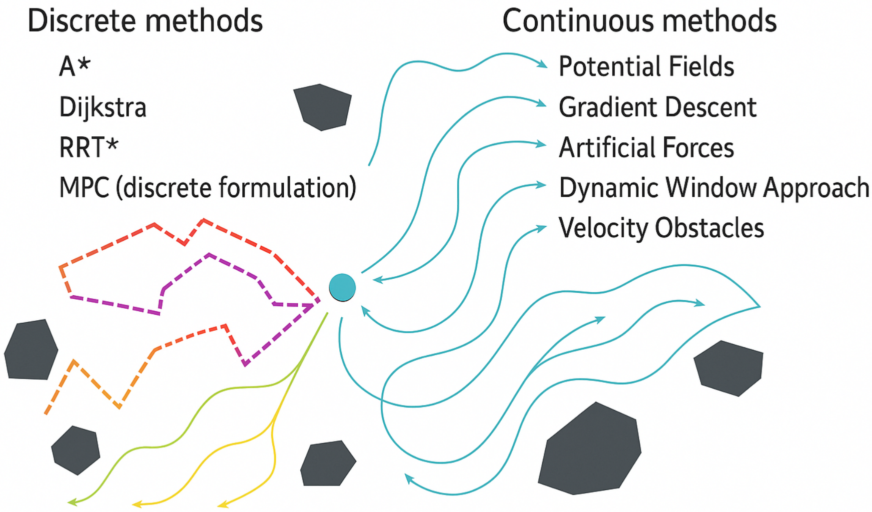
Efficient Path Planning and Task Allocation for Large Robotic Teams Using Petri Net Structural Properties

Cristian Mahulea

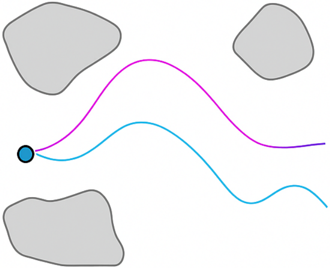
MSR'25 - Reims, France

Dep. of Computer Science and Systems Engineering,
University of Zaragoza, Spain
e-mail: cmahulea@unizar.es

Classical Motion Planning with One Robot

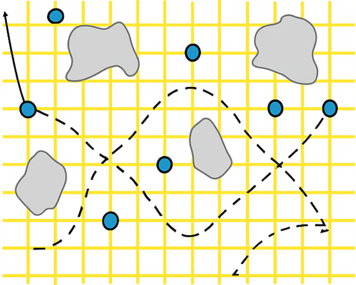


From Continuous Motion to Multi-Agent Discrete Models



Continuous + discrete methods work fine for one robot.

State space = $(\text{cells})^{(\text{robots})}$



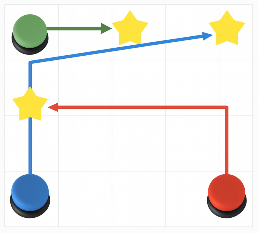
Need for discrete abstraction: graphs, Petri nets, automata

One robot → continuous models are fine.
Many robots → only discrete models survive.

Multi-Agent Path Finding (MAPF)

Outline

Multi-Agent Path Finding (MAPF)



Problem definition:

- Given N robots, each with a **start** and a **goal** cell.
- Find **collision-free paths** on a shared grid or roadmap.
- Robots move step-by-step; conflicts occur if:
  - two robots occupy the same cell at the same time (vertex conflict), or
  - traverse the same edge in opposite directions (edge conflict).

Main methods:

- **Conflict-Based Search (CBS):** 2-level search (high-level resolves conflicts; low-level replans per robot).
- **ECBS / EECBS:** bounded-suboptimal CBS with heuristics (EES) and optimizations (bypass, WDG, cardinal conflicts).

Goal: minimize total cost (sum-of-costs) or makespan while ensuring safety.

Each robot knows its goal; coordination ensures global consistency.

Synchronous Nature of CBS-Based MAPF

Key concept

- Global **time ticks**: move or wait each step.
- Conflicts defined at integer times (vertex/edge).
- Constraints $\Rightarrow$ global ordering of events.

Practice

- Often assume robots *disappear at goal* to keep flow.

Next: Petri nets for **asynchronous**, event-driven coordination.

Outline

Petri nets for MAPF: Decoupling Path Planning and Motion Control

Path Planning:

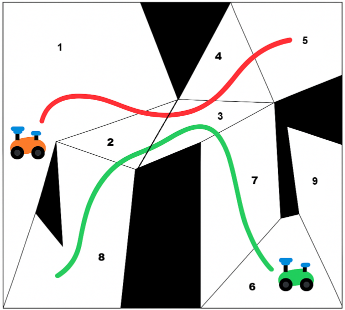
- Each robot computes a shortest path on $G = (Q, E)$ (Dijkstra, A*).
- **It is possible to compute additional feasible paths by using a k-shortest paths algorithm.**
- Paths computed independently, without synchronization.

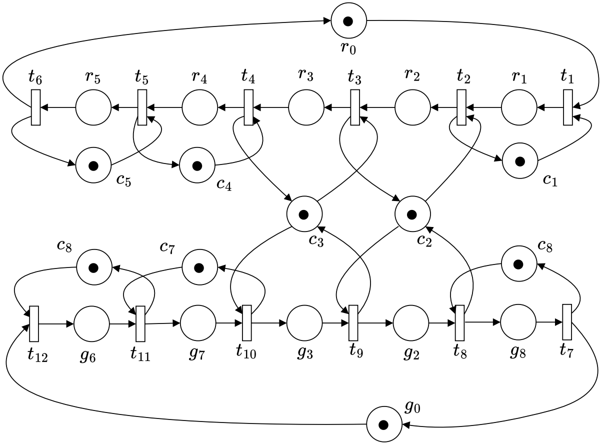
Petri Net Motion Control:

- Modeling the paths result in an S4PR net.
- **Resource places:** ensure mutual exclusion.
- **Monitor places:** prevent deadlocks structurally.
- Robots move asynchronously, each fires transitions when locally enabled.

Result: Global safety and liveness without centralized synchronization.

Motion Control

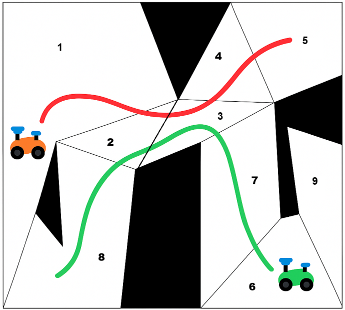




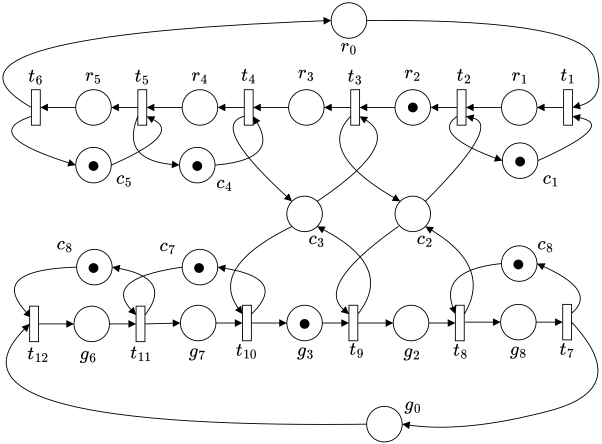
- Shared resources ⇒ **local semaphore**

M. Kloetzer, C. Mahulea and J.M. Colom, "Petri net approach for deadlock prevention in robot planning," In ETFA'2013: 18th IEEE International Conference on Emerging Technologies and Factory Automation, Cagliari, Italy, September 2013.

Motion Control

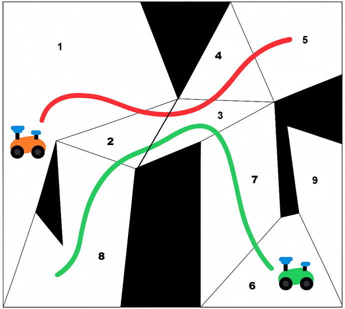


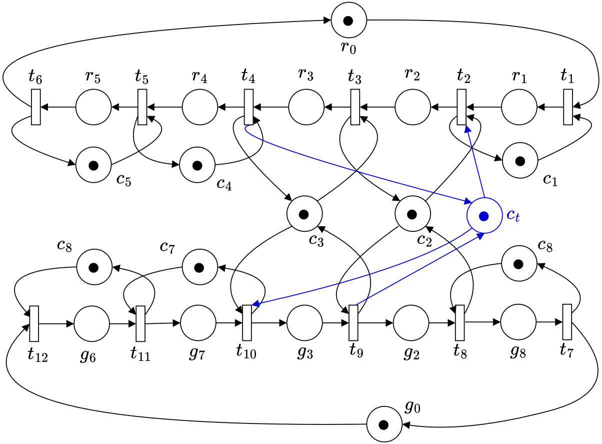
- Shared resources ⇒ **local semaphore**
- Unfortunately, the system may **deadlock**



M. Kloetzer, C. Mahulea and J.M. Colom, "Petri net approach for deadlock prevention in robot planning," In ETFA'2013: 18th IEEE International Conference on Emerging Technologies and Factory Automation, Cagliari, Italy, September 2013.

Motion Control





- Shared resources ⇒ **local semaphore**
- Unfortunately, the system may **deadlock**
- Monitor places ⇒ **local semaphore**

M. Kloetzer, C. Mahulea and J.M. Colom, "Petri net approach for deadlock prevention in robot planning," In ETFA'2013: 18th IEEE International Conference on Emerging Technologies and Factory Automation, Cagliari, Italy, September 2013.

CBS vs. S4PR Petri Net Formulation

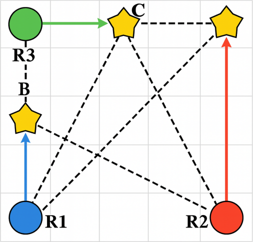
CBS / EECBS	Petri Net (S4PR)
Discrete global steps	Event-driven, asynchronous
Conflicts resolved by constraints	Conflicts prevented structurally (capacities)
Waiting inserted explicitly	Waiting emerges from disabled transitions
Global coordination	Local enabling of transitions

Takeaway: Petri nets provide a scalable, decentralized execution layer ensuring collision and deadlock avoidance for paths computed by fast planners.

Task Assignment and Path Finding (TAPF)

Outline

Task Assignment and Path Finding (TAPF)



Problem definition:

- Robots $R = \{r_1, \ldots, r_N\}$ and **tasks/goals** $G = \{g_1, \ldots, g_N\}$.
- Unlike MAPF, goals are **not pre-assigned**.
- **Objective**: find an **assignment** of robots to tasks *and* corresponding **collision-free paths**.

Previous Approaches:

- **Two-stage:** solve task assignment (Hungarian, auction) $\Rightarrow$ plan paths (CBS, prioritized).
- **Joint optimization:** combinatorial explosion, but ensures global optimality.

Challenge: coupling between assignment and routing.

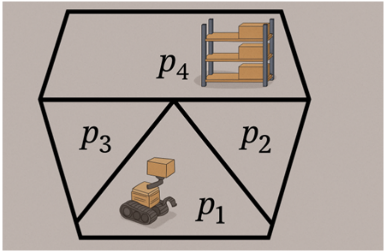
Outline

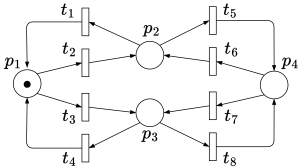
Robot Motion Petri Net System

Definition

A **Robot Motion Petri Net** system (RMPN) is a tuple $\mathcal{Q} = \langle \mathcal{N}, \boldsymbol{m}_0 \rangle$, where:

- $\mathcal{N} = \langle P, T, \boldsymbol{Post}, \boldsymbol{Pre} \rangle$ is a Petri net,
  - One place for each cell, e.g., $P = \{p_1, p_2, p_3, p_4\}$.
  - For each adjacent cells p_i and p_j two transitions are added if possible for a robot to go between regions, e.g., p_1 and p_3 adjacent $\implies t_3$ and t_4 are added.
- $\boldsymbol{m}_0$ is the initial marking, where $\boldsymbol{m}_0[p]$ gives the number of robots initially in $p \in P$.

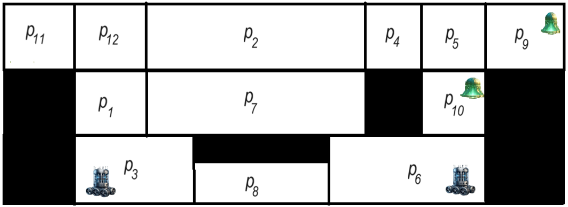


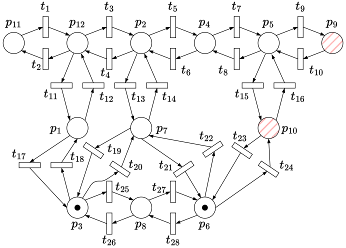


Some advantages of PNs

1. The state is numeric and distributed.
2. The size of the model does not depend on the number of robots.

Path planning using RMPN

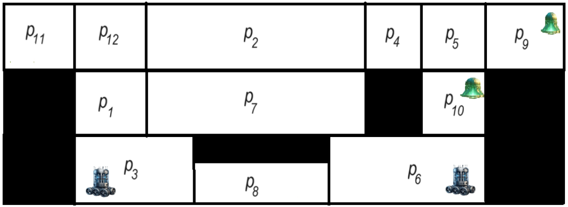


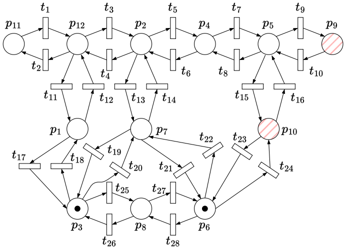


ILP for team of robots

- $\boldsymbol{m}_0 = \left[0,0,1,0,0,1,0,0,0,0,0,0\right]^T$
- $\boldsymbol{m}_f = \left[0,0,0,0,0,0,0,0,1,1,0,0\right]^T$
- $$\begin{array}{ll} \min & sum(\boldsymbol{\sigma}) \\ \text{s.t.:} & \boldsymbol{C} \cdot \boldsymbol{\sigma} = \boldsymbol{m}_f - \boldsymbol{m}_0, \\ & \boldsymbol{\sigma} \in \mathbb{N}^{|T|} \end{array}$$
- $\boldsymbol{\sigma} = [0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,1, 1,1,0,0,2,0,0,0,0]'$
- $\boldsymbol{\sigma} = t_9 + t_{16} + t_{20} + t_{21} + 2' t_{24}$

Path planning using RMPN



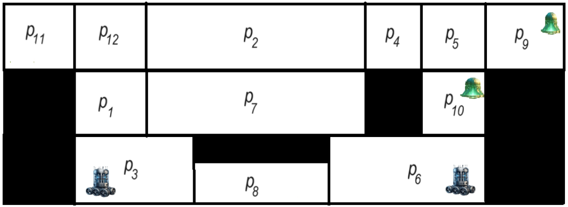


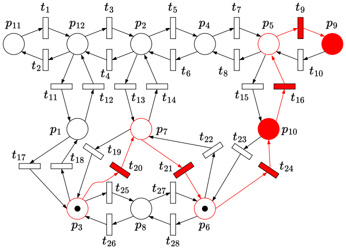
ILP for team of robots

- $\boldsymbol{m}_0 = \left[0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0\right]^T$
- $\boldsymbol{m}_f = \left[0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0\right]^T$
- $$\begin{array}{ll} \min & sum(\boldsymbol{\sigma}) \\ \text{s.t.:} & \boldsymbol{C} \cdot \boldsymbol{\sigma} = \boldsymbol{m}_f - \boldsymbol{m}_0, \\ & \boldsymbol{\sigma} \in \mathbb{N}^{|T|} \end{array}$$

- $\boldsymbol{\sigma} = [0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 2, 0, 0, 0, 0]'$
- $\boldsymbol{\sigma} = t_9 + t_{16} + t_{20} + t_{21} + 2' t_{24}$

Path planning using RMPN





ILP for team of robots

- $\boldsymbol{m}_0 = \left[0,0,1,0,0,1,0,0,0,0,0,0\right]^T$
- $\boldsymbol{m}_f = \left[0,0,0,0,0,0,0,0,1,1,0,0\right]^T$
- $$\begin{aligned} \min \quad & sum(\boldsymbol{\sigma}) \\ \text{s.t.:} \quad & \boldsymbol{C} \cdot \boldsymbol{\sigma} = \boldsymbol{m}_f - \boldsymbol{m}_0, \\ & \boldsymbol{\sigma} \in \mathbb{N}^{|T|} \end{aligned}$$
- $\boldsymbol{\sigma} = [0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,1,1,0,0,2,0,0,0,0]'$
- $\boldsymbol{\sigma} = t_9 + t_{16} + t_{20} + t_{21} + 2' t_{24}$

Path planning using RMPN

Integer Linear Program (ILP)

$$\begin{aligned} \min \quad & sum(\boldsymbol{\sigma}) \\ \text{s.t.:} \quad & \boldsymbol{C} \cdot \boldsymbol{\sigma} = \boldsymbol{m}_f - \boldsymbol{m}_0, \\ & \boldsymbol{\sigma} \in \mathbb{N}^{|T|} \end{aligned}$$

Properties of the ILP

- RMPN is a **state machine** PN
- $\boldsymbol{C}$ is Totally Unimodular (TU) matrix.

Path planning using RMPN

Integer Linear Program (ILP)

$$\begin{aligned} \min \quad & sum(\boldsymbol{\sigma}) \\ \text{s.t.:} \quad & \boldsymbol{C} \cdot \boldsymbol{\sigma} = \boldsymbol{m}_f - \boldsymbol{m}_0, \\ & \boldsymbol{\sigma} \in \mathbb{N}^{|T|} \end{aligned}$$

Properties of the ILP

- RMPN is a **state machine** PN
- $\boldsymbol{C}$ is Totally Unimodular (TU) matrix.

Definition (Totally Unimodular Matrix)

A matrix $\boldsymbol{A}$ is totally unimodular if every square submatrix has determinant 0, $+1$, or -1. In particular, this implies that all entries are 0 or ± 1.

Theorem

If $\boldsymbol{A}$ is TU and $\boldsymbol{b}$ is an integer vector, then $P = \{\boldsymbol{x} | \boldsymbol{A} \cdot \boldsymbol{x} \leq \boldsymbol{b}\}$ has integer vertices.

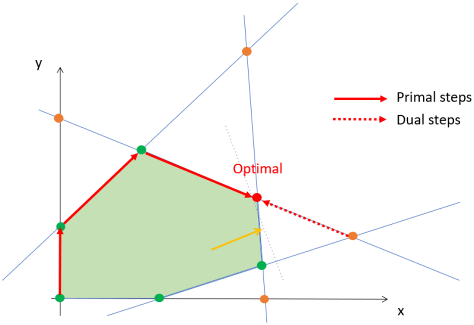
Path planning using RMPN

$$\begin{aligned} \min \quad & sum(\boldsymbol{\sigma}) \\ \text{s.t.:} \quad & \boldsymbol{C} \cdot \boldsymbol{\sigma} = \boldsymbol{m}_f - \boldsymbol{m}_0, \\ & \boldsymbol{\sigma} \in \mathbb{N}^{|T|} \end{aligned}$$

relaxing + dual-simplex

$$\begin{aligned} \min \quad & sum(\boldsymbol{\sigma}) \\ \text{s.t.:} \quad & \boldsymbol{C} \cdot \boldsymbol{\sigma} = \boldsymbol{m}_f - \boldsymbol{m}_0, \\ & \boldsymbol{\sigma} \in \mathbb{R}^{|T|} \end{aligned}$$

- Faster algorithm to solve the optimization problem.



Outline

Infinite Norm and Linear Programming

Definition (Infinite Norm of a Vector)

For a vector $\boldsymbol{x} \in \mathbb{R}^n$, the infinite norm (also known as the maximum norm) is defined as:

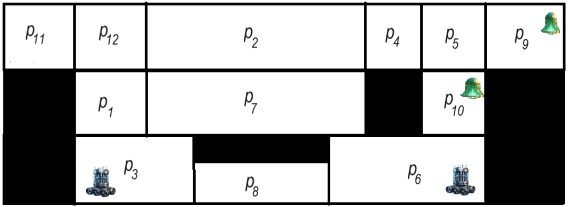
$$\|\boldsymbol{x}\|_\infty = \max_{i=1,\ldots,n} |x_i|$$

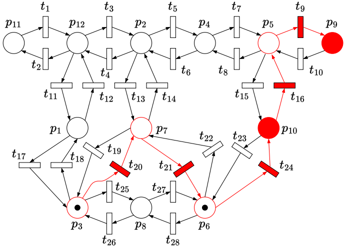
Minimizing the Infinite Norm Using Linear Programming

- Introduce a new variable b to represent the maximum absolute value of the components of $\boldsymbol{x}$.
- Solve the following LP to compute minimum b

$$\begin{aligned} \min \quad & b \\ \text{s.t.} \quad & -b \le x_i \le b, \\ & i = 1, \ldots, n \end{aligned} \iff \begin{aligned} \min \quad & b \\ \text{s.t.} \quad & \begin{cases} \boldsymbol{x} & \le b \cdot \mathbf{1} \\ \boldsymbol{x} & \ge -b \cdot \mathbf{1} \end{cases} \end{aligned} \xrightarrow{\boldsymbol{x} \ge 0} \begin{aligned} \min \quad & b \\ \text{s.t.} \quad & \boldsymbol{x} \le b \cdot \mathbf{1} \end{aligned}$$

Reducing congestion on paths





ILP for team of robots

- $\boldsymbol{m}_0 = \left[0,0,1,0,0,1,0,0,0,0,0,0\right]^T$
- $\boldsymbol{m}_f = \left[0,0,0,0,0,0,0,0,1,1,0,0\right]^T$
- $$\begin{aligned} \min \quad & sum(\boldsymbol{\sigma}) \\ \text{s.t.:} \quad & \boldsymbol{C} \cdot \boldsymbol{\sigma} = \boldsymbol{m}_f - \boldsymbol{m}_0, \\ & \boldsymbol{\sigma} \in \mathbb{N}^{|T|} \end{aligned}$$

- $\boldsymbol{\sigma} = [0,0,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,1,1,1,0,0,2,0,0,0,0]'$
- $\boldsymbol{\sigma} = t_9 + t_{16} + t_{20} + t_{21} + 2' t_{24}$
- $\boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0 = [0,0,1,0,1,2,1,0,1,2,0,0]'$
- $\boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0 = p_3 + p_7 + 2' p_6 + 2' p_{10} + p_5 + p_9$.

Collision avoidance through infinite norm

What about collision avoidance?

- $\boldsymbol{Post} \cdot \boldsymbol{\sigma} \rightarrow$ number of times the robots cross each cell from $\boldsymbol{m}_0$ to $\boldsymbol{m}_f$.
- From all possible σ to reach the desired marking $\boldsymbol{m}_f$ chose the one that minimize $||\boldsymbol{Post} \cdot \sigma + \boldsymbol{m}_0||_\infty \Rightarrow$ **reduce congestion.**

$$
\begin{aligned}
\min \quad & sum(\sigma) + M \cdot b \\
\text{s.t.:} \quad & \boldsymbol{m}_f = \boldsymbol{C} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0, \\
& \boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0 \leq b \cdot \mathbf{1}, \\
& \boldsymbol{\sigma} \in \mathbb{N}^{|T|}, b \in \mathbb{R}
\end{aligned}
$$

Collision avoidance through infinite norm

What about collision avoidance?

- $\boldsymbol{Post} \cdot \boldsymbol{\sigma} \rightarrow$ number of times the robots cross each cell from $\boldsymbol{m}_0$ to $\boldsymbol{m}_f$.
- From all possible $\boldsymbol{\sigma}$ to reach the desired marking $\boldsymbol{m}_f$ chose the one that minimize $||\boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0||_\infty \Rightarrow$ **reduce congestion.**

$$
\begin{aligned}
\min \quad & sum(\boldsymbol{\sigma}) + M \cdot b \\
\text{s.t.:} \quad & \boldsymbol{m}_f = \boldsymbol{C} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0, \\
& \boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0 \leq b \cdot \mathbf{1}, \\
& \boldsymbol{\sigma} \in \mathbb{N}^{|T|}, b \in \mathbb{R}
\end{aligned}
$$

Path planning using RMPN - team of robots

What about collision avoidance?

- If minimum $\|\boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0\|_\infty = 1$, maximum of one robot passes through each region (and based on the cost function no traverse the same edge in opposite directions) $\rightarrow$ **no collision will be possible!**

$$
\begin{aligned}
\min \quad & sum(\boldsymbol{\sigma}) + M \cdot b \\
\text{s.t.:} \quad & \boldsymbol{m}_f = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}, \\
& \boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0 \leq b \cdot \mathbf{1}, \\
& \boldsymbol{\sigma} \in \mathbb{N}^{|T|}, b \in \mathbb{R}
\end{aligned}
$$

Path planning using RMPN - team of robots

What about collision avoidance?

- If minimum $\|\boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0\|_\infty = 1$, maximum of one robot passes through each region (and based on the cost function no traverse the same edge in opposite directions) $\rightarrow$ **no collision will be possible!**

$$
\begin{aligned}
\min \quad & sum(\boldsymbol{\sigma}) + M \cdot b \\
\text{s.t.:} \quad & \boldsymbol{m}_f = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}, \\
& \boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0 \leq b \cdot \mathbf{1}, \\
& \boldsymbol{\sigma} \in \mathbb{N}^{|T|}, b \in \mathbb{R}
\end{aligned}
$$

Path planning using RMPN - team of robots

What about collision avoidance?

- If minimum $\|\boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0\|_\infty = 1$, maximum of one robot passes through each region (and based on the cost function no traverse the same edge in opposite directions) $\rightarrow$ **no collision will be possible!**

$$
\begin{aligned}
\min \quad & sum(\boldsymbol{\sigma}) + M \cdot b \\
\text{s.t.:} \quad & \boldsymbol{m}_f = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}, \\
& \boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0 \leq b \cdot \mathbf{1}, \\
& \boldsymbol{\sigma} \in \mathbb{N}^{|T|}, b \in \mathbb{R}
\end{aligned}
$$

- $\begin{bmatrix} \boldsymbol{C} \\ \boldsymbol{Post} \end{bmatrix}$ is totally unimodular but $\begin{bmatrix} \boldsymbol{C} & \mathbf{0} \\ \boldsymbol{Post} & \mathbf{1} \end{bmatrix}$ is not
- If the LP **relaxation** $(\boldsymbol{\sigma} \in \mathbb{R}^{|T|})$ has b^* integer then the solution is integer.
- If $b^* = 1 \Rightarrow$ no collision.
- If $b^* > 1 \Rightarrow$ add intermediate markings.

Path planning using RMPN - team of robots

What about collision avoidance?

- If minimum $\|\boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0\|_\infty = 1$, maximum of one robot passes through each region (and based on the cost function no traverse the same edge in opposite directions) $\rightarrow$ **no collision will be possible!**

$$
\begin{aligned}
\min \quad & sum(\boldsymbol{\sigma}) + M \cdot b \\
\text{s.t.:} \quad & \boldsymbol{m}_f = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}, \\
& \boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0 \leq b \cdot \mathbf{1}, \\
& \boldsymbol{\sigma} \in \mathbb{N}^{|T|}, b \in \mathbb{R}
\end{aligned}
$$

- $\begin{bmatrix} \boldsymbol{C} \\ \boldsymbol{Post} \end{bmatrix}$ is totally unimodular but $\begin{bmatrix} \boldsymbol{C} & \mathbf{0} \\ \boldsymbol{Post} & \mathbf{1} \end{bmatrix}$ is not
- If the LP **relaxation** $(\boldsymbol{\sigma} \in \mathbb{R}^{|T|})$ has b^* integer then the solution is integer.
- If $b^* = 1 \Rightarrow$ no collision.
- If $b^* > 1 \Rightarrow$ add intermediate markings.

Path planning using RMPN - team of robots

What about collision avoidance?

- If minimum $\|\boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0\|_\infty = 1$, maximum of one robot passes through each region (and based on the cost function no traverse the same edge in opposite directions) $\rightarrow$ **no collision will be possible!**

$$\begin{aligned} \min \quad & sum(\boldsymbol{\sigma}) + M \cdot b \\ \text{s.t.:} \quad & \boldsymbol{m}_f = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}, \\ & \boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0 \leq b \cdot \mathbf{1}, \\ & \boldsymbol{\sigma} \in \mathbb{N}^{|T|}, b \in \mathbb{R} \end{aligned}$$

- $\begin{bmatrix} \boldsymbol{C} \\ \boldsymbol{Post} \end{bmatrix}$ is totally unimodular but $\begin{bmatrix} \boldsymbol{C} & \mathbf{0} \\ \boldsymbol{Post} & \mathbf{1} \end{bmatrix}$ is not
- If the LP **relaxation** $(\boldsymbol{\sigma} \in \mathbb{R}^{|T|})$ has b^* integer then the solution is integer.
- If $b^* = 1 \Rightarrow$ no collision.
- If $b^* > 1 \Rightarrow$ add intermediate markings.

Path planning using RMPN - team of robots

New LP ($ceil(b^*) == 2$)

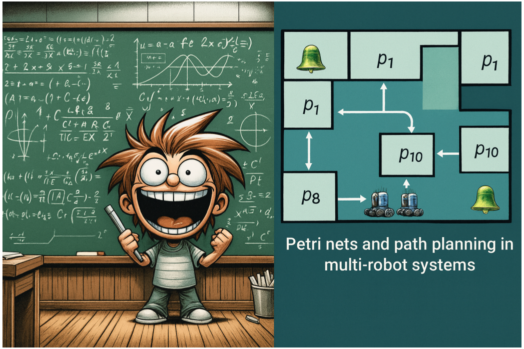
$$
\begin{aligned}
\min \quad & sum(\boldsymbol{\sigma_1}) + sum(\boldsymbol{\sigma_2}) \\
\text{s.t.:} \quad & \boldsymbol{C} \cdot \boldsymbol{\sigma_1} = \boldsymbol{m}_1 - \boldsymbol{m}_0, \\
& \boldsymbol{Post} \cdot \boldsymbol{\sigma_1} + \boldsymbol{m}_0 \leq \mathbf{1}, \\
& \boldsymbol{C} \cdot \boldsymbol{\sigma_2} = \boldsymbol{m}_f - \boldsymbol{m}_1, \\
& \boldsymbol{Post} \cdot \boldsymbol{\sigma_2} + \boldsymbol{m}_1 \leq \mathbf{1}, \\
& \boldsymbol{\sigma_1}, \boldsymbol{\sigma_2} \in \mathbb{R}^{|T|}, \boldsymbol{m}_1 \in \mathbb{R}^{|P|}
\end{aligned}
$$

TU matrix

The constraint matrix:

$$
\begin{bmatrix}
\boldsymbol{C} & \mathbf{0} & -\boldsymbol{I} \\
\boldsymbol{Post} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \boldsymbol{C} & \boldsymbol{I} \\
\mathbf{0} & \boldsymbol{Post} & \mathbf{0}
\end{bmatrix}
$$

, is totally unimodular.



Workflow

$$\begin{aligned} \min \quad & sum(\boldsymbol{\sigma}) + M \cdot b \\ \text{s.t.:} \quad & \boldsymbol{C} \cdot \boldsymbol{\sigma} = \boldsymbol{m}_f - \boldsymbol{m}_0, \\ & \boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0 \leq b \cdot \boldsymbol{1} \\ & \boldsymbol{\sigma} \in \mathbb{R}^{|T|}, b \in \mathbb{R} \end{aligned}$$

if $b^* \neq 1$

$$\begin{aligned} \min \quad & \sum_{i=1}^{\lceil b^* \rceil} sum(\boldsymbol{\sigma}_i) \\ \text{s.t.:} \quad & \boldsymbol{m}_i = \boldsymbol{m}_{i-1} + \boldsymbol{C} \cdot \boldsymbol{\sigma}_i, \quad i = 1, \ldots, \lceil b^* \rceil \\ & \boldsymbol{Post} \cdot \boldsymbol{\sigma}_i + \boldsymbol{m}_{i-1} \leq \boldsymbol{1}, \quad i = 1, \ldots, \lceil b^* \rceil \\ & \boldsymbol{\sigma}_i \in \mathbb{R}^{|T|}, \quad i = 1, \ldots, \lceil b^* \rceil \end{aligned}$$

I. Hustiu and R. Abolpour, M. Kloetzer and C. Mahulea, "Efficient Path Planning and Task Allocation Algorithm for Boolean Specifications," *arXiv*, `https://arxiv.org/abs/2506.04881`, 2025.

Task-Assignment and Path Finding - Simulations

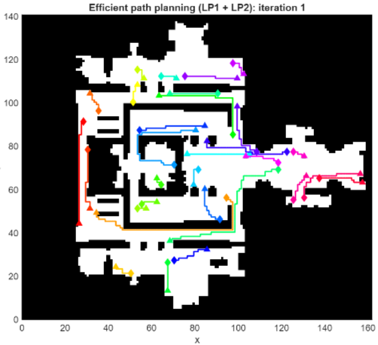


TABLE I
TAPF PROBLEM: MEAN VALUES FOR PROPOSED ALGORITHM VS. ILP FORMULATION

n_R	runtime (s) ((6) + (7)) ours (LPs)	ILPs	cost	synchronizations $\bar{s}$ mean	min	max	SR %
10	0.39	0.44	450.7	1.00	1	1	100
50	1.40	2.37	1329.1	1.40	1	2	100
100	4.75	5.85	2029.3	2.00	1	3	100
250	6.30	8.47	3003.0	2.65	2	5	100
500	9.98	15.83	4413.2	3.52	2	5	85
750	12.77	20.83	5320.5	4.06	2	5	80
1000	13.15	31.87	5794.3	4.50	4	5	50
1250	12.50	20.48	6300.1	4.77	4	5	45
1500	13.08	23.49	6462.5	5.00	5	5	10
1750	10.34	38.61	6724.5	5.00	5	5	10
2000	3.83	37.99	5837.0	4.50	4	5	10
2250	3.52	22.37	7009.0	5.00	5	5	5
2500	–	–	–	–	–	–	0

- Image of 141 $\times$ 162 pixels; after removing obstacles (black pixels), the environment comprises 7, 461 free pixels (places) and 27, 926 transitions (adjacency arcs).
- 20 random start-goal configurations were generated from the benchmark map.

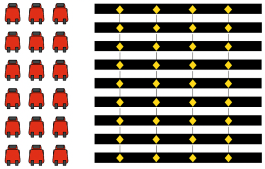
R. Stern, N. Sturtevant, A. Felner, S. Koenig, H. Ma, T. Walker, J. Li, D. Atzmon, L. Cohen, T. Kumar *et al.*, "Multi-agent pathfinding: definitions, variants, and benchmarks," *Proc. of the Int. Symposium on Combinatorial Search (SoCS)*, vol. 10, no. 1, 2019, pp. 151-158.

Path Planning with Boolean Specifications

Outline

Boolean Specifications for Multi-Goal Missions

Example: warehouse environment



Problem definition:

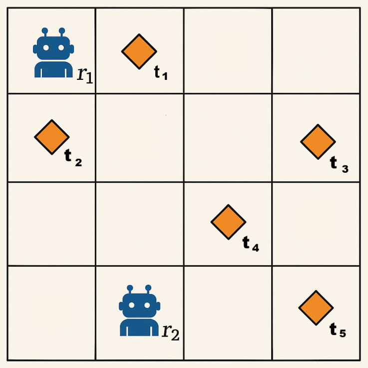
- Robots $R = \{r_1, \ldots, r_N\}$ and goals $G = \{g_1, \ldots, g_M\}$, typically with $M \geq N$.
- Given a planning horizon H, tipcially $\lceil \frac{M}{N} \rceil \leq H \leq M$, representing the number of service "rounds".

- For each goal g_i and each step $k \in \{1, \ldots, H\}$, introduce a Boolean variable y_i^k:

$$y_i^k = 1 \iff \text{goal } g_i \text{ is scheduled in round } k.$$

- **Mission formulation:** Boolean expressions encode **goal–round feasibility constraints** (e.g., every goal must be scheduled in at least one round).
- **Objective:** compute **collision-free** robot paths that satisfy the Boolean mission.

Example: Boolean Specification for Multi-Goal Missions



Scenario (2 robots, 5 goals)

- Robots: r_1, r_2
- Goals: $g_1, \ldots, g_5$

- Use $H = \lceil \frac{5}{2} \rceil = 3$ scheduling rounds.
- Introduce Boolean variables y_i^k:

$$y_i^k = 1 \iff \text{goal } g_i \text{ is scheduled in round } k.$$

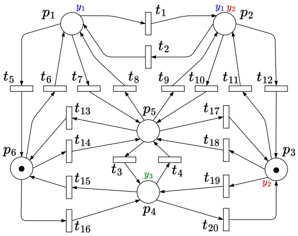
Boolean mission:

$$\varphi = \bigwedge_{i=1}^{5} (y_i^1 \vee y_i^2 \vee y_i^3)$$

Each goal must be scheduled in at least one round.

Outline

Idea 1: Reachability of a Labeled Region



Observation

$$\boldsymbol{v}_1 \cdot \boldsymbol{m} = \boldsymbol{m}[p_1] + \boldsymbol{m}[p_2] \geq 1 \iff$$

y_1 is active at $\boldsymbol{m}$.

Characteristic vector of y_i

- Let $\boldsymbol{v}_i \in \{0,1\}^{1\times|P|}$ be the characteristic row vector of y_i, defined as

$$\boldsymbol{v}_i[p_j] = \begin{cases} 1, & \text{if } p_j \text{ is labeled by } y_i, \\ 0, & \text{otherwise.} \end{cases}$$

- For y_1, for instance:

$$\boldsymbol{v}_1 = [1 \;\; 1 \;\; 0 \;\; 0 \;\; 0 \;\; 0].$$

Computing a marking that activates y_1

$$\min \quad \text{sum}(\boldsymbol{\sigma})$$
$$\text{s.t.} \quad \begin{cases} \boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}, \\ \boldsymbol{v}_1 \cdot \boldsymbol{m} \geq 1, \\ \boldsymbol{m} \in \mathbb{R}^{|P|}, \;\; \boldsymbol{\sigma} \in \mathbb{R}^{|T|}. \end{cases}$$

Idea 2: From Boolean Formula to Linear Constraints

For each atomic proposition y_i, define a Boolean variable

$$x_i = \begin{cases} 1, & \text{if } y_i \text{ is required to be active,} \\ 0, & \text{otherwise.} \end{cases}$$

Theorem (Classical result)

A Boolean formula φ is satisfied if and only if the variables $\mathbf{x}$ satisfy a corresponding set of linear constraints that can be automatically generated.

Example 1: conjunction

$$\varphi_1 = y_1 \wedge y_2 \wedge \neg y_3$$

$$\iff \begin{cases} x_1 \geq 1, \\ x_2 \geq 1, \\ x_3 \leq 0. \end{cases}$$

Example 2: disjunction

$$\varphi_2 = y_1 \vee y_2 \vee \neg y_3$$

$$\iff x_1 + x_2 + (1 - x_3) \geq 1.$$

Idea 3: PN Reachability with Boolean Constraints

- **Goal.** Compute a marking $\boldsymbol{m}$ at which the active observations satisfy

$$\varphi_3 = y_1 \wedge y_2 \wedge \neg y_3.$$

- **Reachability constraint**

$$\boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}, \qquad \boldsymbol{m} \in \mathbb{N}^{|P|}, \ \boldsymbol{\sigma} \in \mathbb{N}^{|T|}.$$

- **Boolean constraints (from previous slide)**

$$x_1 \geq 1, x_2 \geq 1, x_3 \leq 0.$$

- **Enforcing observation activation at marking $\boldsymbol{m}$.** For each $y_i/\boldsymbol{v}_i$,

$$x_i = 1 \implies \boldsymbol{v}_i \cdot \boldsymbol{m} \geq 1, \qquad x_i = 0 \implies \boldsymbol{v}_i \cdot \boldsymbol{m} = 0.$$

Big-M encoding:

$$\begin{cases} \boldsymbol{v}_i \cdot \boldsymbol{m} \leq M \cdot x_i, \\ \boldsymbol{v}_i \cdot \boldsymbol{m} \geq x_i, \end{cases} \qquad M \text{ sufficiently large.}$$

Path planning with Boolean Specifications (H=1)

$$\min \quad \alpha \cdot \mathbf{1}^T \cdot \boldsymbol{\sigma} \; + \quad \beta \cdot ||\boldsymbol{m} - \boldsymbol{m}_0||_1 \quad + \; M \cdot b$$

$$\text{s.t.} \quad \begin{cases} \boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}, \\ \text{Linear inequalities of the formula } \varphi, \\ x_i \leq \boldsymbol{v}_i \cdot \boldsymbol{m} \leq M \cdot x_i, \quad \forall y_i \in \mathcal{Y} \\ \boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0 \leq b \cdot \mathbf{1} \\ \boldsymbol{m} \in \mathbb{N}^{|P|}, \boldsymbol{\sigma} \in \mathbb{N}^{|T|}, \boldsymbol{x} \in \{0,1\}^{|\mathcal{Y}|}, b \in \mathbb{N}. \end{cases}$$

Path planning with Boolean Specifications (H=1)

travelled regions

$$\min \quad \alpha \cdot \mathbf{1}^T \cdot \boldsymbol{\sigma} + \beta \cdot ||\boldsymbol{m} - \boldsymbol{m}_0||_1 + M \cdot b$$

$$\text{s.t.} \quad \begin{cases} \boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}, \\ \text{Linear inequalities of the formula } \varphi, \\ x_i \leq \boldsymbol{v}_i \cdot \boldsymbol{m} \leq M \cdot x_i, \quad \forall y_i \in \mathcal{Y} \\ \boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0 \leq b \cdot \mathbf{1} \\ \boldsymbol{m} \in \mathbb{N}^{|P|}, \boldsymbol{\sigma} \in \mathbb{N}^{|T|}, \boldsymbol{x} \in \{0,1\}^{|\mathcal{Y}|}, b \in \mathbb{N}. \end{cases}$$

Path planning with Boolean Specifications (H=1)

moving robots

$$\min \quad \alpha \cdot \mathbf{1}^T \cdot \boldsymbol{\sigma} \;+\; \beta \cdot ||\boldsymbol{m} - \boldsymbol{m}_0||_1 \;+\; M \cdot b$$

$$\text{s.t.} \quad \begin{cases} \boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}, \\ \text{Linear inequalities of the formula } \varphi, \\ x_i \leq \boldsymbol{v}_i \cdot \boldsymbol{m} \leq M \cdot x_i, \quad \forall y_i \in \mathcal{Y} \\ \boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0 \leq b \cdot \mathbf{1} \\ \boldsymbol{m} \in \mathbb{N}^{|P|}, \boldsymbol{\sigma} \in \mathbb{N}^{|T|}, \boldsymbol{x} \in \{0,1\}^{|\mathcal{Y}|}, b \in \mathbb{N}. \end{cases}$$

Path planning with Boolean Specifications (H=1)

crossing cells

$$\min \quad \alpha \cdot \mathbf{1}^T \cdot \boldsymbol{\sigma} \;+\; \beta \cdot ||\boldsymbol{m} - \boldsymbol{m}_0||_1 \;+\; M \cdot b$$

$$\text{s.t.} \quad \begin{cases} \boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}, \\ \text{Linear inequalities of the formula } \varphi, \\ x_i \leq \boldsymbol{v}_i \cdot \boldsymbol{m} \leq M \cdot x_i, \quad \forall y_i \in \mathcal{Y} \\ \boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0 \leq b \cdot \mathbf{1} \\ \boldsymbol{m} \in \mathbb{N}^{|P|}, \boldsymbol{\sigma} \in \mathbb{N}^{|T|}, \boldsymbol{x} \in \{0,1\}^{|\mathcal{Y}|}, b \in \mathbb{N}. \end{cases}$$

Path planning with Boolean Specifications (H=1)

$$\min \quad \alpha \cdot \mathbf{1}^T \cdot \boldsymbol{\sigma} \; + \quad \beta \cdot ||\boldsymbol{m} - \boldsymbol{m}_0||_1 \quad + \; M \cdot b$$

$$\text{s.t.} \quad \begin{cases} \boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}, \\ \text{Linear inequalities of the formula } \varphi, \\ x_i \leq \boldsymbol{v}_i \cdot \boldsymbol{m} \leq M \cdot x_i, \quad \forall y_i \in \mathcal{Y} \\ \boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0 \leq b \cdot \mathbf{1} \\ \boldsymbol{m} \in \mathbb{N}^{|P|}, \boldsymbol{\sigma} \in \mathbb{N}^{|T|}, \boldsymbol{x} \in \{0,1\}^{|\mathcal{Y}|}, b \in \mathbb{N}. \end{cases}$$

Assumption

Assume each atomic proposition is assigned to one an only one region, i.e.,

$$\forall y_i \in \mathcal{Y}, sum(\boldsymbol{v}_i) = 1.$$

Path planning with Boolean Specifications (H=1)

$$\min \quad \alpha \cdot \mathbf{1}^T \cdot \boldsymbol{\sigma} \; + \quad \beta \cdot ||\boldsymbol{m} - \boldsymbol{m}_0||_1 \quad + \; M \cdot b$$

$$\text{s.t.} \quad \begin{cases} \boldsymbol{m} = \boldsymbol{m}_0 + \boldsymbol{C} \cdot \boldsymbol{\sigma}, \\ \text{Linear inequalities of the formula } \varphi, \\ x_i \leq \boldsymbol{v}_i \cdot \boldsymbol{m} \leq M \cdot x_i, \quad \forall y_i \in \mathcal{Y} \\ \boldsymbol{Post} \cdot \boldsymbol{\sigma} + \boldsymbol{m}_0 \leq b \cdot \mathbf{1} \\ \boldsymbol{m} \in \mathbb{R}^{|P|}, \boldsymbol{\sigma} \in \mathbb{R}^{|T|}, \boldsymbol{x} \in \{0,1\}^{|\mathcal{Y}|}, b \in \mathbb{N}. \end{cases}$$

Assumption

Assume each atomic proposition is assigned to one an only one region, i.e.,

$$\forall y_i \in \mathcal{Y}, sum(\boldsymbol{v}_i) = 1.$$

Efficient optimization

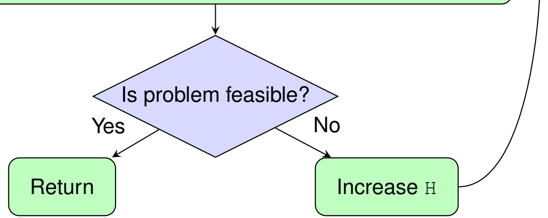
MILP with $|\mathcal{Y}| + 1$ integer variables!

Multi-Robot Gathering Problem: Optimal Algorithm

$$\min \quad \sum_{i=1}^{H} \boldsymbol{\sigma}_i$$

$$\text{s.t.:} \quad \begin{cases} \boldsymbol{m}_i = \boldsymbol{m}_{i-1} + \boldsymbol{C} \cdot \boldsymbol{\sigma}_i, & i = 1, \ldots, H \\ \boldsymbol{Post} \cdot \boldsymbol{\sigma}_i + \boldsymbol{m}_{i-1} \leq \mathbf{1}, & i = 1, \ldots, H \\ \boldsymbol{V} \cdot \boldsymbol{m} - M \cdot \boldsymbol{x}_i \leq \mathbf{0}, & i = 1, \ldots, H \\ \boldsymbol{x}_i - \boldsymbol{V} \cdot \boldsymbol{m}_i \leq \mathbf{0}, & i = 1, \ldots, H \\ \sum_{i=1}^{H} \boldsymbol{x}_j^i = 1, & j = 1, \ldots, \texttt{num\_objects} \end{cases}$$

$$\boldsymbol{\sigma}_i \in \mathbb{R}_{\geq 0}^{|T|}, \boldsymbol{m}_i \in \mathbb{R}_{\geq 0}^{|P|}, \boldsymbol{x}_i \in \{0, 1\}^{|\mathcal{Y}|}, i = 1, \ldots, H$$



Multi-robot gathering: `den312d.map`

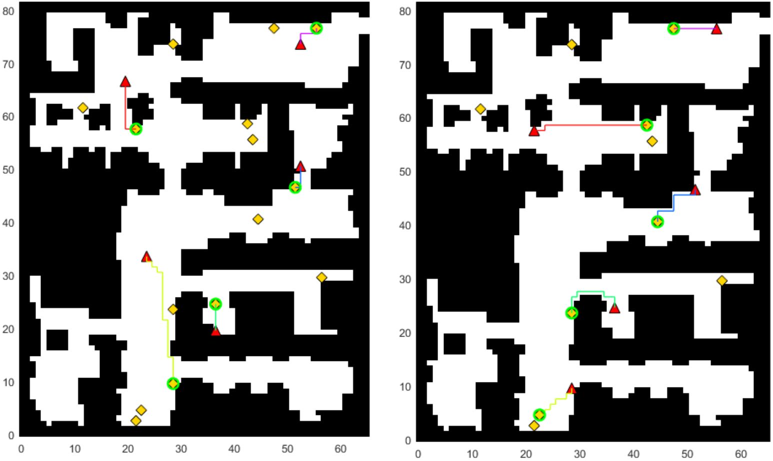
# Robots	# Tasks	Integer var.	MILP [s]
5	5	5	0.5235
5	10	20	6.8666
5	15	45	109.0247
5	20	80	2087.0085
5	25	125	-

Table 1: Timing for different number of tasks.

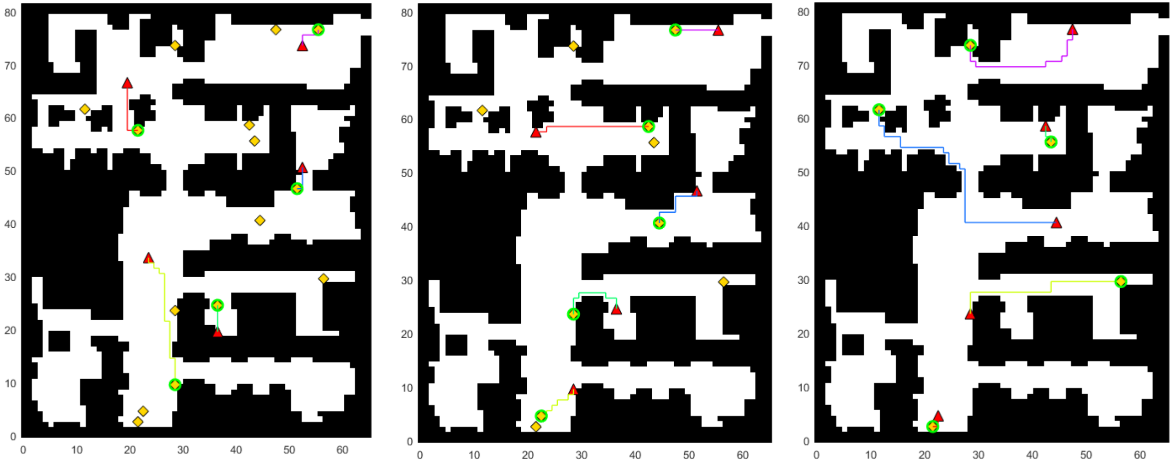
- `den312d.map` $\Rightarrow$ 7,461 free cells and 27,926 possible transitions.
- Initial and goal positions for the robots are randomly generated.

Multi-robot gathering: `den312d.map`: with 5 robots and 15 tasks

Multi-robot gathering: `den312d.map`: with 5 robots and 15 tasks

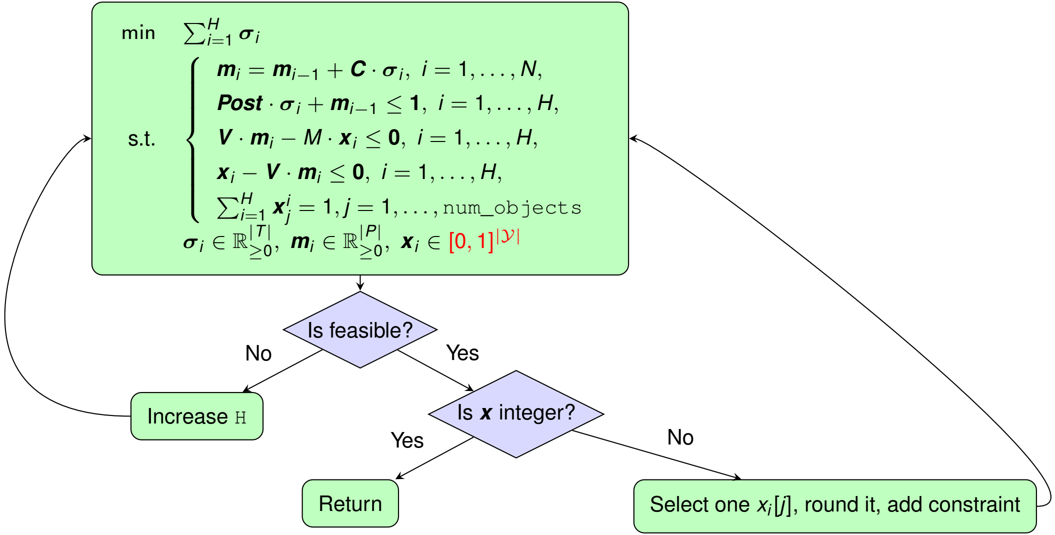


Multi-robot gathering: `den312d.map`: with 5 robots and 15 tasks



Multi-Robot Gathering Problem: Relaxing Algorithm

$$
\begin{aligned}
\min \quad & \sum_{i=1}^{H} \boldsymbol{\sigma}_i \\
\text{s.t.} \quad & \begin{cases}
\boldsymbol{m}_i = \boldsymbol{m}_{i-1} + \boldsymbol{C} \cdot \boldsymbol{\sigma}_i, \; i = 1, \ldots, N, \\
\boldsymbol{Post} \cdot \boldsymbol{\sigma}_i + \boldsymbol{m}_{i-1} \leq \boldsymbol{1}, \; i = 1, \ldots, H, \\
\boldsymbol{V} \cdot \boldsymbol{m}_i - M \cdot \boldsymbol{x}_i \leq \boldsymbol{0}, \; i = 1, \ldots, H, \\
\boldsymbol{x}_i - \boldsymbol{V} \cdot \boldsymbol{m}_i \leq \boldsymbol{0}, \; i = 1, \ldots, H, \\
\sum_{i=1}^{H} \boldsymbol{x}_j^i = 1, j = 1, \ldots, \texttt{num\_objects}
\end{cases} \\
& \boldsymbol{\sigma}_i \in \mathbb{R}_{\geq 0}^{|T|}, \; \boldsymbol{m}_i \in \mathbb{R}_{\geq 0}^{|P|}, \; \boldsymbol{x}_i \in [0, 1]^{|\mathcal{Y}|}
\end{aligned}
$$



Multi-robot gathering: `den312d.map`

# Robots	# Tasks	Integer var.	MILP [s]	**# iterations**	**LPs [s]**
5	5	5	0.5235	**1**	**0.3019**
5	10	20	6.8666	**6**	**9.6056**
5	15	45	109.0247	**12**	**47.3758**
5	20	80	2087.0085	**25**	**170.2436**
5	25	125	-	**42**	**612.0738**
50	**200**	**800**	**-**	**326**	**3304.3094**
50	**250**	**1250**	**-**	**558**	**12745.2440**

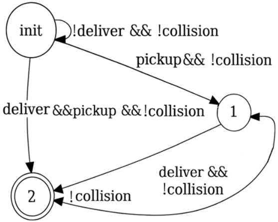
Table 2: Timing for different number of tasks.

Path Planning with Temporal Logic Specifications

Outline

Temporal Logic Specifications: From Logical Goals to Real-Time Missions

Temporal logic extends Boolean goals with **ordering** and **quantitative timing** constraints.



LTL example (untimed):

$$\mathbf{G}\neg\text{collision} \ \wedge \ \mathbf{F}\,\text{deliver} \ \wedge \ (\neg\text{deliver}\ \mathbf{U}\ \text{pickup})$$

"Always avoid collisions, eventually deliver, and deliver only after pickup."

MITL example (timed, same logic with intervals):

$$\mathbf{G}_{[0,\infty)}\,\neg\text{collision} \ \wedge \ \mathbf{F}_{[0,10]}\,\text{deliver} \ \wedge \ \left(\neg\text{deliver}\ \mathbf{U}_{[0,\infty)}\ \text{pickup}\right)$$

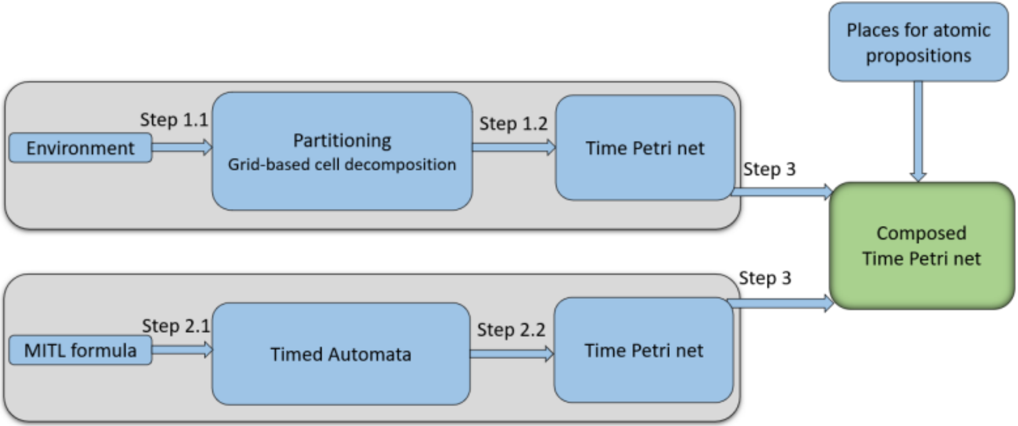
"Always avoid collision; delivery must occur within 10 seconds; and as in LTL, delivery is forbidden before pickup."

Idea

MITL preserves the structure of LTL while adding explicit metric timing constraints to temporal operators, enabling real-time mission specification.

Outline

Overview of the Methodology



S. Hustiu, D.V. Dimarogonas, C. Mahulea and M. Kloetzer, "Multi-robot Motion Planning under MITL Specifications based on Time Petri Nets," in *2023 European Control Conference (ECC)*, Bucharest, Romania, June 2023.

From MITL to Real-Time Models: TBA and Time Petri Nets

MITL Semantics Through Automata

- MITL formulas can be translated into a **Timed Büchi Automaton (TBA)** that captures their exact real-time semantics.
- The construction preserves the metric constraints of the temporal operators.

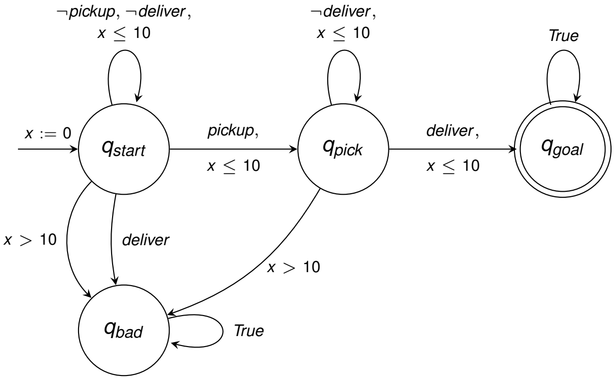
From TBA to Time Petri Nets

- Using the classical translation of Bérard et al., a TBA can be converted into a **Time Petri Net (TPN)**.
- The translation is **timed bisimulation preserving**: the TPN and the TBA have equivalent real-time behaviors.
- TPNs provide:
  - compact real-time modeling,
  - explicit transition intervals $[\alpha(t), \beta(t)]$,
  - efficient partial-state exploration (ROMEO, etc.).

B. Bérard, F. Cassez, S. Haddad, D. Lime, and O. H. Roux, "Comparison of the expressiveness of timed automata and time Petri nets," in *Int. Conf. on formal modeling and analysis of timed systems*. Springer, 2005, pp. 211–225.

Timed Büchi Automaton for the MITL Formula

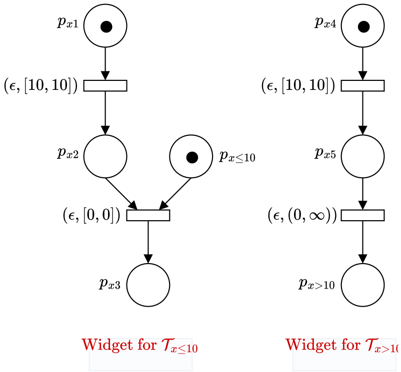
~~$\mathbf{G}_{[0,\infty)}\ \neg\text{collision}$~~ $\wedge$ $\mathbf{F}_{[0,10]}\ \text{deliver}\ \wedge\ \left(\neg\text{deliver}\ \mathbf{U}_{[0,\infty)}\ \text{pickup}\right)$, i.e., collision avoidance is guaranteed by the path-planning module.



R. Alur, T. Feder, and T. A. Henzinger, "The benefits of relaxing punctuality," *Journal of the ACM (JACM)*, vol. 43(1):116-146, 1996.

Time Petri Net of the Specification

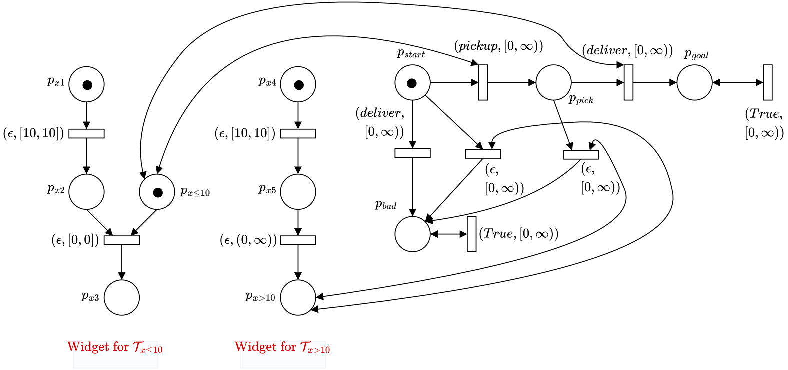
Step 1: Modeling atomic constraints on *x*



B. Bérard, F. Cassez, S. Haddad, D. Lime, and O. H. Roux, "Comparison of the expressiveness of timed automata and time Petri nets," in *Int. Conf. on formal modeling and analysis of timed systems*. Springer, 2005, pp. 211–225.

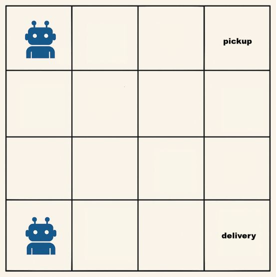
Time Petri Net of the Specification

Step 2: Model of the specification

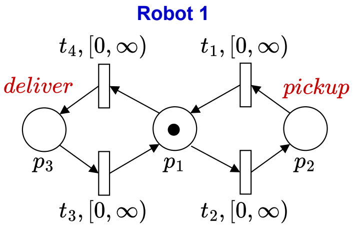


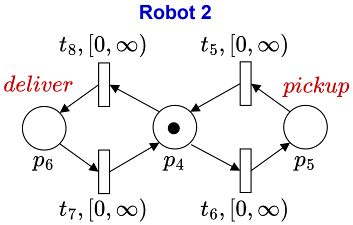
B. Bérard, F. Cassez, S. Haddad, D. Lime, and O. H. Roux, "Comparison of the expressiveness of timed automata and time Petri nets," in *Int. Conf. on formal modeling and analysis of timed systems*. Springer, 2005, pp. 211–225.

Reduced Petri Net

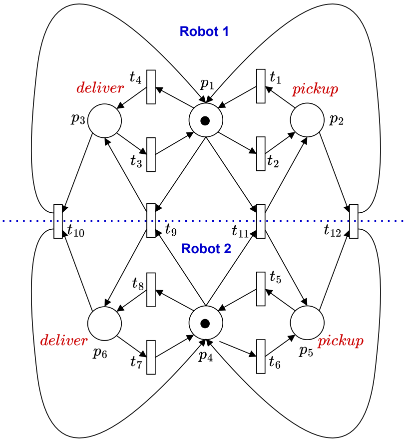


- A marked place actives all its outputs (i.e., all the atomic propositions that labeled the marked place are evaluated to true)





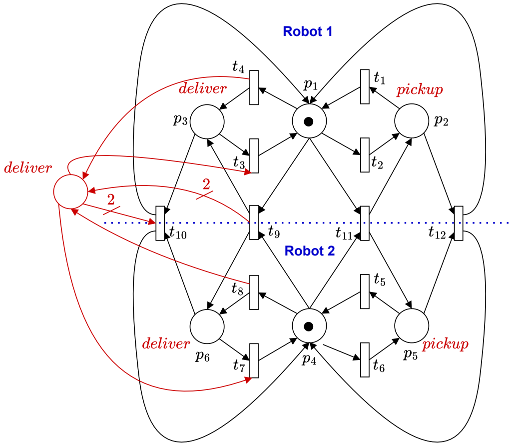
Reduced Time Petri Net of the Team



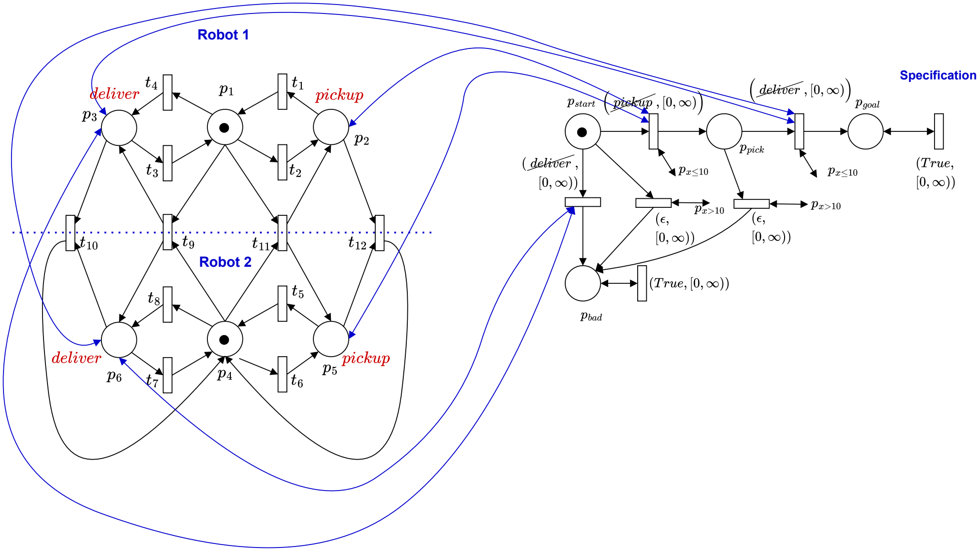
- In general, we define a common *delivery* place ($p_3 + p_6$) and a common *pickup* place ($p_2 + p_5$).
- We also consider their complementary places (representing the negated propositions).
- For simplicity, we assume that *deliver* and *pickup* cannot occur at the same time instant (although they may occur simultaneously for both robots).
- The outputs of the Team Model (i.e., changes in the marking) serve as inputs to the Specification Net.

Time Petri Net of the Team and Logic Places

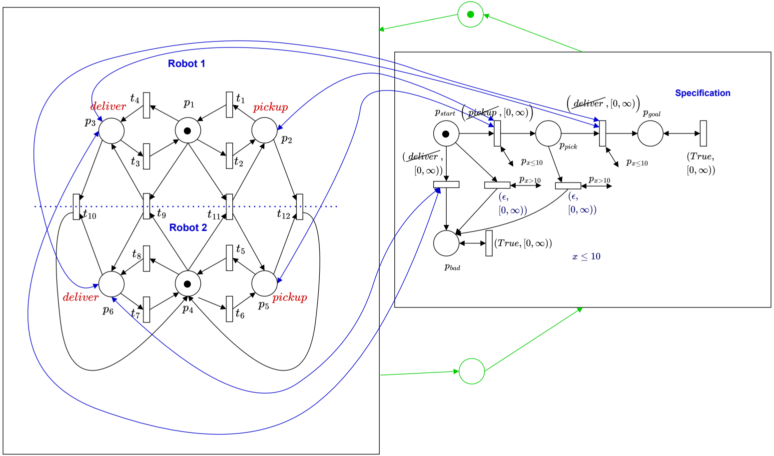
Example of the *deliver* place.



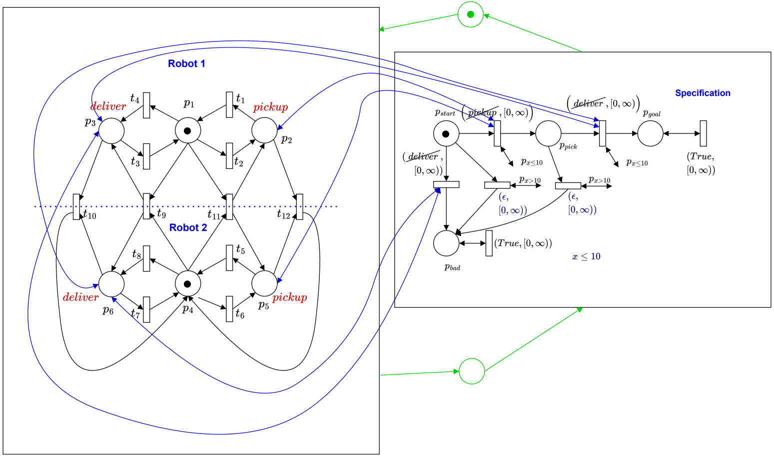
Time Petri Net of the Composed System



Time Petri Net for Timed Formal Verification



Time Petri Net for Timed Formal Verification



Is the marking $m[p_{goal}] \geq 1$ reachable within 10 seconds? ROMEO provides the answer.

Conclusions

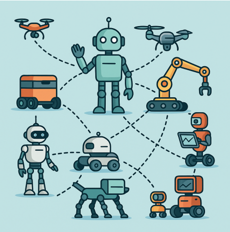
Conclusions - From Structure to Scalability

- **Petri nets** provide a compact and compositional framework to model coordination and concurrency.
- **Optimization replaces search:** the reachability problem becomes a structured MILP or LP formulation.
- **Scalability through structure:** the same model extends from a few to thousands of robots.

"When structure replaces simulation, complexity becomes manageable."

Beyond Thousands of Robots - Future Directions

- **Dynamic and uncertain environments:** planning under evolving maps and disturbances.
- **Distributed Petri-net control:** scalable decision-making with local autonomy.
- **Learning meets structure:** combining data-driven estimation with logic-based control.



Acknowledgments and Closing Remarks

- Research supported by: **PID2024-159284NB-I00**, **ONR N62909-24-1-2081**, and **TED2021-130449B-I00**.
- Work developed at the **COSMOS group - I3A, University of Zaragoza**.
- Special thanks to the Laboratoire d'Informatique et des Systèmes (LIS), Aix-Marseille Université (AMU), for inviting me as a visiting professor.
- Collaborations with partners from Romania, France, Italy, Sweden, and China.

"From modeling concurrency to orchestrating cooperation - the journey of Petri nets continues." (Petri, 1962; Silva & Recalde, 2015)

Slides, References & Feedback

Thank you for your attention!

- Online form for feedback and comments.
- Access to **references** and related material.
- Link to **slides**.

MSR'25 Slides & Feedback

Scan the QR code to access everything.