Path-planning for Multiple Robots

RNDr. **Pavel Surynek**, Ph.D.

Department of Theoretical Computer Science and Mathematical Logic

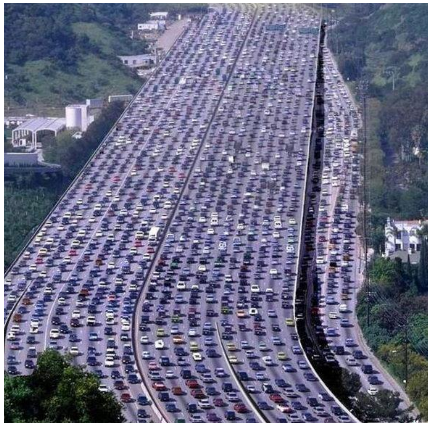
Faculty of Mathematics and Physics

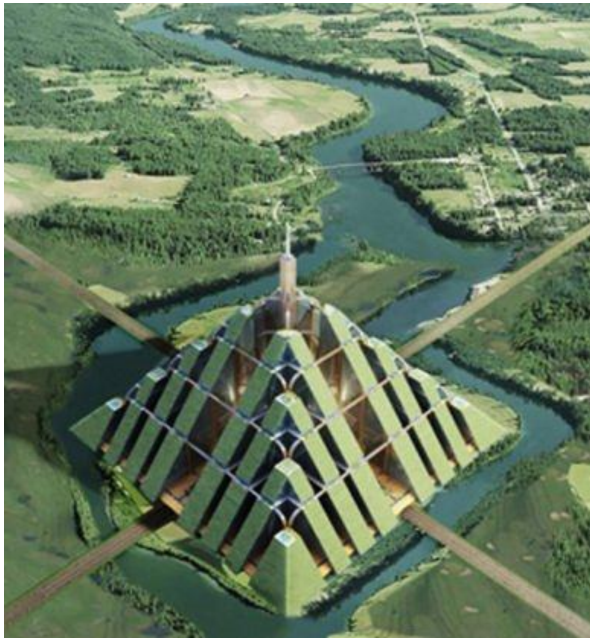
Charles University in Prague

http://ktiml.mff.cuni.cz/~surynek

Motivation (1)

- Container rearrangement
  (robot = **container**)
- Heavy traffic
  (robot = **automobile** (in a jam))
- Data transfer
  (robot = **data packet**)
- Generalized lifts
  (robot = **lift**)



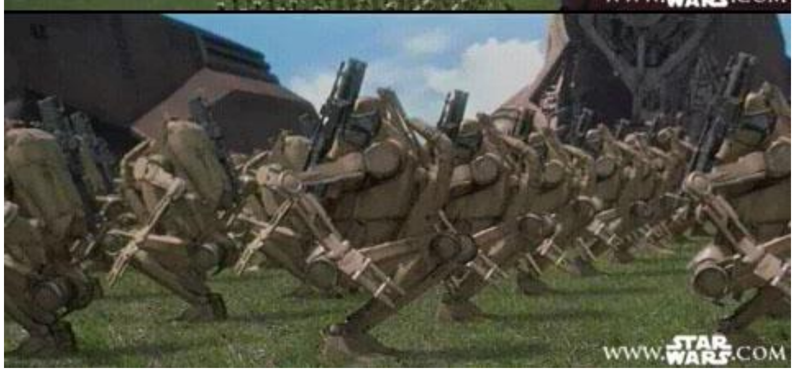


Motivation (2)

STAR WARS

(Lucasfilm Ltd., 1999-2009)

- Computer generated **imagery** (mass scenes)
- Computer **games** (unit navigation in real-time strategies)



Path planning for multiple robots (1)

Ryan, 2007; Surynek, 2009

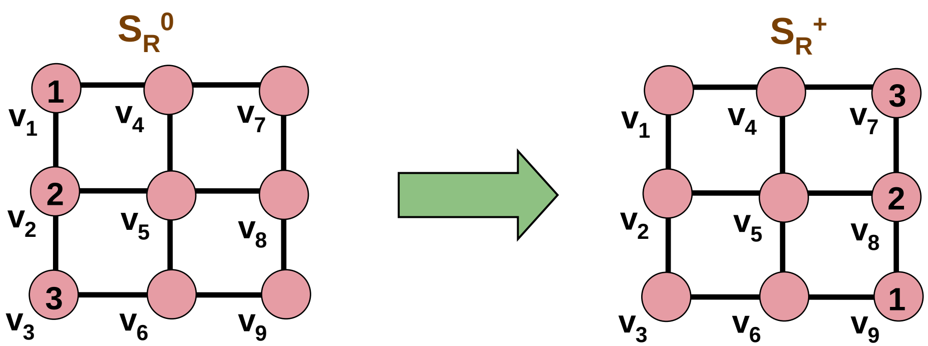
- **Formal description** of an instance of the problem of path planning for multiple robots:
  - The environment is modeled as an undirected graph, where **vertices represent locations** and **edges represent possibility of traversal** between locations.
  - The instance is a quadruple $\Sigma = (G, R, S_R^0, S_R^+)$, where:
    - $G=(V,E)$ is an undirected **graph**,
    - $R = \{r_1,r_2,\ldots,r_\nu\}$, where $\nu<|V|$ is a **set of robots**,
    - S_R^0: $R \rightarrow V$ is a uniquely invertible function representing **initial arrangement of robots** in vertices of the graph, and
    - S_R^+: $R \rightarrow V$ is a uniquely prostá funkce representing the **goal arrangement of robots** in vertices of the graph.
- The **time** is discrete. Time steps and their ordering is isomorphic to the structure of **natural numbers**.

Path planning for multiple robots(2)

Ryan, 2007; Surynek, 2009

- The **dynamicity** of the task is as follows:
  - A robot occupying a vertex at the time step **t** can move into the neighboring vertex (the robot will occupy the target vertex at time step **t+1**) if this movement is **allowed** and **no other** robot is trying to enter the same target vertex.
  - A movement commenced at the time step **t** and finished at the time step **t+1** is **allowed,** if and only if:
    - the target vertex of the movement is **unoccupied** at the time step **t**, **or**
    - the target vertex is **being left** by another robot at time step **t** by the **allowed movement**.
- Given $\Sigma = (G, R, S_R^0, S_R^+)$, the task is to find:
  - A **sequence of movements** scheduled over time for each robot such that each robot **reach its goal** vertex and the condition on **dynamicity is always preserved**.

An **example instance** and **remarks**



A solution of an instance of multi-robot path planning, where R={1,2,3}

solution length=5
$\mathbf{O_1}$=[$\mathbf{v_1}$, v_4, v_7, v_8, $\mathbf{v_9}$]
$\mathbf{O_2}$=[$\mathbf{v_2}$, v_1, v_4, v_7, $\mathbf{v_8}$]
$\mathbf{O_3}$=[$\mathbf{v_3}$, v_2, v_1, v_4, $\mathbf{v_7}$]

Time step: **1 2 3 4 5**

- **Properties** implied by the dynamicity condition:
  - forbids **collisions** among robots
  - allows **high parallelism**
- Compare with **pebble motion on a graph**:
  - The movement is allowed into the **currently unoccupied** vertex only.
  - Lloyd's 9, 15, (n^2-1)-puzzle
    - The **parallelism** is substantially **lower**.

Known results (1)

- Originally obtained for **pebble motion on a graph**
- The case with **bi-connected graph** is especially interesting – the instance is **almost always solvable** – we will restrict on this case:
  - Suppose there is a **single unoccupied vertex** (the most difficult case)
  - Rearrangement of robots in vertices of the graph can be regarded as a **permutation** (the unoccupied vertex is fixed within $\mathbf{S_R^0}$ a v $\mathbf{S_R^+}$).
  - Permutation can be either **even** or **odd** (can be expressed using the even or odd number of transpositions of pair of robots respectively).
- Wilson, 1974:
  - If a bi-connected graph (not isomorphic to a cycle) with a single unoccupied vertex **contains cycle of the odd length**, then **every instance** of path planning for multiple robots over this graph is **solvable**.
  - If a bi-connected graph (not isomorphic to a cycle) with a single unoccupied vertex **does not contain cycle of the odd length**, then an instance of path planning for multiple robots over this graph is solvable, if and only if $\mathbf{S_R^+}$ represents an **even permutation** with respect to $\mathbf{S_R^0}$.
  - Solution of the length of $\mathbf{O(|V|^5)}$ can be generated in the worst case time of $\mathbf{O(|V|^5)}$.

Known results (2)

- Kornhauser, Miller, Spirakis, 1984 (**MIT** algorithm):
  - Again designed for **bi-connected** graphs with **single unoccupied** vertex.
  - Solution of the length of **$O(|V|^3)$** can be generated in the worst case time of **$O(|V|^3)$**.
  - There exist instances, where the length of shortest possible solution is $\Omega(|V|^3)$.
- Ratner, Warmuth, 1986:
  - The decision variant of the problem of pebble motion on a graph when the shortest possible solution required, is an **NP-complete** problem.
    - movements are into unoccupied vertices only – **low parallelism**
  - Shown for the generalized Lloyd's 15 on the board of the size of N x N
- New results for **multi-robot path planning** on bi-connected graphs:
  - Generating solution of the length of $O(|V|^3)$ can be done in the worst case time of $O(|V|^3)$ with **lower constants** in the asymptotic estimation than MIT.
    - at least **two unoccupied vertices** are required
    - **single unoccupied**: solution length $O(|V|^4)$ in the time of $O(|V|^4)$, however practically still **better than MIT**
  - The decision variant of the problem of multi-robot path planning, when a shortest possible solution is required, is an **NP-complete** problem.
    - movements can be done into currently unoccupied vertices – **higher parallelism**

The **MIT** Algorithm – main idea (1)

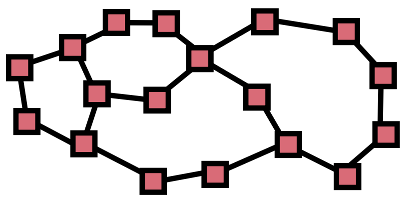
Kornhauser, Miller, Spirakis, 1984

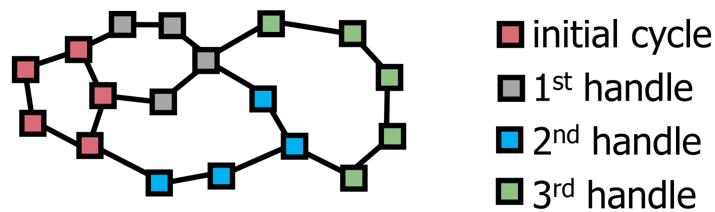
- The set of possible rearrangements of pebbles over the graph forms a permutation group $\mathcal{G}$ with elements $P=\{p_1,p_2,\ldots,p_\mu\}$
  - **Definition:** Permutation group $\mathcal{G}$ for elements $P=\{p_1,p_2,\ldots,p_\mu\}$ is **k-transitive** for $k\leq\mu$, if for every pair of k-tuples of elements $a_1,a_2,\ldots,a_k$ and $b_1,b_2,\ldots,b_k$, where $a_i\in P$, $b_i\in P$ for $i=1,2,\ldots,k$ there exists a permutation $\pi\in\mathcal{G}$, such that $\pi(a_i)=b_i$ for $i=1,2,\ldots,k$.
- **Proposition:** If a permutation group $\mathcal{G}$ contains a **cycle of the length k** and it is **k-transitive**, then it **contains all the cycles of the length k.**
- **Proposition:** Every even permutation of elements $P=\{p_1,p_2,\ldots,p_\mu\}$ can be obtained as a composition of at most μ-2 **cycles of the length 3** (3-cycle).
- We will show a **sketch** of the proof that a bi-connected graph not isomorphic to a cycle induces 3-cycle and it is 3-transitive:
  - This is sufficient to create **every even permutation**.

The **MIT** Algorithm – main idea (2)

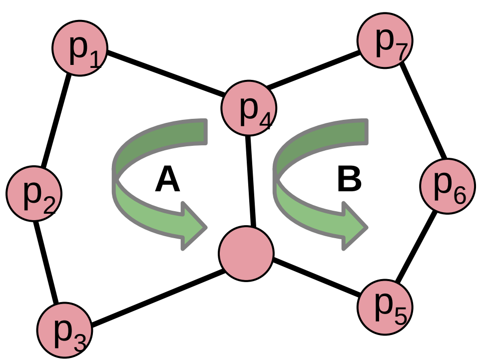
Kornhauser, Miller, Spirakis, 1984

- **Definition:** Graph G=(V,E) is bi-connected, if $|V| \geq 3$ and $\forall v \in V$ $G=(V-\{v\},E')$, where $E'=\{\{x,y\} \in E \mid x,y \neq v\}$, is **connected**.
- **Proposition:** Every bi-connected graph can be constructed from a cycle by adding **handles**.





Cycle + 1 handle



- Illustration of a **3-cycle:** $ABA^{-1}B^{-1}=(p_4,p_7,p_3)$
- Illustration of **3-transitivity:**
  - Every 3 pebbles (x,y,z) can be arranged into a handle of the length at least 3 … permutation P
  - Similarly for pebbles (u,v,w)…permutation Q
  - PQ^{-1} gives 3-transitivity (x,y,z) on (u,v,w)

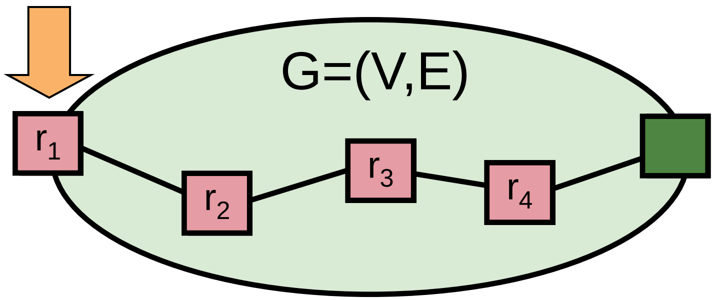
The **MIT** Algorithm – remarks

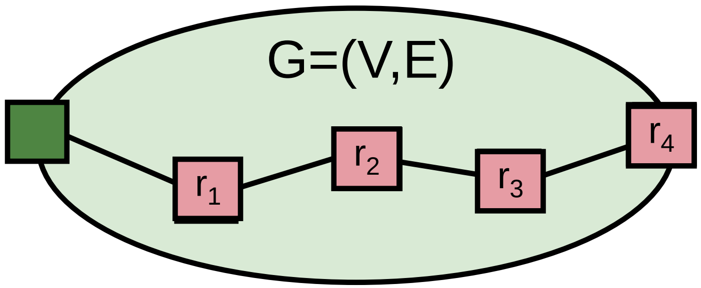
- The **simplest case** has been illustrated only:
  - Case analysis for various situations (length of handles, special sub-graphs)
  - Main idea is the same as in the illustrated example.
- The **length** of the solution and worst case **time complexity**:
  - Generating 3-cycle
    - the constant number of rotations of handles
    - a rotation of the handle consumes $O(|V|)$ movements, can be determined in the worst case time of $O(|V|)$
    - $O(|V|)$ movements and time is required in total
  - Generating 3-transitivity
    - 3 relocations of pebbles to a handle, 3 rotations of the handle
    - relocation of a pebble along an edge consumes $O(|V|)$ movements
    - a pebble move along at most $|V|$ edges
    - $O(|V|^2)$ movements and time is required in total
  - We need to compose μ-2 3-cycles in total, where each 3-cycle consumes $O(|V|^2)$ movements and time. $O(|V|^3)$ movements and time is required in total.
- **Drawback:** relatively high constants in asymptotic estimations

The **BIBOX** Algorithm – main idea (1)

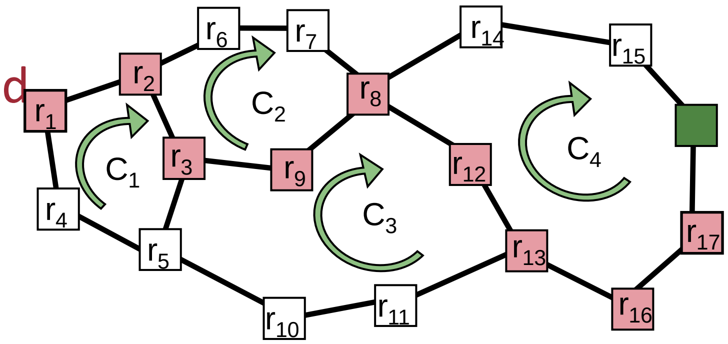
Surynek, 2009

- Suppose that we have a decomposition of a bi-connected graph $G=(V,E)$ into handles $H_1,H_2,\ldots,H_l$ and an initial cycle C_0.
- Further suppose that a sequence of handles is associated with a sequence of cycles $C_1,C_2,\ldots,C_l$ - can be obtained by connecting endpoints of a handle.





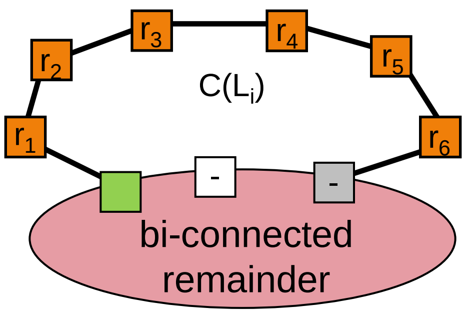
- **Relocation primitives:**
  - **We are able to:** „relocate“ an unoccupied vertex to any location in G.
  - **We are able to:** relocate any robot into any vertex (bi-connectivity is exploited)

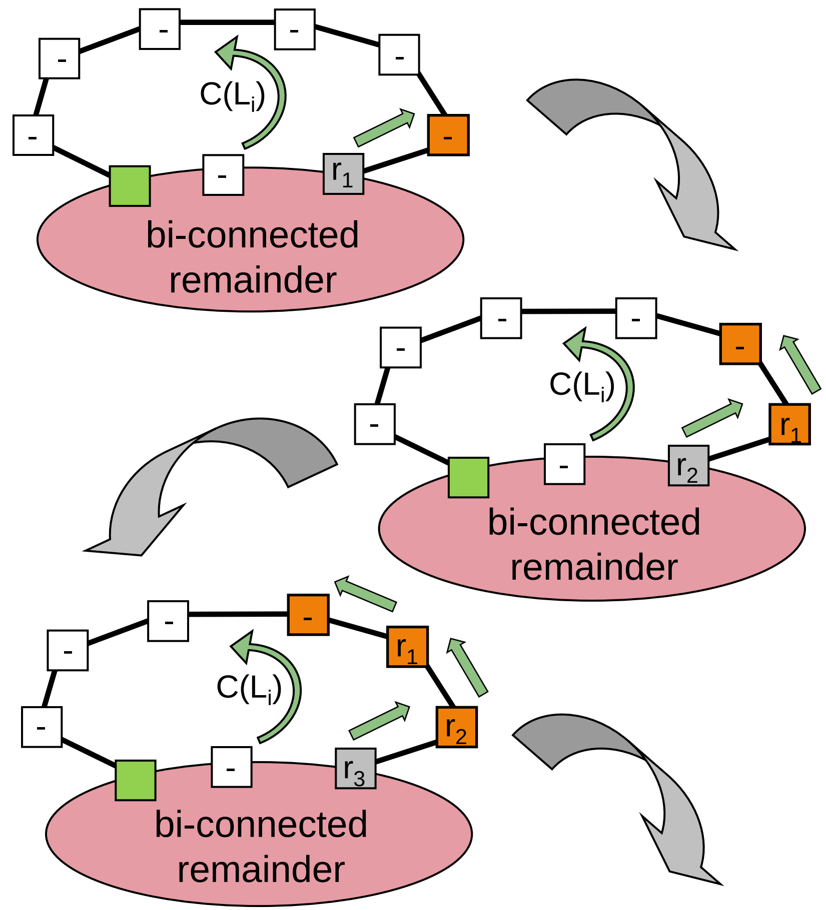


The **BIBOX** Algorithm – main idea(2)

Surynek, 2009

- **More complex relocations** of robots are possible using mentioned relocation primitives:
  - Stacking robots into handle in the right ordering.





- **Stacking robots**:
  - Take the **last handle**.
    - Relocate robot into **the grey** vertex.
    - Perform rotation of the handle (using **the green** unoccupied vertex).
    - ...

...

The **BIBOX** Algorithm – remarks

- **Stacking robots** into handles has been illustrated
  - Does not work for the initial cycle and the first handle
    - **Special approach** must be used
    - The **second unoccupied** vertex is used to exchange pairs of robots
    - **Every permutation** of robots can be achiever using robot exchanges
- The length of the solution and worst case time complexity:
  - **Placing** a single robot into the handle requires:
    - O(|V|) rotations of a handle, where each rotation consumes O(|V|) move
    - relocation of a robot along a path of the length O(|V|), where transition along an edge consumes O(|V|) movements
    - $O(|V|^3)$ movements in total
    - worst case time is $O(|V|^3)$ as well
  - The same asymptotic estimation as in the case of MIT
    - however, constants in the estimations are **better in the case BIBOX**

NP-completeness of multi-robot path planning (1)

Surynek, 2010

- The **decision** version of the **optimization** variant of the problem of path planning for multiple robots:
  - For a given instance $\Sigma = (G=(V,E), R, S_R^0, S_R^+)$ and a number η, we need to answer whether there exists a solution of Σ which **makespan** is at most η.
    - Makespan is typically lower than the total number of movements within the solution ← **parallelism.**
- **Lemma:** The problem is in the **NP** class as we can construct solution of the size of $O(|V|^3)$ (for example by the **BIBOX** algorithm).
  - The size of the oracle/certificate we need to guess in the non-deterministic model is at most $O(|V|^3)$, thus **polynomially bounded**.
  - **Remark**: this is not always known even for similar tasks (such as **Sokoban**)

NP-completeness of multi-robot path planning (2)

Surynek, 2010

- **Lemma:** The decision version of the optimization variant of multi-robot path planning is an **NP-hard** problem.
  - Using a reduction of **Boolean satisfiaility** to **multi-robot path planning**
    - extremely **complicated**
    - **main ideas** will be shown only
- Let *F* be a Boolean formula in CNF
  - **Formula** = conjunction of clauses
  - **Clause** = disjunction of literals
  - **Literal** = Boolean variable or a negation of a Boolean variable
- An instance of multi-robot path planning problem $\Sigma = (G=(V,E), R, S_R^0, S_R^+)$ will be constructed.
- Some **vertices** of the graph G will be associated with literals of the formula *F*.
- The valuation of variables *e* of *F* will be defined as follows:
  - If a vertex corresponding to a literal does not contain a robot at the given time step, then the literal is assigned the value ***TRUE***.
  - Otherwise the literal is assigned the value ***FALSE***.

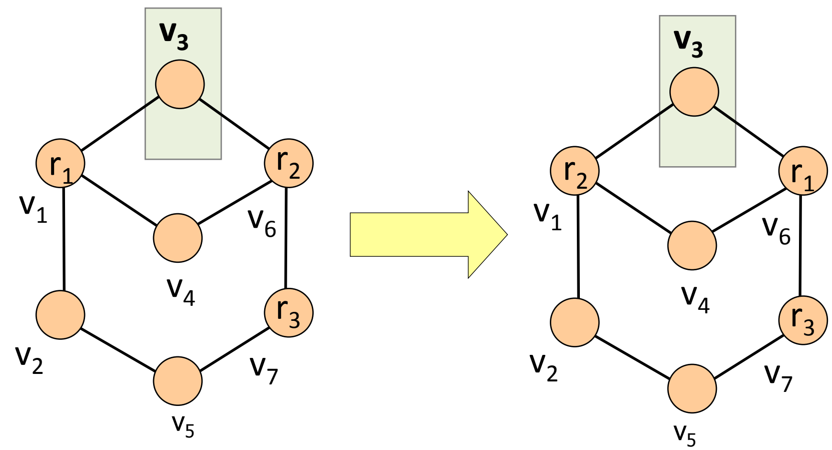
NP-completeness of multi-robot path planning (3)

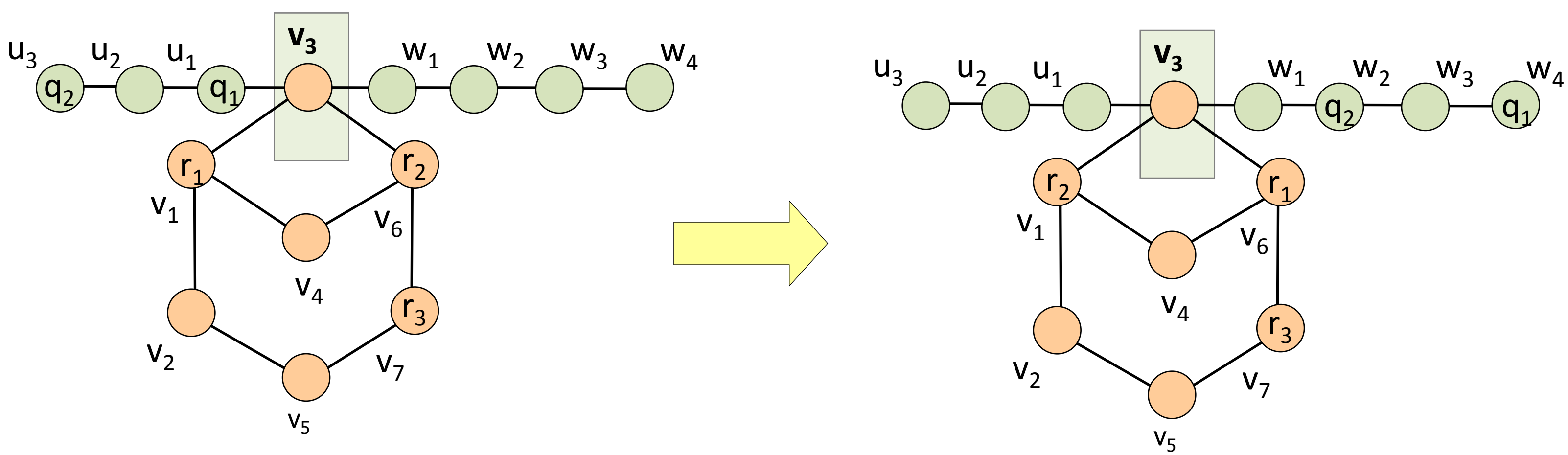
Surynek, 2010

- **Additional vertices** and **edges** in G serve to enforce the following constraints:
  - **Boolean consistency**
    - **literals** corresponding to the same Boolean variable should be **consistently** evaluated using the valuation ***e***
  - **Clause satisfaction**
    - all the clauses should be **satisfied** by the valuation ***e***
- Boolean consistency and clause satisfaction constraints together with some other techniques ensures the following:
  - The formula *F* is satisfiable if and only if there exists a solution of Σ of the makespan at most η.
- The following techniques are used to ensure above constraints:
  - **Vertex locking**
    - a robot cannot enter certain vertex at given time step
  - **Conjugation of robots**
    - a group of robots must be still together

The technique of **vertex locking**

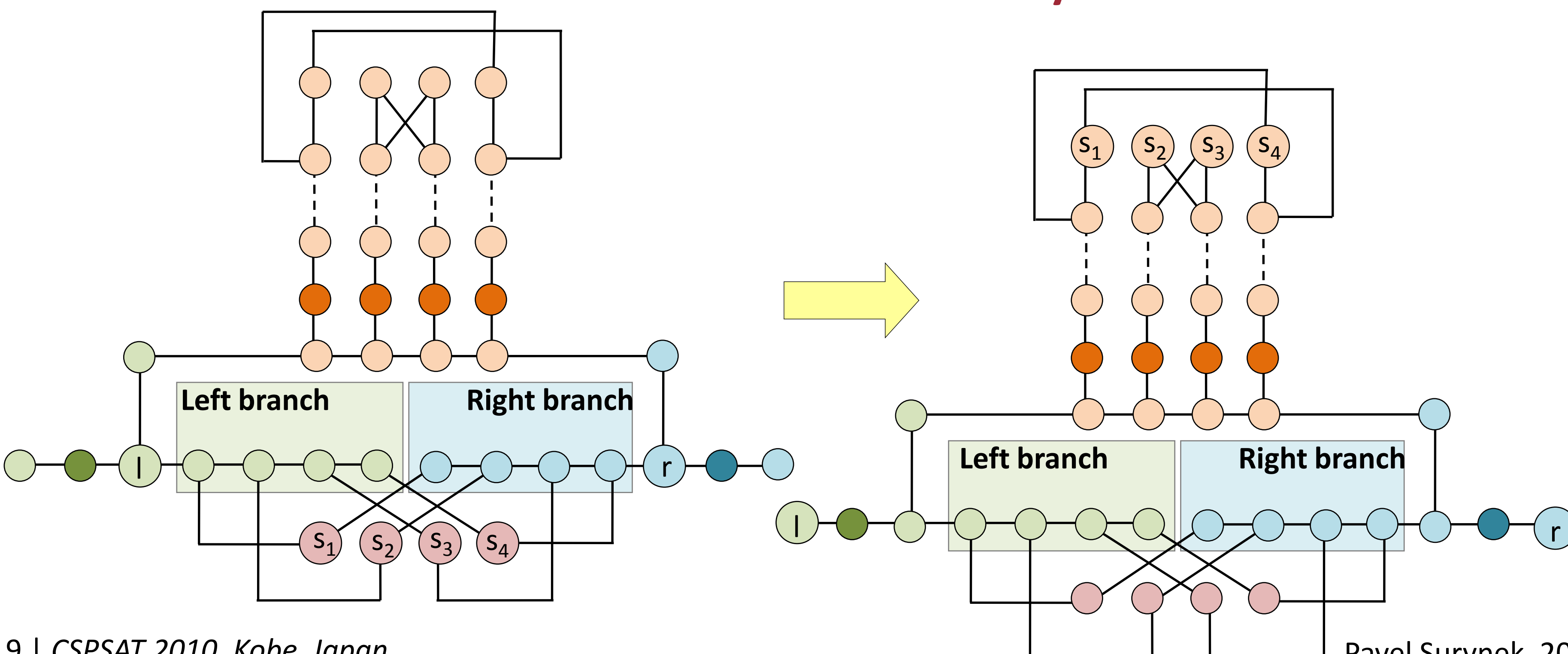
- An instance with the **optimal** makespan of 3
- Vertex v_3 will be **locked** for time steps 2 and 4 in all the optimal solutions
  - can be extended to **multiple locked vertices**
  - it is possible to lock **sets of vertices**





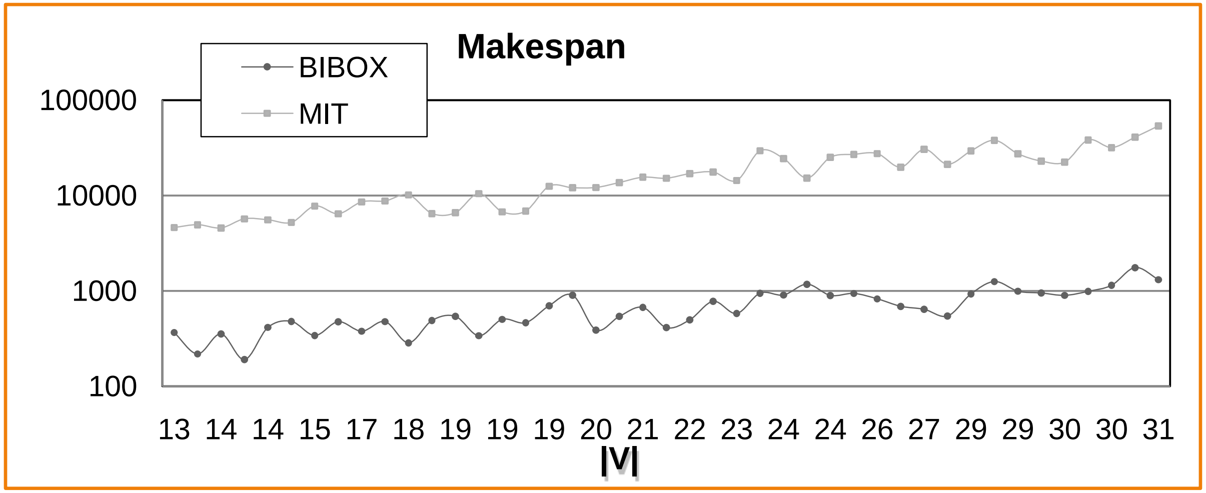
The technique of **conjugation of robots**

- A group of 4 agents must move together in all the optimal solutions – vertex locking is exploited
  - The conjugated group of robots move either through the **left** or through the **right** branch
  - Serves for simulation of **Boolean consistency**



Experimental evaluation (1)

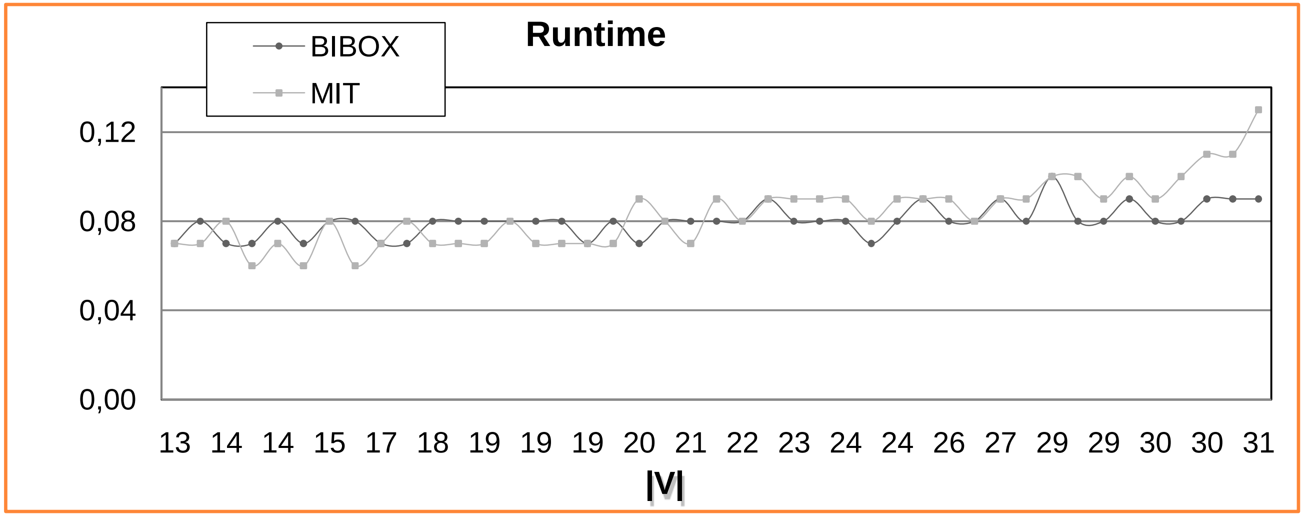
- Comparison of algorithms MIT and BIBOX – **makespan**
  - Tested on random graphs with random arrangements of robots.
    - Graphs containing up to 30 vertices
    - Generated by adding handles of random length



- Algorithm BIBOX produces order of magnitude shorter solutions than MIT.

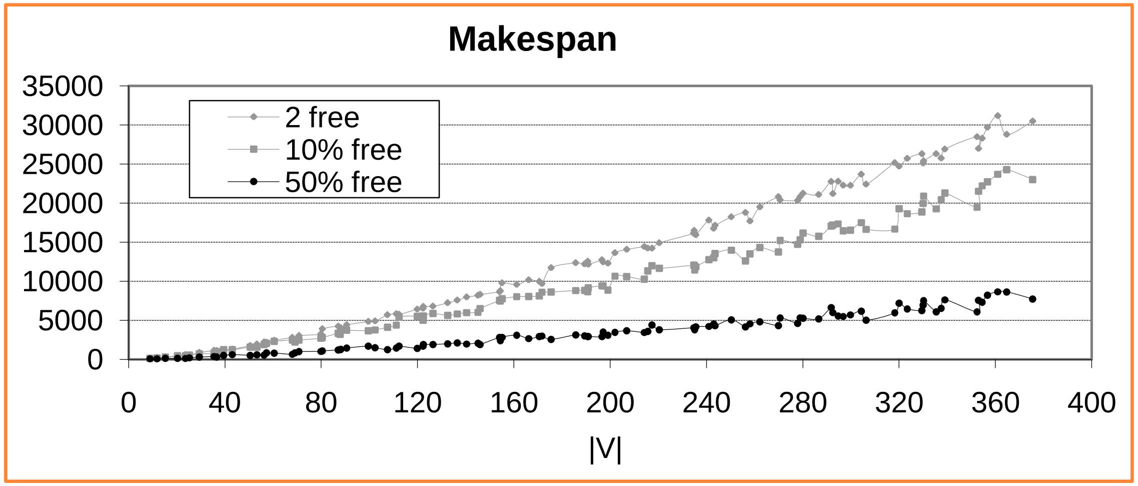
Experimental evaluation (2)

- Comparison of algorithms MIT and BIBOX - **runtime**
  - The same set of testing instances as in the previous test

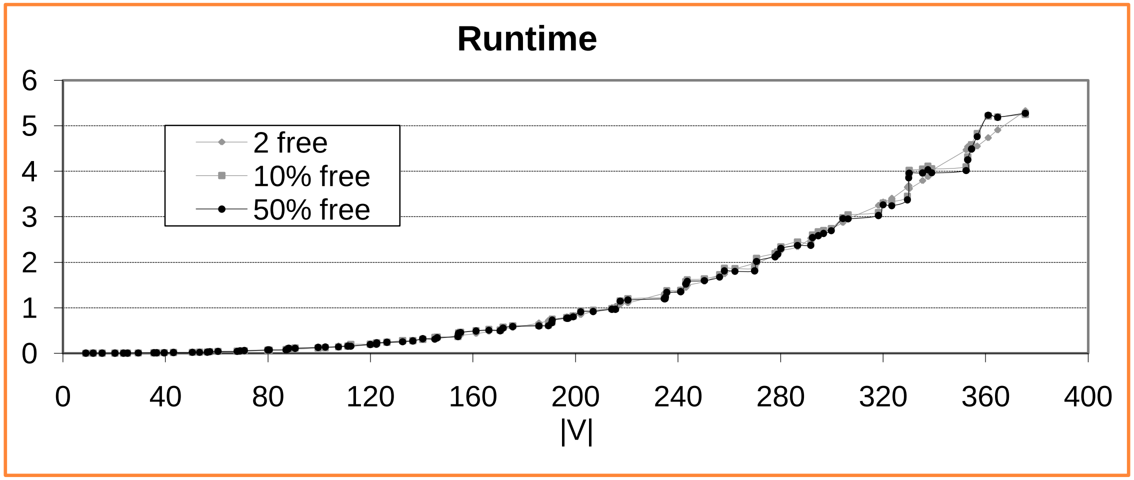


  - Algorithm BIBOX proved to be slightly faster than MIT.

Experimental evaluation (3)



- Tests of the BIBOX algorithm on large instances
  - Random instances with up to 400 vertices
  - Varying number of unoccupied vertices



Conclusions

- The existent algorithm MIT for pebble motion on a graph has been described – exploits 3-transitivity and 3-cycles
  - bi-connected graphs, single unoccupied vertex
- New algorithm BIBOX for multi-robot path planning – uses direct placement of robots into verteces of handles of the handle decomposition
  - bi-connected graphs, two unoccupied vertices (can be augmented to single unoccupied)
  - Experiments shown that BIBOX produces better solutions than MIT
  - Runtime of BIBOX is slightly lower than that of MIT
- NP-completeness of the optimization variant of the problem of multi-robot path planning
  - Reduction of Boolean satisfiability to multi-robot path planning
  - Various techniques for enforcing correspondence between both instances have been shown

Demo

GraphRec Software (by **Petr Koupý**) for visualizing solutions

References

- Wilson, R. M., 1974. *Graph Puzzles, Homotopy, and the Alternating Group.* Journal of Combinatorial Theory, Ser. B 16, pp. 86-96, Elsevier.
- Kornhauser, D., Miller, G. L., Spirakis, P. G., 1984. *Coordinating Pebble Motion on Graphs, the Diameter of Permutation Groups, and Applications.* Proceedings of the 25th Annual Symposium on Foundations of Computer Science (FOCS 1984), pp. 241-250, IEEE Press.
- Ratner, D., Warmuth, M. K., 1986. *Finding a Shortest Solution for the N × N Extension of the 15-PUZZLE Is Intractable.* Proceedings of the 5th National Conference on Artificial Intelligence (AAAI 1986), pp. 168-172, Morgan Kaufmann Publishers.
- Ryan, M. R. K., 2007. *Graph Decomposition for Efficient Multi-Robot Path Planning.* Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007), Hyderabad, India, pp. 2003-2008, 2007.
- Surynek, P. 2009. *A Novel Approach to Path Planning for Multiple Robots in Bi-connected Graphs.* Proceedings of the 2009 IEEE International Conference on Robotics and Automation (ICRA 2009), pp. 3613-3619, Kobe, Japan, IEEE Press, 2009.
- Surynek, P. 2009. *Towards Shorter Solutions for Problems of Path Planning for Multiple Robots in ϑ-like Environments.* Proceedings of the 22nd International Florida Artificial Intelligence Research Society Conference (FLAIRS 2009), pp. 207-212, Sanibel Island, FL, USA, AAAI Press, 2009.
- Surynek, P. 2010. *An Optimization Variant of Multi-Robot Path Planning is Intractable.* Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI 2010), pp. 1261-1263, Atlanta, GA, USA, AAAI Press.