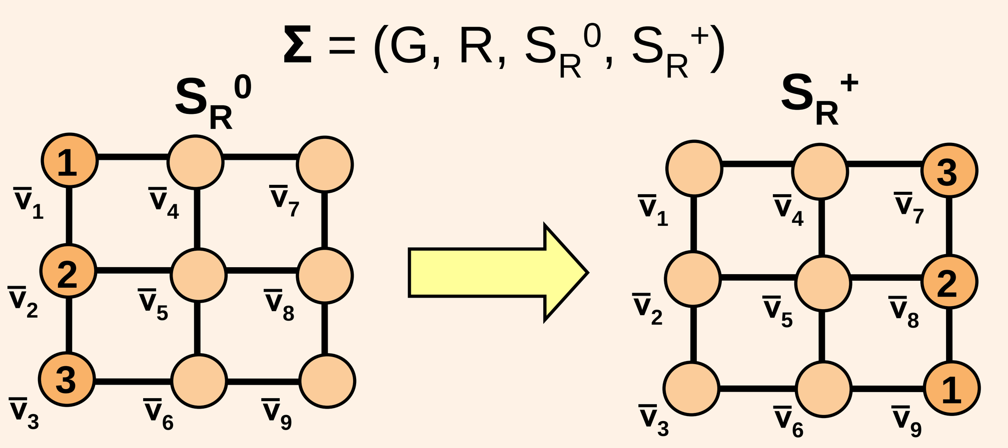
An Optimization Variant of Multi-robot Path Planning is Intractable

Pavel Surynek
Department of Theoretical Computer Science and Mathematical Logic
Faculty of Mathematics and Physics, Charles University in Prague, Czech Republic

Problem of Multi-robot Path Planning

- **Input:** $\Sigma = (G, R, S_R^0, S_R^+)$
  - an undirected graph **G** = (V,E)
  - a set of robots **R** = $\{\bar{r}_1, \bar{r}_2, \ldots, \bar{r}_\nu\}$, where $|V| > \nu$
  - a uniquely invertible function S_R^0: R → V determining an **initial arrangement** of robots in vertices of G
  - another uniquely invertible function S_R^+: R → V determining a **goal arrangement** of robots
- **Dynamicity:**
  - a move into a **currently** unoccupied vertex is ***allowed***
  - a move into a vertex **currently** vacated by an allowed move is ***allowed***
- **Output:** $[S_R^0, S_R^1, S_R^2, \ldots, S_R^\zeta = S_R^+]$
  - S_R^i: R → V is a uniquely invertible function $\forall i \in \{0,1,\ldots,\zeta\}$
  - S_R^{i+1} is obtained from S_R^i by allowed moves $\forall i \in \{0,1,\ldots,\zeta-1\}$
  - ζ is the **makespan** of the solution

Example of Multi-robot Path Planning



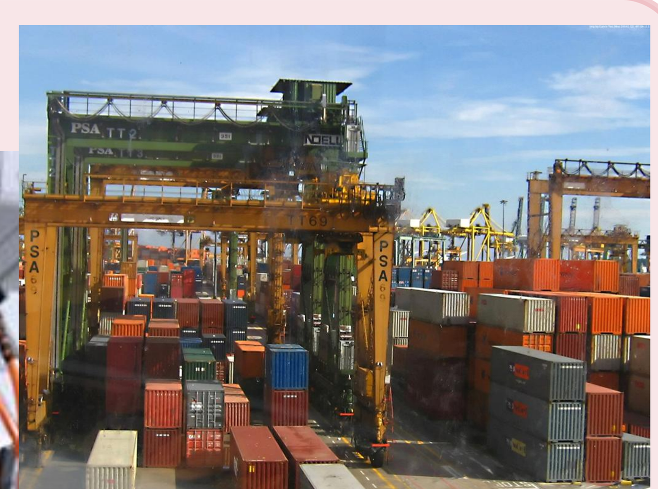
Solution of an instance of *path planning for multiple robots* **Σ** with R = {1,2,3}

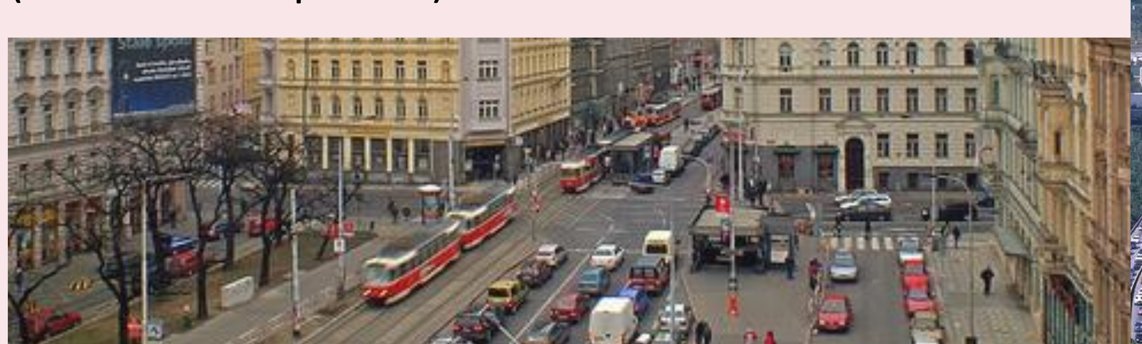
ζ = 4 R	S_R^0	S_R^1	S_R^2	S_R^3	$S_R^4 = S_R^+$
1	$\bar{v}_1$	$\bar{v}_4$	$\bar{v}_7$	$\bar{v}_8$	$\bar{v}_9$
2	$\bar{v}_2$	$\bar{v}_1$	$\bar{v}_4$	$\bar{v}_7$	$\bar{v}_8$
3	$\bar{v}_3$	$\bar{v}_2$	$\bar{v}_1$	$\bar{v}_4$	$\bar{v}_7$

- a **solution** of the **makespan ζ = 4** is shown
- **columns** represent arrangements of robots in vertices at individual time steps
- **rows** represent sequences of moves of individual robots

Motivation for the Problem

- **rearranging** containers (robot = container)
- heavy traffic **control** (robot = car)
- **data transfer** planning (robot = data packet)

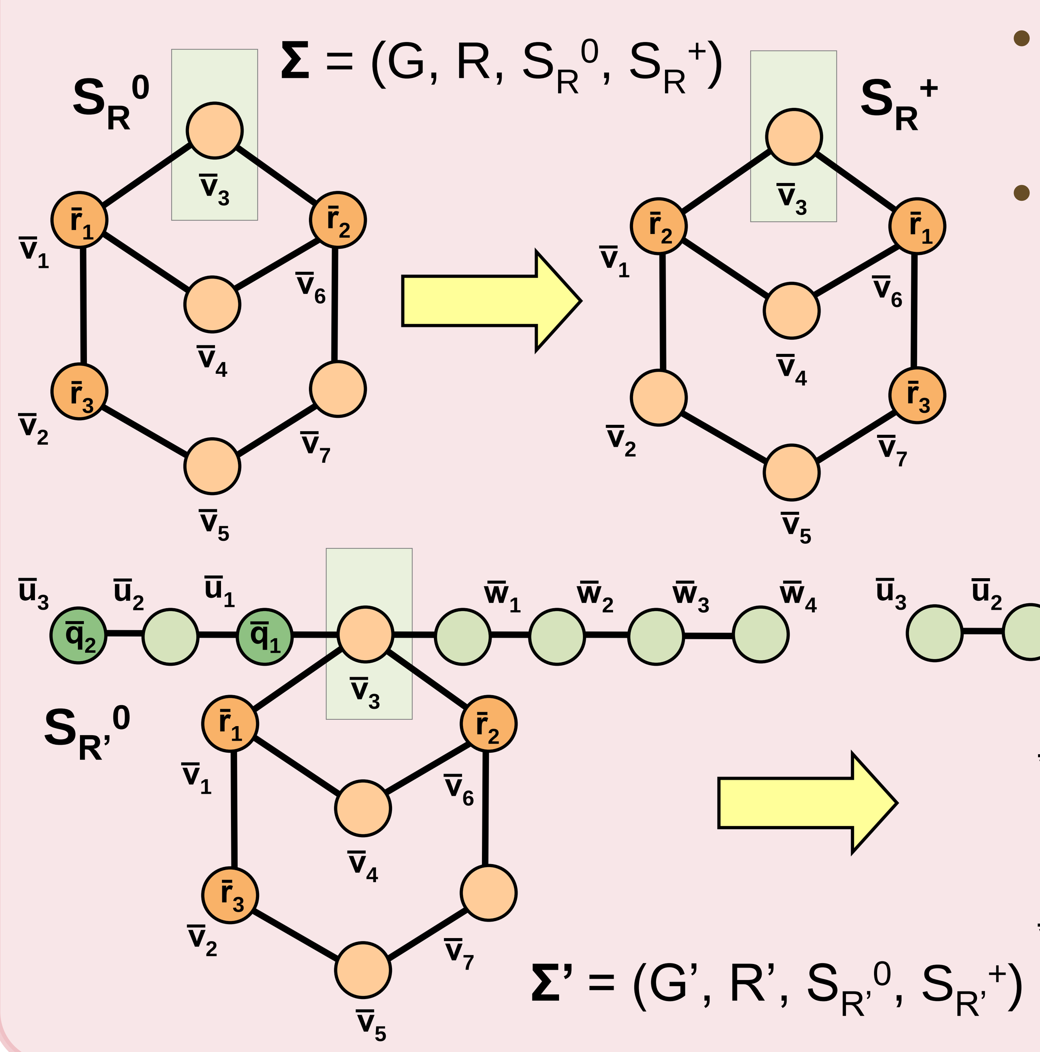




Optimization Variant of the Problem

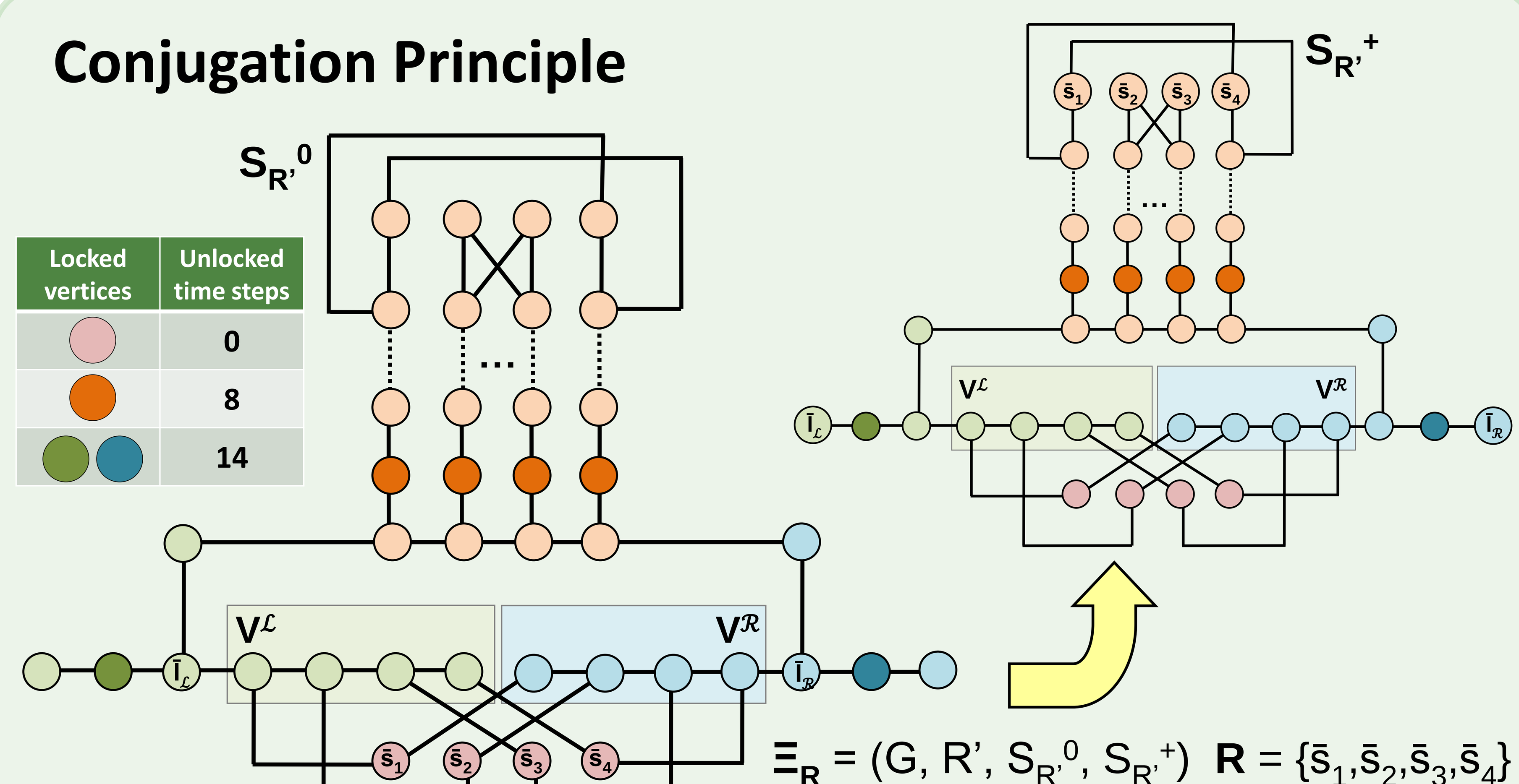
- The **makespan** of the solution must be as **small** as possible.
- **Decision** version … $MRPP_{OPT}$
  - **Input:** **Σ** = (G=(V,E), R, S_R^0, S_R^+), **η**
  - Is there a **solution** to **Σ** of the makespan at most η?
- Decision version of the optimization variant is **NP-complete**.
  - $MRPP_{OPT}$ ∈ **NP**: a solution of the makespan $O(|V|^3)$ can be generated - **polynomial** upper bound of the size of a solution to guess in non-deterministic model (with oracle)
  - $MRPP_{OPT}$ is **NP-hard**: $SAT_=$ polynomial-time reduced to $MRPP_{OPT}$
    - the same number of positive and negative occurrences of each variable

Vertex Locking Mechanism



- vertex $\bar{v}_3$ will be **locked** for time steps **1** and **3**
- original robots $\bar{r}_1$, $\bar{r}_2$, and $\bar{r}_3$ cannot enter $\bar{v}_3$ at time steps 1 and 3 in any **optimal solution**

Conjugation Principle

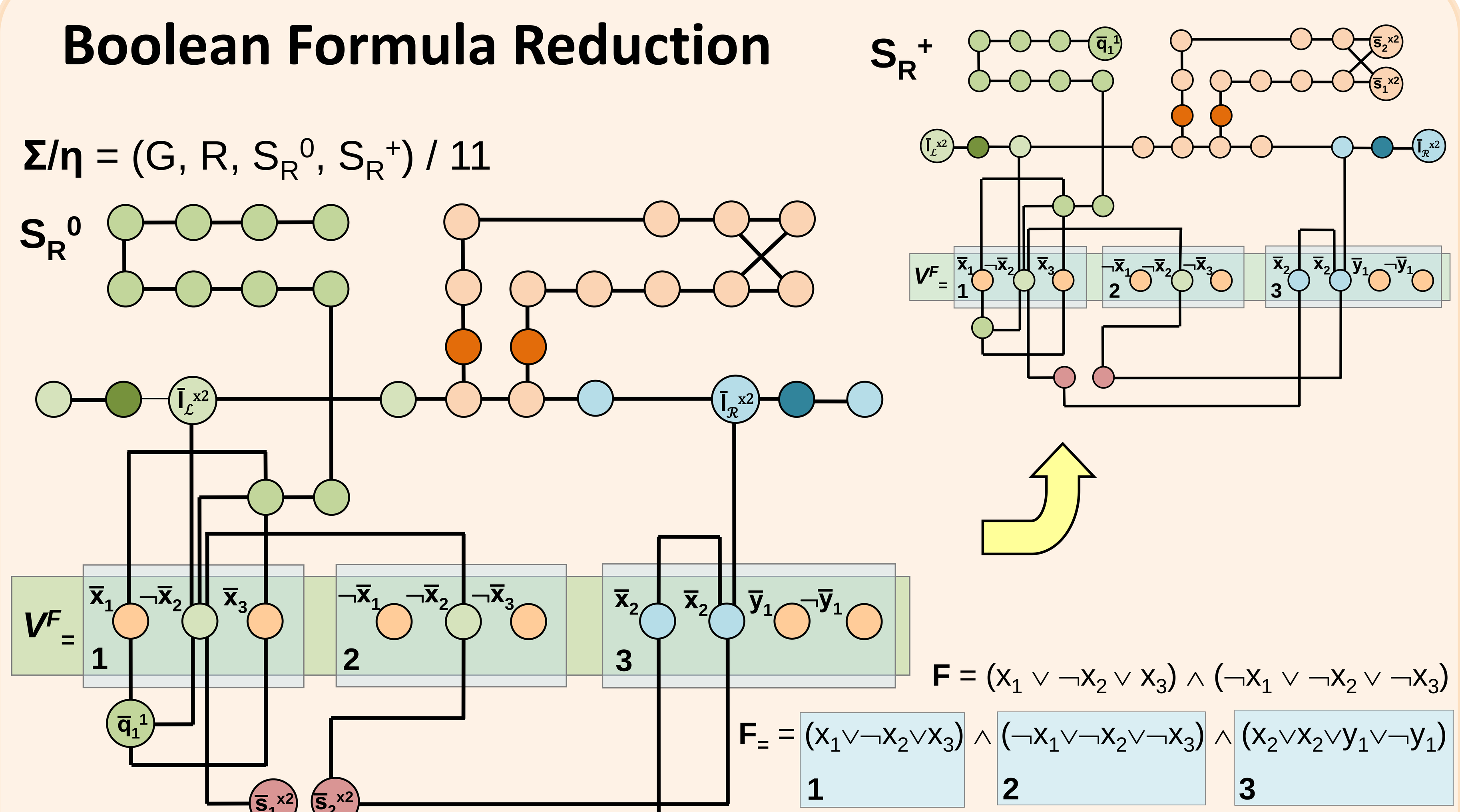


$\Xi_R = (G, R', S_{R'}^0, S_{R'}^+)$ **R** = $\{\bar{s}_1, \bar{s}_2, \bar{s}_3, \bar{s}_4\}$
R' = $\{\bar{s}_1, \bar{s}_2, \bar{s}_3, \bar{s}_4\} \cup \{\bar{l}_\mathcal{L}, \bar{l}_\mathcal{R}\}$

- either $V^\mathcal{L}$ or $V^\mathcal{R}$ is visited by robots $\bar{s}_1$, $\bar{s}_2$, $\bar{s}_3$, and $\bar{s}_4$ at time step **1** in any **optimal solution**
- **positive** literals mapped to $V^\mathcal{L}$; **negative** literals mapped to $V^\mathcal{R}$
- conjugation principle is used to simulate **Boolean consistency** over different occurrences of the same variable

Boolean Formula Reduction

$\Sigma/\eta = (G, R, S_R^0, S_R^+) / 11$



$F = (x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_3)$

$F_= = (x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_3) \wedge (x_2 \vee x_2 \vee y_1 \vee \neg y_1)$
1 2 3

- each variable of $F_=$ is associated with a **conjugation instance**
  - vertices corresponding to literals form a set $V^F_=$
  - literals that correspond to vertices through which a robot goes at **time step 1** are set to FALSE (complementary are TRUE)
- vertex locking mechanism is used to enforce **clause satisfaction**
  - at least one vertex of clause vertices is locked at time step 1

AAAI 2010 | The Twenty-Fourth AAAI Conference on Artificial Intelligence | July 2010 | Atlanta, GA, USA