Comparison of A* and RRT-Connect Motion Planning Techniques for Self-Reconfiguration Planning

David Brandt

presented by Jan Mrázek

ParaDiSe
Parallel & Distributed Systems Laboratory

Masaryk University

Brno, Czech Republic

26th October 2020

The Problem of Reconfiguration

- given an initial and target configuration
  - structure
  - position in space
- find a sequence of atomic actions that lead from the initial to the target configuration
  - collision-free
  - feasible (e.g., consider limited strength of joints)
  - optimal vs. feasible solution

The Problem of Reconfiguration

- given an initial and target configuration
  - structure
  - position in space
- find a sequence of atomic actions that lead from the initial to the target configuration
  - collision-free
  - feasible (e.g., consider limited strength of joints)
  - optimal vs. feasible solution

This paper:

- tackles reconfiguration for ATRONs
- uses state-space search
  - A*
  - RRT-Connect
- provides comparison

ATRONs

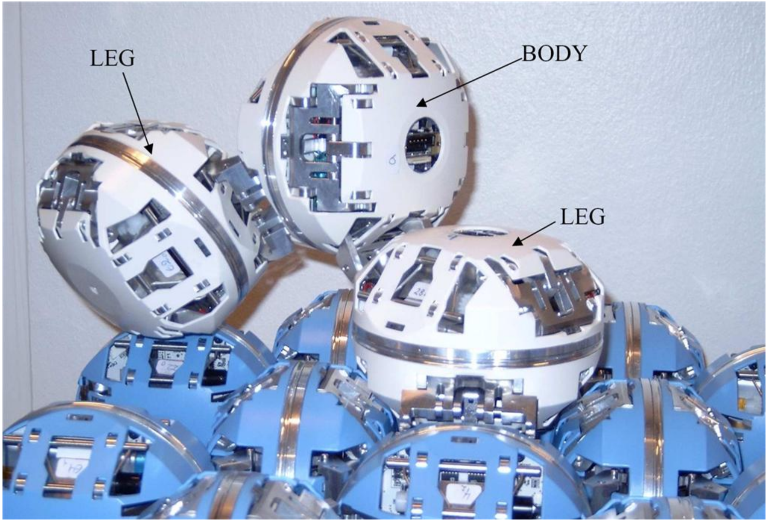

Figure 1:ATRON

Sidenote

- state-space of reconfiguration have high number of dimensions
- all algorithms we will present work on unlimited number of dimensions
- we will illustrate them in 2D for simplicity

- build a discrete graph over the state space by random sampling
- use, e.g., Dijkstra to find shortest paths in the discrete graph

Building the graph:

- take random configuration
- if invalid, discard
- find path to an already sampled points
  - when not found - discard
  - add new vertex and edge to the graph
  - use heuristics
- repeat until dense enough graph is built

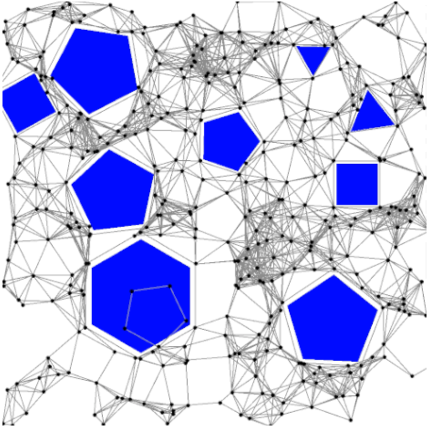

Figure 2:Discrete navigation graph

Preliminary: Navigation Graph

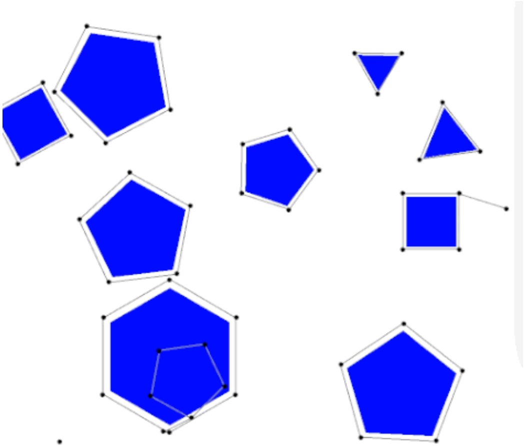

Figure 3:Step 1 (Source Wikipedia)

Preliminary: Navigation Graph

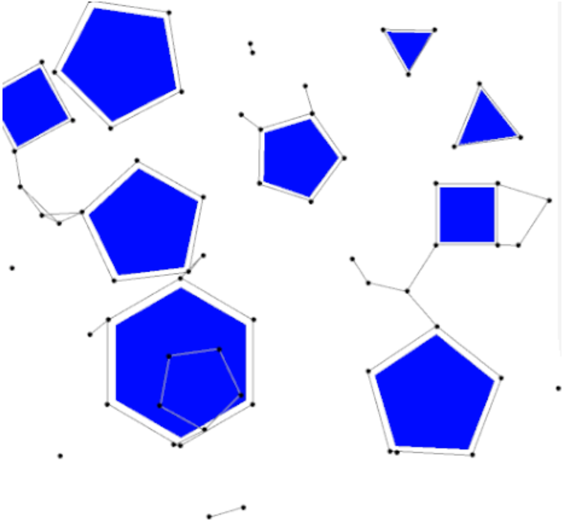

Figure 4:Step 2 (Source Wikipedia)

Preliminary: Navigation Graph

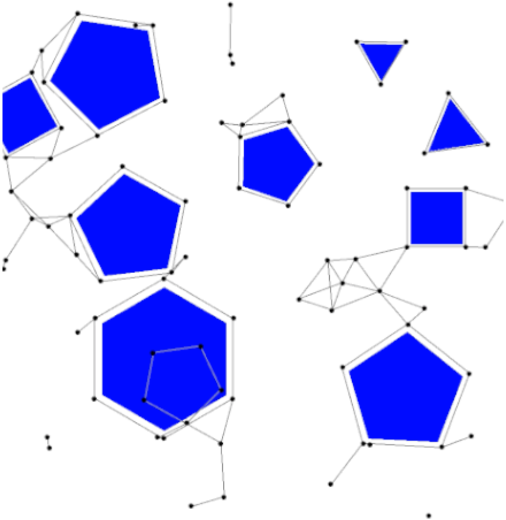

Figure 5:Step 3 (Source Wikipedia)

Preliminary: Navigation Graph

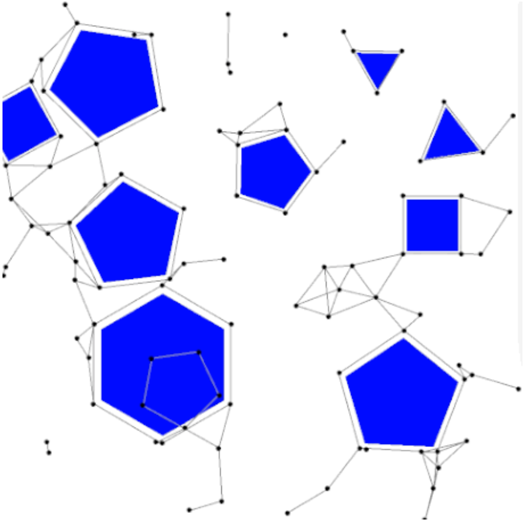

Figure 6:Step 4 (Source Wikipedia)

Preliminary: Navigation Graph

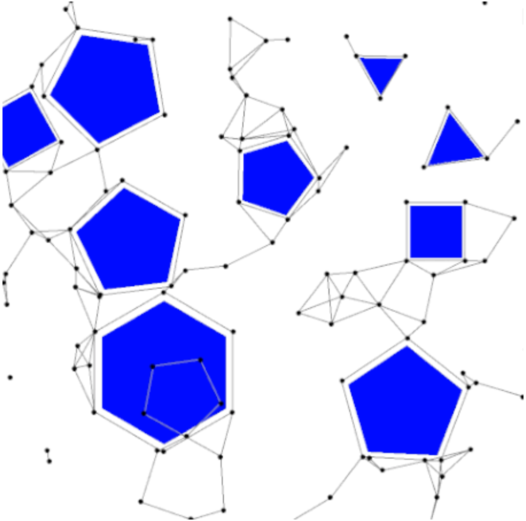

Figure 7:Step 5 (Source Wikipedia)

Preliminary: Navigation Graph

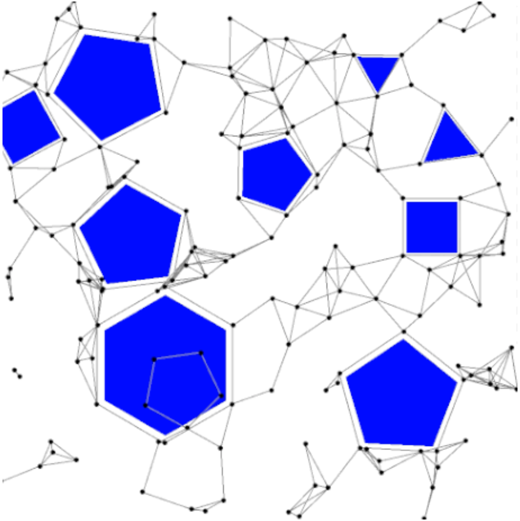

Figure 8:Step 6 (Source Wikipedia)

Preliminary: Navigation Graph

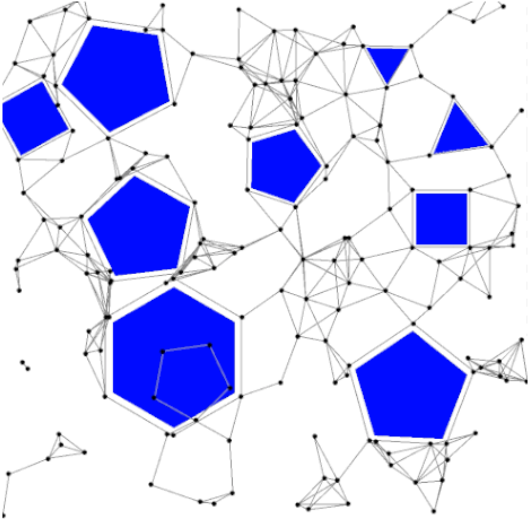

Figure 9:Step 7 (Source Wikipedia)

Preliminary: RRT

- find only path to the nearest point
- $\rightarrow$ tree

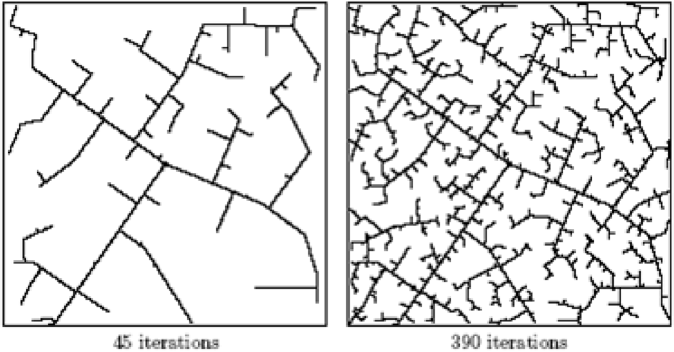

Figure 10:RRT (Source Wikipedia)

It is hard to a find path between two configurations

- make only a step towards random configuration
- build two trees and try to connect them

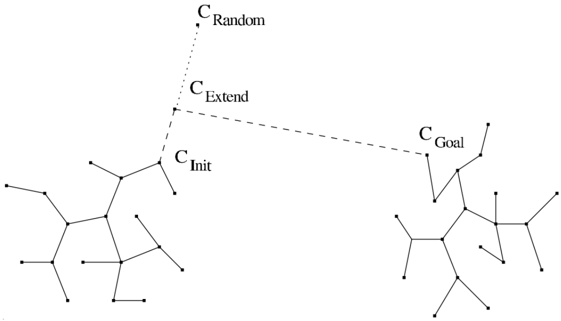

Figure 11:RRT - Connect

It is hard to a find path between two configurations

- make only a step towards random configuration
- build two trees and try to connect them

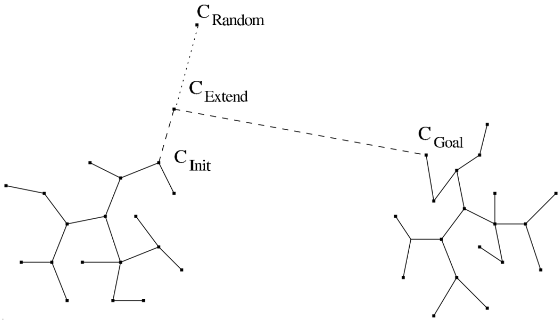

Figure 11:RRT - Connect

What is “a step towards a configuration?”

Configuration Similarity Metric

- optimal solution: minimal number of atomic steps to change one configuration to another
  - the problem is NP-complete (proven in 2016)

Configuration Similarity Metric

- optimal solution: minimal number of atomic steps to change one configuration to another
  - the problem is NP-complete (proven in 2016)

Approximation:

$$\mathrm{dist}(a_1, a_2) = \max(|a_{1_x}, a_{2_x}|, |a_{1_y}, a_{2_y}|, |a_{1_z}, a_{2_z}|) + \frac{\mathrm{diffOri}(a_1, a_2)}{2} + \frac{\mathrm{diffConn}(a_1, a_2)}{16}$$

where:

$$\mathrm{diffOri}(a_1, a_2) = \begin{cases} 1 & \text{if orientation of } a_1 \text{ differs from } a_2 \\ 0 & \text{otherwise} \end{cases}$$

$$\mathrm{diffConn}(a_1, a_2) = \text{number of connectors in different state}$$

Find pairing of modules from one configuration the other minimizing sum of distances

Hungarian Algorithm

- solve assignment between workers and tasks
- input is matrix $n \times n$:

	Clean bathroom	**Sweep floors**	**Wash windows**
Paul	2$	3$	3$
Dave	3$	2$	3$
Chris	3$	3$	2$

- minimum cost: $6
  - Paul clean the bathroom
  - Dave sweep the floors
  - Chris wash the windows
- runs in $\mathcal{O}(n^3)$

Configuration Similarity Metric: Putting it All Together

- prepare matrix $n \times n$ representing possible pairings
  - values in the matrix are $\mathrm{dist}(a_x, a_y)$
  - can be computed in $\mathcal{O}(n^2)$
- find the minimal pairing using the Hungarian algorithm
  - can be computed in $\mathcal{O}(n^3)$

- finding optimal C_{Extend} is expensive
- the authors use 3 atomic steps as an approximation

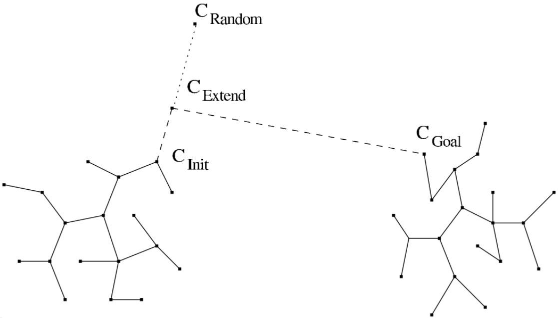

Figure 12:RRT - Connect

Recap: A*

- create a set `open` with the initial configuration
- create an empty set `closed`
- repeat until `open` is not empty and path have not been found:
  - get a configuration from `open` that is closest to the target configuration
  - if the distance is zero, path have been found
  - generate all successors and put them `open` if they are not in `closed`
  - put configuration into `closed`

Successors are generated as a sequence of 3 atomic steps to overcome small local minima.

Comparison

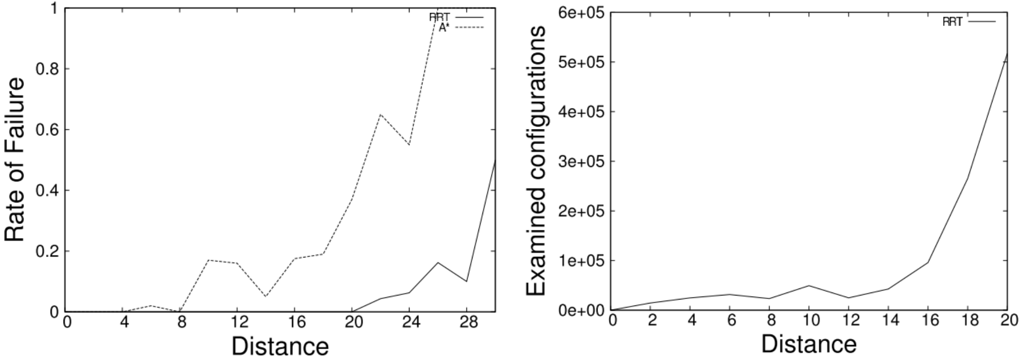

Figure 13:Results

- limit 10^7 states examined (~30 minutes of compute time)
- note the clear exponential increase in examined states

Conclusion

- RRT-Connect can be perceived as “randomized A*”
  - it helps to move to another location before examining the whole local minima
- A* finds optimal solution, RRT-Connect finds long paths
  - RRT paths can be post-processed

Conclusion

- RRT-Connect can be perceived as “randomized A*”
  - it helps to move to another location before examining the whole local minima
- A* finds optimal solution, RRT-Connect finds long paths
  - RRT paths can be post-processed

What might be interesting for RoFI:

- ignore module ids by performing matching
- came up with similar similarity metrics
  - experimentally fine-tune parameters
- implement RRT-Connect and benchmark it