Research Project Proposal:
Robustness in Multi-Agent Pickup and Delivery

Giacomo Lodigiani
giacomo.lodigiani@mail.polimi.it
T2A Track

HONOURS PROGRAMME HP-SR in Information Technology

POLITECNICO MILANO 1863

Index

POLITECNICO MILANO 1863

Introduction

MAPF

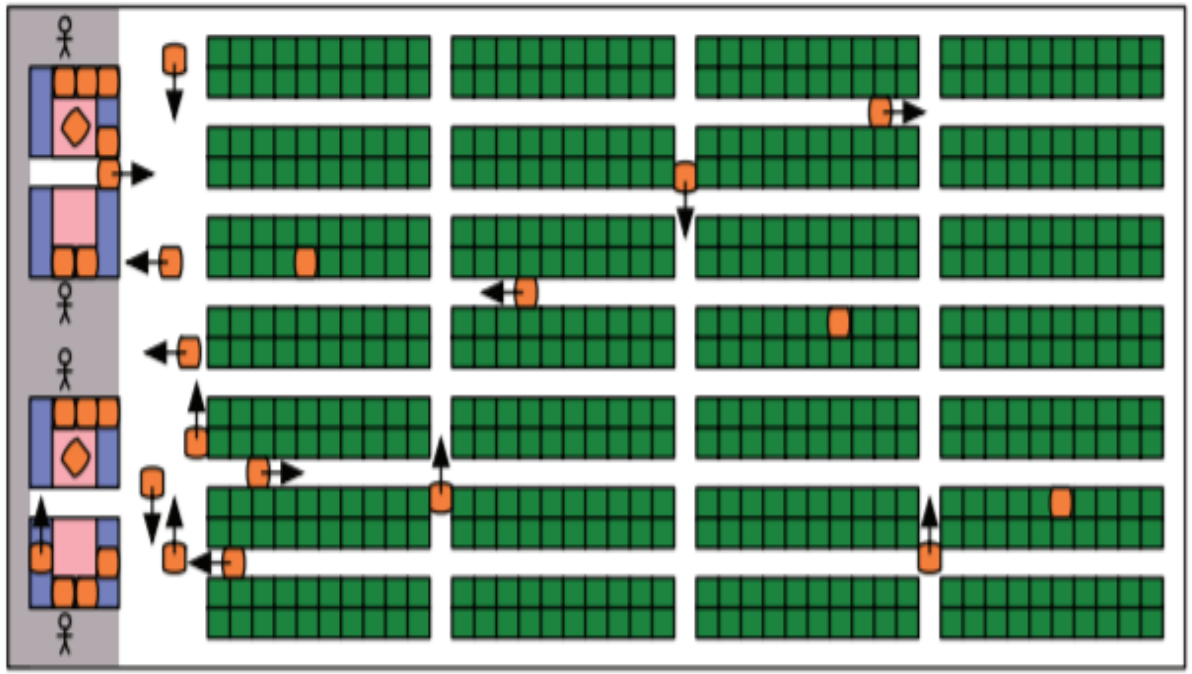
Multi-Agent Path Finding (**MAPF**) is a problem in the broader field of Multi-Robot Systems in which multiple agents must plan paths to preassigned targets, avoiding collisions and optimizing some cost function.

MAPD

Multi-Agent Pickup and Delivery (**MAPD**) is an extension of MAPF in which agents have also the freedom to assign themselves to targets which, unlike in the MAPF problem, are not a fixed set but may **change** at any time step.

POLITECNICO
MILANO 1863

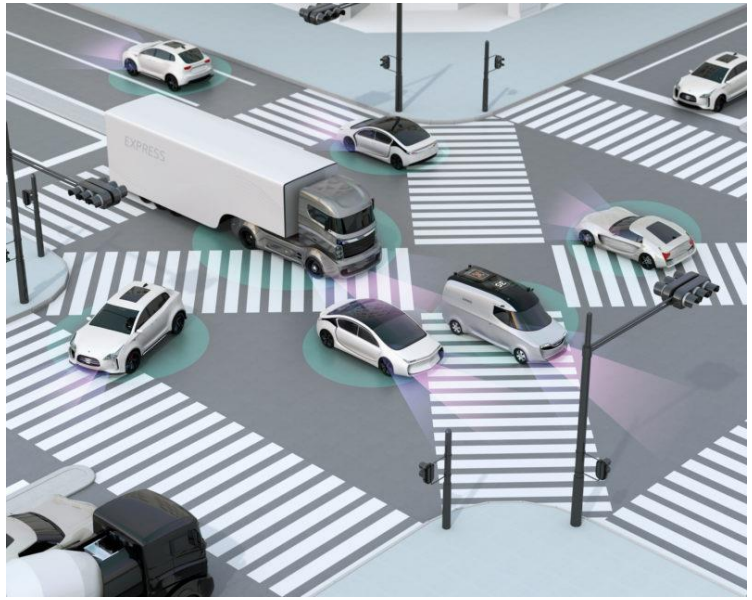
Applications



Logistics

POLITECNICO
MILANO 1863

Applications



Autonomous vehicles

Aircraft towing

Videogames

POLITECNICO MILANO 1863

Problem formulations - Preliminaries

Assumptions

- Agents are points, no kinematic constraints.
- Discretized environment, represented as an undirected graph $G = (V, E)$ whose vertices V correspond to locations and whose edges E correspond to connections between locations that the agents can traverse.
- Discretized time steps.

POLITECNICO
MILANO 1863

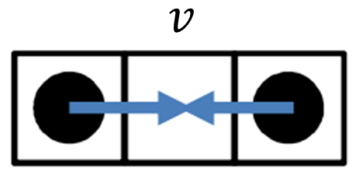
Problem formulations - Preliminaries

Constraints

Agents are not completely free when moving on the graph, some **constraints** need to be enforced to avoid collisions between them.

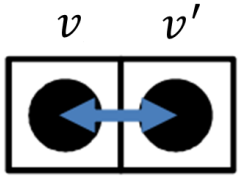
Vertex collision

A *vertex collision* is a tuple $\langle a_i, a_j, v, t \rangle$ where agents a_i and a_j occupy the same vertex v at the same time step t.



Edge collision

An *edge collision* is a tuple $\langle a_i, a_j, u, v, t \rangle$ where agents a_i and a_j traverse the same edge (u, v) in opposite directions between time steps t and $t + 1$.

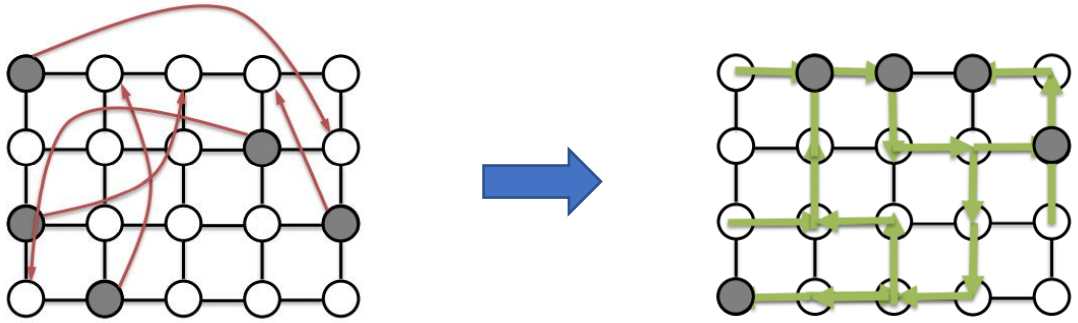


POLITECNICO
MILANO 1863

Problem formulations - MAPF

MAPF

- A MAPF problem instance consists of a given finite connected undirected graph $G = (V, E)$ and a given set of M agents $\{a_i | i \in [M]\}$. Each agent a_i has a start vertex $a_i \in V$ and a goal vertex $g_i \in V$.
- At each discrete time step, each agent a_i either moves to an adjacent vertex or waits at the same vertex.
- A MAPF plan consists of a path (sequence of actions leading an agent from its start location to its goal) π_i for each agent a_i. A MAPF solution is a MAPF plan whose paths are **collision-free**. The problem of MAPF is to find a solution minimizing a given cost measure, like *makespan* (maximum of the arrival times of all agents at their goal vertices) or *flowtime* (sum of the arrival times of all agents at their goal).



POLITECNICO
MILANO 1863

Problem formulations - MAPD

MAPD

- A MAPD problem instance consists of a given finite connected undirected graph $G = (V, E)$, a given set of M agents $\{a_i | i \in [M]\}$, each with an assigned initial vertex, and a **task set** T that contains the set of unexecuted tasks in the system. The task set changes **dynamically** as, at each time step, new tasks can be added to the system.
- At each discrete time step, each agent a_i either moves to an adjacent vertex or waits at the same vertex.
- A task can be assigned to one agent at a time. A free agent (agent not executing any task) can be assigned any task $\tau_j \in T$. In order to execute task τ_j, it then has to move first from its current vertex to the pickup vertex s_j of the task and then from there to the delivery vertex g_j of the task.
- The problem of MAPD is to find **collision-free** paths for the agents to finish executing all tasks, minimizing a cost measure like *makespan* or *service time* (average number of time steps needed to finish each task).

POLITECNICO
MILANO 1863

Complexity

MAPF complexity

While single agent path finding is tractable [Dijkstra, 1959], MAPF is **NP-hard** to solve optimally on general graphs for both makespan minimization and flowtime minimization [Yu and LaValle, 2013]. The optimal makespan and optimal flowtime of any MAPF problem instance are both bounded by $O(|V|^3)$ [Yu, 2015].

MAPD complexity

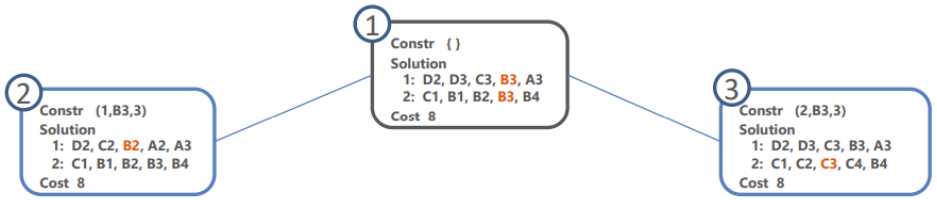
Being a generalization of MAPF, also MAPD is **NP-hard** to solve optimally [Ma, 2020].

POLITECNICO
MILANO 1863

Algorithms

MAPF algorithms

Various algorithms exists to solve MAPF problems optimally, the two of the most notable examples are *Conflict-Based Search* and *Increasing Cost Tree Search.* Since MAPF is NP-hard to solve optimally, bounded sub-optimal algorithms were developed to address applications in time constrained or real-time situations.



MAPD algorithms

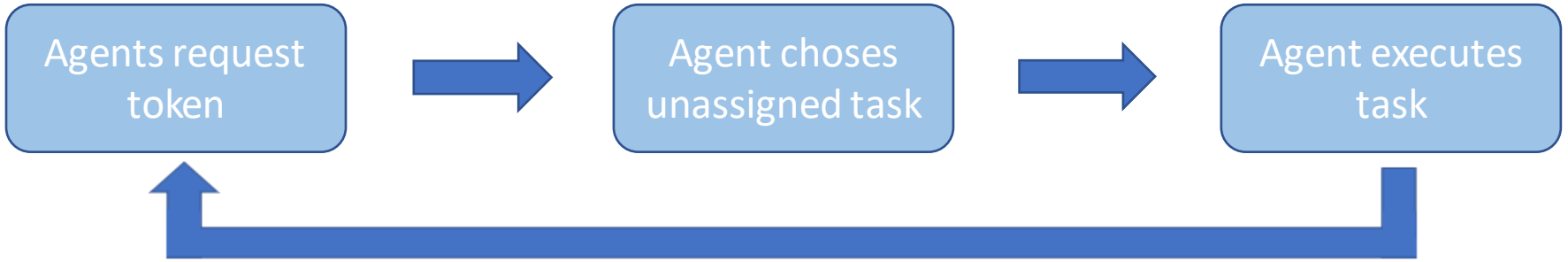
Two categories of online algorithms have been proposed: **decoupled** (where each agent assigns itself to tasks and computes its own collision-free paths given some global information) and **centralized**.

POLITECNICO
MILANO 1863

Algorithms - MAPD

Token Passing [Ma et al., 2017]

Token Passing (TP) is a **decoupled** algorithm based on a token, a synchronized shared block of memory that contains the current paths of all agents, the task set, and the task assignments that record which tasks are currently assigned to which agent.

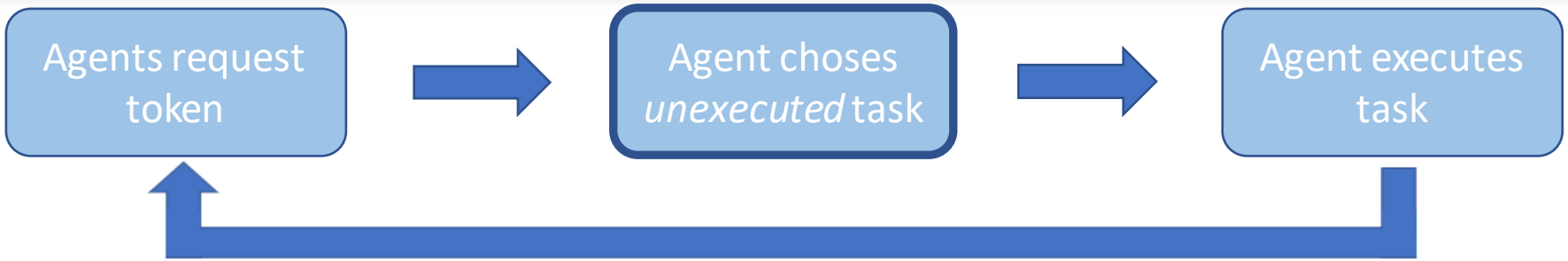


POLITECNICO MILANO 1863

Algorithms - MAPD

Token Passing with Task Swaps [Ma et al., 2017]

Token Passing with Task Swaps (TPTS) is a **decoupled** algorithm similar to TP except that its task set now contains all unexecuted tasks (agent is still moving to the pickup vertex of the task), rather than only unassigned tasks.



Central [Ma et al., 2017]

Central is a **centralized** algorithm that makes decisions for multiple agents at a time. Central allows agents to consider all unexecuted tasks like TPTS but uses a centralized target-assignment algorithm, the Hungarian method, to assign (or reassign) tasks to agents.

POLITECNICO MILANO 1863

Open challenges

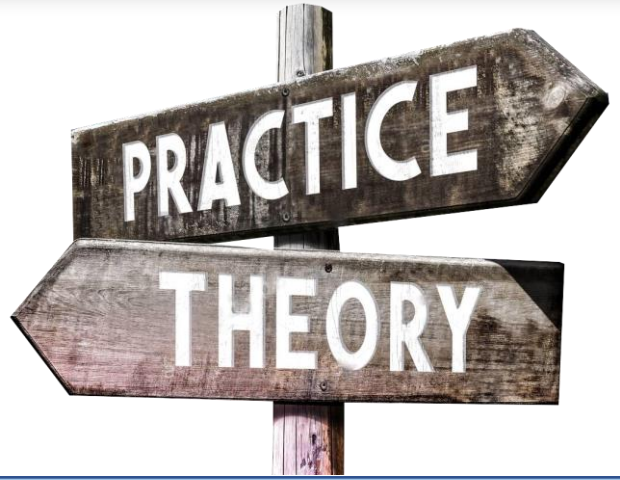
Theory vs. applications

MAPF offers a great theoretical foundation for improvement of multi-agent systems in many important areas.

- Direct implementations are difficult due to the many assumptions.

MAPD is a promising attempt trying to bring theory one step closer to applications.

- Theoretical framework still needs some work.
- Still **no guarantees** in a real context where execution is not perfect and the environment could be, in some cases, adversarial.



POLITECNICO
MILANO 1863

Robustness

Robustness

Robustness describes the property of a MAPF or MAPD algorithm of being able to **complete all** the planned **tasks** even in case some unexpected event, forces some deviation of the execution from the original plan.

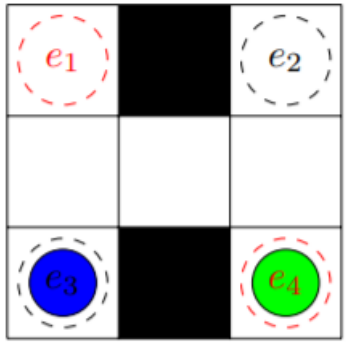
Long-term robustness

Long term robustness is the property of a MAPD algorithm of returning a solution, if one exists, that finishes a finite set of tasks in a finite amount of time. Well-formedness can provide a sufficient condition to allow for long-term robustness [Ma, 2020]. A MAPD problem instance is well-formed if and only if:

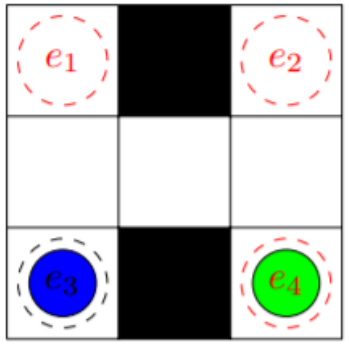
- The number of tasks is finite.
- There are no fewer non-task endpoints (designated vertices that allow the agents to rest) than the number of agents.
- For any two endpoints, there exists a path between them that traverses no other endpoints.

POLITECNICO
MILANO 1863

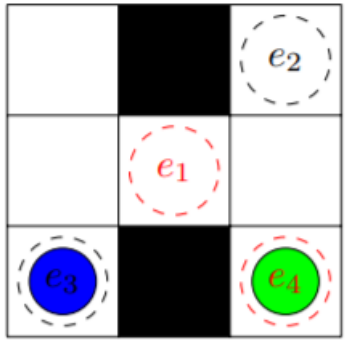
Long-term robustness examples



Well-formed



Non well-formed



Non well-formed

POLITECNICO
MILANO 1863

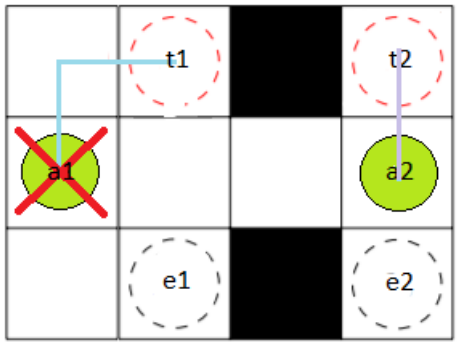
Research project - Goals

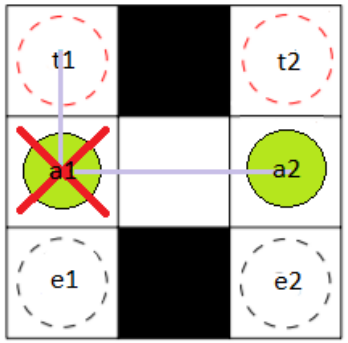
MAPD closer to applications

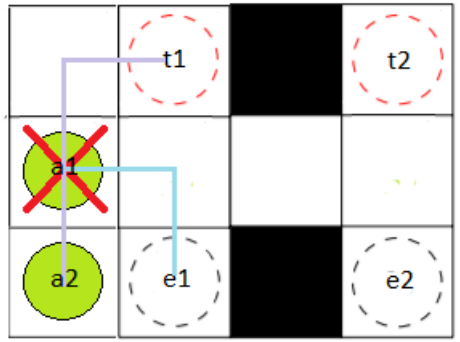
Long-term robustness in the MAPD framework has been studied only inside the original theoretical setting of the problem; long-term robustness in real application settings, where delays, failures and adversarial events may occur, still remains an open question.

Robust planning

Instead of simple recovery routines in case of failure or attack, we would like to create algorithms robust at **planning time**, that should in any case complete all tasks in a finite time.







POLITECNICO
MILANO 1863

Research project - Plan

Baseline algorithms

Analyze existing MAPD algorithms in terms of long-term robustness and extend them with simple recovery routines to allow the completion of plans even in case of unexpected events.

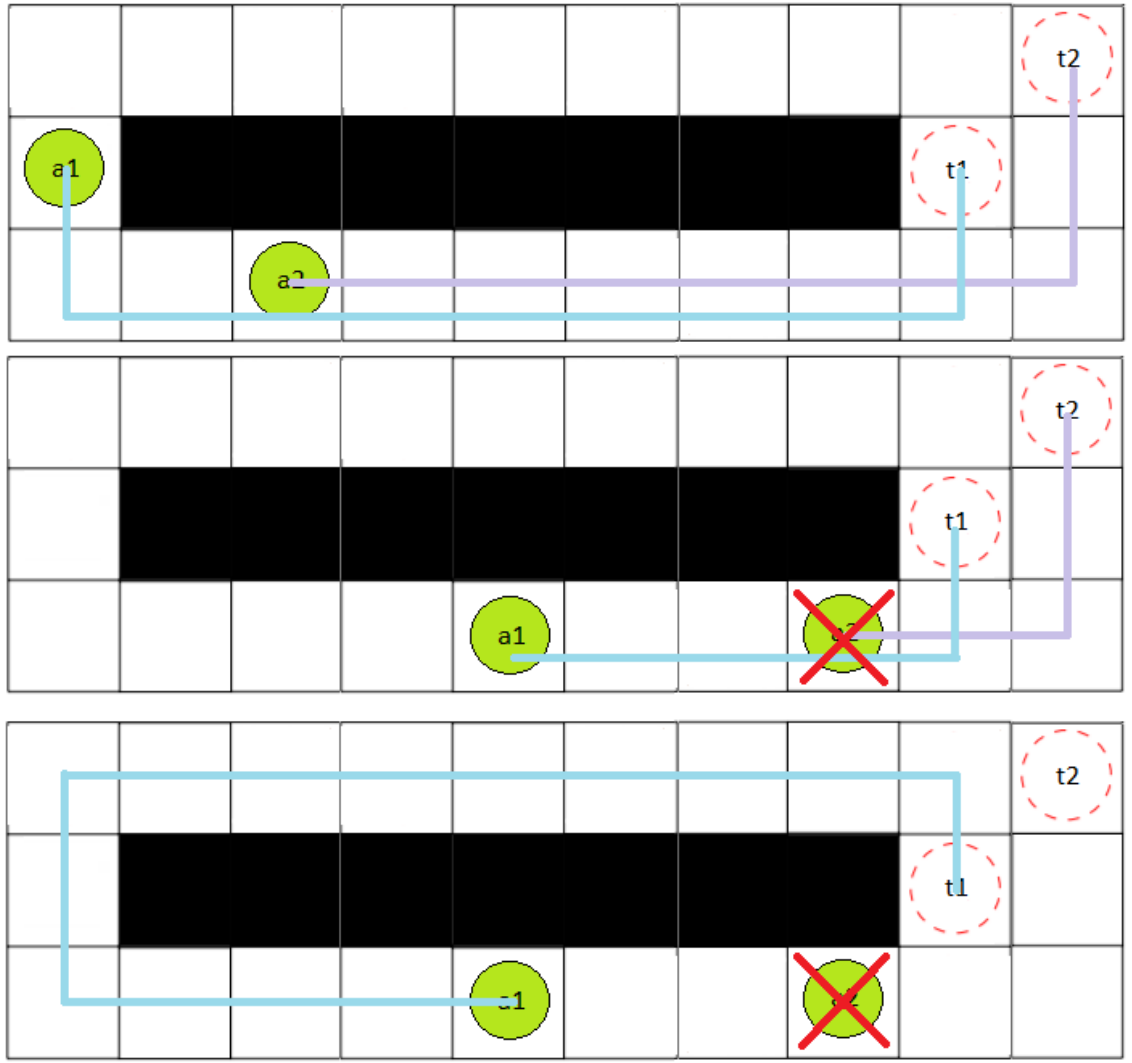
Frameworks

Formalize probabilistic and game theory frameworks to describe failures and attacks in the MAPD context and use them to evaluate the performance impact on MAPD algorithms with recovery routines.

POLITECNICO
MILANO 1863

Research project - Example

Recovery routine



POLITECNICO MILANO 1863

Research project - Plan

New robust algorithms

Design and implement **new robust** MAPD algorithms, even in case of failures or attacks.

- Focus on robustness at planning time, avoiding time expensive recovery routines.
- Validate the approach comparing performance with algorithms implementing naive recovery routines.
- Study the trade-off between robustness and performance.
- Estimate theoretical bounds for complexity and plan length.

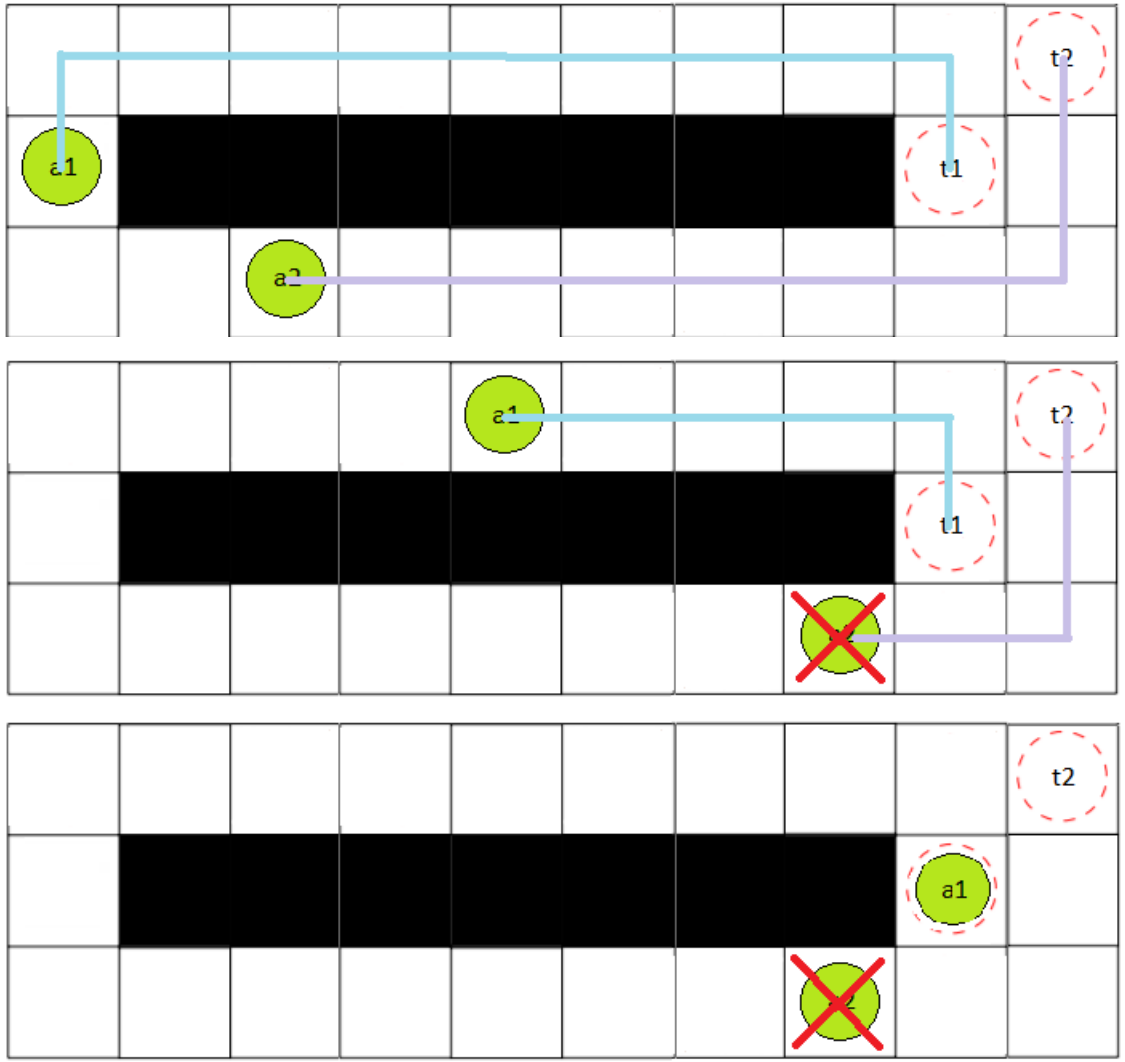
Evaluation metrics

To compare our algorithms, we will use classical metrics employed in the MAPF and MAPD domain, like makespan and service time. In particular, we will study the impact of adverse events and we will analyze how different algorithms manage to reduce the performance loss caused by an increasing rate of said events.

POLITECNICO
MILANO 1863

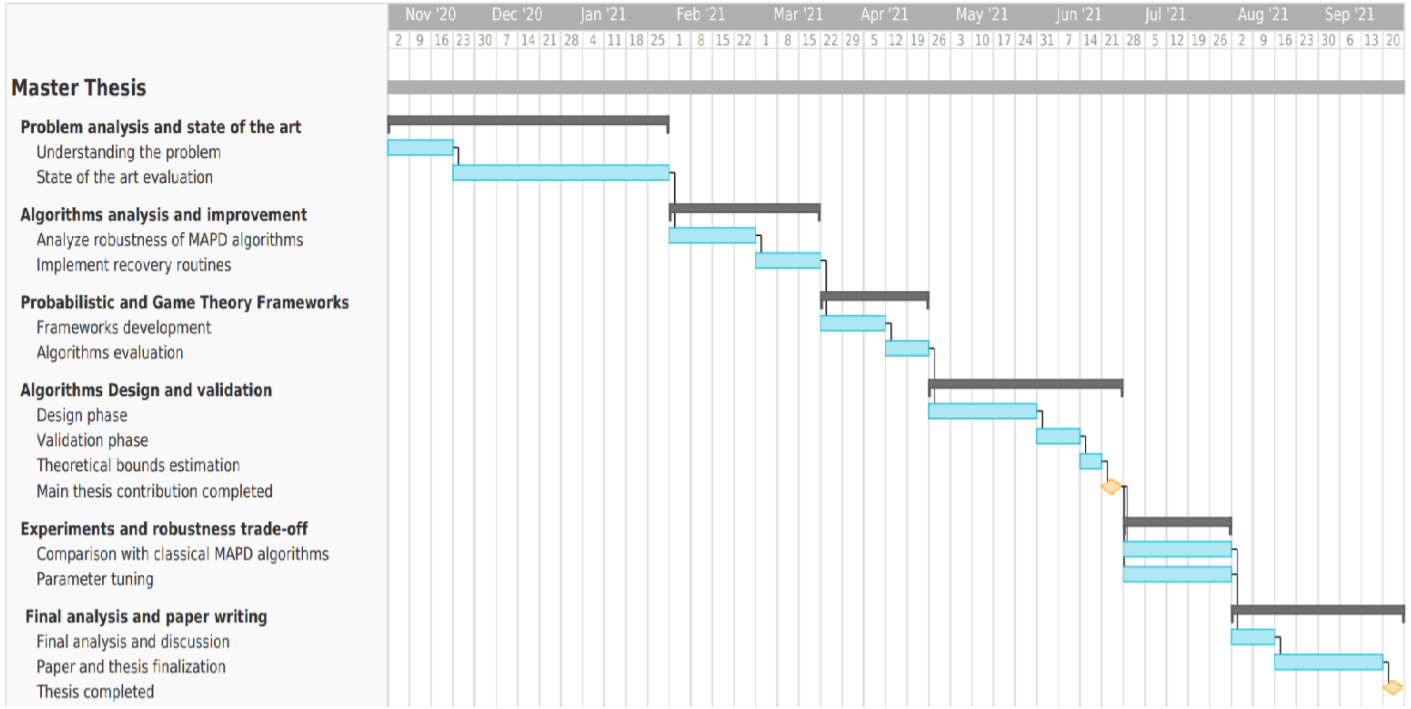
Research project - Example

Robust planning



POLITECNICO
MILANO 1863

Research project - Timeline



Important deadlines

- End of thesis work: September 2021.
- AAAI 2022: deadline towards the beginning of September.
- AAMAS 2022 and ICAPS 2022: deadline in mid-November.

POLITECNICO
MILANO 1863

Thanks for your attention!

POLITECNICO
MILANO 1863