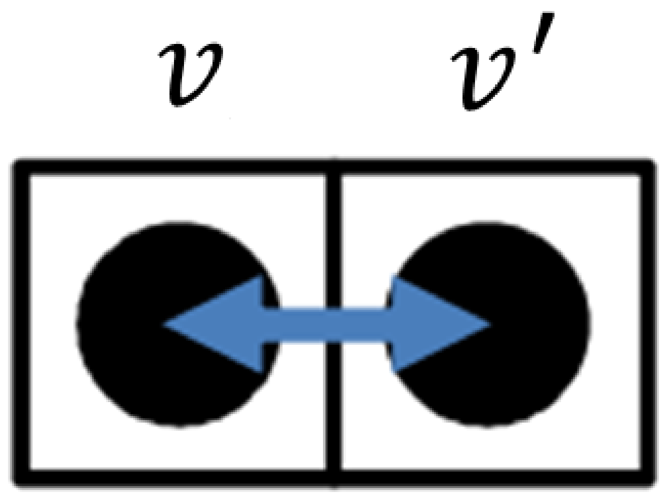
Preliminaries

- 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 timestamp t



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



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

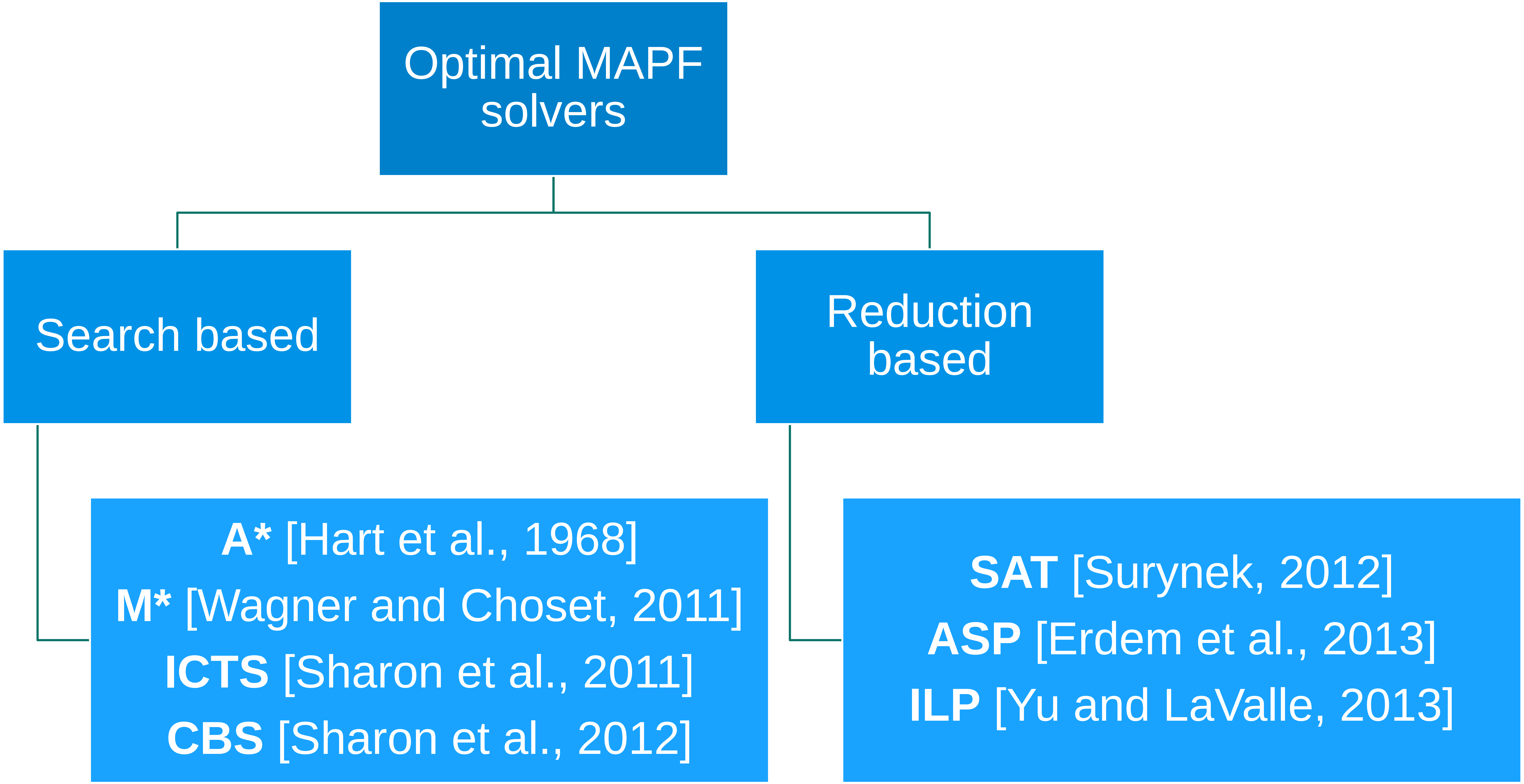
- A **valid solution** is defined as a set of k conflict-free paths (no collisions)

NP-hardness

- Theorem [Yu and LaValle, 2013]: MAPF problems are NP-hard to solve optimally for both makespan and flowtime minimization

- NP-hardness proof is based on a direct reduction from the 3-satisfiability problem

Optimal MAPF Solvers

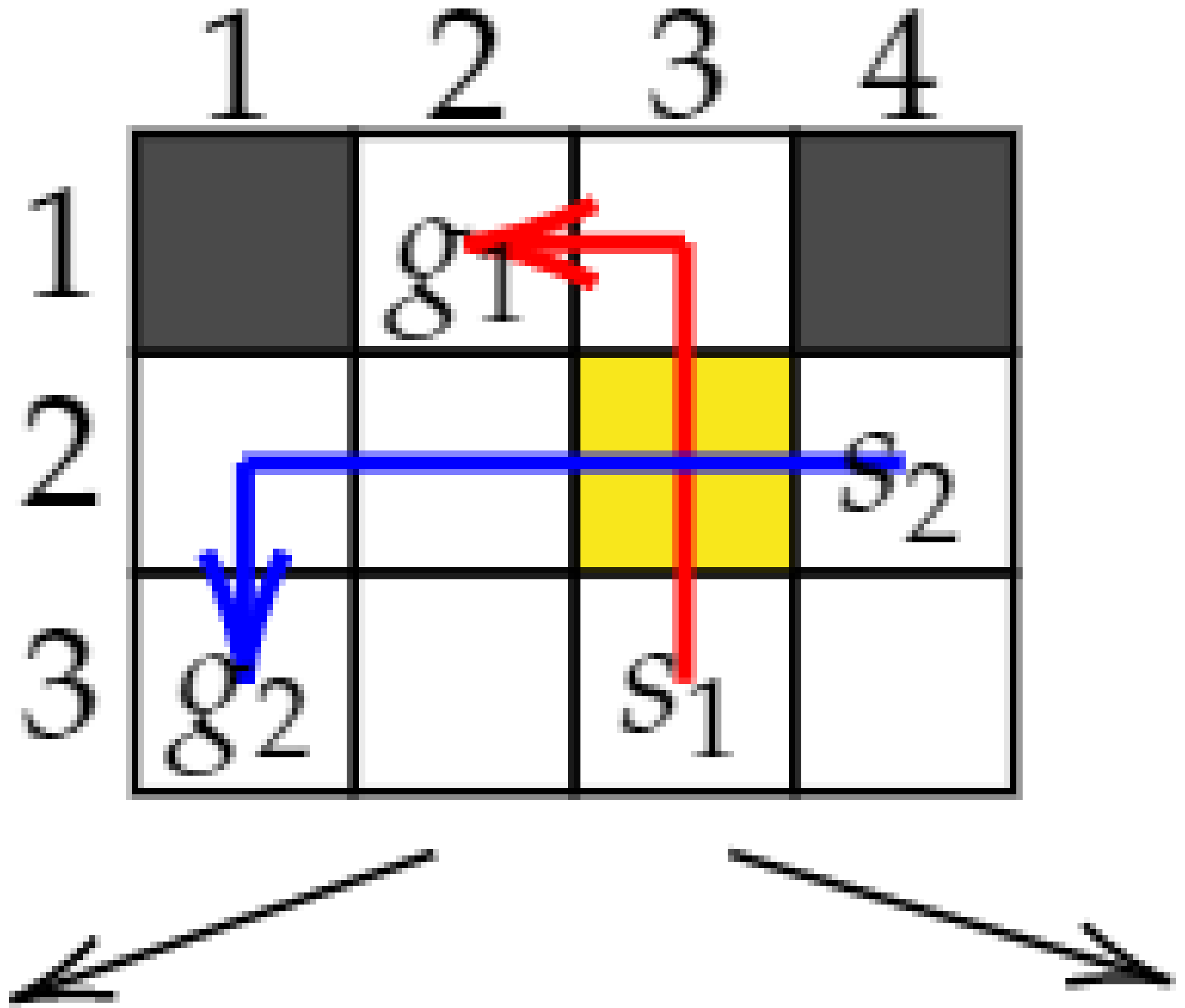


Conflict-Based Search

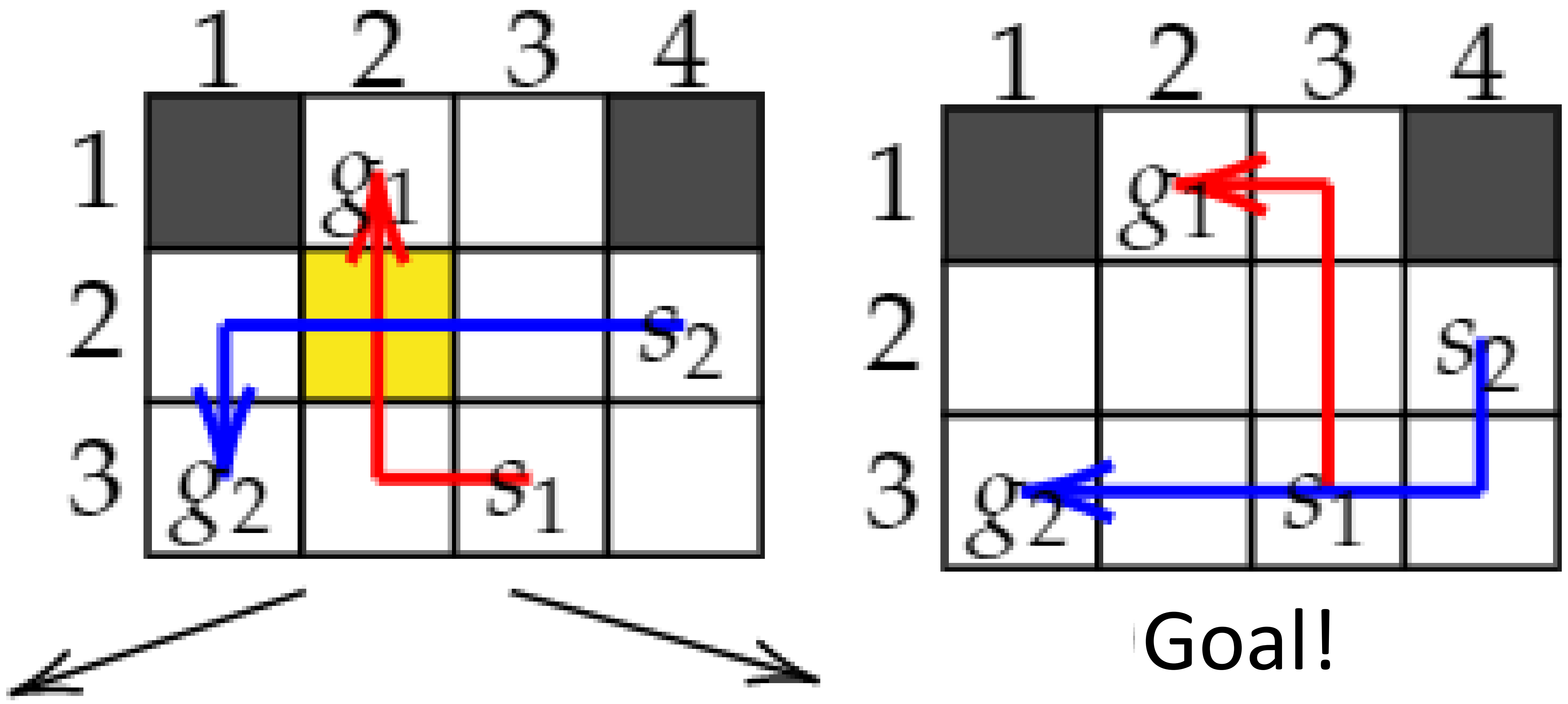
- Two-level structure: high and low level
- The low-level searches for an optimal path for each agent individually taking into account constraints imposed by the high-level
- When a conflict between individual paths is found, the high-level expands a constraint tree via a split action, keeping all the information about the conflict, and imposes new constraints to the agents in order to avoid the conflict
  - The idea is to explore all possible ways to solve the conflict
  - A constraint prohibits an agent from being in a certain position at a given time

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

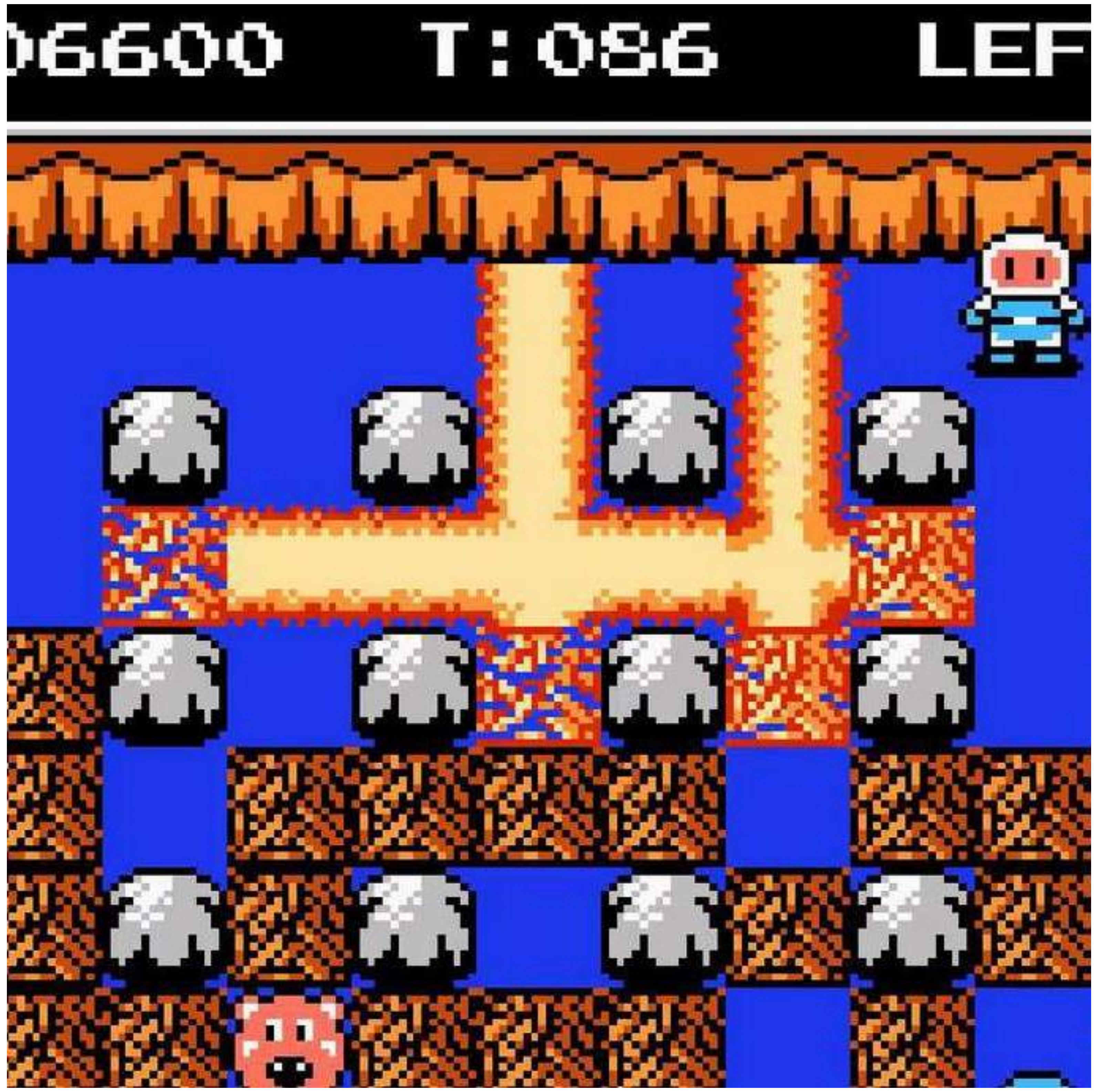
- In classical MAPF problems, we make strong assumptions concerning the environment

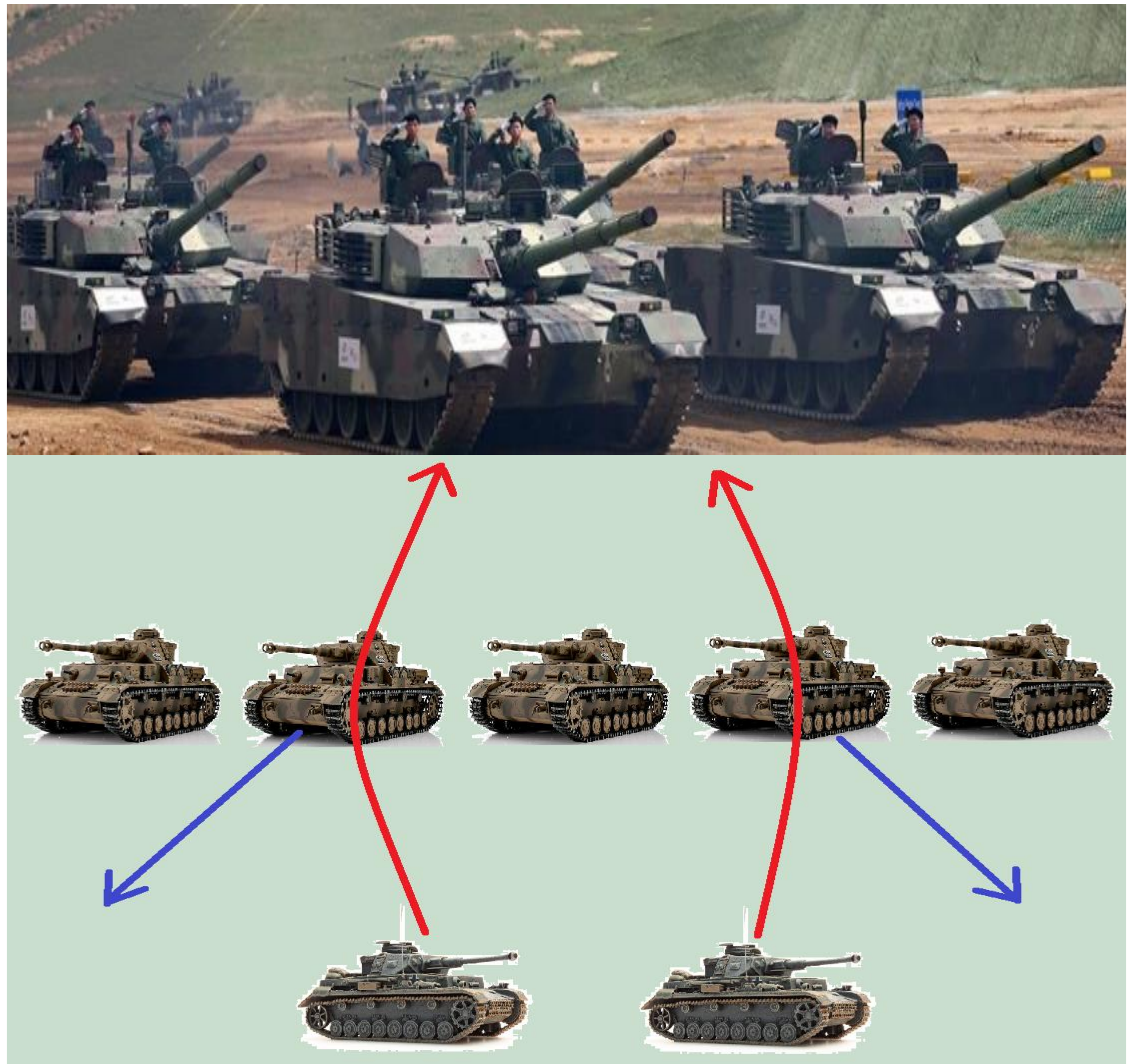
- Real-world environments may have different features and properties
  - Dynamic obstacles
  - Destructible obstacles
  - Shrinking maps
  - Toll booths

Reconfigurable environments

- We want to introduce the concept of *environment reconfiguration*
  - The idea is that something in the environment can be changed, at will, under certain conditions
  - In general, an environment that is mutable in this sense is called a **reconfigurable environment**
- Our first goal is to introduce and clearly define the concept of environment reconfiguration in the MAPF context

Environments with removable obstacles





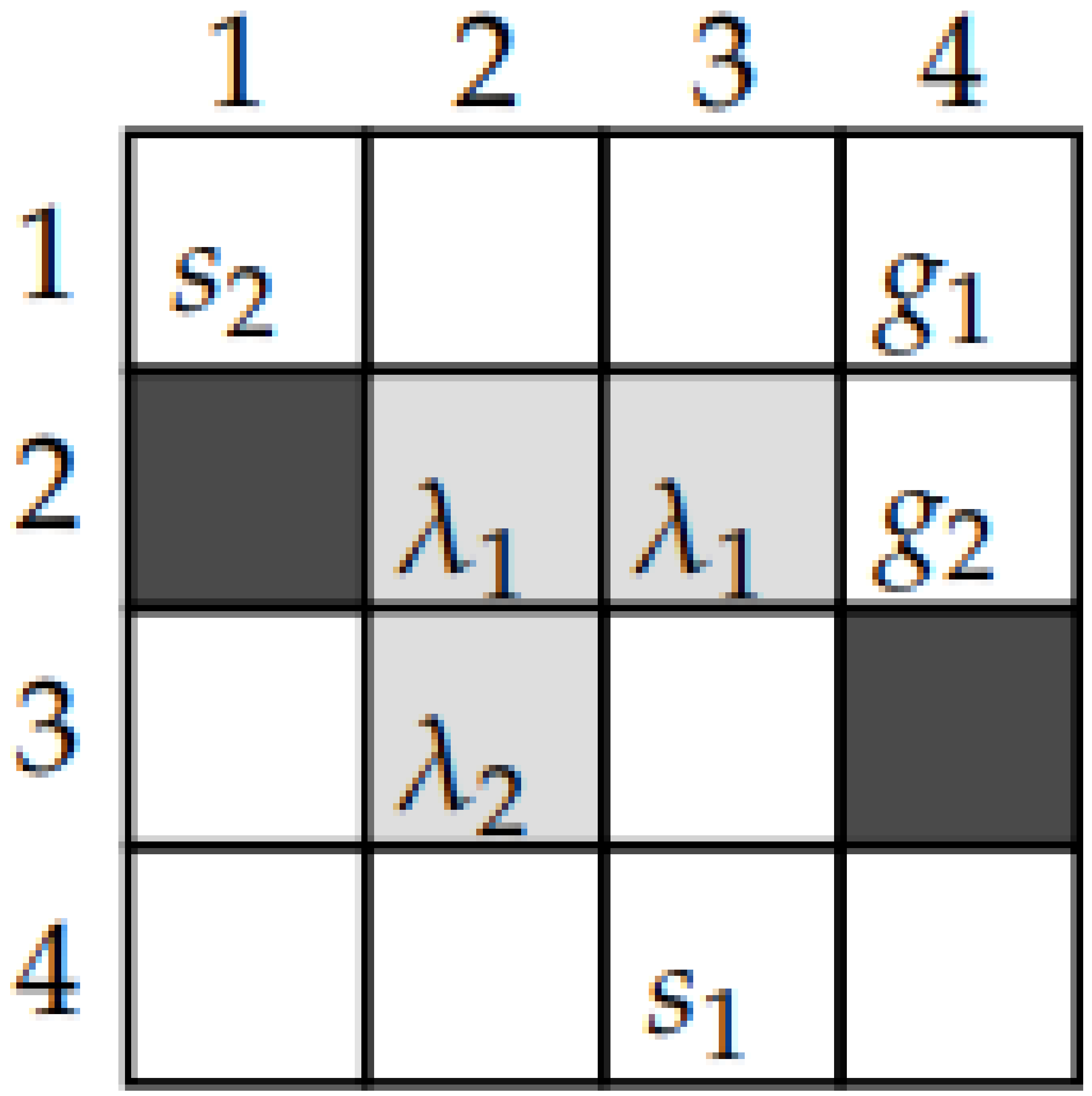
MAPF with removable obstacles

- We focus on reconfigurable environments with destructible obstacles, so we assume that some obstacles can just be removed from the environment
- For simplicity we focus on grids, composed by free and blocked cells, but the problem could be extended to arbitrary graphs
- A **removable obstacle** is defined as a set of (blocked) cells and each removable obstacle has an associated cost called removal cost
  - We assume that there are no cells shared by multiple obstacles
- We are given a budget that can be spent on removing obstacles

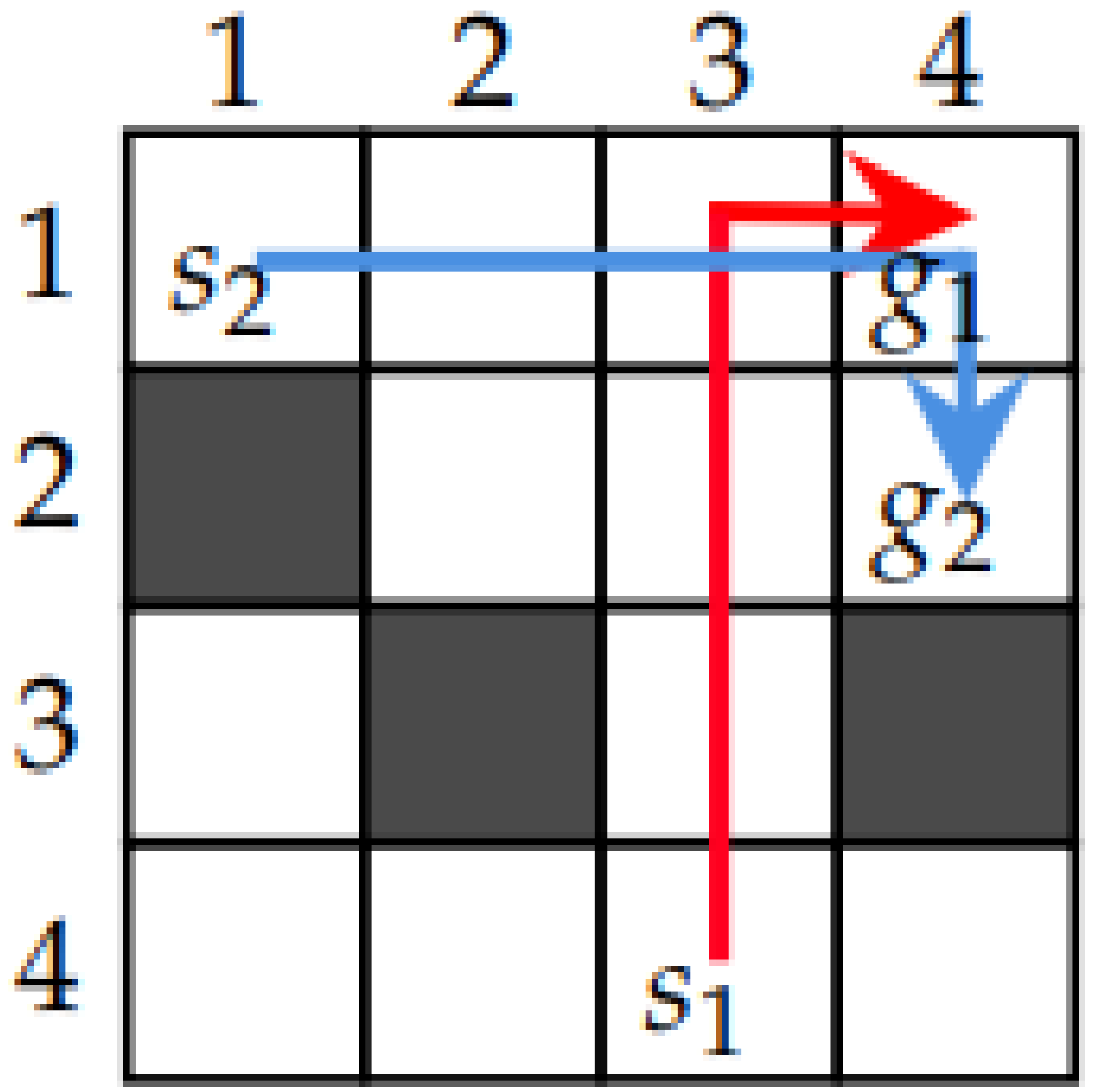
MAPF with removable obstacles

- For optimally solving a MAPF with removable obstacles (RO-MAPF) problem we mean returning the solution with the minimum cost and a set of removable obstacles to remove (within the budget) in order to allow agents to follow their paths without any collision

- A unitary budget and two removable obstacles:
  - λ_1 composed by two cells and a unitary cost
  - λ_2 composed by one cell and a unitary cost



(a)

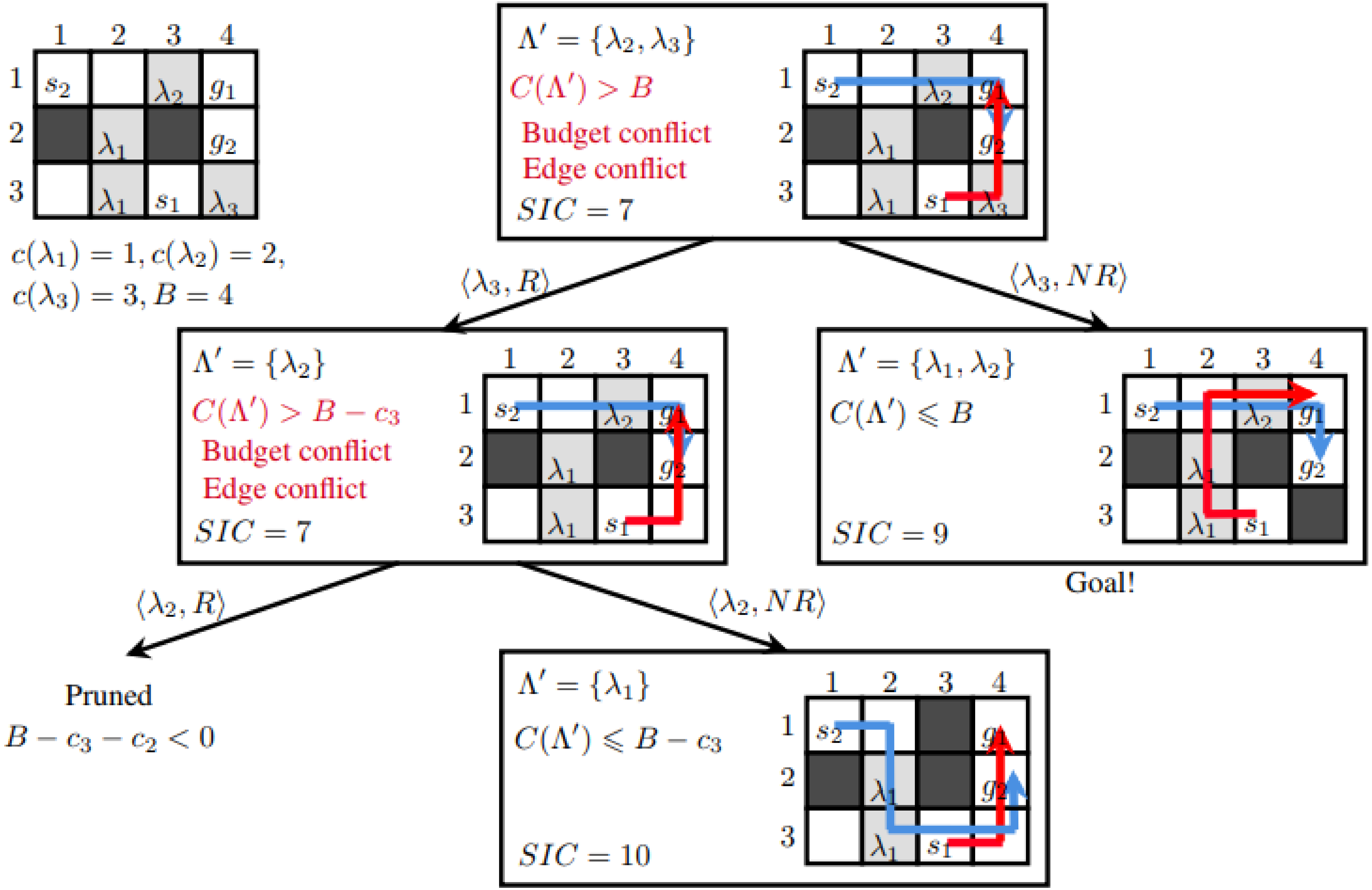


(b)

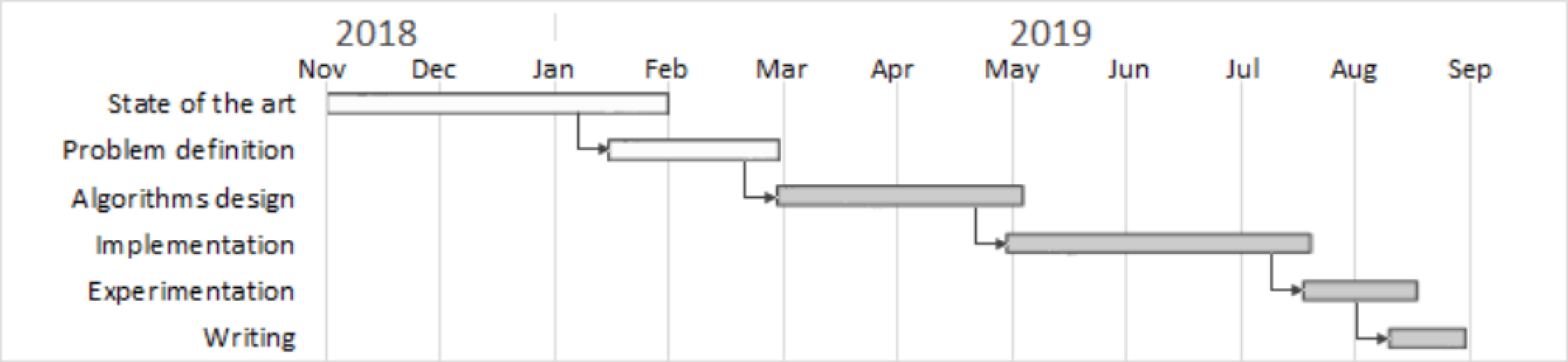
MAPF with removable obstacles

- MAPF problem is NP-hard to solve optimally, even on grid-based environments
- RO-MAPF problem is NP-hard to solve optimally too since it is a generalization of the MAPF problem
- Despite this intractability issue, our goal is to develop an effective CBS-based algorithm able to solve RO-MAPF problems optimally
  - For now, we have designed a draft version of our final solver

Draft version of our RO-MAPF solver



Research timeline



- We are keeping in mind some conference deadlines
  - AAAI 2020: deadline towards the beginning of September
  - AAMAS 2020 and ICAPS 2020: deadline in mid-November

Thank you

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