# Some Thoughts On Robustness in Multi-Agent Path Finding

**Roman Barták**
Charles University, Czech Republic

# Somewhere in Europe


TARZAN
42

# Somewhere in America

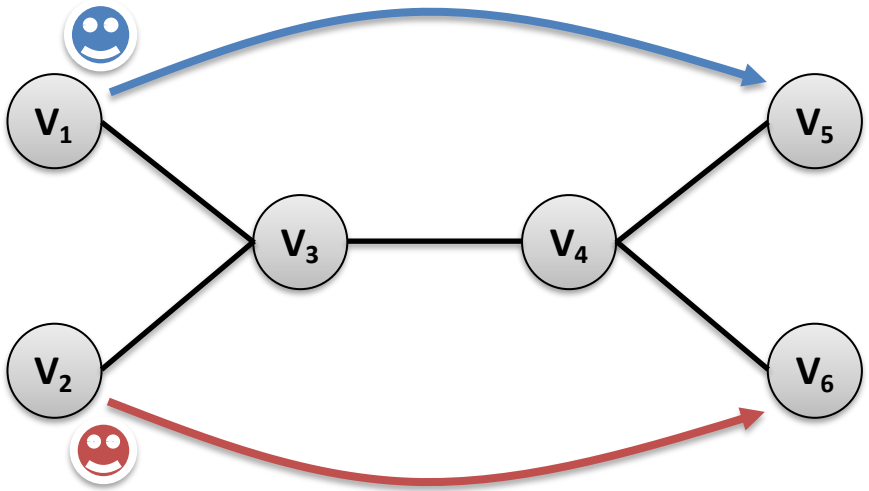
# Somewhere in Africa (or Asia)

# Somewhere in your computer


5100
8050
74/17

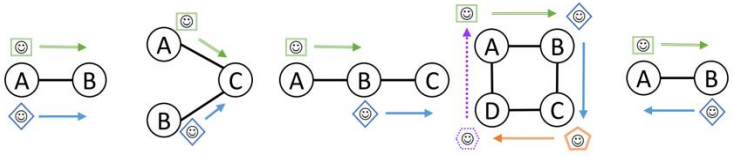
# Multi-agent Path Finding

- an (undirected) **graph**
- a set of **agents**, each agent is assigned to two locations (nodes) in the graph (start, destination)
- agents can **move** (to a neighboring node) or **wait**
- find a **collision free path** for each agent




Stern, R.; Sturtevant, N. R.; Felner, A.; Koenig, S.; Ma, H.; Walker, T. T.; Li, J.; Atzmon, D.; Cohen, L.; Kumar, T. K. S.; Barták, R.; and Boyarski, E.: Multi-Agent Pathfinding: Definitions, Variants, and Benchmarks. In SoCS 2019, pp. 151–159. AAAI Press.

Abstract plans are not always executed perfectly as planned due to external disturbances and imprecision of execution.

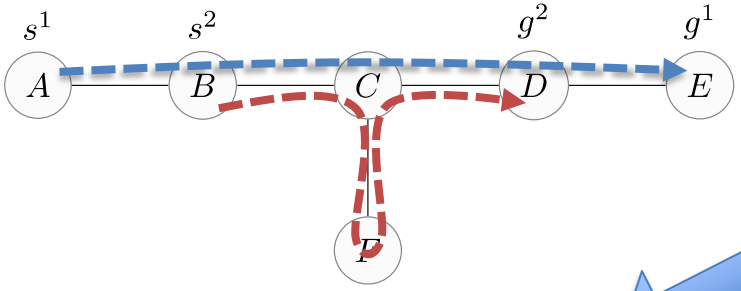



What can we do with it?

- re-planning
- **including robustness in plans or in their execution**

Barták, R.; Švancara, J.; Škopková, V.; Nohejl, D.; and Krasičenko, I.: Multi-agent path finding on real robots. AI Commun., 32(3): 175–189.

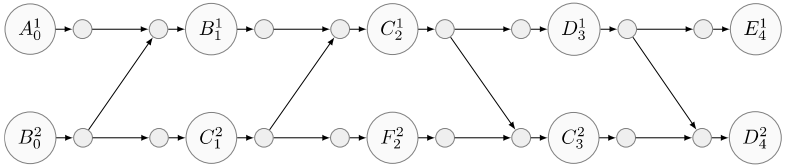
Plans for execution assume **kinematic constraints** and include points of **local synchronization**.




Abstract plans:

A B C D E

B C F C D



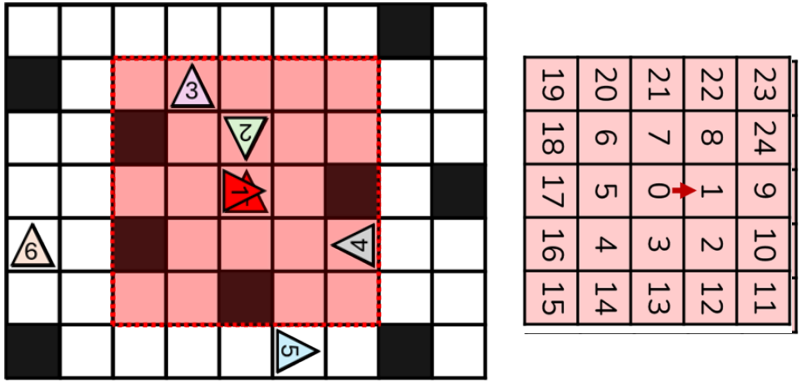

**Issue**: when one agent is delayed (possibly infinitely due to malfunction), the other agent must wait for it.

**Temporal Plan Graph** (TPG) describes synchronization of agents (if any agent is delayed the other agent can wait to prevent collision).

Hönig, W.; Kumar, T. K. S.; Cohen, L.; Ma, H.; Xu, H.; Ayanian, N.; and Koenig, S.: Multi-Agent Path Finding with Kinematic Constraints. In ICAPS 2016, pp. 477–485. AAAI Press.

**Social laws** (aka traffic rules) describe what action to select in the current situation.

- an agent follows own route but modifies it based on current neighborhood




(A1) (N3) (P3) (N4) (N13) -> (R100)

conjunction of prepositions -> action (with probability)

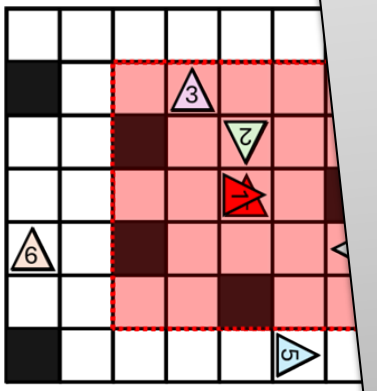
prepositions: **A**gent, **N**o agent, **P**assable (no obstacle), **O**bstacle

actions: **R**ight, **L**eft, **B**ackward, **S**tay
relative to agent's position and orientation

J. Slezák, J. Mestek J. and R. Barták: Social Laws for Multi-Agent Pathfinding. In ICAART 2025, pp. 436-470.

**Social laws** (aka traffic rules) describe what action to select in the current situation

– an a … d on curr…




1: (A1) (N3) (P3) (N4) (N13) > (R100)
2: (A1) (N5) (P5) (N6) (N17) > (B100)
3: (A1) > (S100)
4: (A2) (N3) (P3) (N4) (N13) > (R100)
5: (A2) (N5) (P5) (N6) (N17) > (B100)
6: (A2) > (S100)
7: (A9) (N3) (P3) (N4) (N13) > (R100)
8: (A9) (N5) (P5) (N6) (N17) > (B100)
9: (A9) > (S100)
10: (A10) (N3) (P3) (N4) (N13) > (R100)
11: (A12) (N3) (P3) (N4) (N13) > (R100)
12: (A3) (N5) (P5) (N6) (N17) > (B100)
13: (A11) (N3) (P3) (N4) (N13) > (R100)
14: (A13) (N5) (P5) (N6) (N17) > (B100)

do not enter occupied vertex

priority to the right

agent incoming to 1 from opposite direction

better coordination & deadlock prevention

(N13) ->

(with probability)

able (no obstacle),

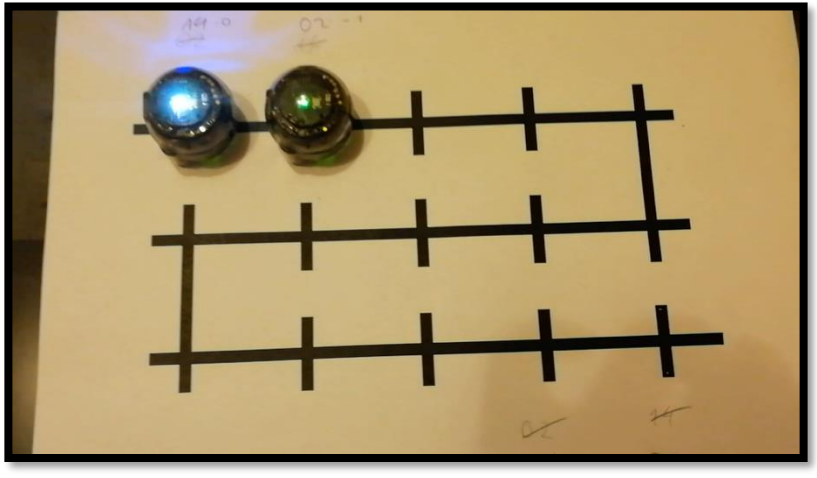
**B**ackward, **S**tay

relative to agent's position and orientation

J. Slezák, J. Mestek J. and R. Barták: Social Laws for Multi-Agent Pathfinding. In ICAART 2025, pp. 436-470.

Plans can be generated to proactively assume possible issues (delays) during exaction.

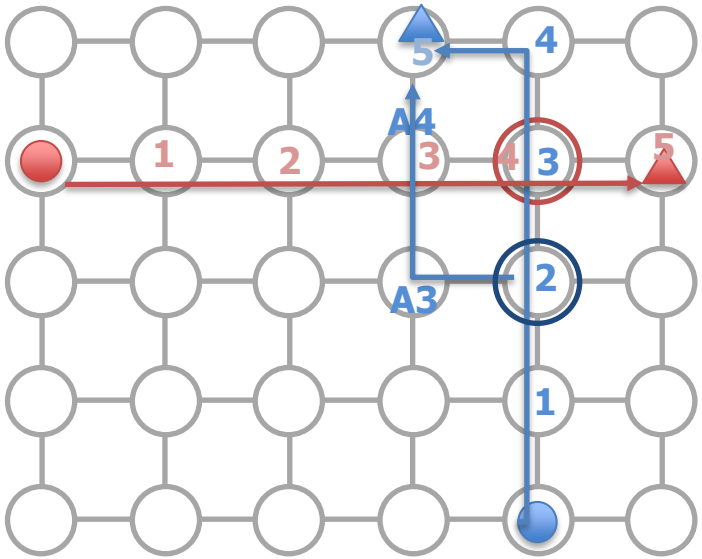
**k-robustness** ensures that plans can be executed even if any agent is delayed at most k steps (agent can only enter node which has been unoccupied for last k steps).



**Issue:** always assumes the worst case -> longer plans or plans may even not exist (in dense environments).

Atzmon, D.; Stern, R.; Felner, A.; Wagner, G.; Barták, R.; and Zhou, N.: Robust Multi-Agent Path Finding and Executing. J. Artif. Intell. Res., 67: 549–579.

Main plans may contain alternative paths in case of delay to minimize influence on other agents and prolonging plans (**contingency planning**).



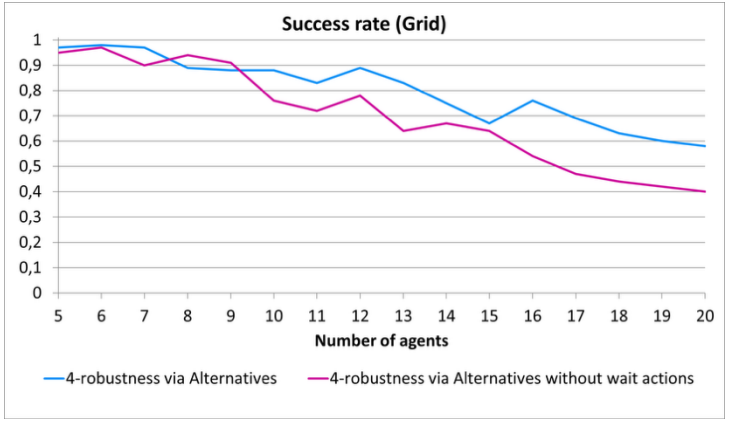

1. Take a collision-free plan (main routes)
2. Identify locations of possible collisions
3. Plan alternative routes to divert robots before the collision will happen

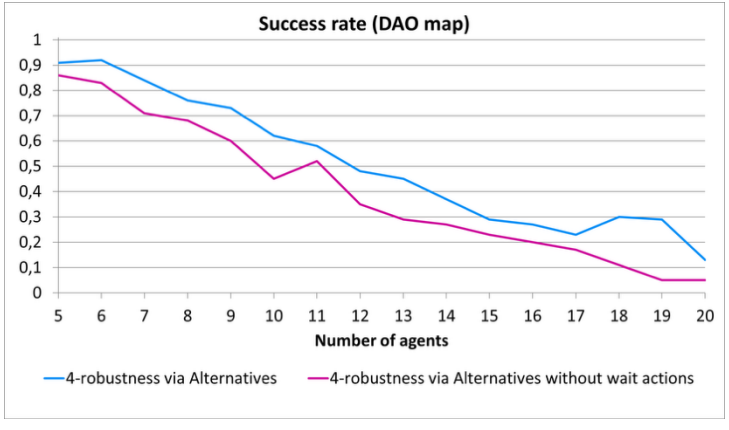
**Issue**: when more agents are delayed, the alternative plans do not guarantee collision-free execution (too computationally expensive).

Nekvinda, M.; and Barták, R.: Contingent Planning for Robust Multi-Agent Path Finding. In ICTAI 2021, pp. 487–492. IEEE.

**Observation**: k-robustness always requires a larger gap between agents

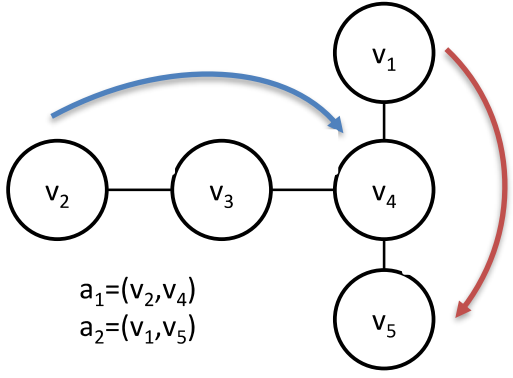
**Idea**: if the robot is waiting at its destination, the wait actions can be used earlier in the plan to enlarge gap between agents



**MAPF with Temporal Uncertainty** (Shahar et al., 2021):

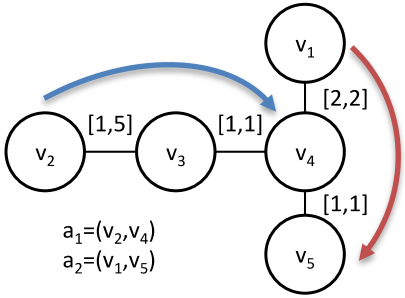
- *classical MAPF*: agents (on a graph) are moving from their starting positions to destinations without (vertex, edge, and swapping) collisions
- each *edge has minimal ($w^-$) and maximal ($w^+$) duration* for traversing
- it is unknown what the actual duration of moving will be (uncertainty)
- **safe solution** guarantees no collisions independently of actual traversal times of agents
  - *optimistic solution* = the earliest time in which the agent can reach its goal (sum of $w^-$)
  - *pessimistic solution* = the latest time in which the agent will reach its goal (sum of $w^+$)



# Plans vs. policies

**Plan** = a sequence of move/wait actions (can be represented as a sequence of vertices/edges)

- useful for blind execution (agent cannot sense location, cannot measure time, cannot sense other agents, cannot communicate)




*plan*

| | |
|---|---|
| $a_1$ | ↻ → → |
| $a_2$ | ↓ ↓ |

**Policy** = recommendation of action based on location and time

- agent knows location and measures time, but cannot sense other agents and cannot communicate

*policy*

| | | | |
|---|---|---|---|
| $a_1$ | 0 | $v_2$ | → |
| $a_1$ | 1 | $v_3$ | ↻ |
| $a_1$ | 2 | $v_3$ | → |
| $a_1$ | 3 | $v_3$ | → |
| | | | ... |
| $a_2$ | 0 | $v_1$ | ↓ |
| $a_2$ | 2 | $v_4$ | ↓ |

J. Švancara, D. Zahrádka, M. Subramanian, R. Barták, M. Kulich: Generating Safe Policies for Multi-Agent Path Finding with Temporal Uncertainty. ICAART 2025, pp. 1238-1244

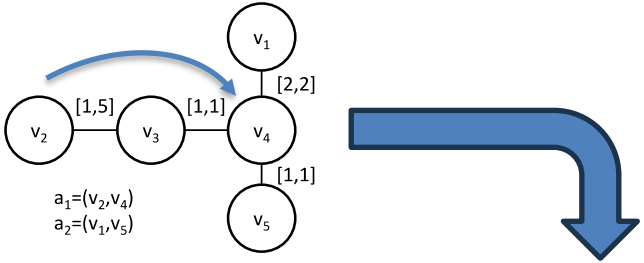
**Proposition 1.** If there exists a classical MAPF solution, there also exists a policy-based solution to MAPF-TU with any positive $w^-$ and $w^+$.

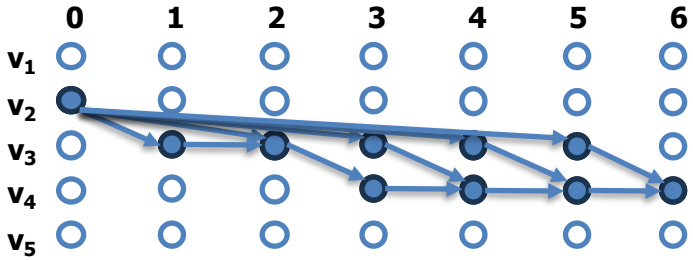
**Proposition 2.** A policy-based solution always produces a solution with equal or better cost compared to a plan-based solution when optimizing optimistic soc or pessimistic soc.

J. Švancara, D. Zahrádka, M. Subramanian, R. Barták, M. Kulich: Generating Safe Policies for Multi-Agent Path Finding with Temporal Uncertainty. ICAART 2025, pp. 1238-1244

**At(V,A,T)**: agent A is at vertex V at time T

**Pass(E,A,T)**: agent A is using edge E at time T




$$\forall a_i \in A : At(s_i, a_i, 0) = 1 \quad (1)$$

$$\forall a_i \in A : At(g_i, a_i, T) = 1 \quad (2)$$

$$\forall v \in V, \forall a_i \in A, \forall t \in \{0, \ldots, T-1\} :$$
$$At(v, a_i, t) \implies \sum_{(v,u)\in E} Pass((v,u), a, t) = 1 \quad (3)$$

$$\forall (v,u) \in E, \forall a_i \in A, \forall t \in \{0, \ldots, T-1\} :$$
$$Pass((v,u), a, t) \implies At(v, a_i, t) \quad (4)$$

$$\forall (v,u) \in E, \forall a_i \in A,$$
$$\forall t \in \{0, \ldots, T - w^+((v,u))\},$$
$$\forall w \in \{w^-((v,u)), w^+((v,u))\} :$$
$$Pass((v,u), a, t) \implies At(u, a, t+w) \quad (5)$$

$$\forall v \in V, \forall t \in \{0, \ldots, T\} : \sum_{a_i \in A} At(v, a_i, t) \leq 1 \quad (6)$$

$$\forall (u,v) \in E : u \neq v, \forall t \in \{0, \ldots, T-1\} :$$
$$\sum_{\substack{a_i \in A, \\ (x,y)\in\{(u,v),(v,u)\}, \\ w\in\{w^-((x,y)),w^+((x,y))\}}} (Pass((x,y), a_i, t+w) \leq 1 \quad (7)$$

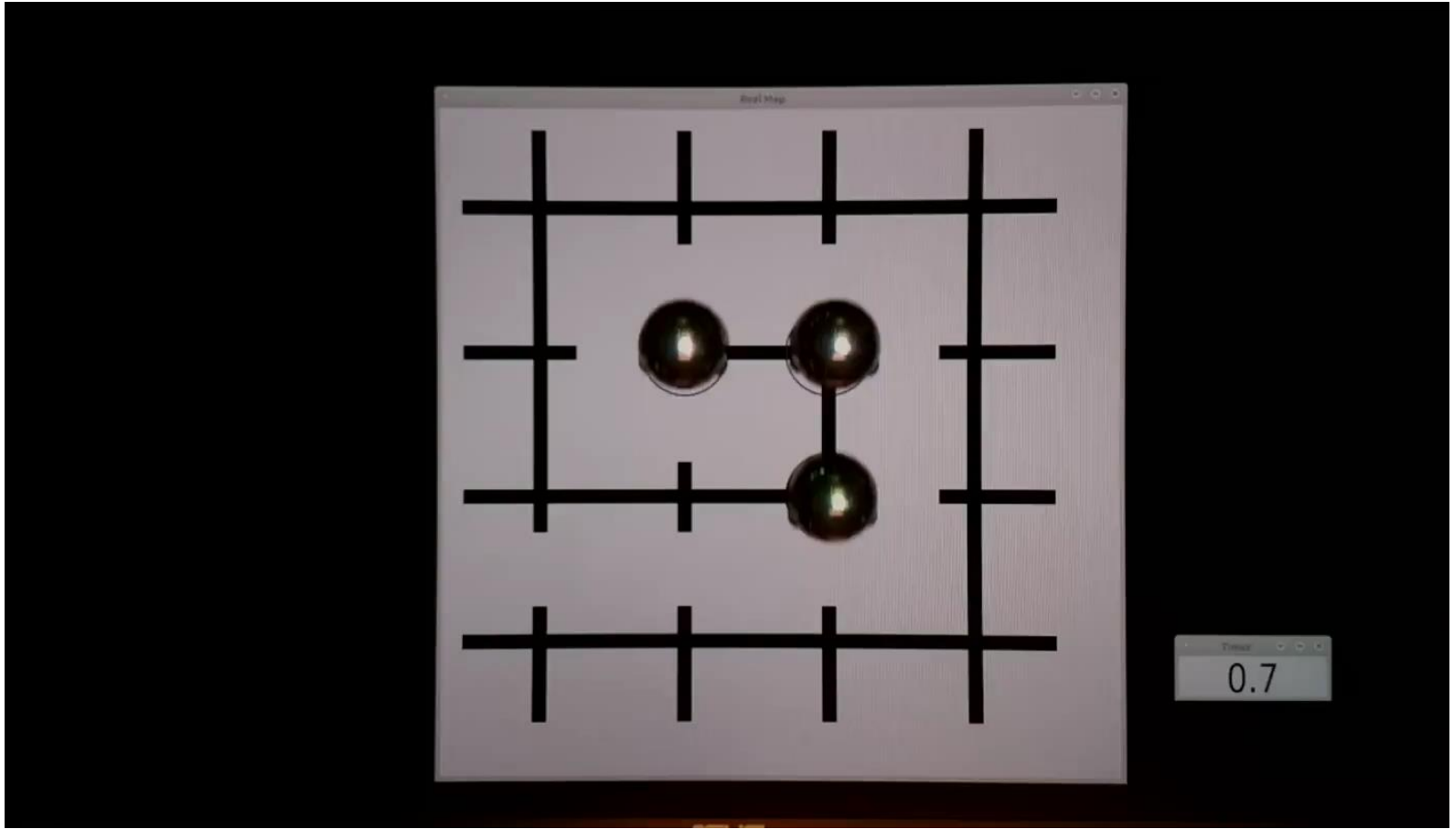
J. Švancara, D. Zahrádka, M. Subramanian, R. Barták, M. Kulich: Generating Safe Policies for Multi-Agent Path Finding with Temporal Uncertainty. ICAART 2025, pp. 1238-1244

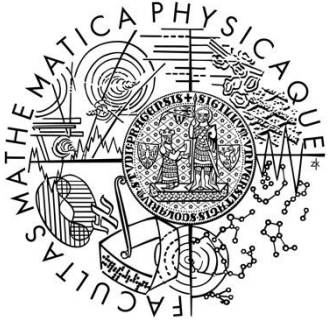
Better **integration of plans and policies**

- **plans** are easier to find and understand and better for achieving long term goals, but too rigid for execution
- **policies** are harder to find and understand but more flexible in case of uncertainties

Exploitation of **machine learning**

- learning policies from plans


Real Map
Timer
0.7




**Roman Barták**
Charles University, Faculty of Mathematics and Physics
bartak@ktiml.mff.cuni.cz