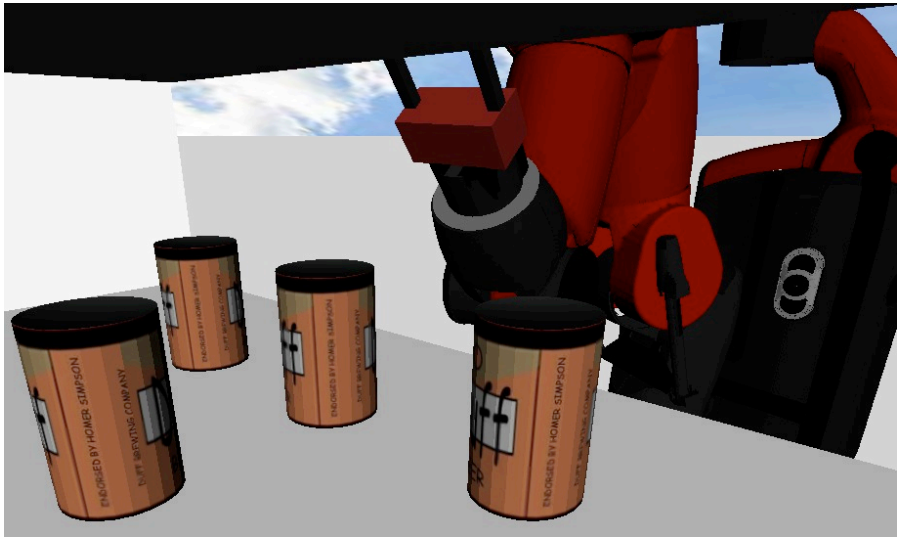
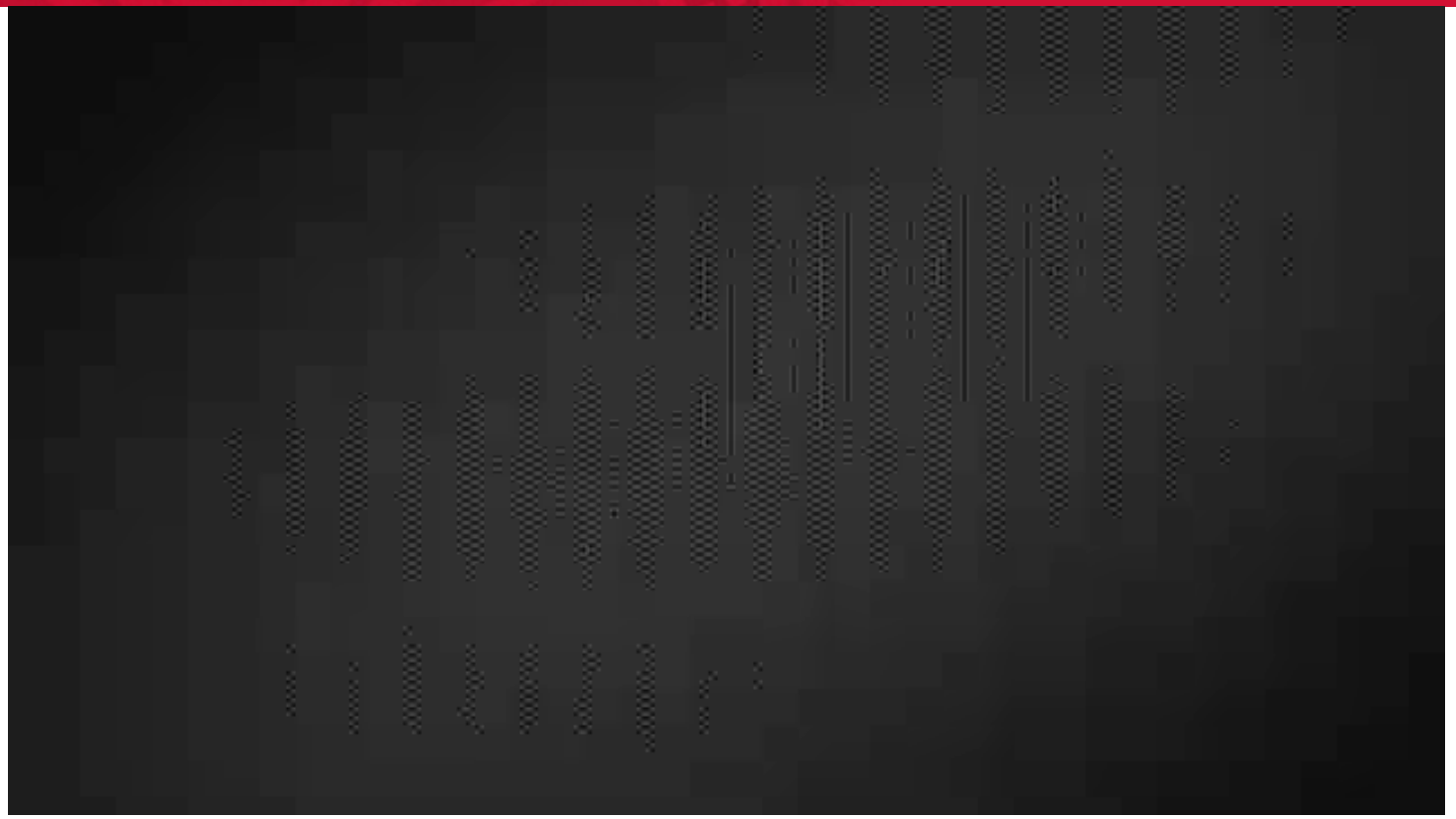
Similar Part Rearrangement With Pebble Graphs



Athanasios Krontiris, Rahul Shome, Andrew Dobson, Andrew Kimmel and Kostas E. Bekris
Department of Computer Science
Rutgers University

Example



Warehouse management

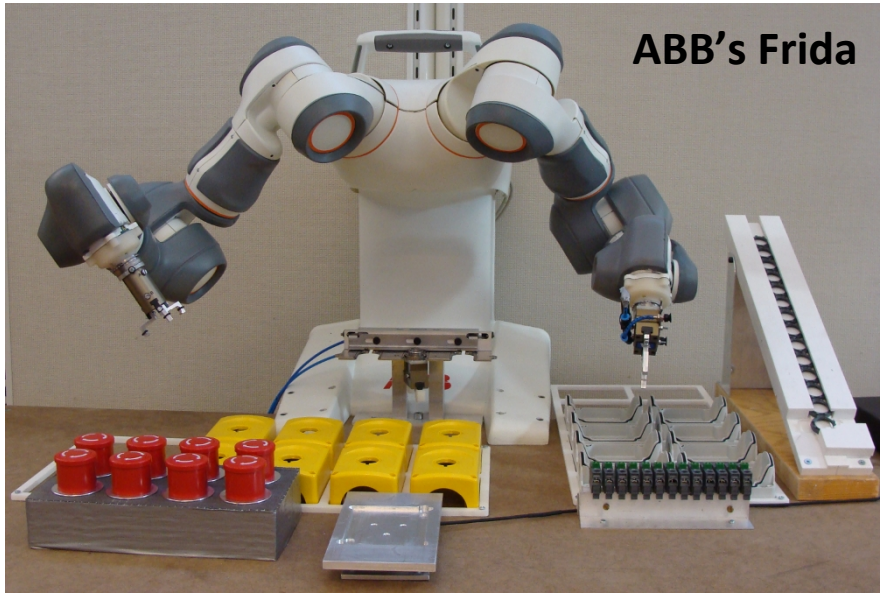
Cluttered manipulation

Motivation

- Humanoid/compliant robots become available for industrial applications

- They need to operate in less structured, cluttered, narrow workspaces where reaching every object is not easy

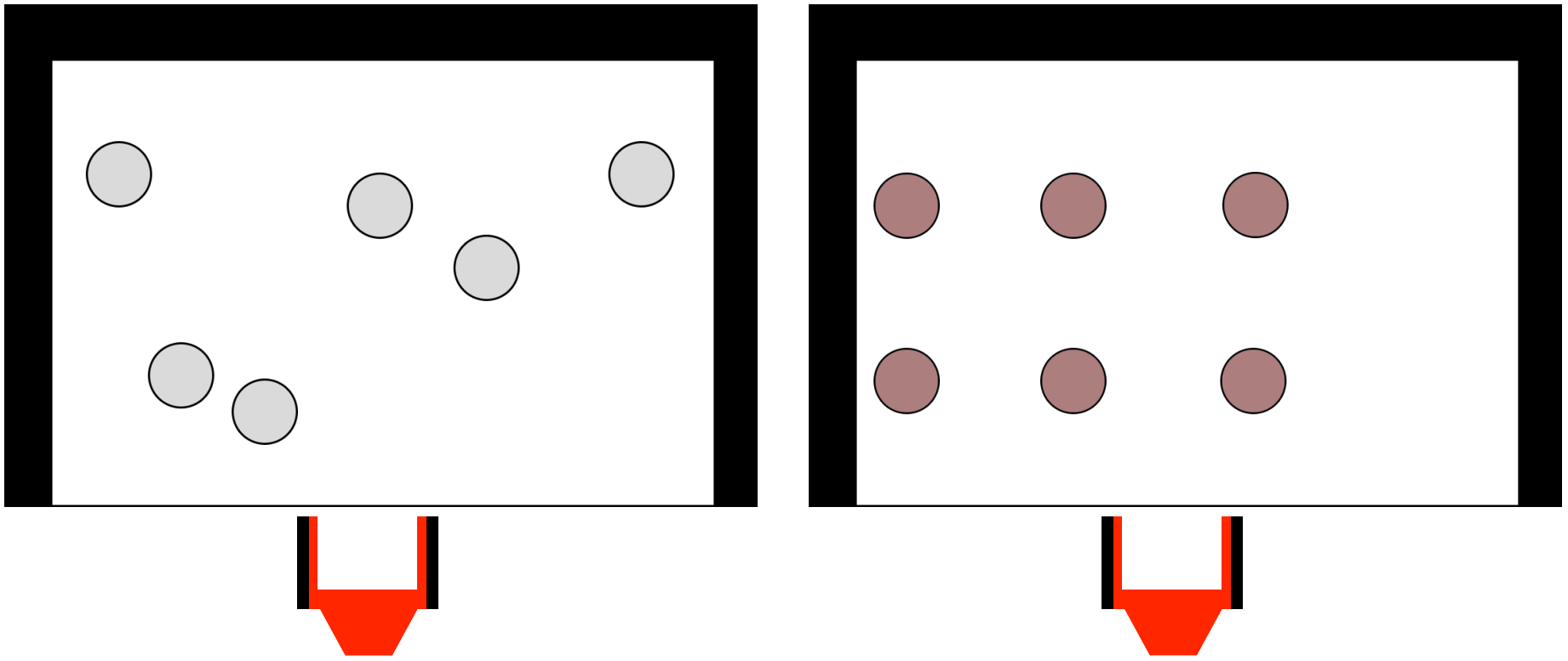
- One capability that is needed:
  - Rearrange products or components in a desired way



Unlabeled Rearrangement Problem

Given an initial and a final object arrangement compute a continuous manipulation path to move the objects from the initial to the final arrangement without collisions.

– Focus on motion planning and combinatorial aspects



Challenging Problem

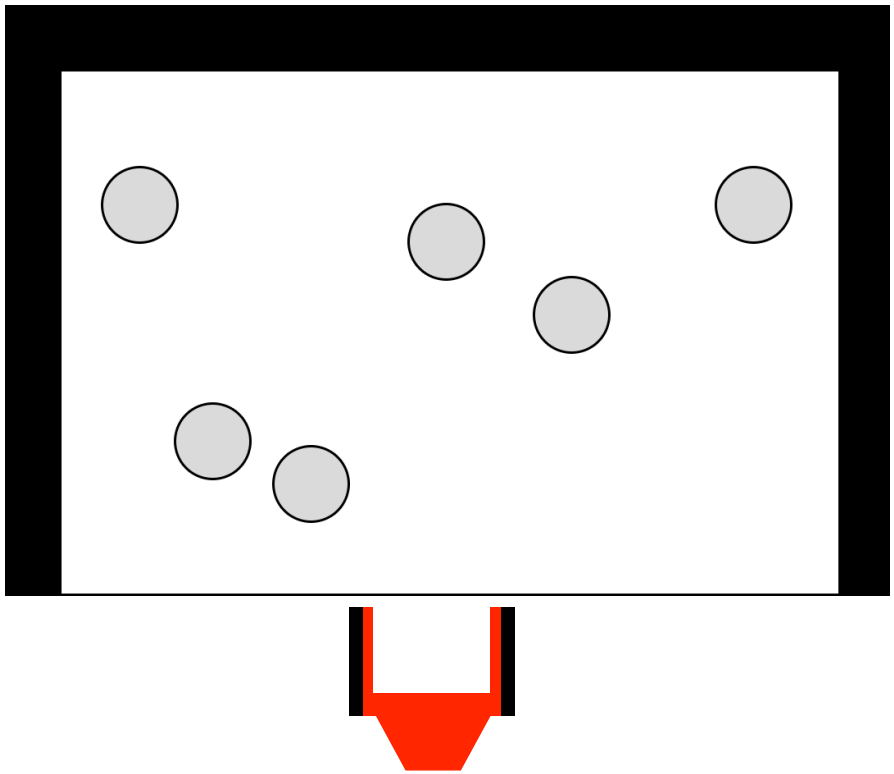
- High-dimensional challenge

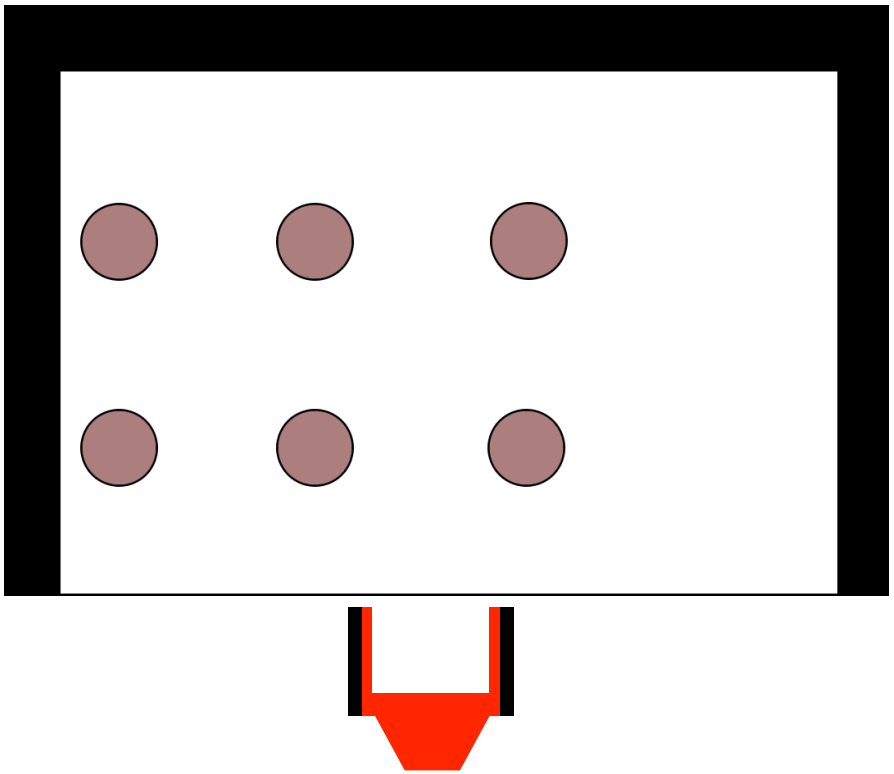
  7 DOF arms + 6 DOF per object

- Brings together many other problems:
  - Manipulation and grasping
  - Multi-robot motion planning

Easy instances:
Monotone Challenges

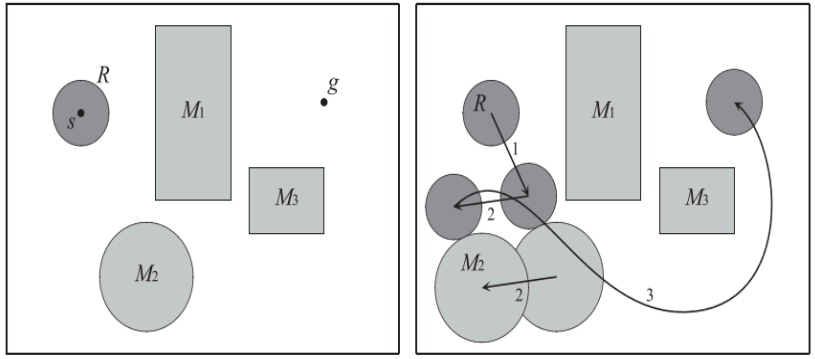
Hard instances:
Non-Monotone
(an object must be grasped more than once)



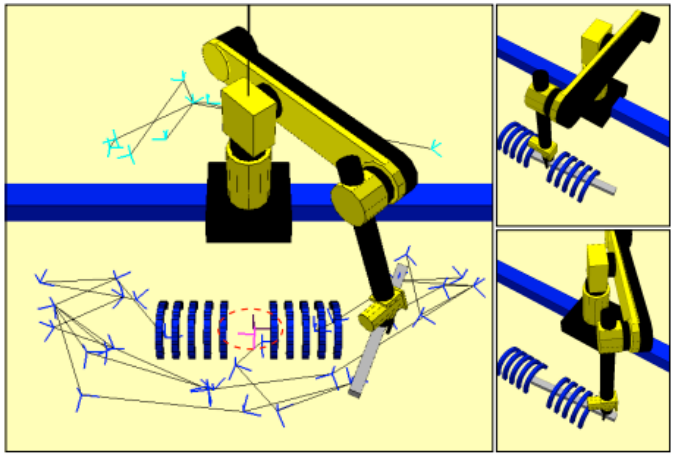


Background

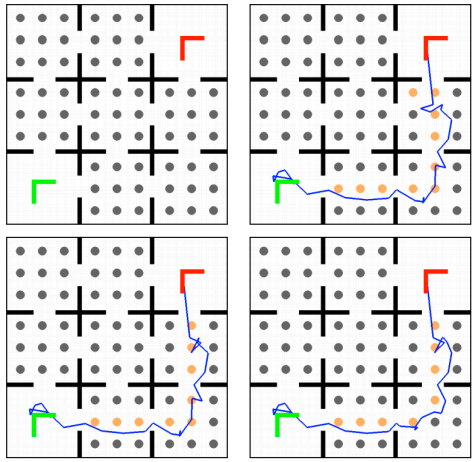
- Planning among movable obstacles & Manipulation
  - Navigation Among Movable Obstacles.
    [J. van den Berg, M. Stilman et al, 2008]
  - Manipulation Among Movable Obstacles.
    [M. Stilman et al, 2007]
  - Minimum Constraint Displacement.
    [K. Hauser, 2012]
  - Manipulation Planning with Probabilistic Roadmaps.
    [T. Simeon, J.-P. Laumond, et al., 2004]



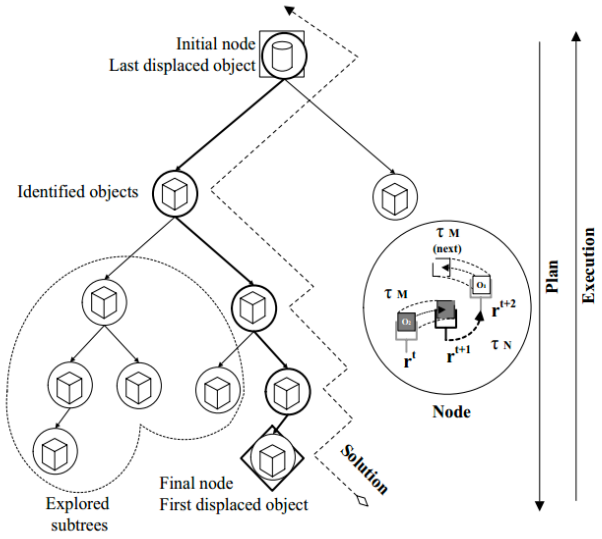
[J. van den Berg, M. Stilman et al, 2008]



[T. Simeon, J.-P. Laumond, et al., 2004]



[K. Hauser, 2012]



[M. Stilman et al, 2007]

Task and Motion Planning

Rearrangement planning is a prototypical example of

- integrating task and motion planning
- combining continuous and discrete reasoning

- Motion planning:
  - The continuous motions for reaching to and grasping motions
- Task planning:
  - Placing objects to different locations (combinatorial aspect)

Idea:

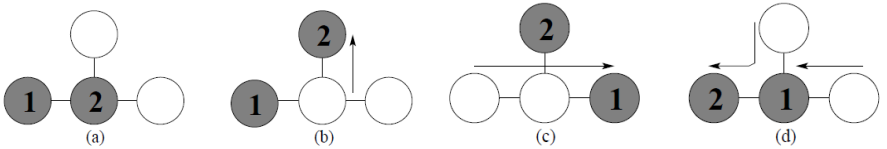
- Implicitly represent entire sets of states in a compact manner without the need to enumerate them
- Use multi-robot path planning tools in this direction

Seeking Inspiration

- Multi-robot planning.
  - Discrete solvers for multi-robot motion planning.

    [Luna, R., Bekris, K.E., 2011]

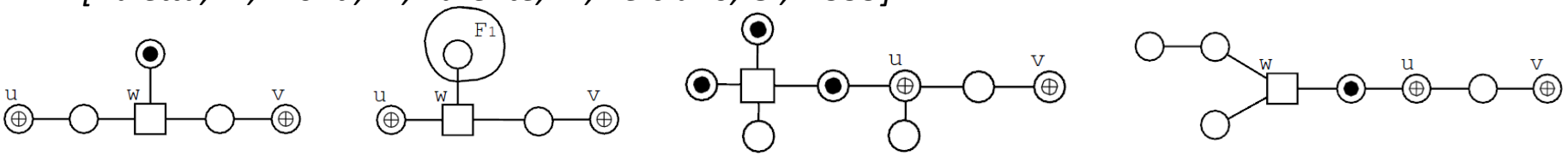
    [Yu, J., LaValle, S.M., 2012]



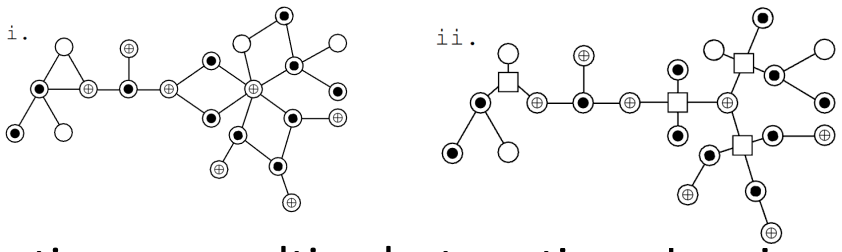
[Luna, R., Bekris, K.E., 2011]

  - Feasibility test for pebble motion on graphs.

    [Auletta, V., Monti, A., Parente, D., Persiano, G., 1999]

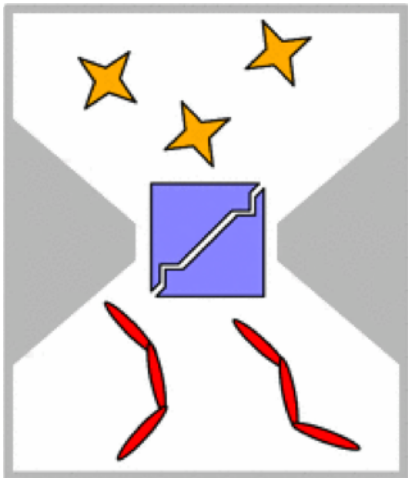


[Goraly, G., Hassin, R., 2010]



  - Continuous multi-robot motion planning.

    [Solovey, K., Halperin, D. 2012]

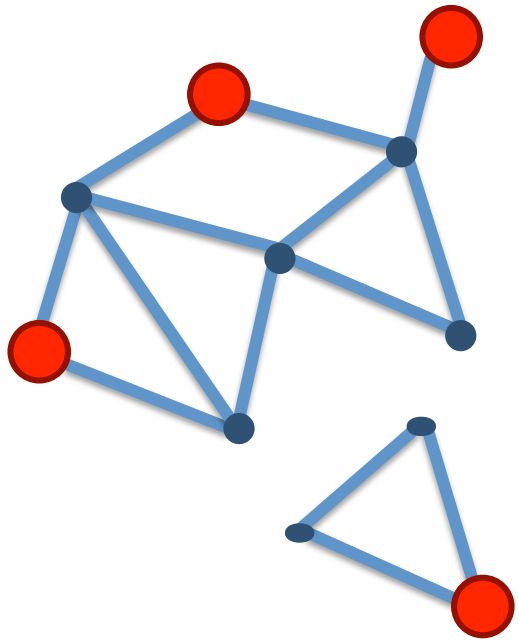


[Solovey, K., Halperin, D. 2012]

Approach

Unlabeled Pebble Graphs

- Given a graph with multiple connected components
- And k unlabeled pebbles that can occupy vertices



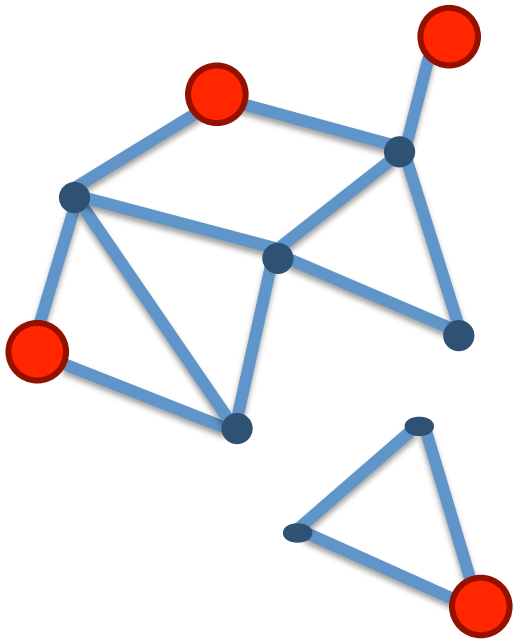
K = 4
Signature = {3,1}

- A set of reachable arrangements can be represented compactly by:
  - the graph
  - and a "signature" indicating the number of pebbles on each connected component

Linear time to find a path between any two reachable arrangements

PGs for Manipulation

- Difference between discrete abstraction and manipulation challenge:
  - Requires no collisions between objects occupying two nodes and between nodes/edges
  - The manipulator must be able to move objects along edges



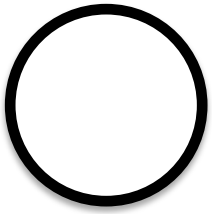
K = 4

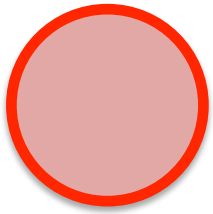
Signature = {3,1}

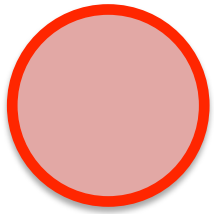
- Approach: Try to abstract out the motion of the manipulator and reason directly about the movement of objects

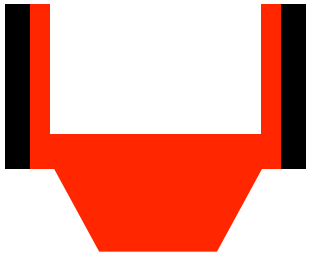
Computing the Graph

- Given a safe configuration for the robot.
- Given initial arrangement.
- And the target arrangement.

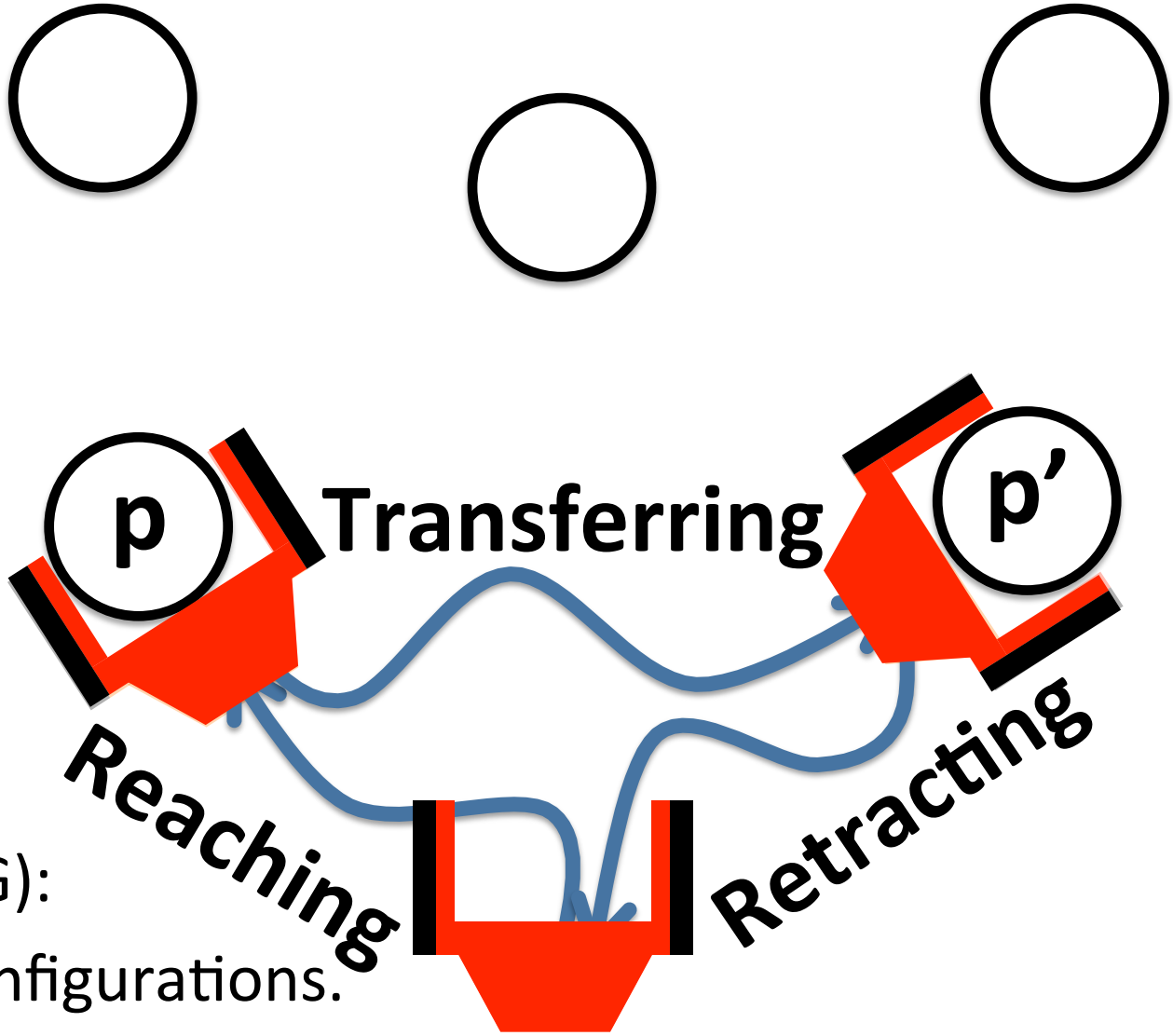








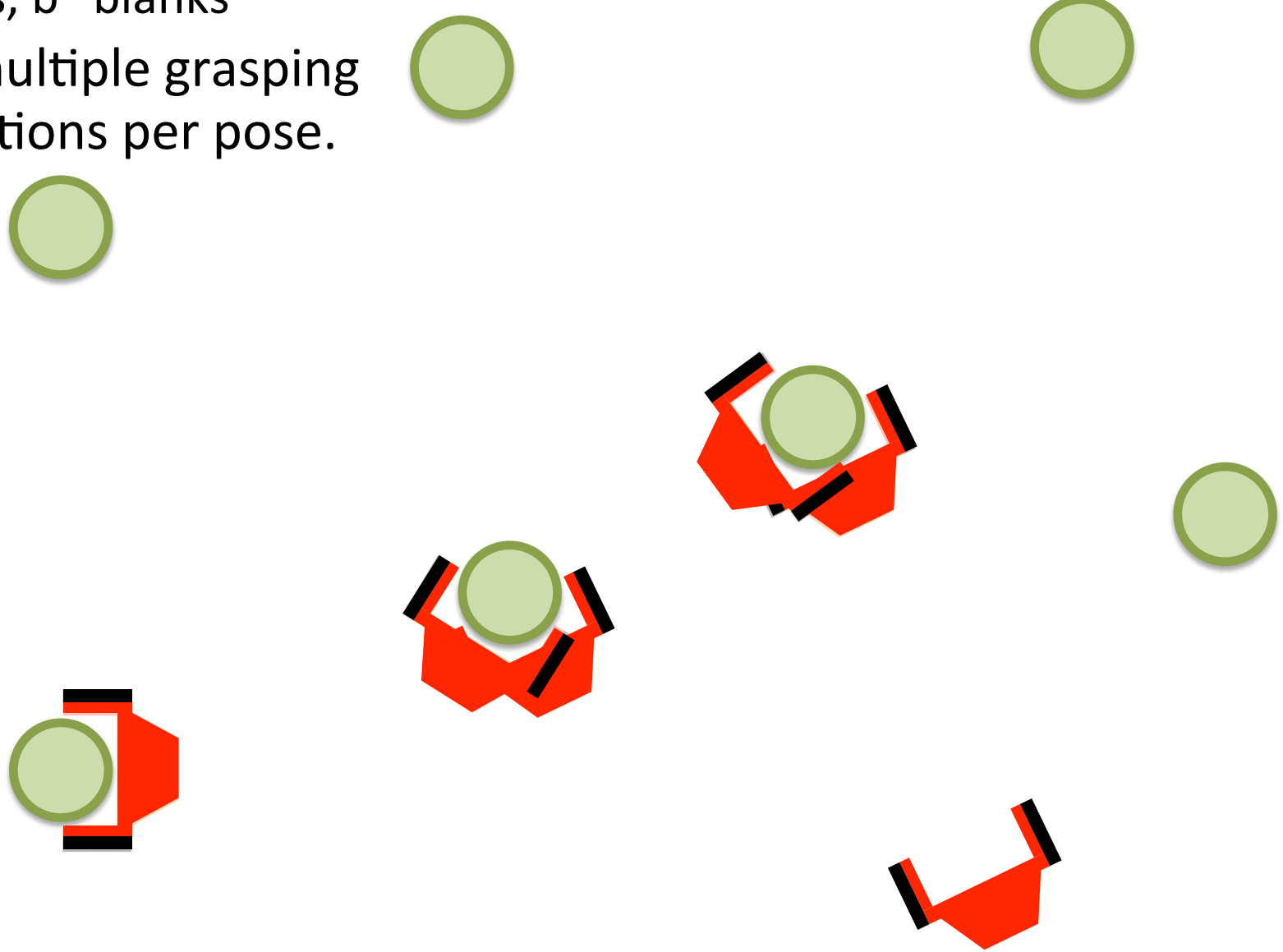
Computing the Graph



- Rearrangement Pebble Graph (RPG):
  - Nodes: Stable configurations.
  - Edges: Collision free motions of the arm that transfer an object between poses.

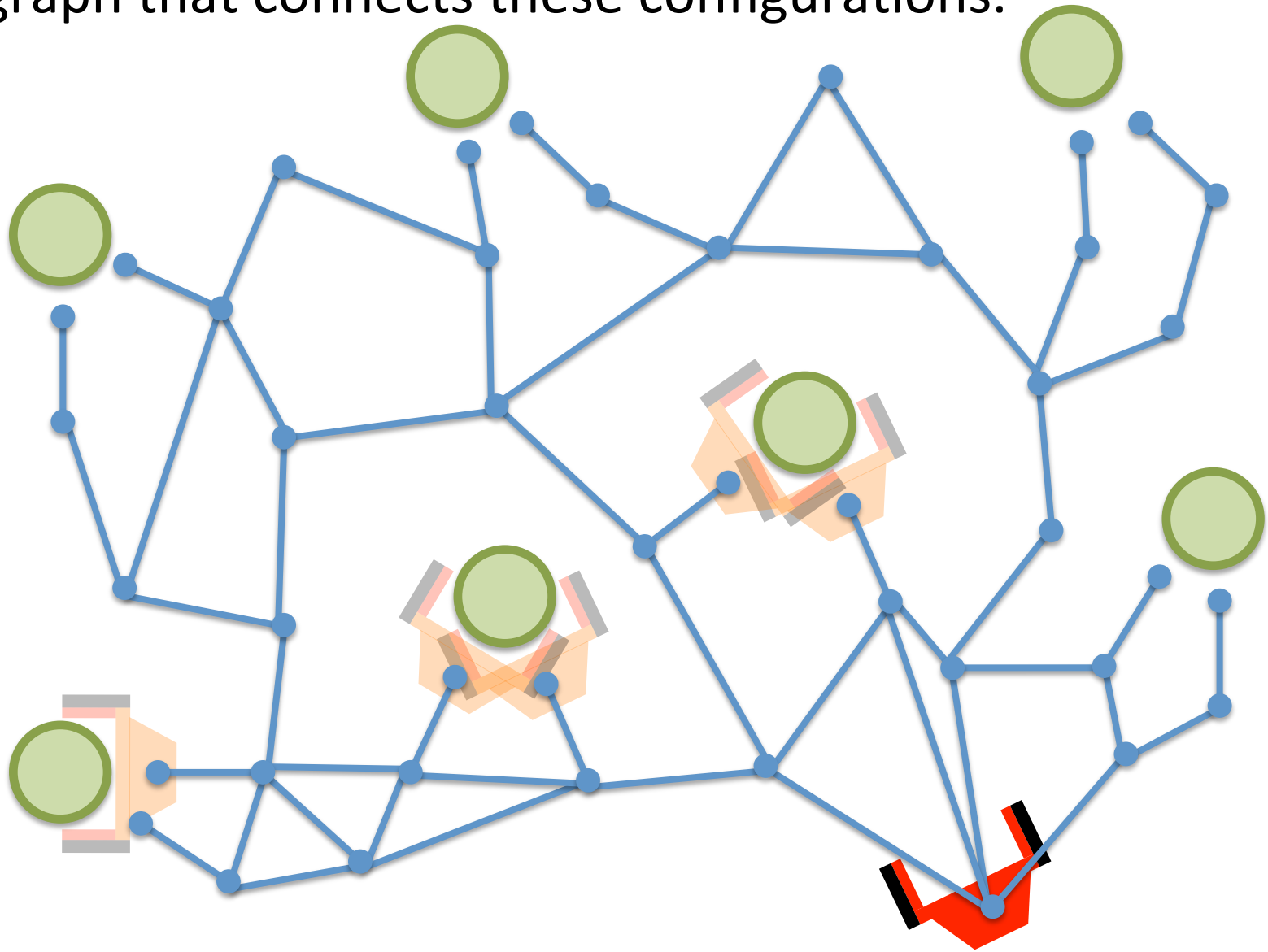
Use of a Manipulation Graph for RPGs

- Sample k+b poses
  - k objects, b "blanks"
- Sample multiple grasping configurations per pose.



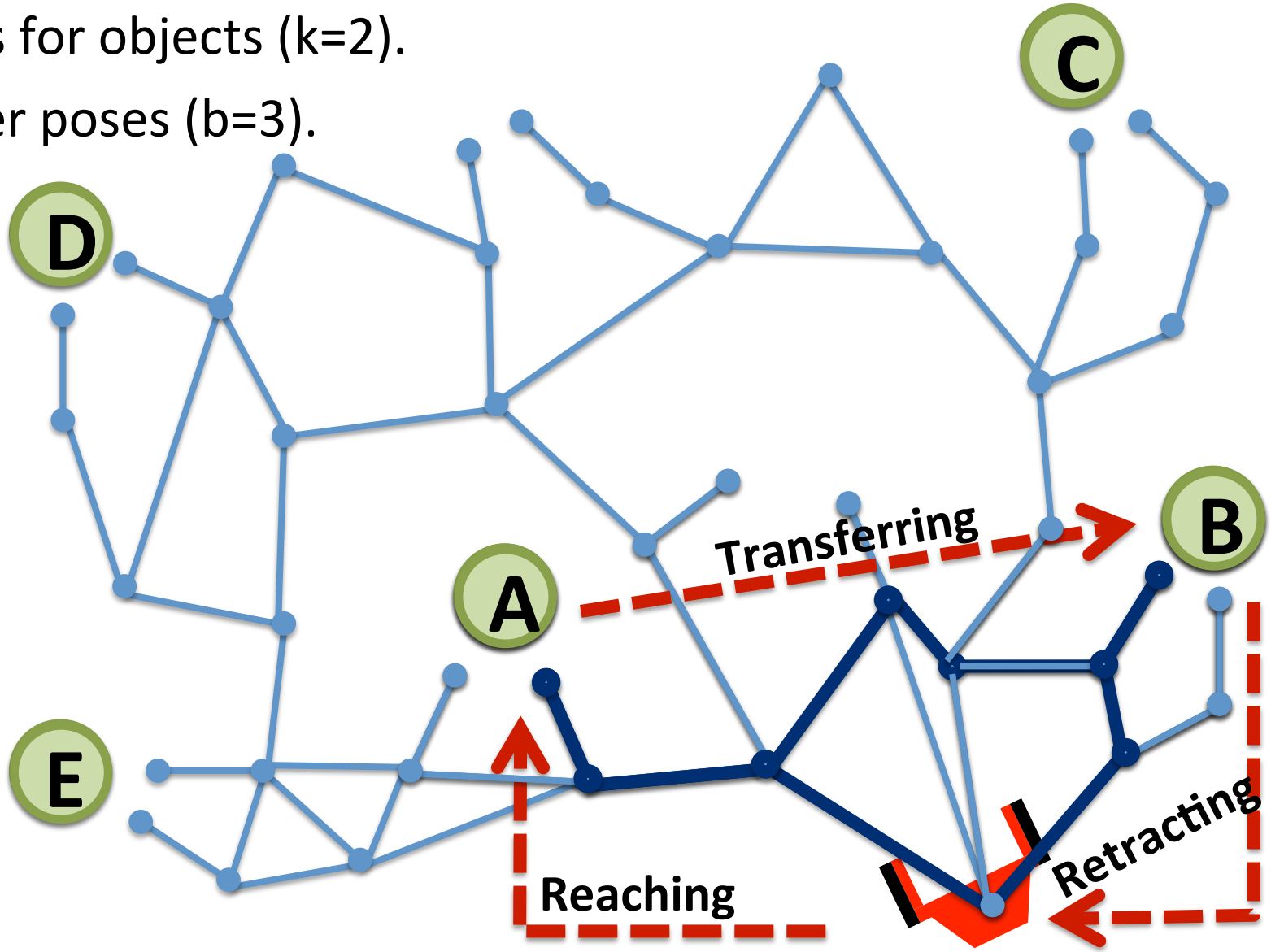
Use of a Manipulation Graph for RPGs

- Build a graph that connects these configurations.

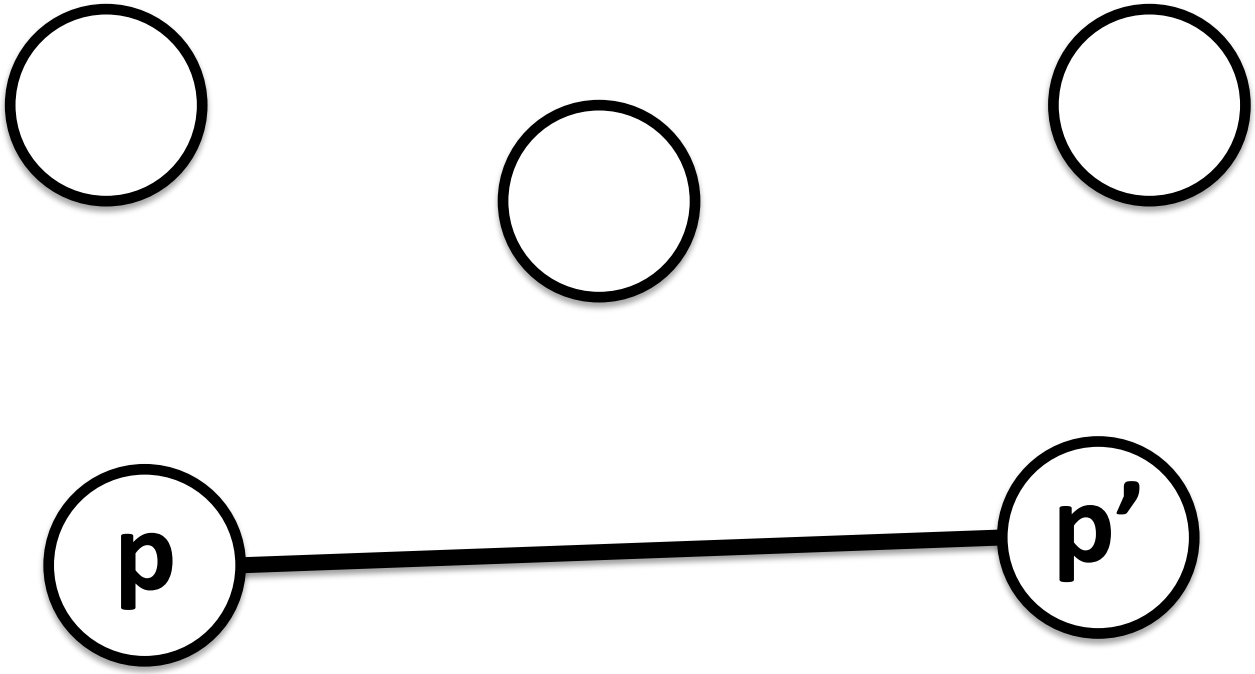


Use of a Manipulation Graph for RPGs

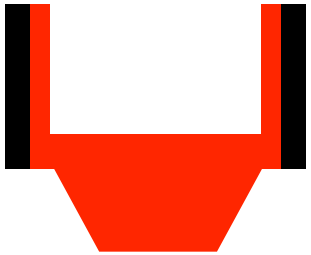
- Select k+b poses.
  - k poses for objects (k=2).
  - b helper poses (b=3).



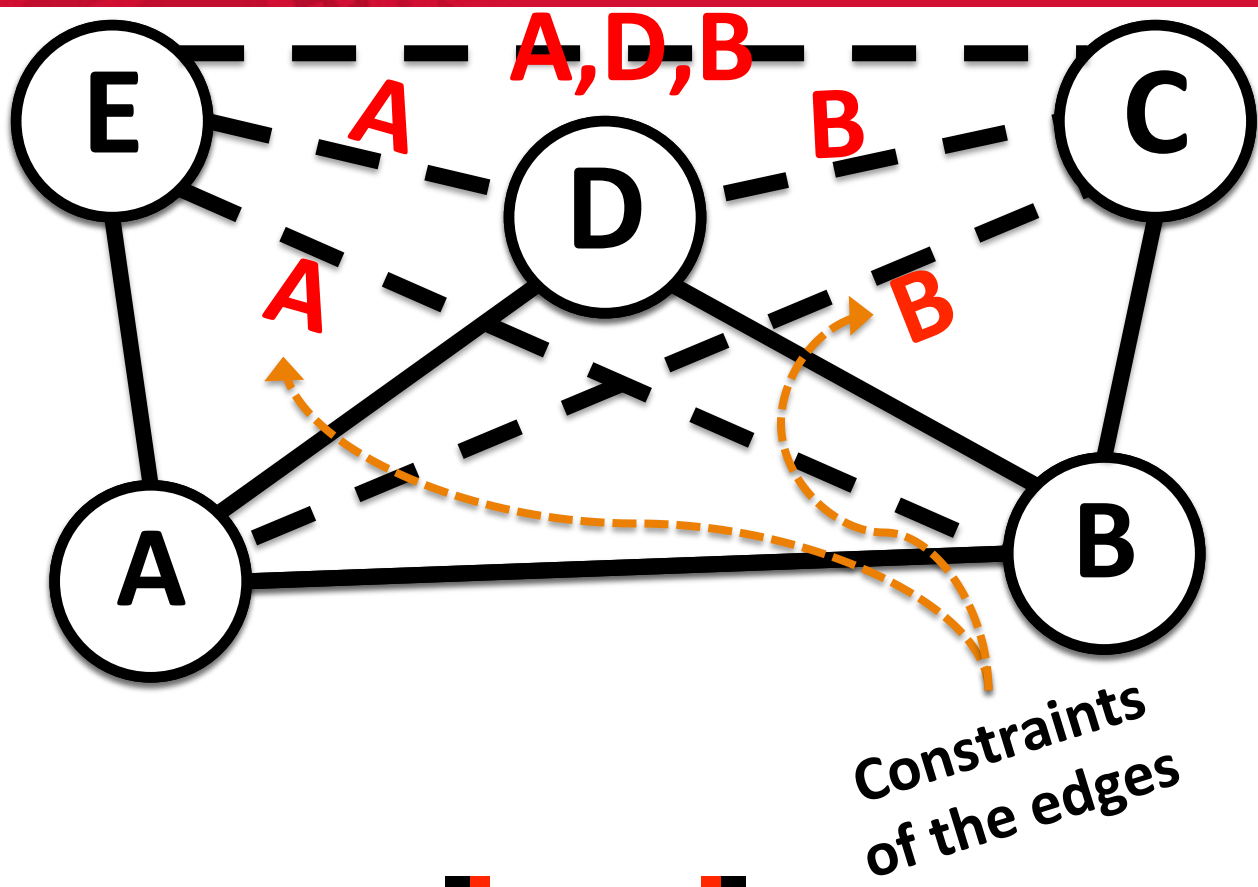
Abstract out Manipulator's motion



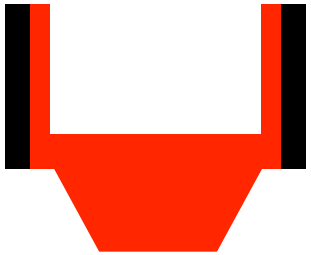
- Rearrangement Pebble Graph (RPG):
  - Nodes: Stable configurations.
  - Edges: Collision free motions of the arm that transfer an object between poses.



Computing the Graph



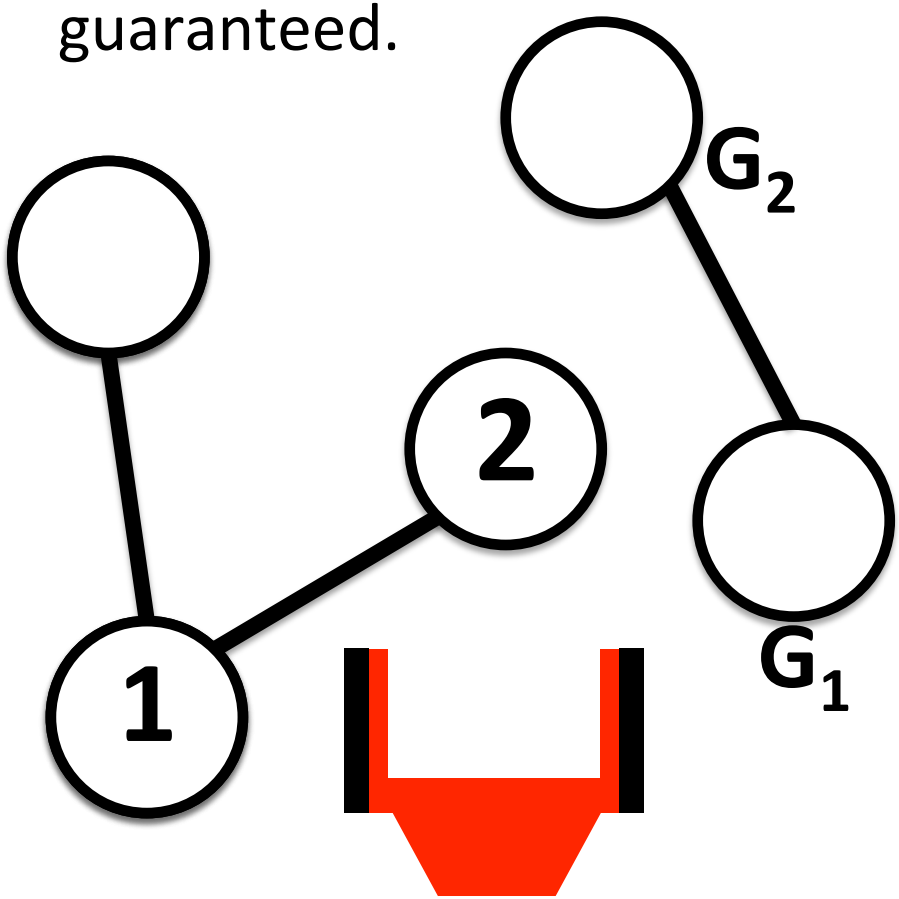
- Rearrangement Pebble Graph (RPG):
  - Nodes: Stable configurations.
  - Edges: Collision-free motions of the arm that transfer an object between poses regardless of object placements.



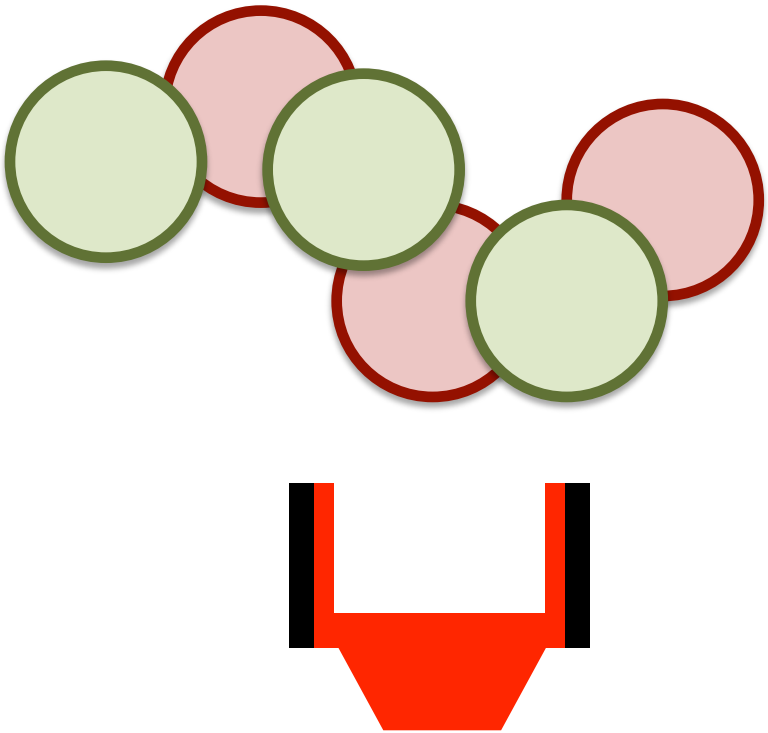
A Single RPG is not Enough!

- If initial/target arrangement on same component: Victory!
- A rearrangement may not be solvable with a single RPG

Issue 1: Connectivity is not guaranteed.



Issue 2: Poses in the initial and target arrangements could be overlapping.

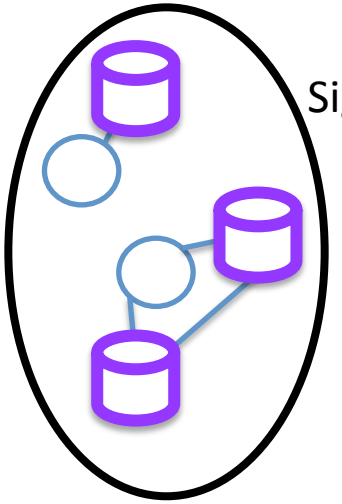


Super-Graph of RPGs

- Motivated by multi-robot motion planning work.
  - *Solovey, Halperin: k-Color Multi-Robot Motion Planning. In: WAFR (2012)*

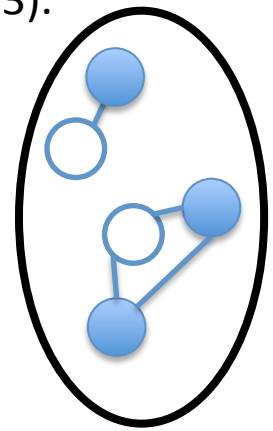
- Build multiple such graphs (RPGs)
  - Within each RPG a discrete solver can be used to achieve all feasible arrangements given constraint-free edges.
  - Allow transitions between graphs

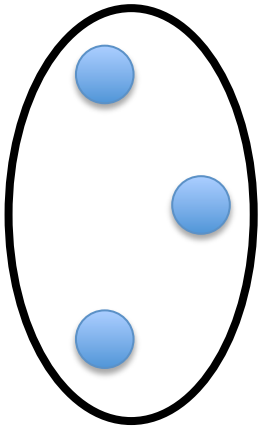
- Build a super-graph:
  - Nodes: an RPG with a signature
  - Edge: If we can transit from one RPG to the other, given their signatures.



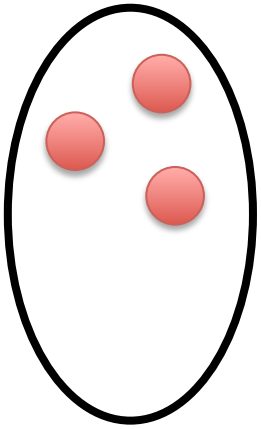
High Level Idea

- k poses for objects (k=3).
- b helper poses (b=2).



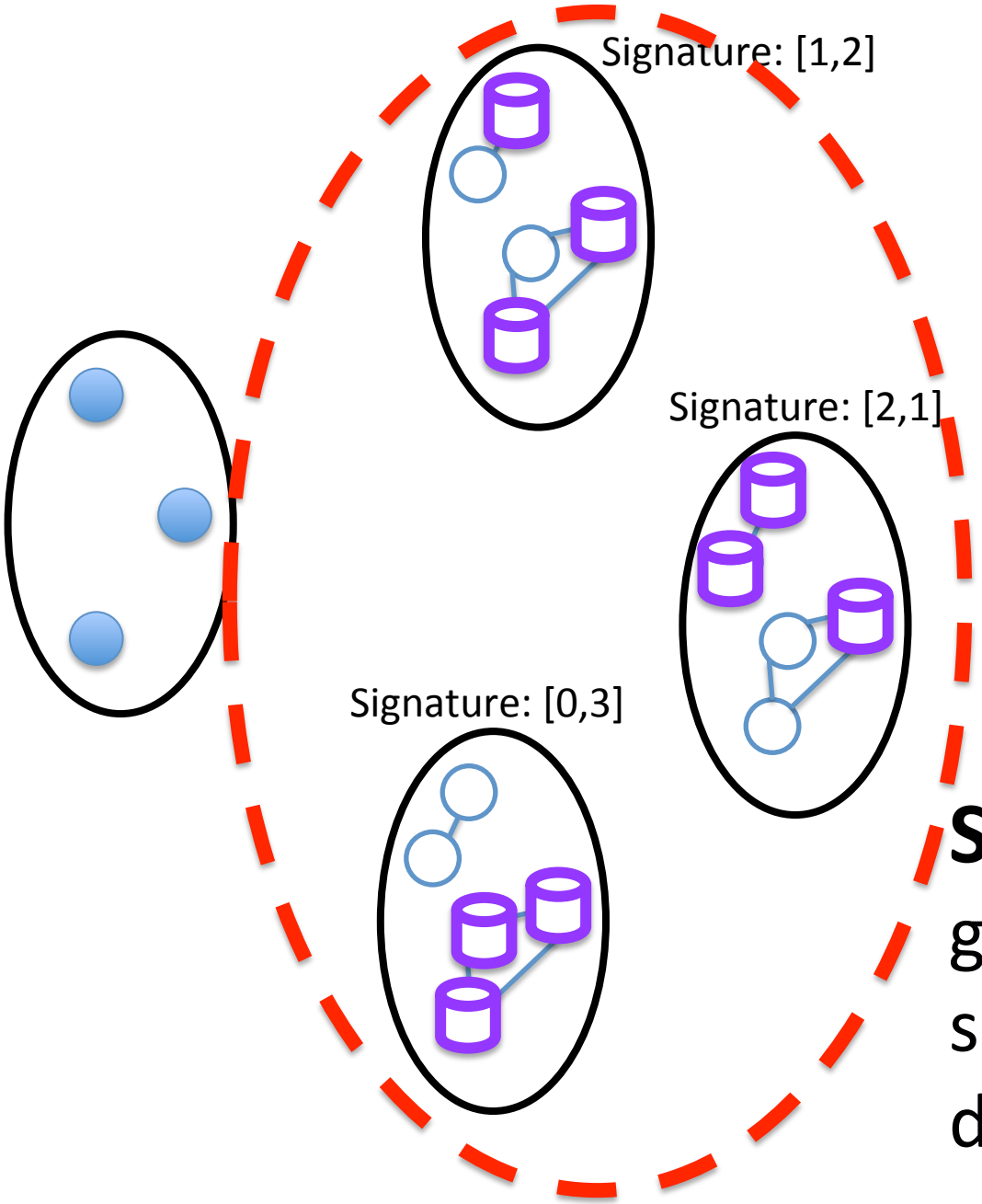


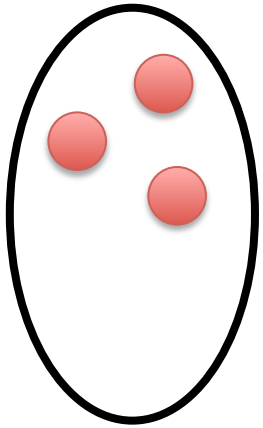
Initial arrangement



Target arrangement

High Level Idea

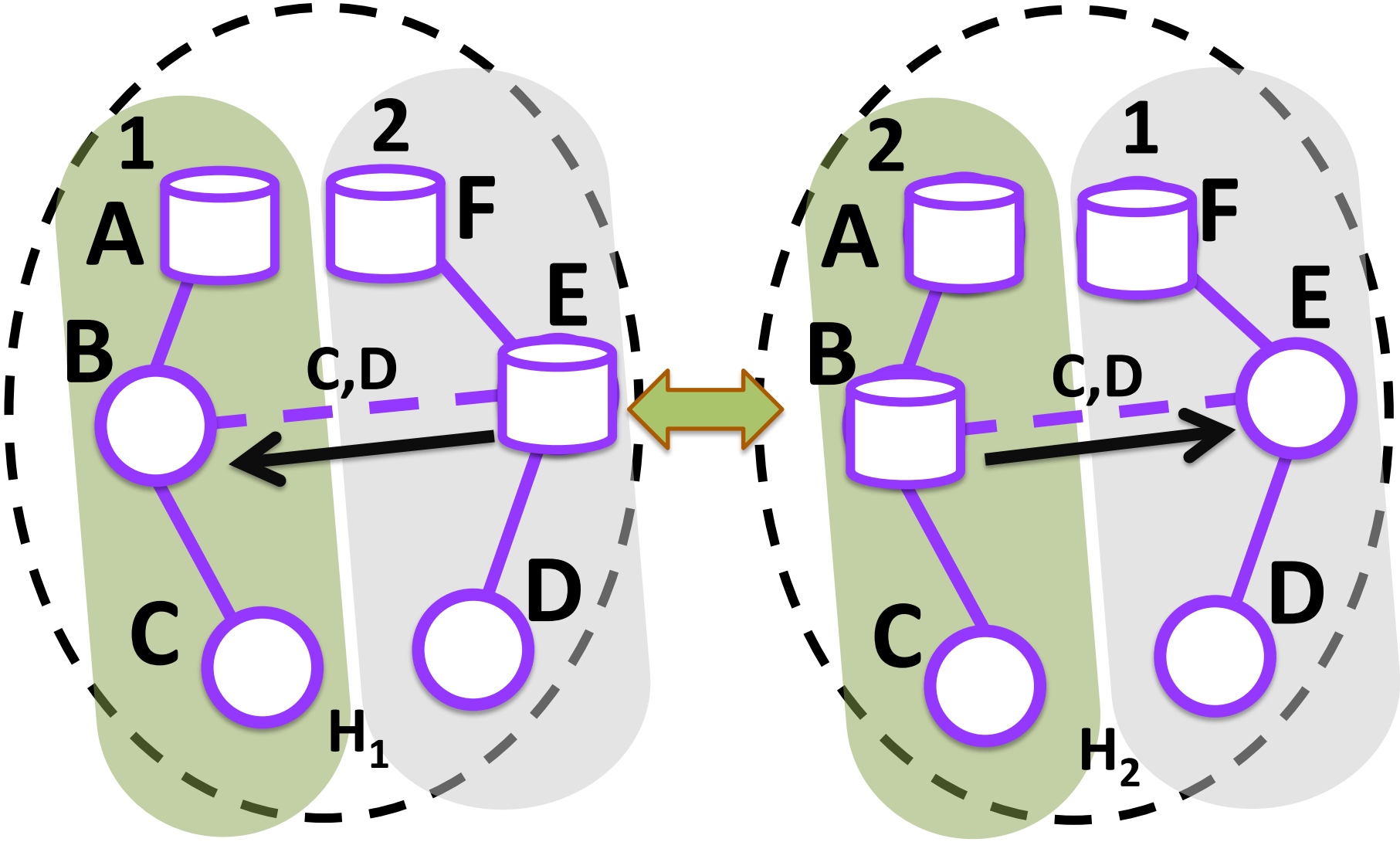




Sibling RPGs: are generated by the same graph but have different signatures.

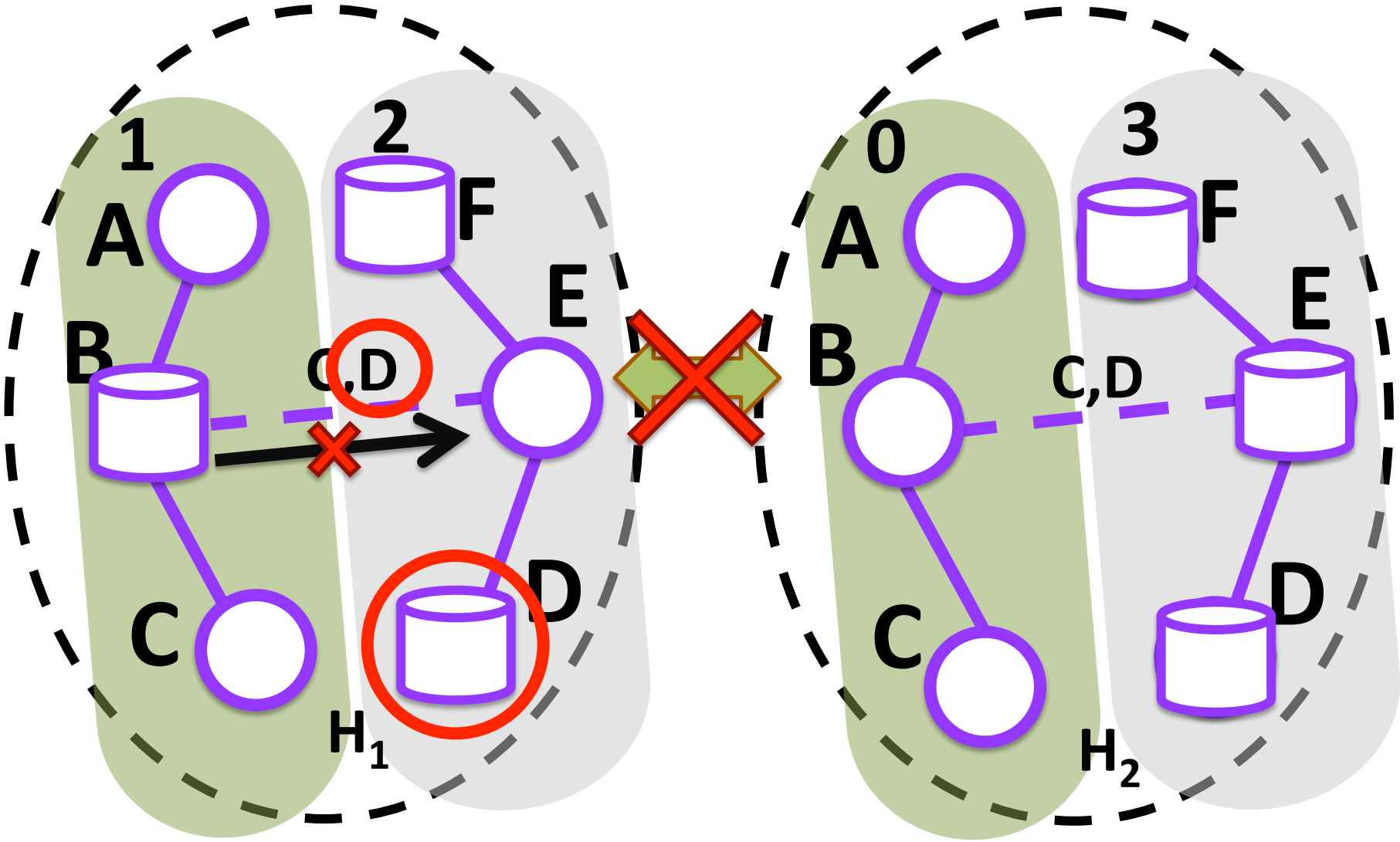
Connect RPGs over Constrained Edges

- Use signatures to connect super-nodes corresponding to the same graph (siblings).

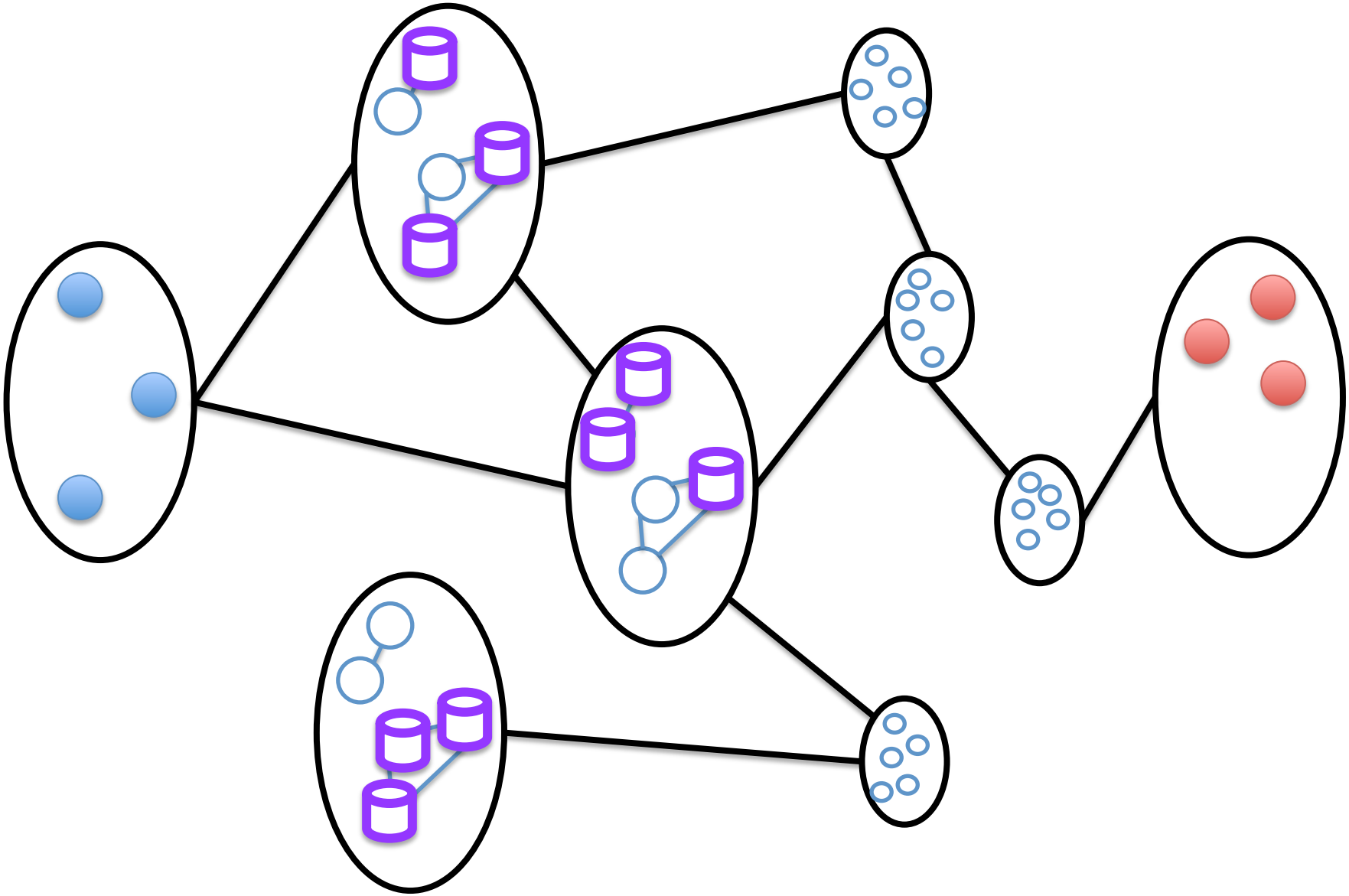


Connect RPGs With Motion

- Siblings that will not be connected because of the signatures.

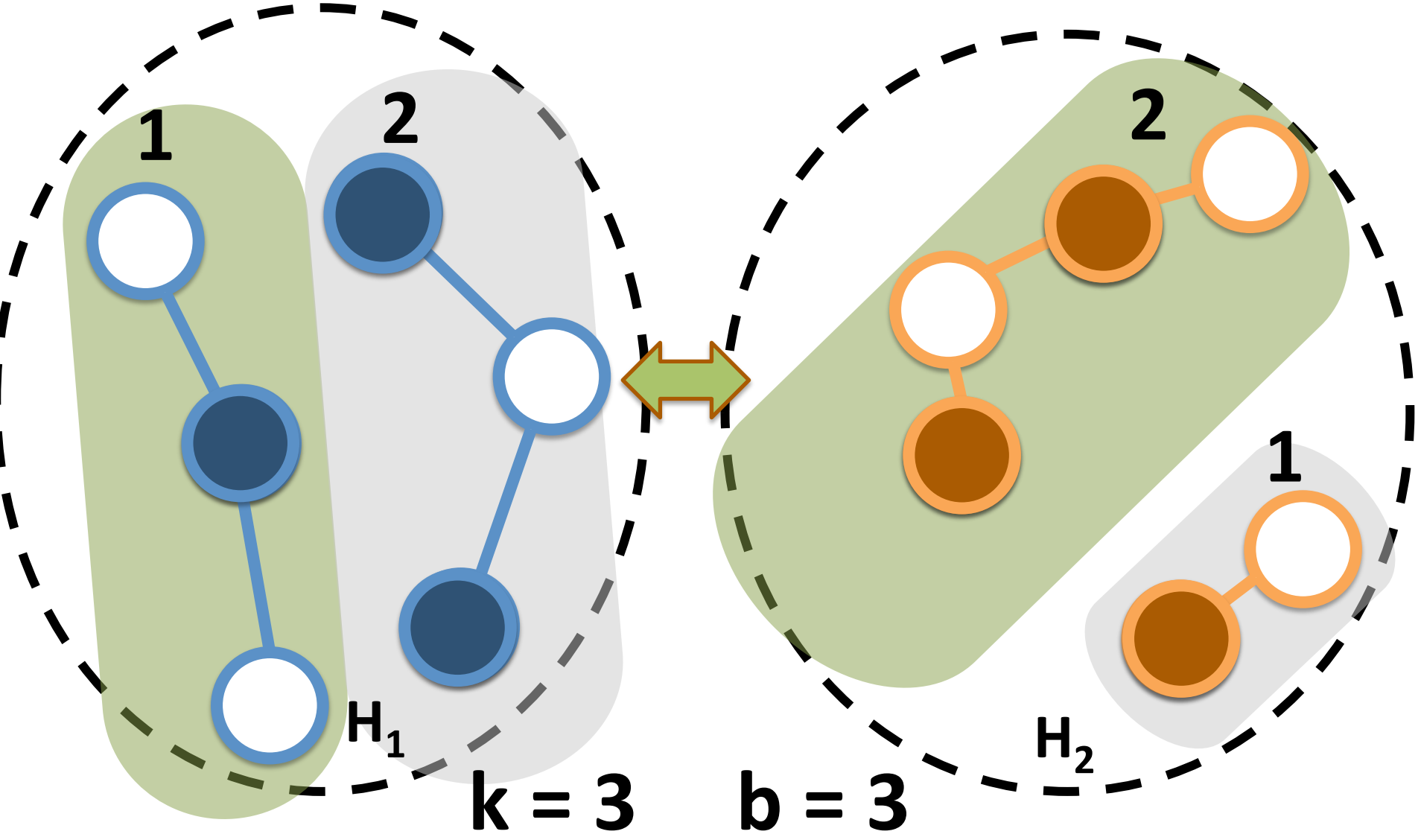


High Level Idea



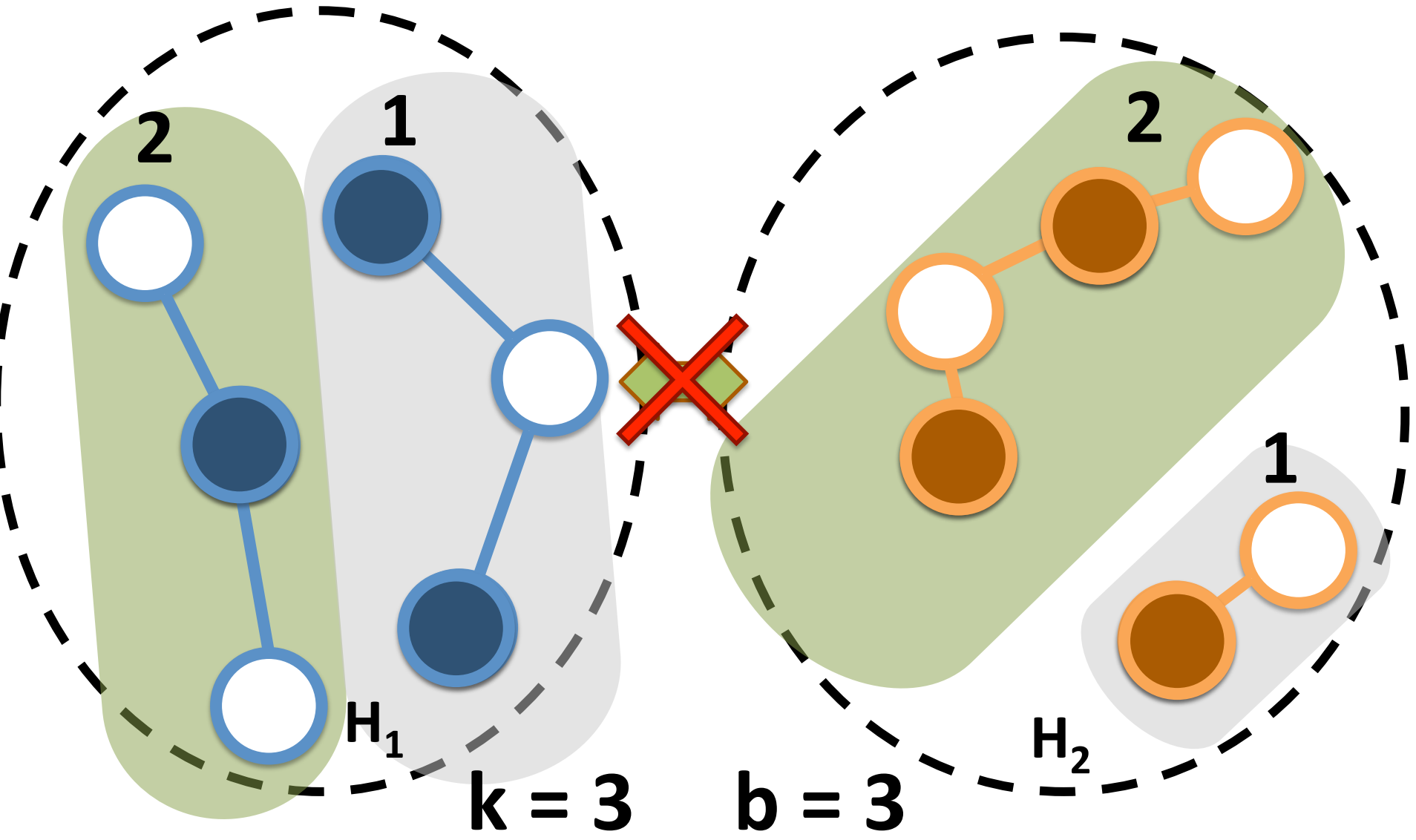
Connect RPGs

- Objects within each connected component can be rearranged, since the edges within an RPG correspond to valid motions.

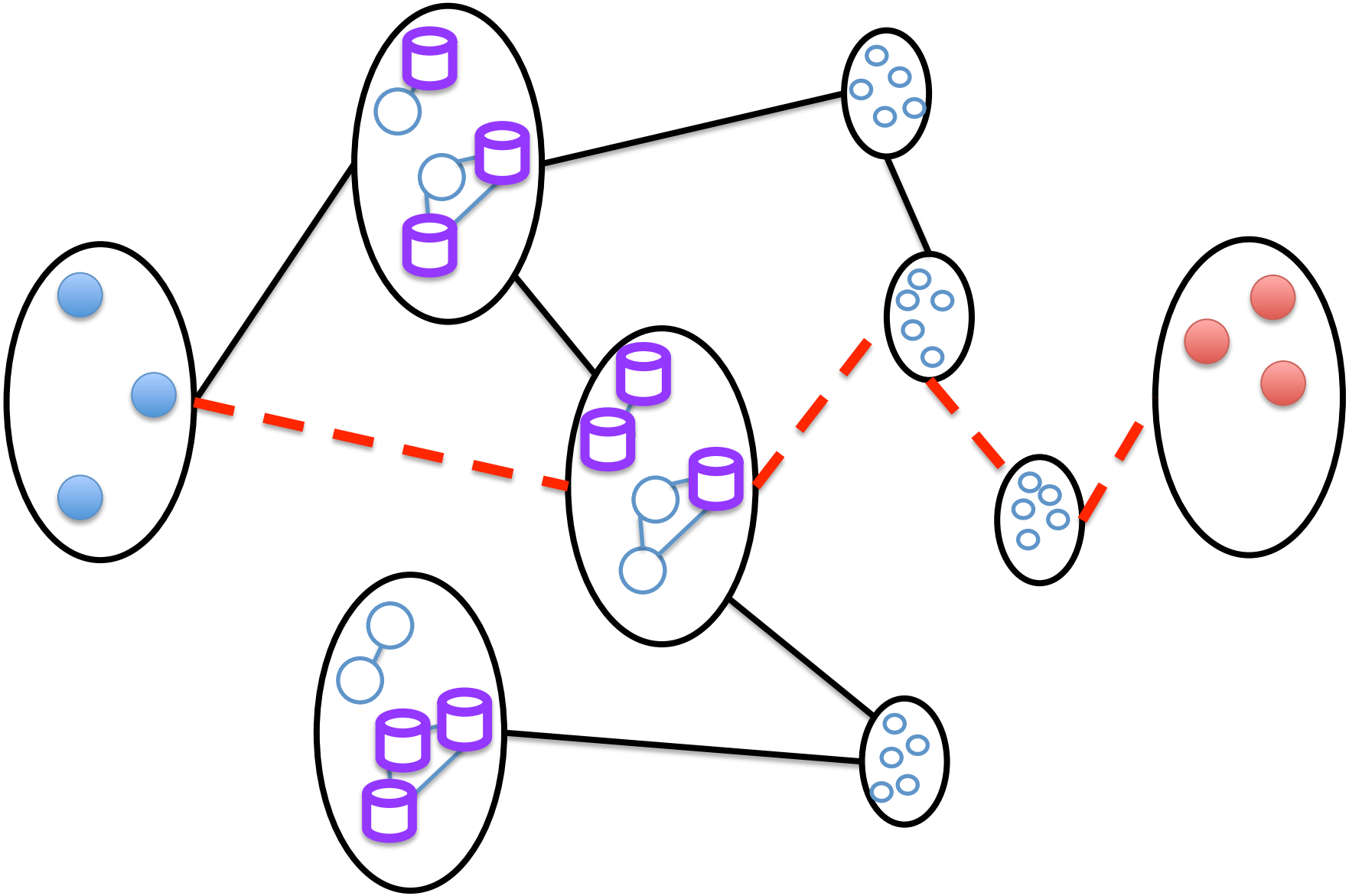


Connect RPGs

- Hyper nodes that will fail to connect because of the signatures.



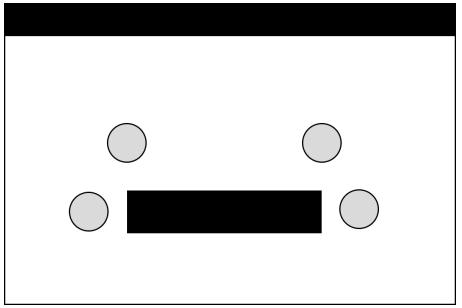
Reconstruct a Path



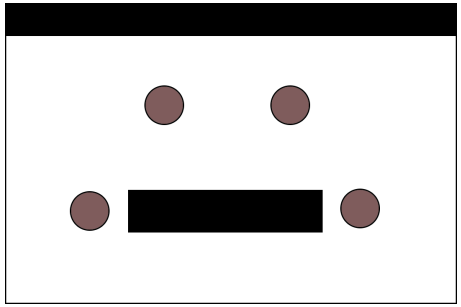
RUTGERS

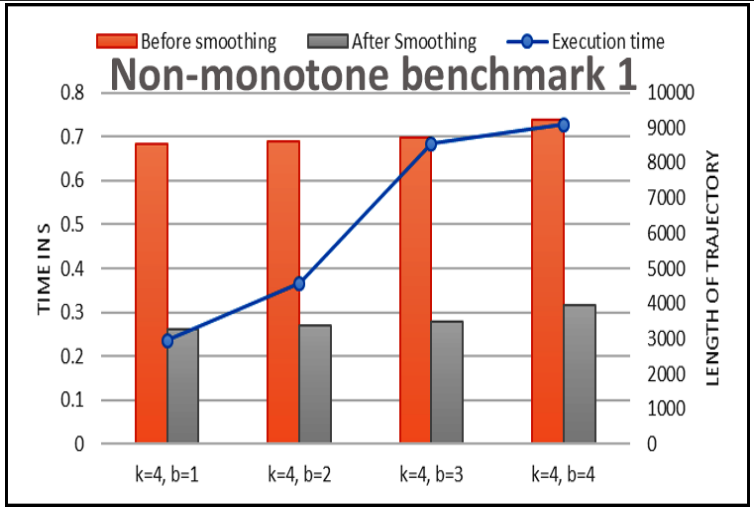
Non-Monotone Benchmark 1

Start



Goal

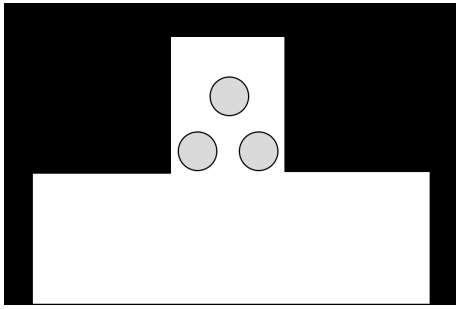




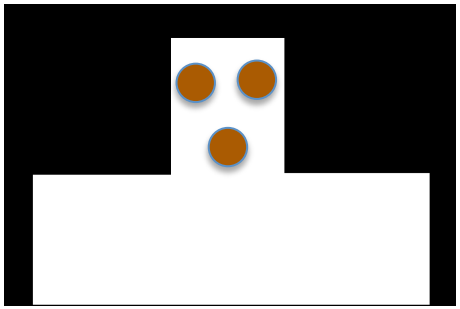
RUTGERS

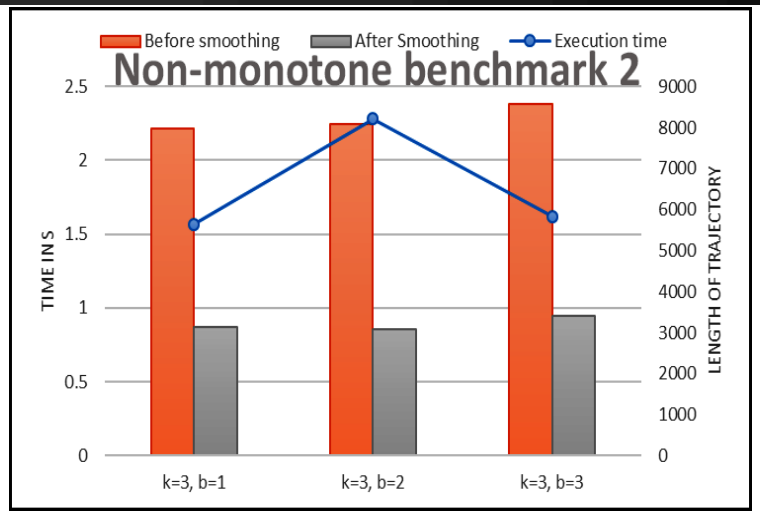
Non-Monotone Benchmark 2

Start

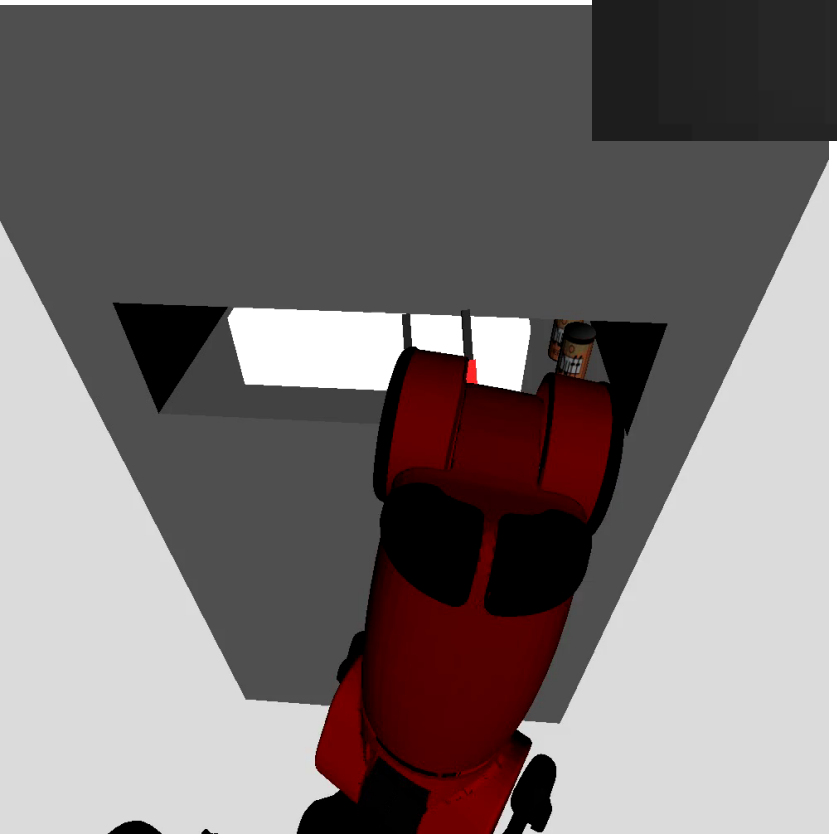


Goal





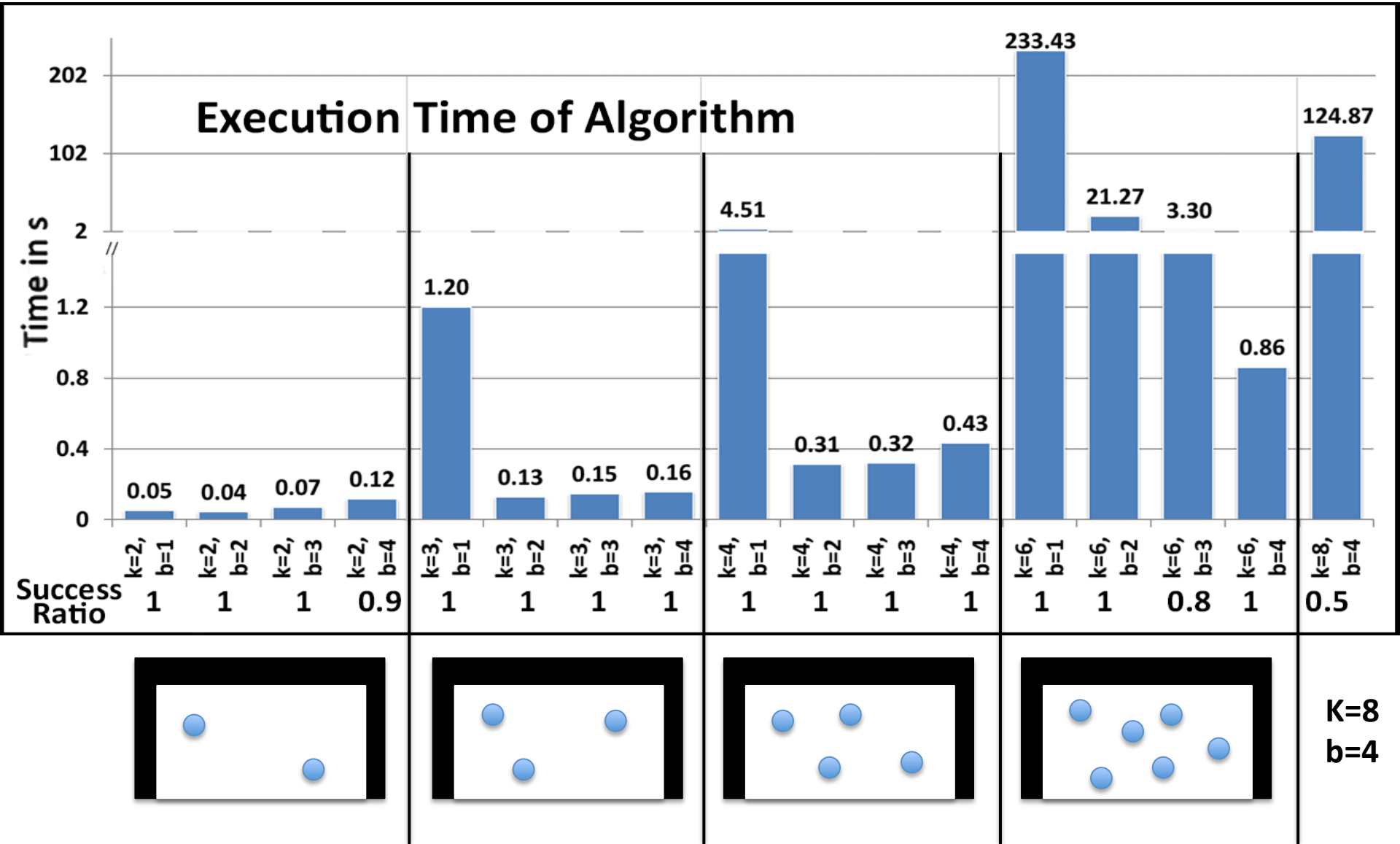
RUTGERS



From Simulation to Baxter

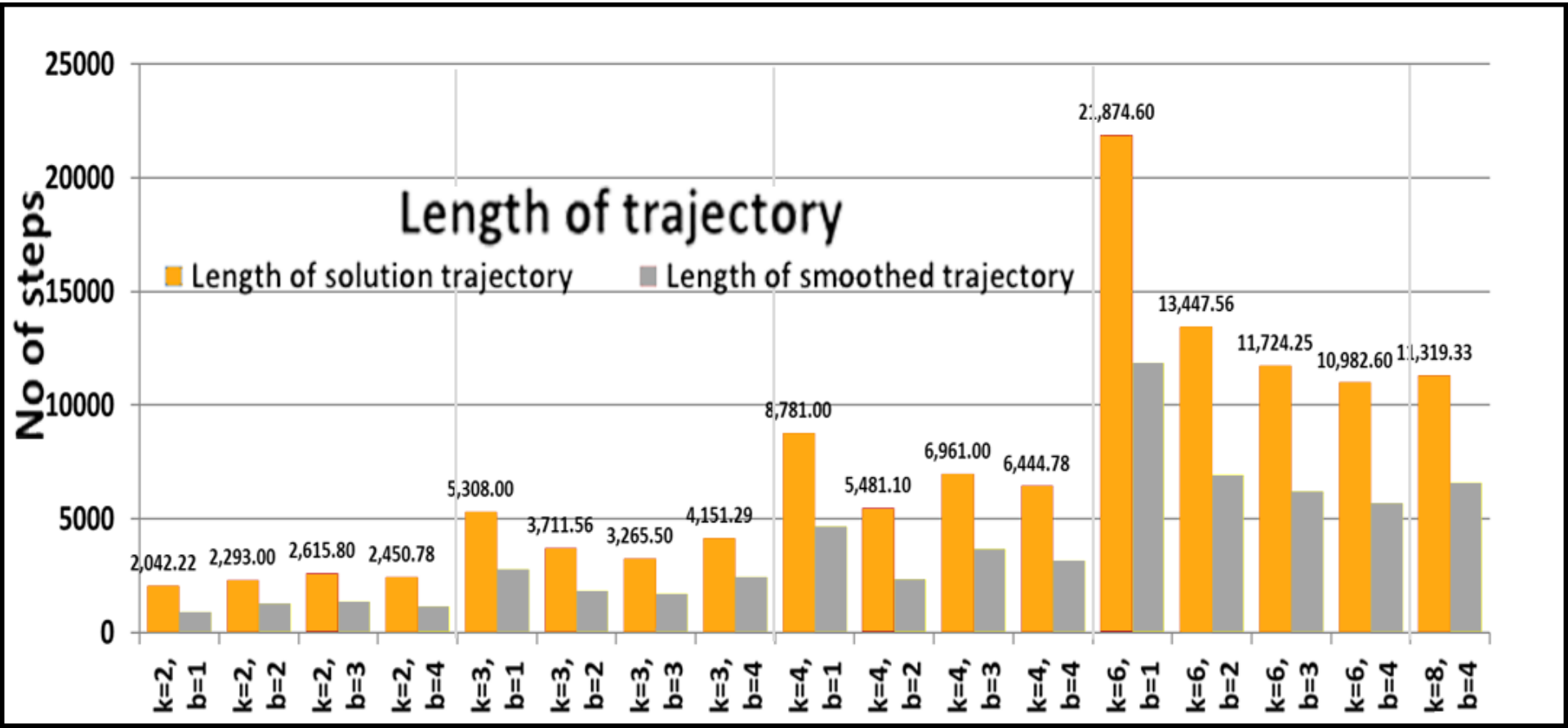
Execution Time

- Randomized placement of poses



Solution Quality

- Smaller values of b reduces the exploration rate, which results in longer trajectories.
- Smoothing shortens, on average, the trajectory by 48%



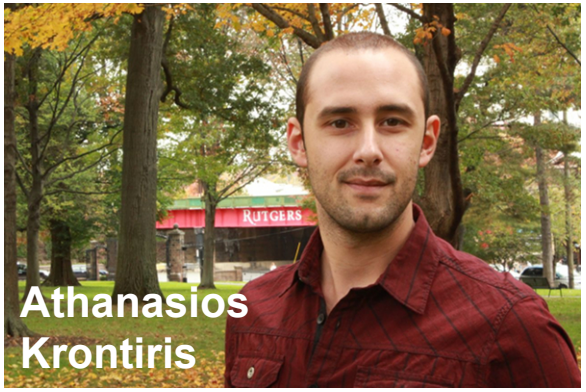
Discussion

Summary:

- Given a precomputed manipulation graph for a single object
  - We can solve rearrangement problems using RPGs in seconds
- Probabilistic completeness can be argued for a class of instances
  - We must be able to retract the arm to the safe configuration

Future steps:

- Labeled case: Objects with different geometries/labels
- Consider ways to increase connectivity of super-graph
- Improve path quality
- Allow general grasps and resting poses in implementation
- Dual-arm manipulation



Thank you for your attention!

Our research efforts have been supported by:

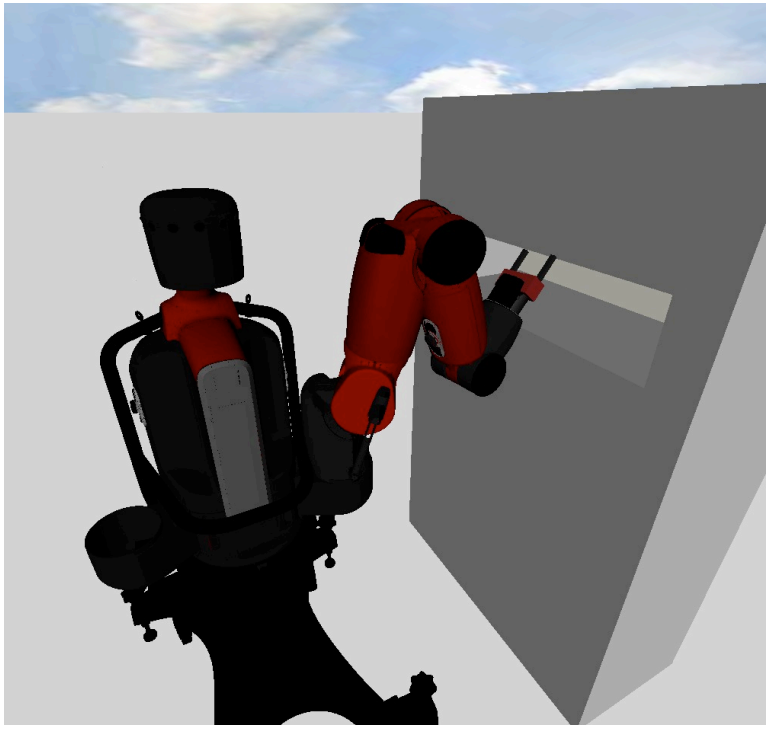
- the National Science Foundation (NSF),
- the National Aeronautics and Space Administration (NASA),
- the Department of Defense (ONR & DoD TARDEC).

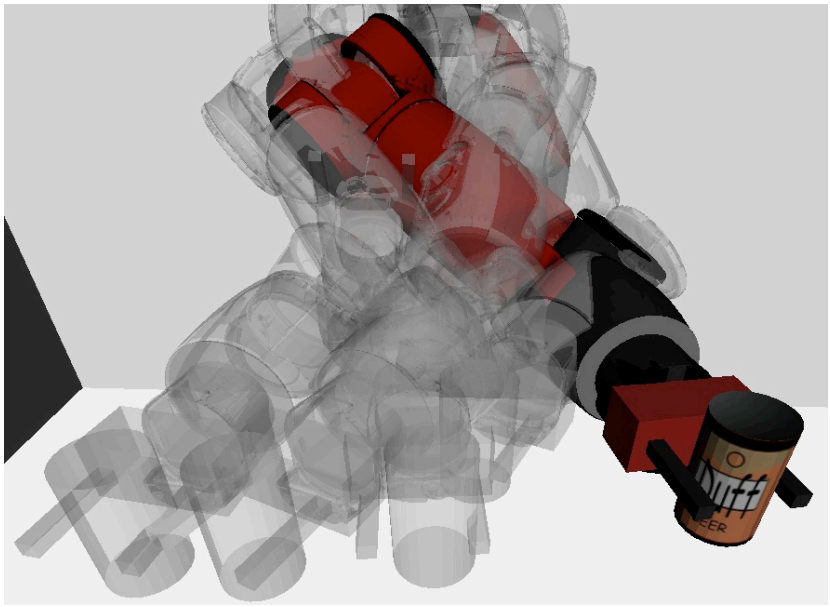
Backup Slides

Useful Notions

- Safe Arm Configuration.

  A collision-free configuration that the arm is safe to return regardless the placement of objects.





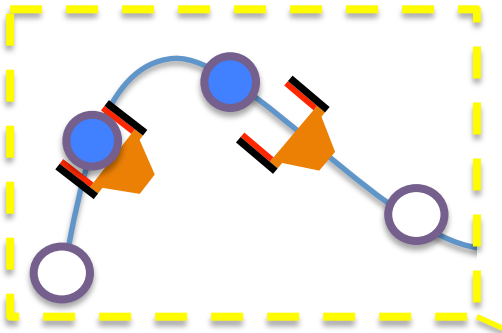
- Stable Object Configuration

  A placement of object so they can stay at rest and can be grasped by the robot

Example

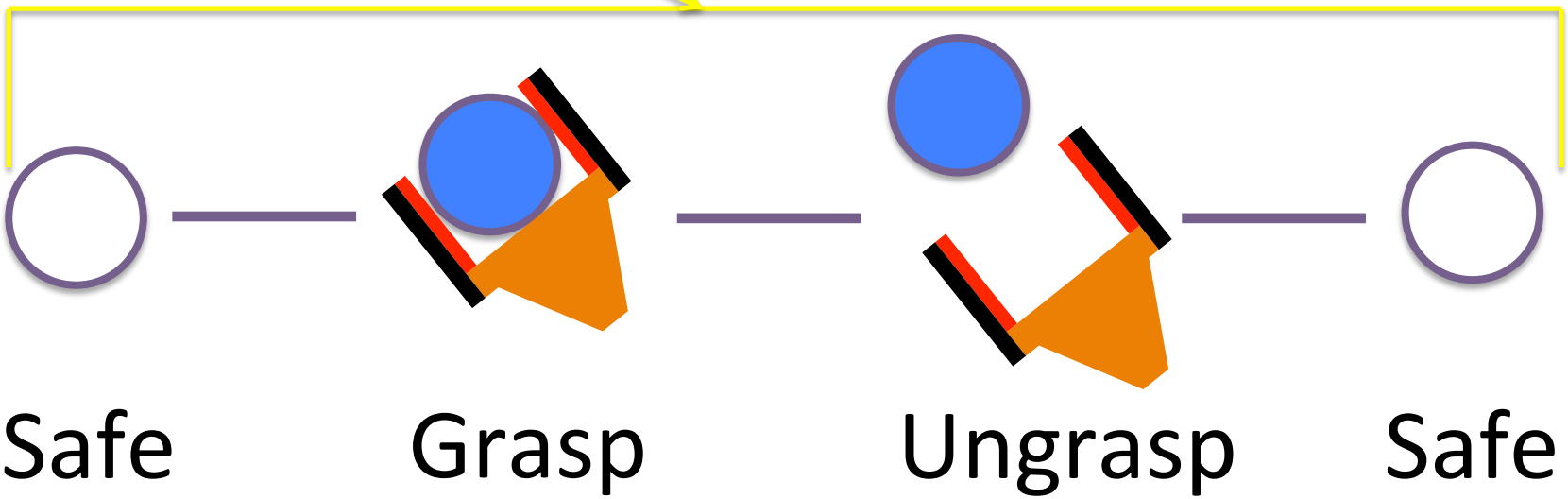
Smoothing

Phase 0: Trajectory Decomposition

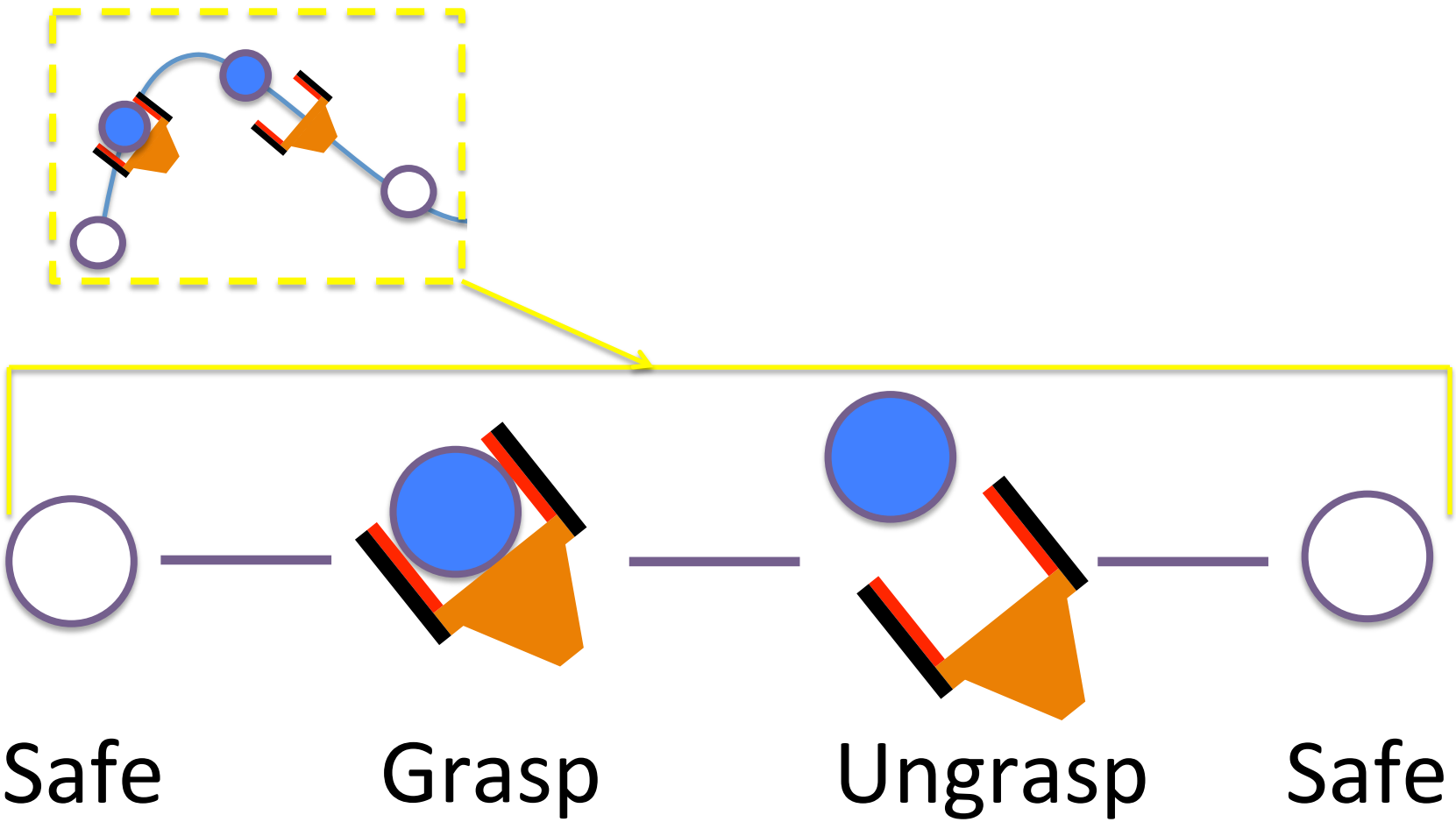


The solution trajectory can be decomposed into 3 separate categories of states

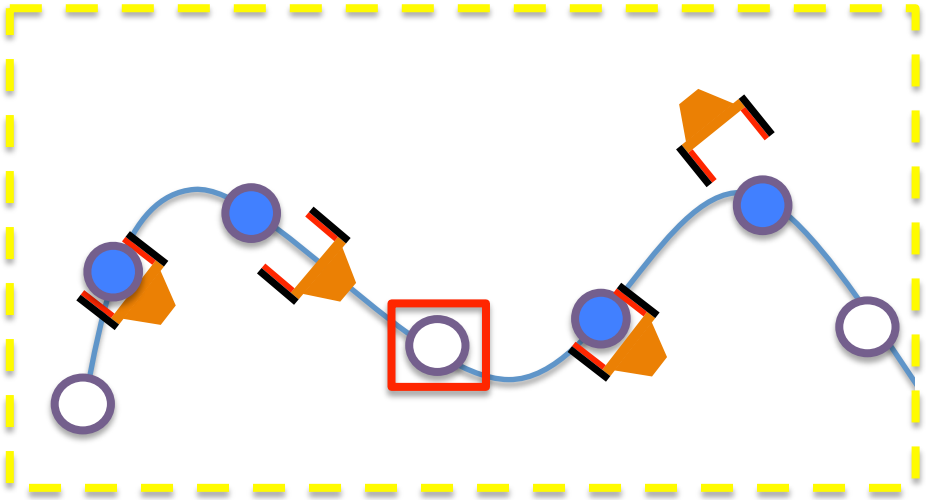
1. Safe State
2. Grasp States
3. Ungrasp States



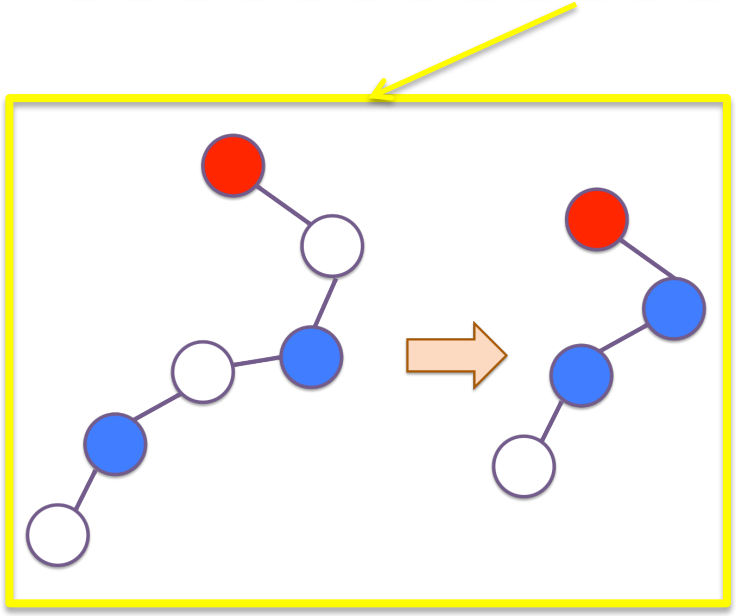
Phase 1: Consecutive Identical Grasps



Phase 1: Consecutive Identical Grasps

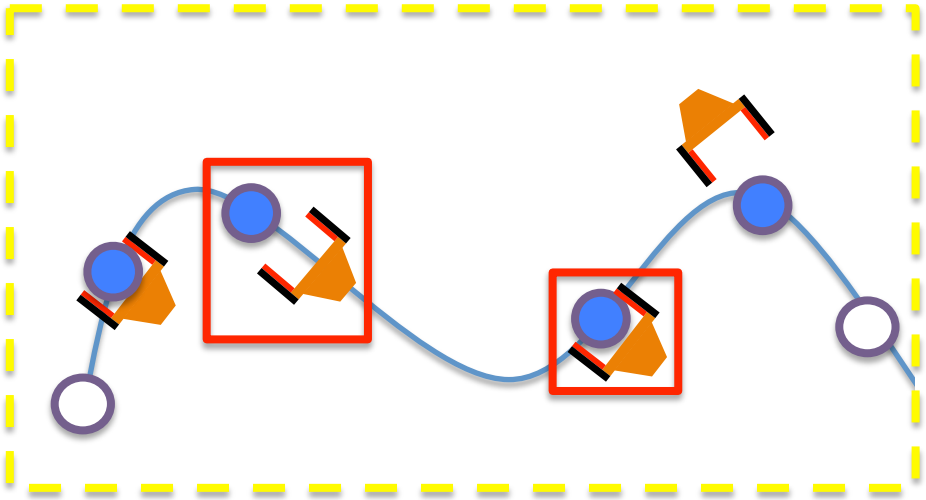


We identify parts of the trajectory where the robot grasps the same object consecutively.



Any intermediate safe states are removed

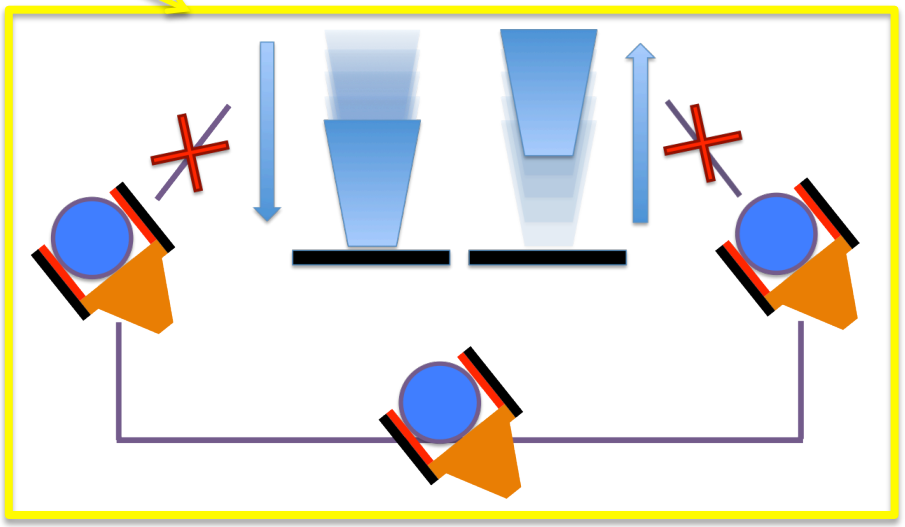
Phase 1: Consecutive Identical Grasps



We identify parts of the trajectory where the robot grasps the same object consecutively.

Next, we identify redundant movements in this sequence.

For example, the robot dropping and picking up the same object before moving it again.

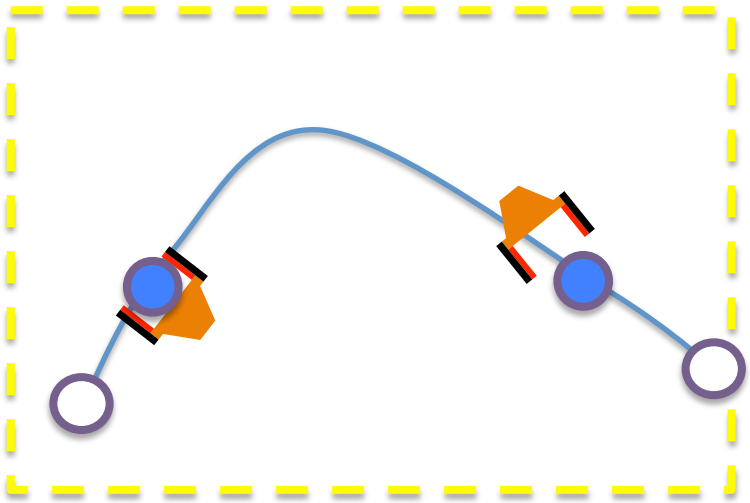


Phase 2: Trajectory Smoothing

We now can apply standard trajectory smoothing, given a sequence of states from one safe state to another safe state.

1. Check pairs of states (s1, s2)
2. Compute the shortest trajectory between them
3. If this trajectory is collision free, replace the original trajectory with this new one.

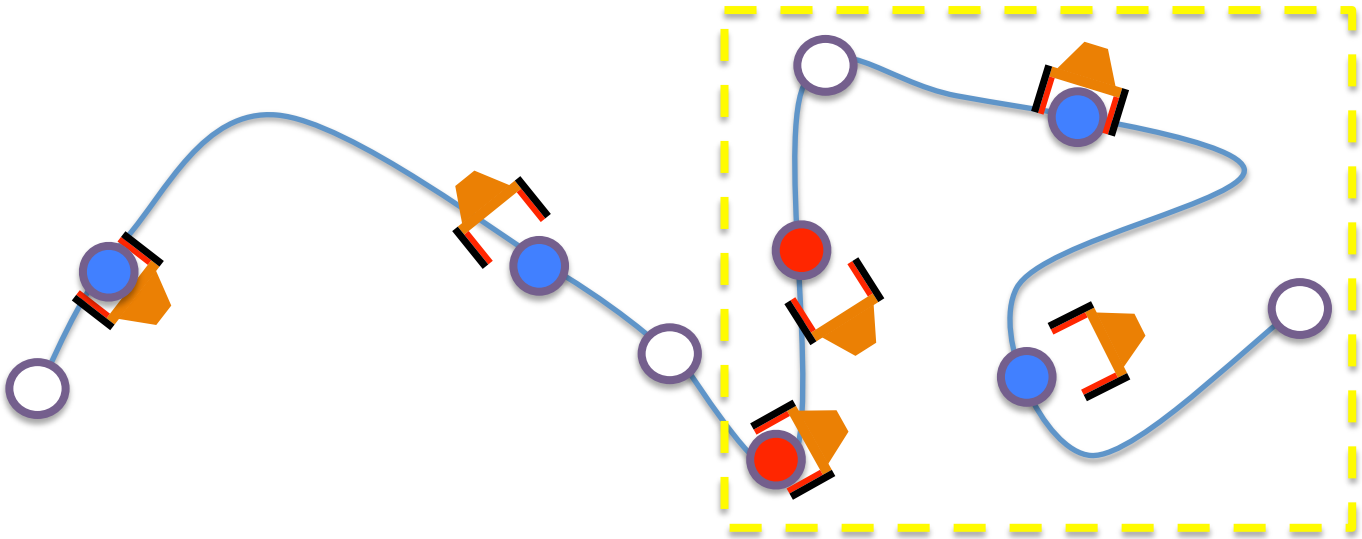
Phase 2: Trajectory Smoothing

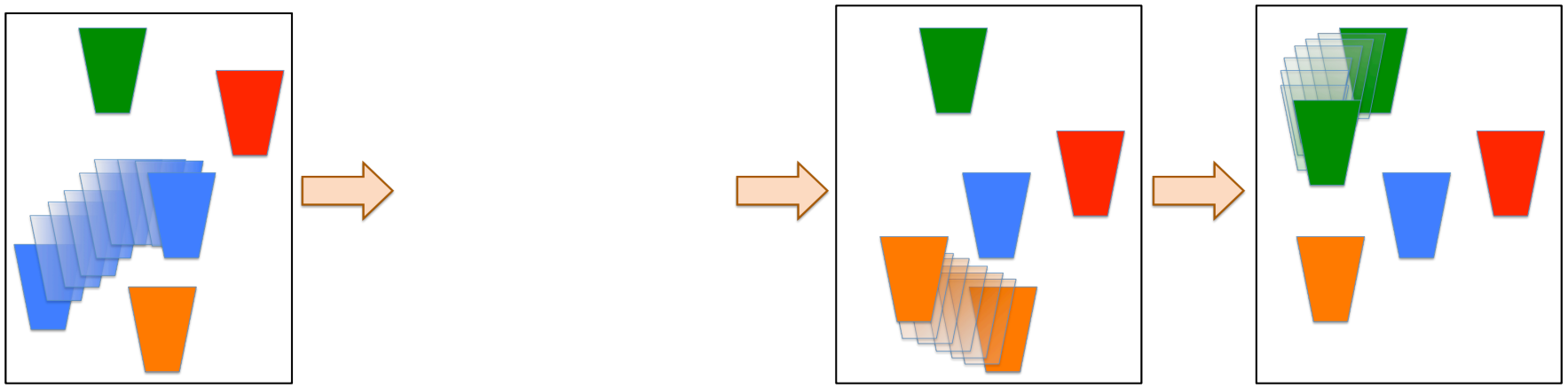


We now can apply standard trajectory smoothing, given a sequence of states from one safe state to another safe state.

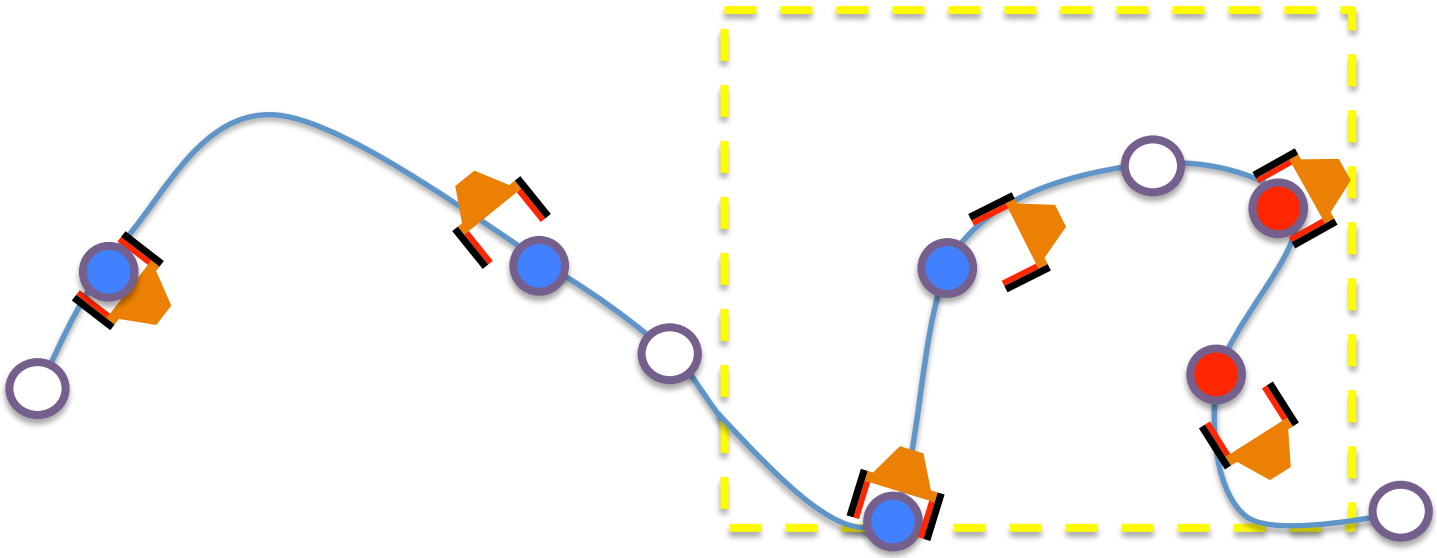
1. Check pairs of states (s1, s2)
2. Compute the shortest trajectory between them
3. If this trajectory is collision free, replace the original trajectory with this new one.

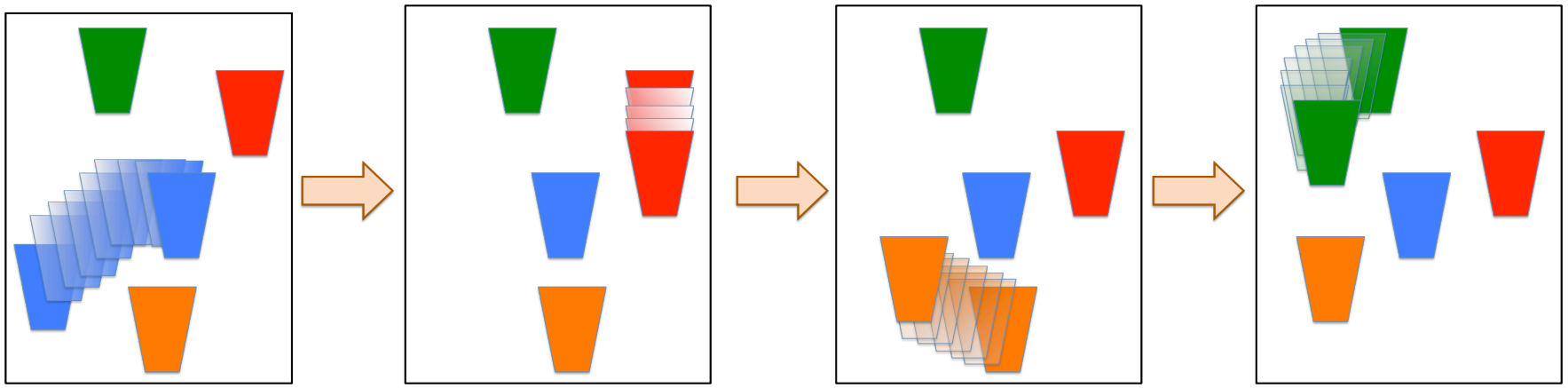
Phase 3: Maximize Consecutive Grasps



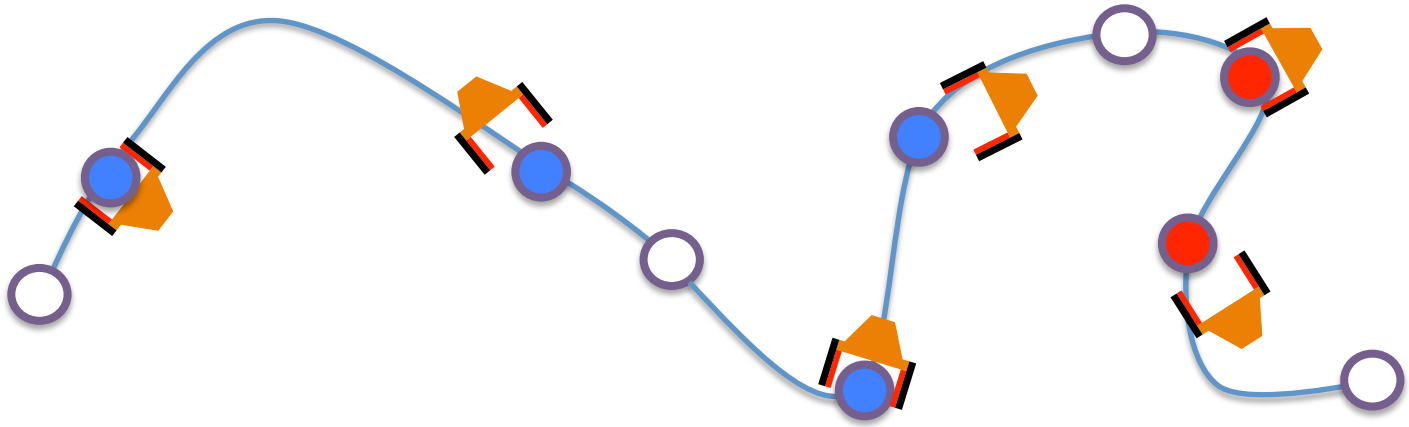


Phase 3: Maximize Consecutive Grasps





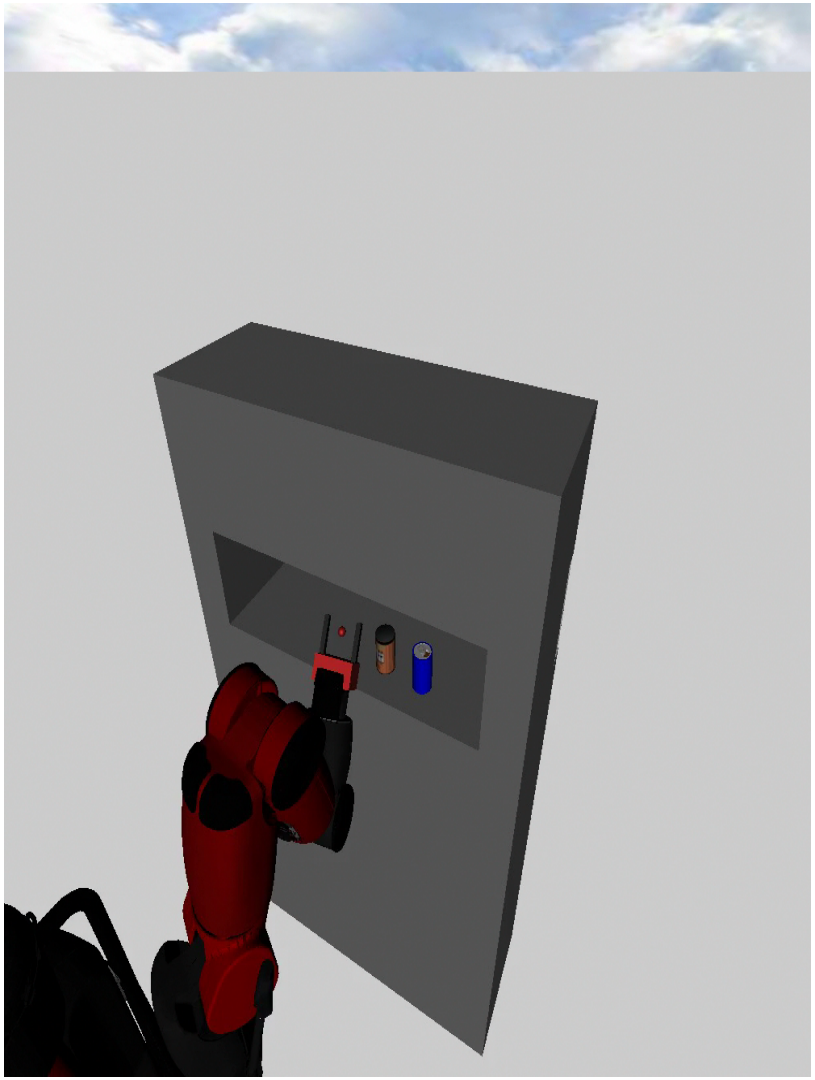
Repeat Everything?



- Individual phases, such as the standard smoothing, can be applied.
- Or, the entire smoothing process could also be repeated.
- This has the potential of yielding a new, shorter solution.
- In practice, one iteration of the smoothing process will provide the largest reduction in the solution trajectory.

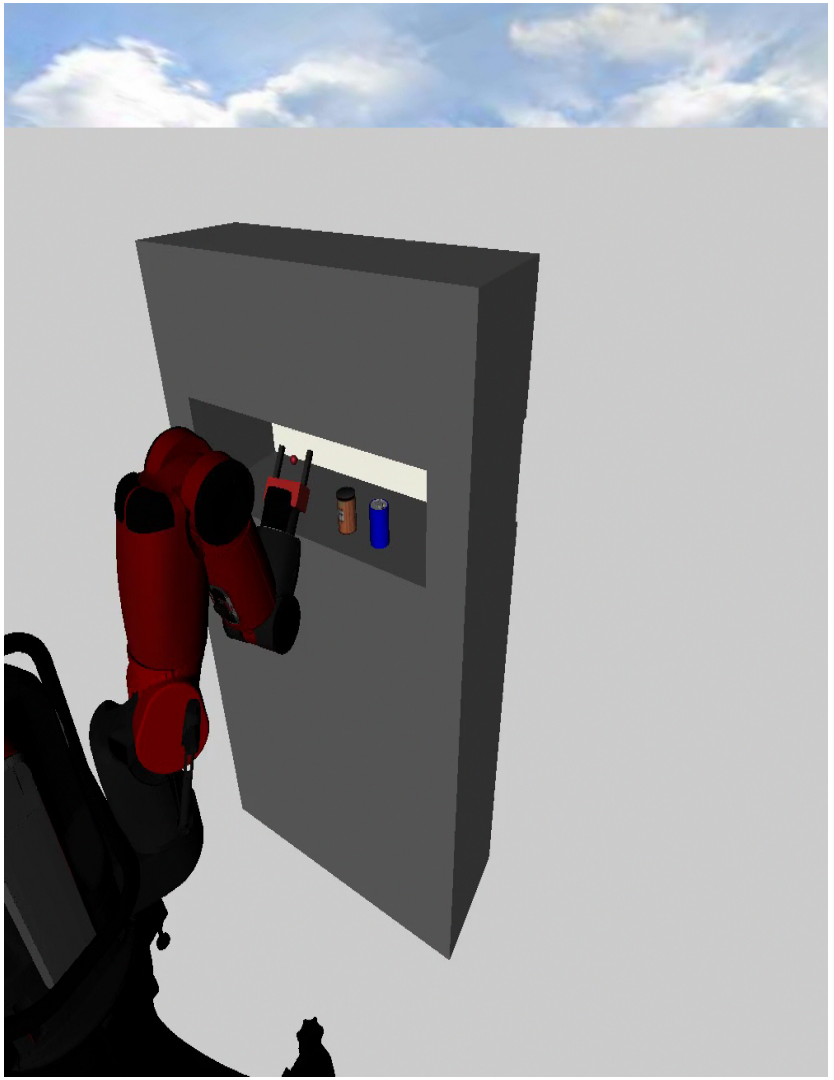
Smoothing

WITHOUT SMOOTHING



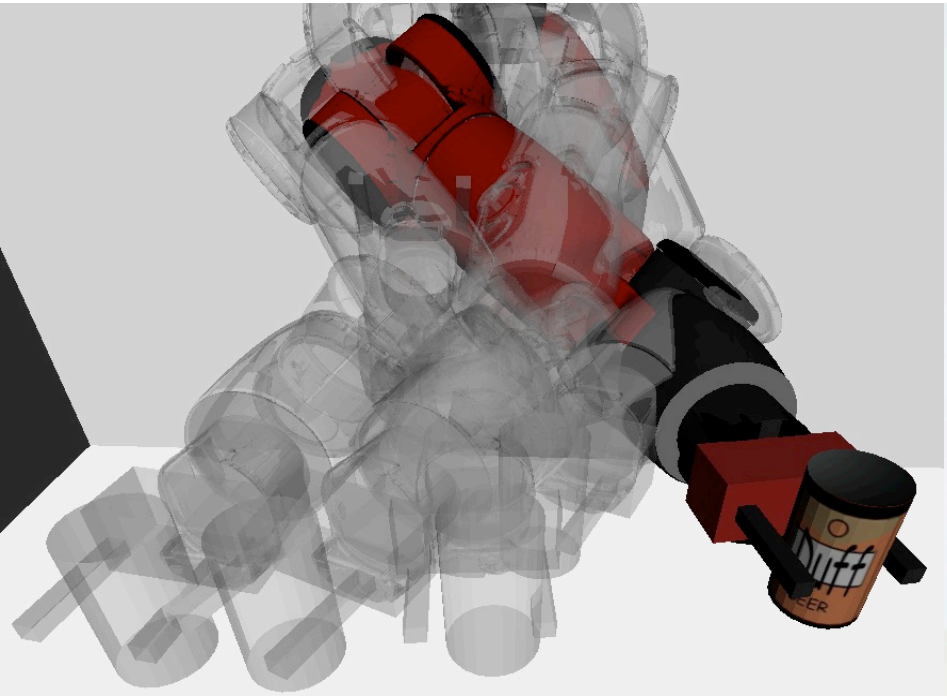
VS

WITH SMOOTHING



Experimental Setup

- The experiments were conducted both in simulation and on a Baxter robot (7 DOF).



- We evaluate the scalability of the approach, and also showcase its ability to solve non-monotone problems.

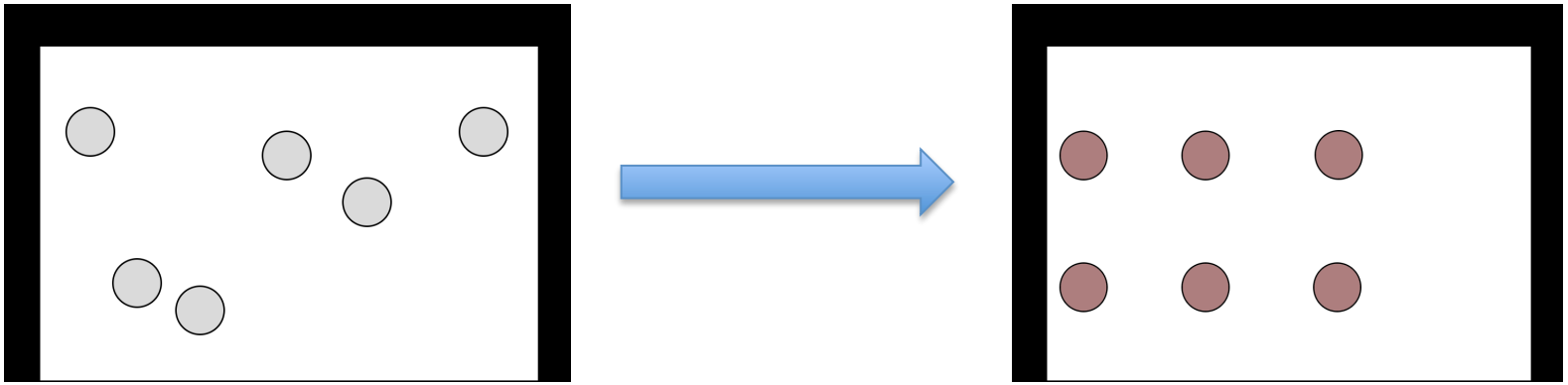
Experimental Setup

- Here are the metrics we measured…

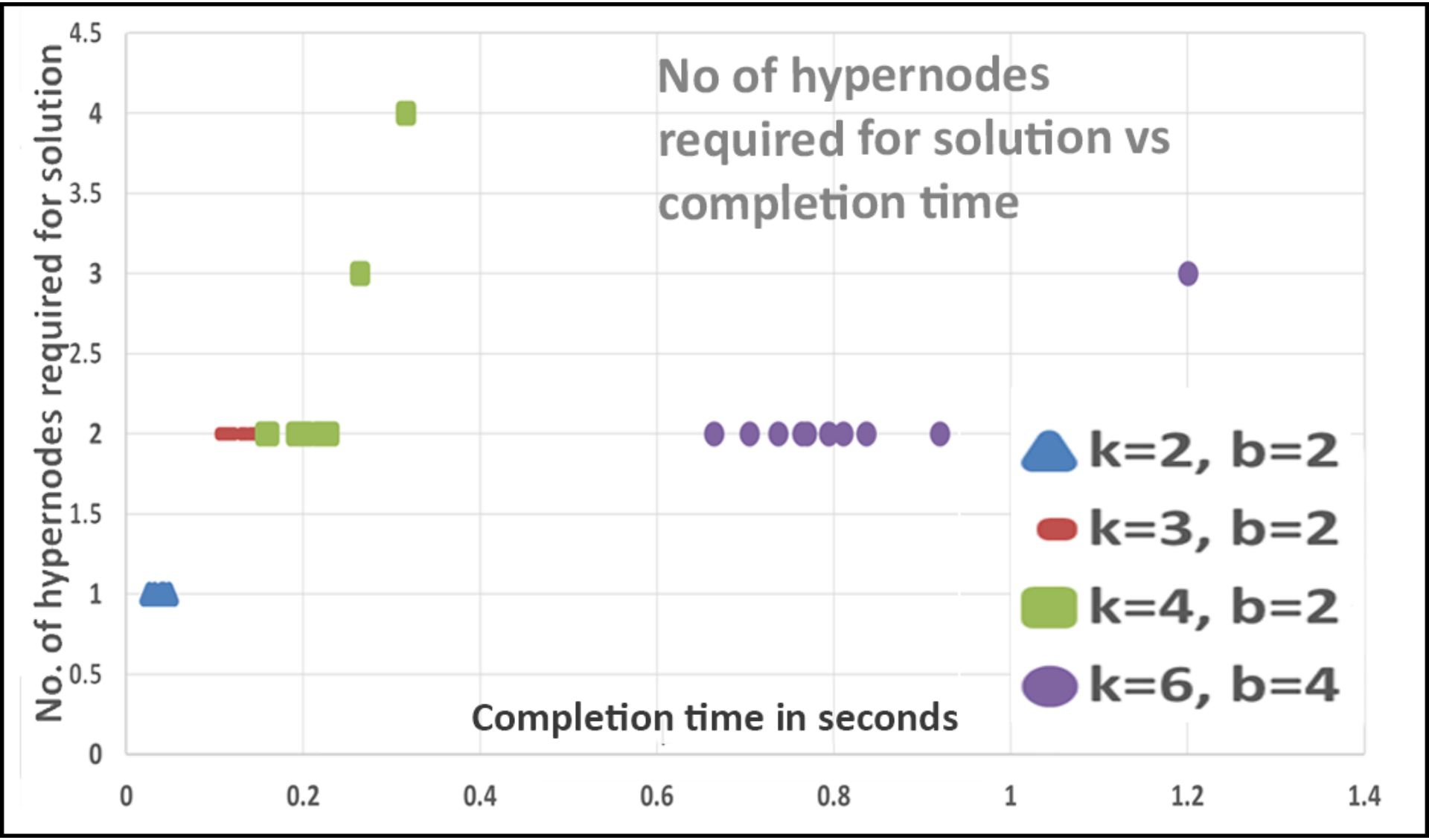
1. **Execution Time**: How fast were we able to solve the problem, and with what success rate?

2. **Solution Quality:** How short were the trajectories?

3. **Connectivity:** How many edges connect our hypernodes?

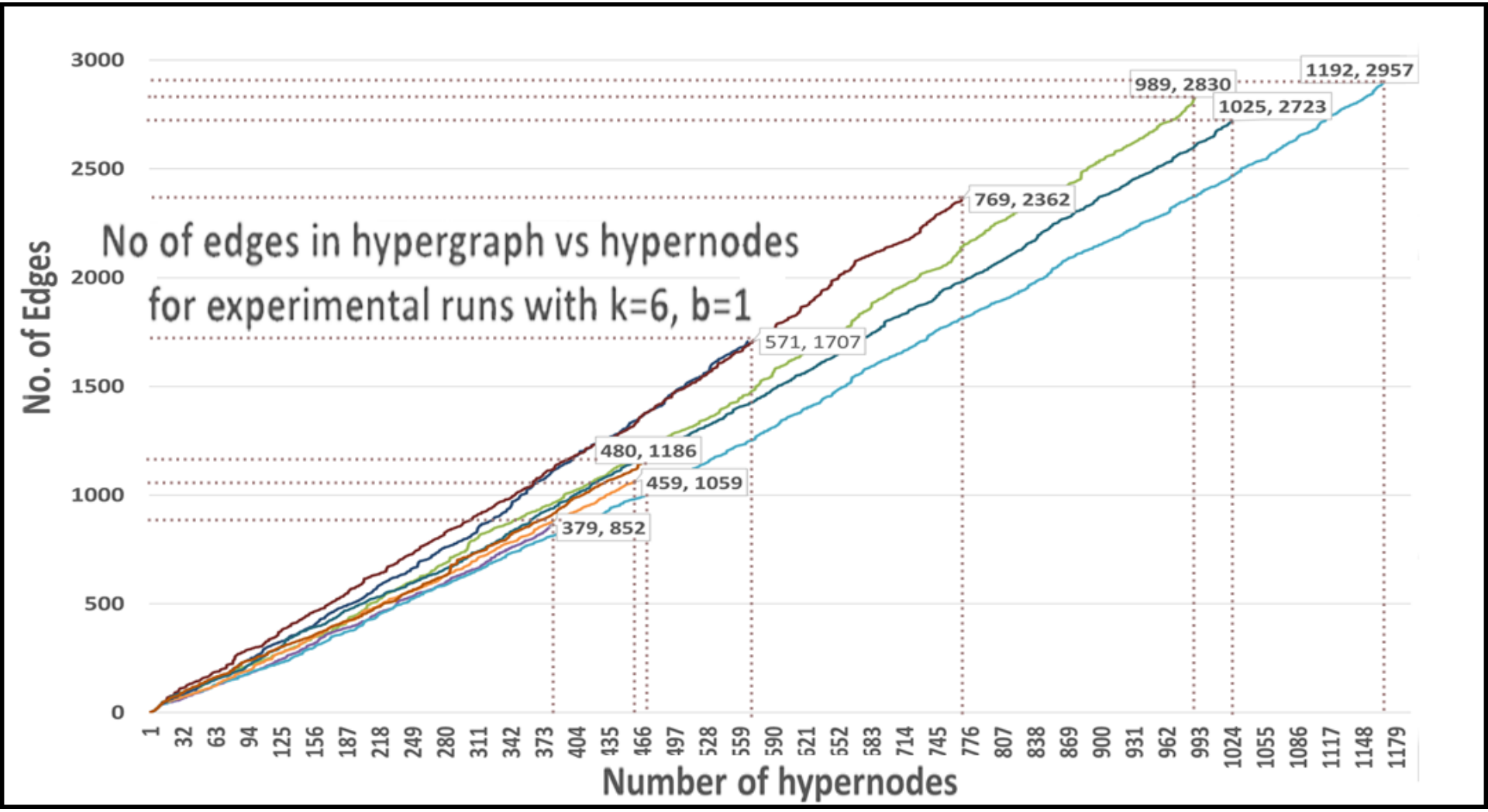


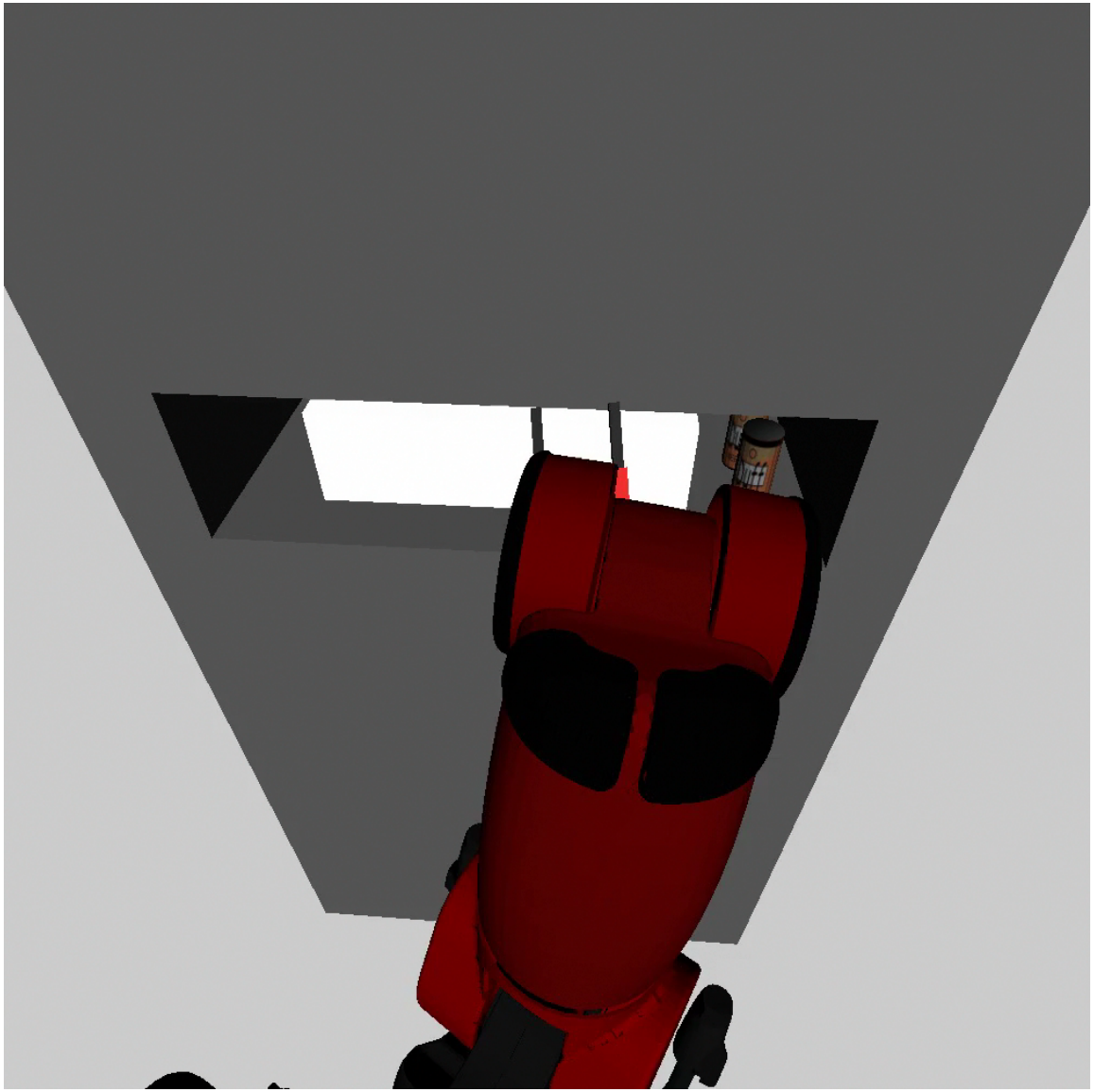
Number of Required Hypernodes



Connectivity

- Randomized placement of nodes





Thank you!

http://www.pracsyslab.org