Task and Motion Planning

Tapomayukh Bhattacharjee
University of Washington

Modified slides from Maxim Likhachev, CMU; Siddhartha Srivastava, ASU

Planning in Robotics

- Since Shakey, planning evolved along two largely independent dimensions

- Solving real tasks like setting a table requires both:
- Task planning
  - Figure out long-term strategies (use tray?), stacking order
  - Input: STRIPS-like initial state, goal, actions
- Motion planning
  - Compute a continuous path in the space of configurations of the robot
  - Input: initial & target configurations
    - Configuration: high-dimension vector
  - E.g., 20-dimensions for the PR2
    - Arms (2x8)
    - Base (3)
    - Torso height (1)

Combining Task and Motion Planners

- Task planners:
  - Effective algorithms for implicitly defined discrete transition systems (HSP [Bonet & Geffner '98], FF [Hoffmann & Nebel '01], FD [Helmert '06])
  - Not directly applicable to continuous state and action spaces

- Motion planners:
  - Effective algorithms for search in high-dimensional continuous space (PRM [Kavraki et al., '96], RRT [LaValle '98], CHOMP [Ratliff et al., 2009], Trajopt [Schulman et al., 2013])
  - Not directly applicable to discrete action sequencing

Blocksworld - Pick and Place

- Planning to re-order the blocks

Assume the arm can reach for any block, can move any block if required

Blocksworld - Pick and Place

- Planning to re-order the blocks

What is a state?
What is an action?

Blocksworld - Pick and Place

- Planning to re-order the blocks

Actions:

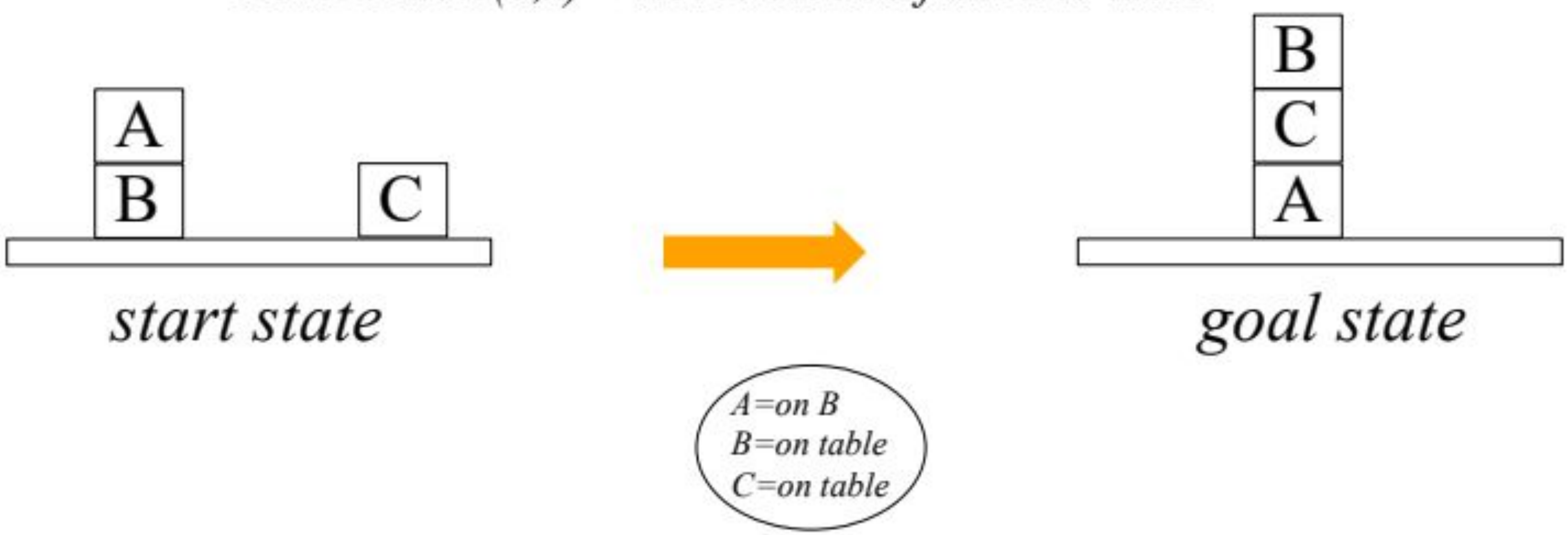
Move(b,x,y) – moves block b from x to y
MoveToTable(b,x) – moves block b from x to table y

Blocksworld - Towards Graph Search

- Planning to re-order the blocks

Actions:

Move(b,x,y) – moves block b from x to y
MoveToTable(b,x) – moves block b from x to table

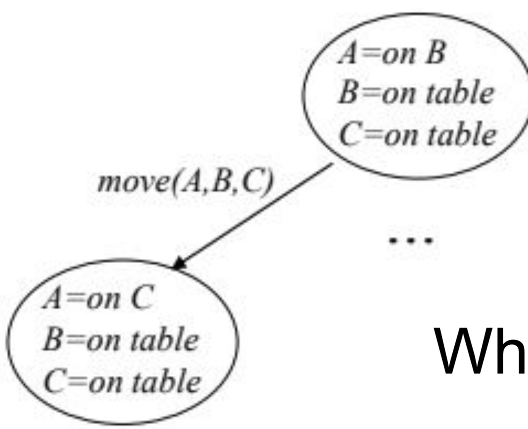


Blocksworld - Towards Graph Search

- Planning to re-order the blocks

Actions:

Move(b,x,y) – moves block b from x to y
MoveToTable(b,x) – moves block b from x to table



What is the cost of this edge?

Symbolic Planning

Can we specify any task planning problem with a representational language that allows definition of states, actions and goals?

Generic Representation

- STRIPS (=Stanford Research Institute Problem Solver)

  State Representation:

  Goal Representation:

  Action Representation:

Generic Representation

- STRIPS (=Stanford Research Institute Problem Solver)

State Representation:

conjunction of positive(true) literals

(e.g, On(A,B)^On(B,Table)^On(C,Table)^Block(A)^Block(B)^Block(C)^Clear(A)^Clear(C))

Goal Representation:

Action Representation:

Closed-World Assumption: Unspecified information is FALSE!

Generic Representation

- STRIPS (=Stanford Research Institute Problem Solver)

State Representation:

conjunction of positive(true) literals

(e.g, On(A,B)^On(B,Table)^On(C,Table)^Block(A)^Block(B)^Block(C)^Clear(A)^Clear(C))

Goal Representation:

desired conjunction of positive(true) literals

Action Representation:

What is the goal representation for the Block-World Example?

Can be partially specified.

Generic Representation

- STRIPS (=Stanford Research Institute Problem Solver)

State Representation:

conjunction of positive(true) literals

(e.g, On(A,B)^On(B,Table)^On(C,Table)^Block(A)^Block(B)^Block(C)^Clear(A)^Clear(C))

Goal Representation:

desired conjunction of positive(true) literals

Action Representation:

***Preconditions**: conjunction of positive(true) literals that must be held true in order for the action to be applicable*

***Effect**: conjunction of positive(true) literals showing how the state will change (what should be deleted and added)*

Generic Representation

MoveToTable(b,x)
Precond: On(b,x)^Clear(b)^Block(b)^Block(x)
Effect: On(b,Table)^Clear(x)^~On(b,x)

Move(b,x,y)
Precond: On(b,x)^Clear(b)^Clear(y)^Block(b)^Block(y)^(b~=y)
Effect: On(b,y)^Clear(x)^~On(b,x)^~Clear(y)

Generic Representation

- Representing it with STRIPS

Start state:

On(A,B)^On(B,Table)^On(C,Table)^Block(A)^Block(B)^Block(C)^Clear(A)^Clear(C)

***Goal state**:*

On(B,C)^On(C,A)^On(A,Table)

Actions:

<u>MoveToTable(b,x)</u>

Precond: On(b,x)^Clear(b)^Block(b)^Block(x)

Effect: On(b,Table)^Clear(x)^~On(b,x)

<u>Move(b,x,y)</u>

Precond: On(b,x)^Clear(b)^Clear(y)^Block(b)^Block(y)^(b~=y)

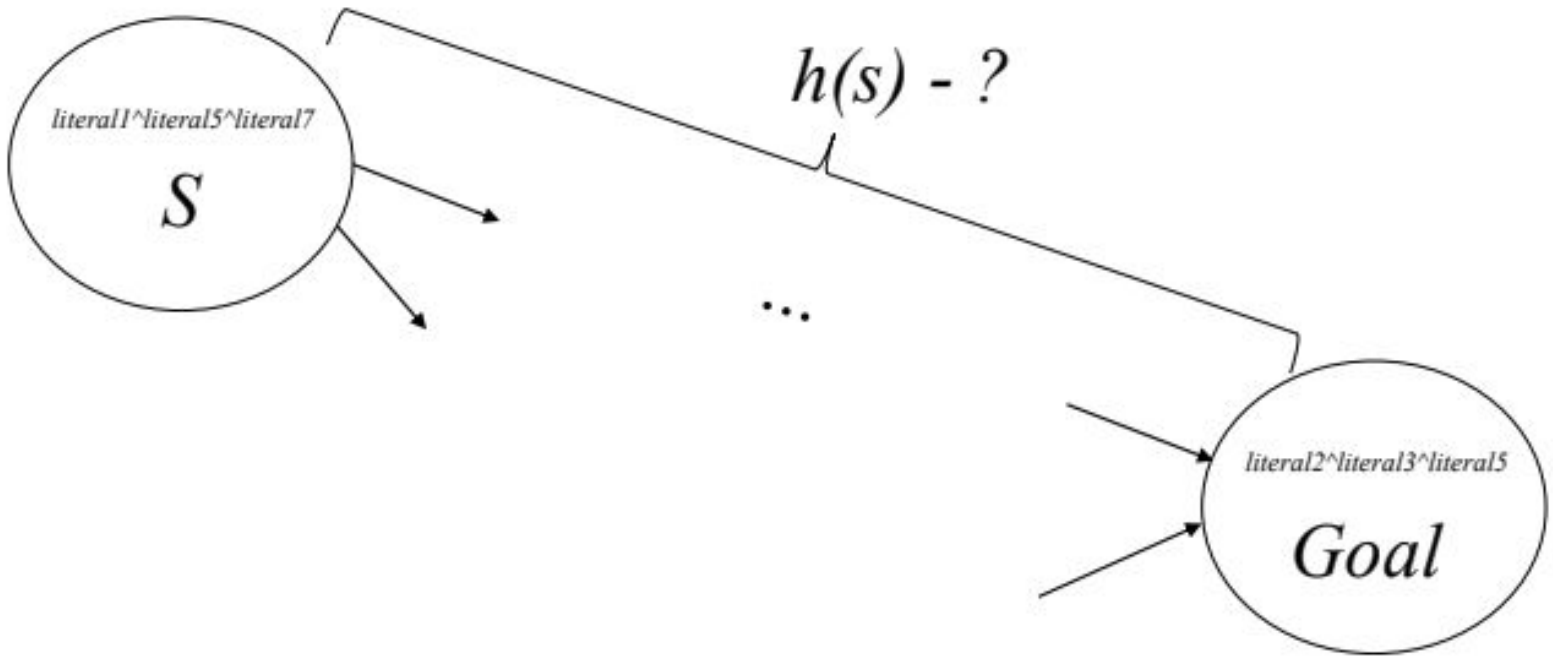
Effect: On(b,y)^Clear(x)^~On(b,x)^~Clear(y)

Generic Representation

1. Represent Problem using STRIPS
2. Design a domain-independent program
3. Define GetSuccessor() Function to work with Implicit Graphs
4. Implement A* or similar algorithms
5. What could be “domain-independent” heuristics?

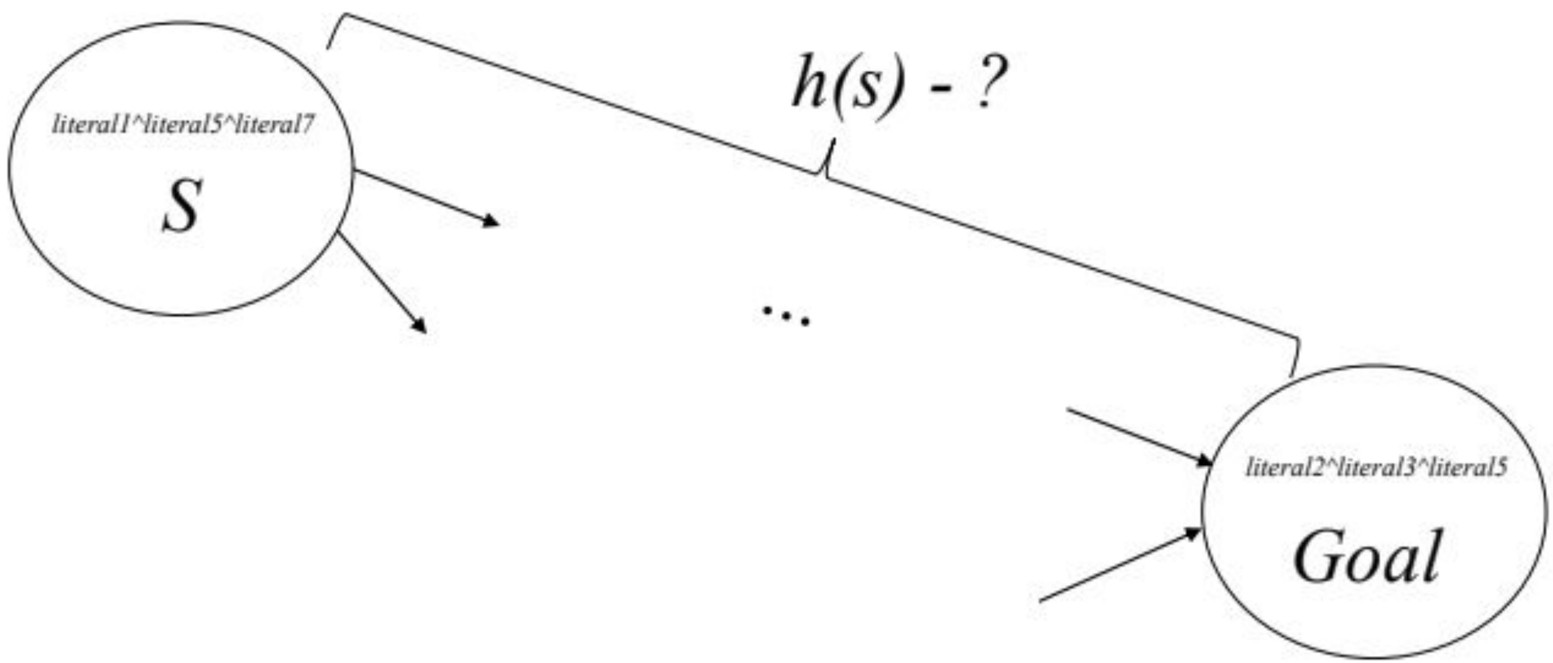
Domain-Independent Heuristics

- Computing heuristics



Domain-Independent Heuristics

- Computing heuristics

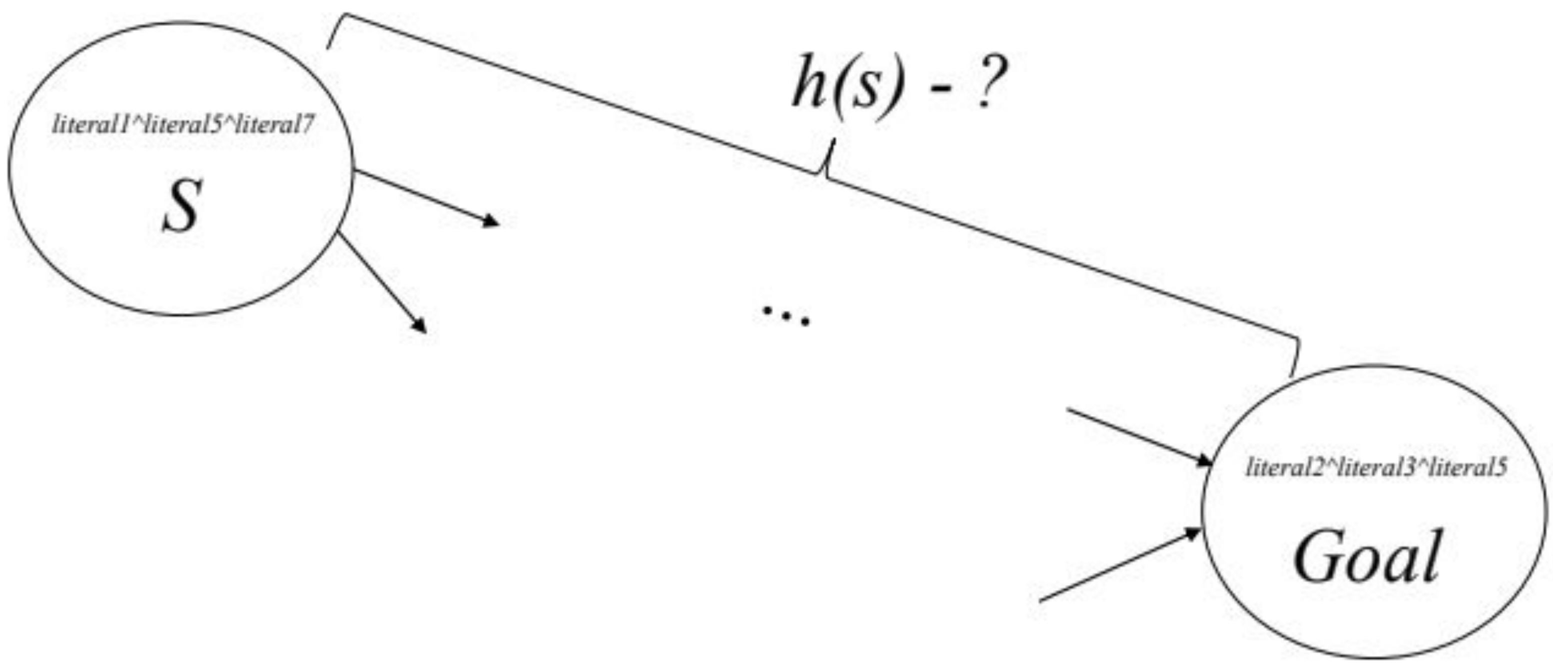


Option 1: Number of literals not yet satisfied?

Admissible? Can we use it?

Domain-Independent Heuristics

- Computing heuristics

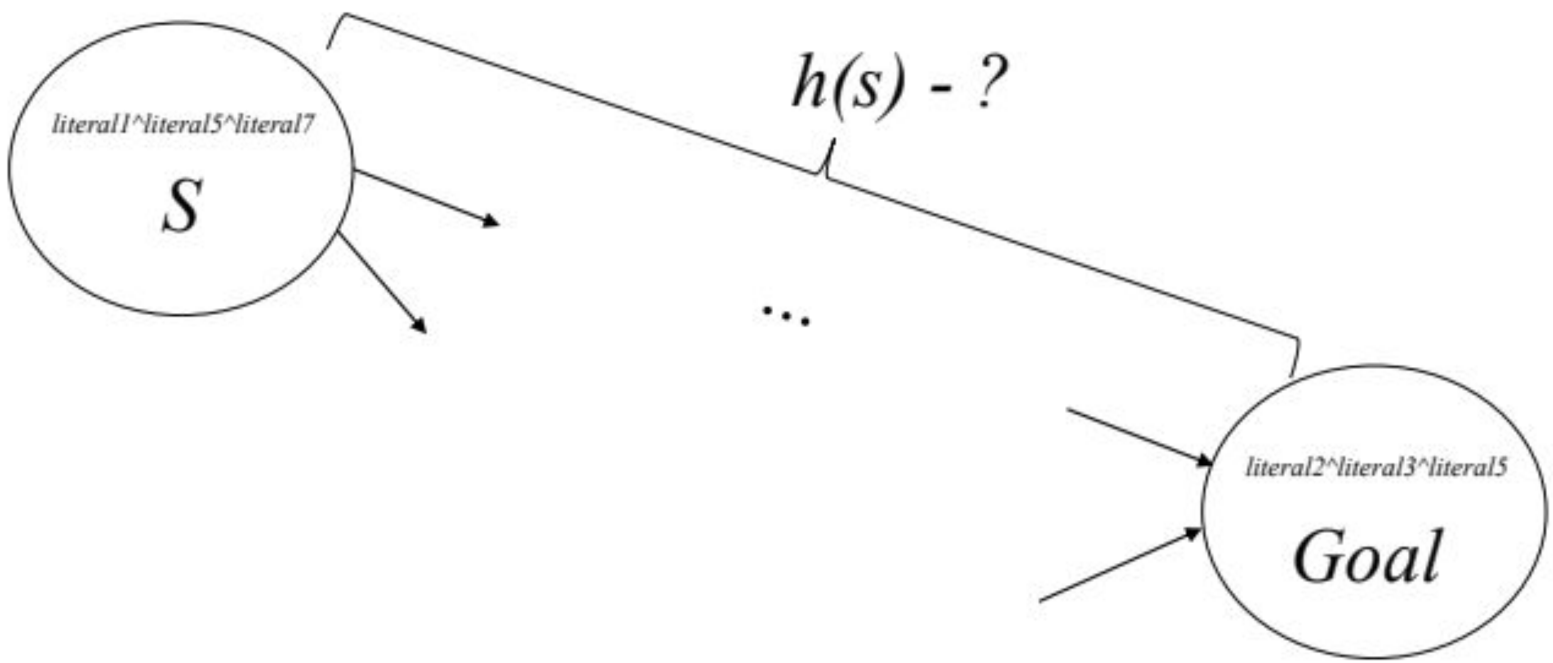


Option 2: Compute heuristic using a relaxed problem.

Ex: Actions don't have -ve effects (empty-delete-list heuristic)

Domain-Independent Heuristics

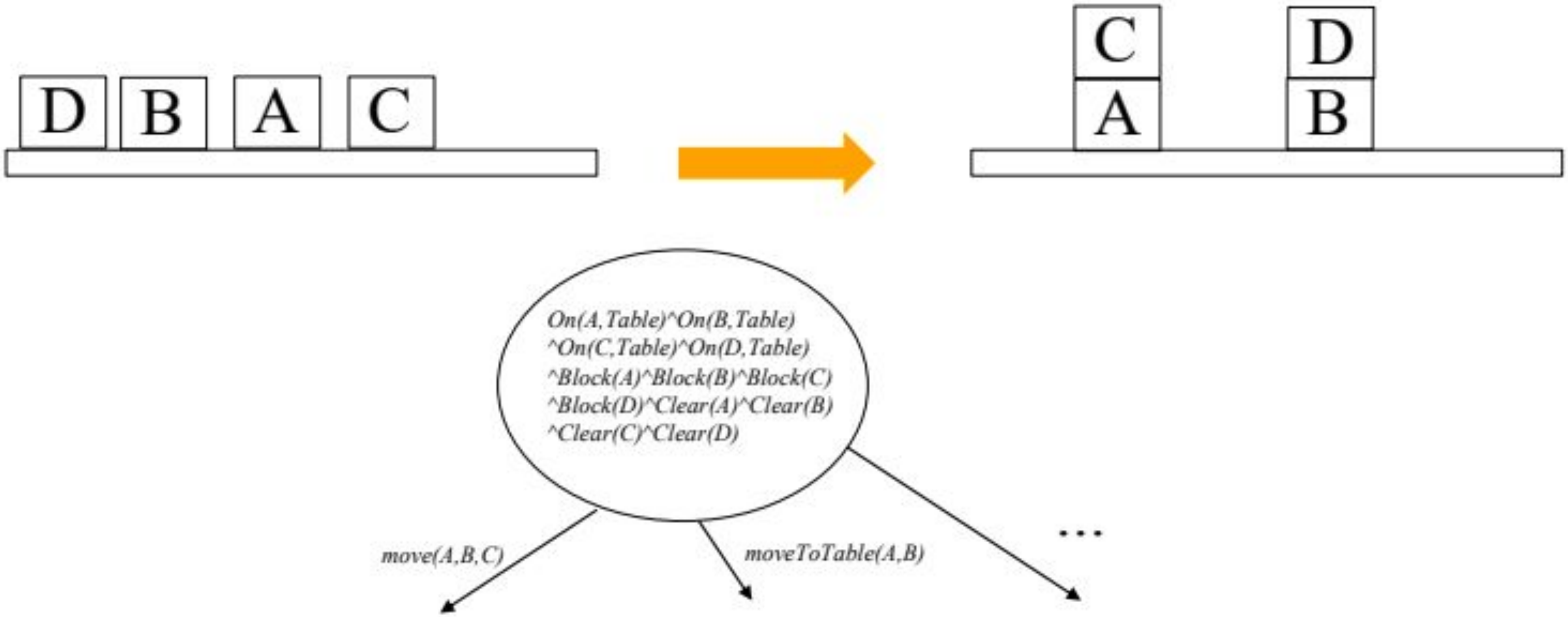
- Computing heuristics



Option 2: Compute heuristic using a relaxed problem.

Admissible? Pros and Cons?

Challenges in Naive Graph Search



Graphs can blow up!!

Possible solution: Planning Graphs! [See the other slide deck]

Task and Motion Planning

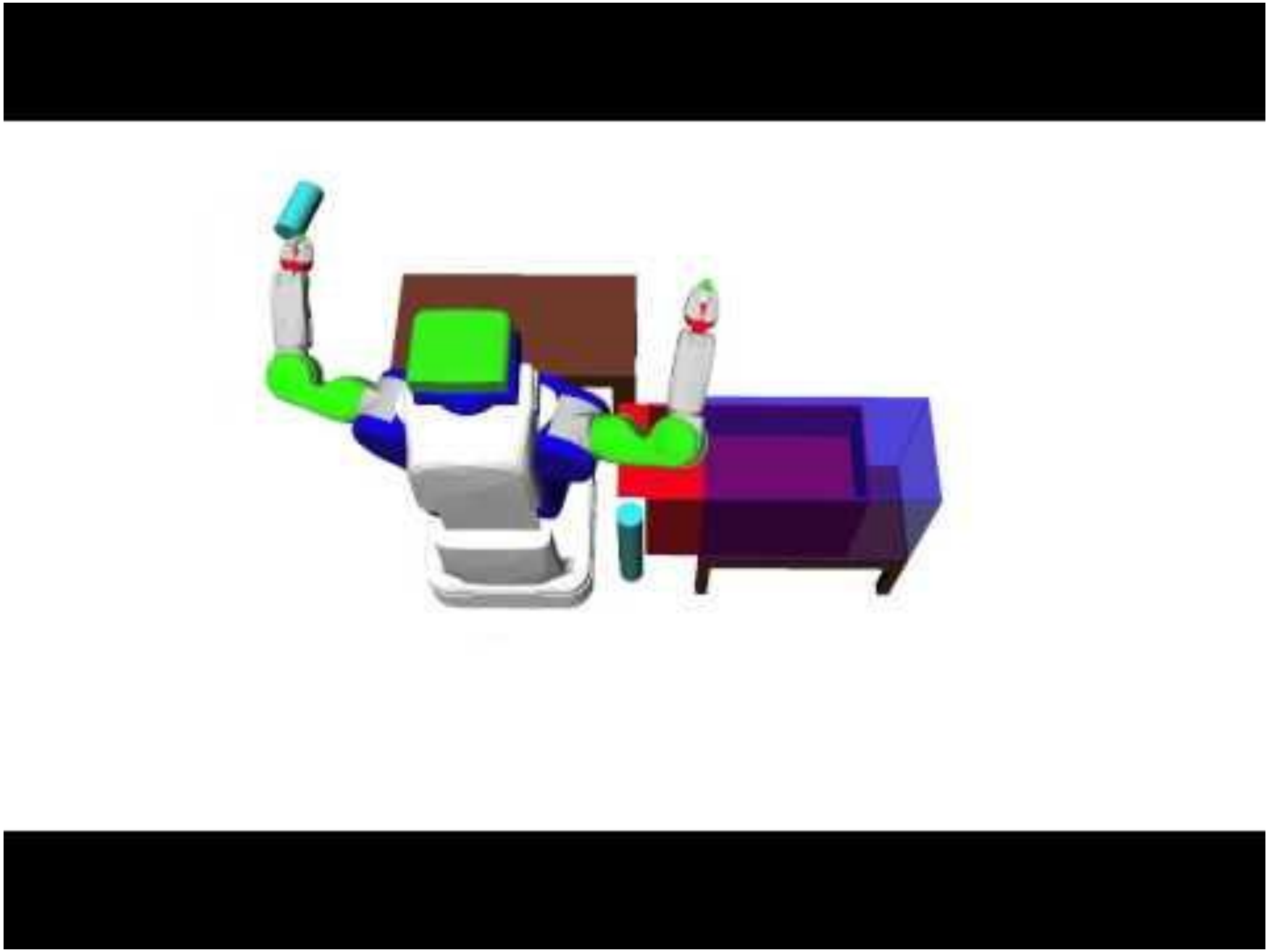
Where does Motion Planning come into play?

Task and Motion Planning

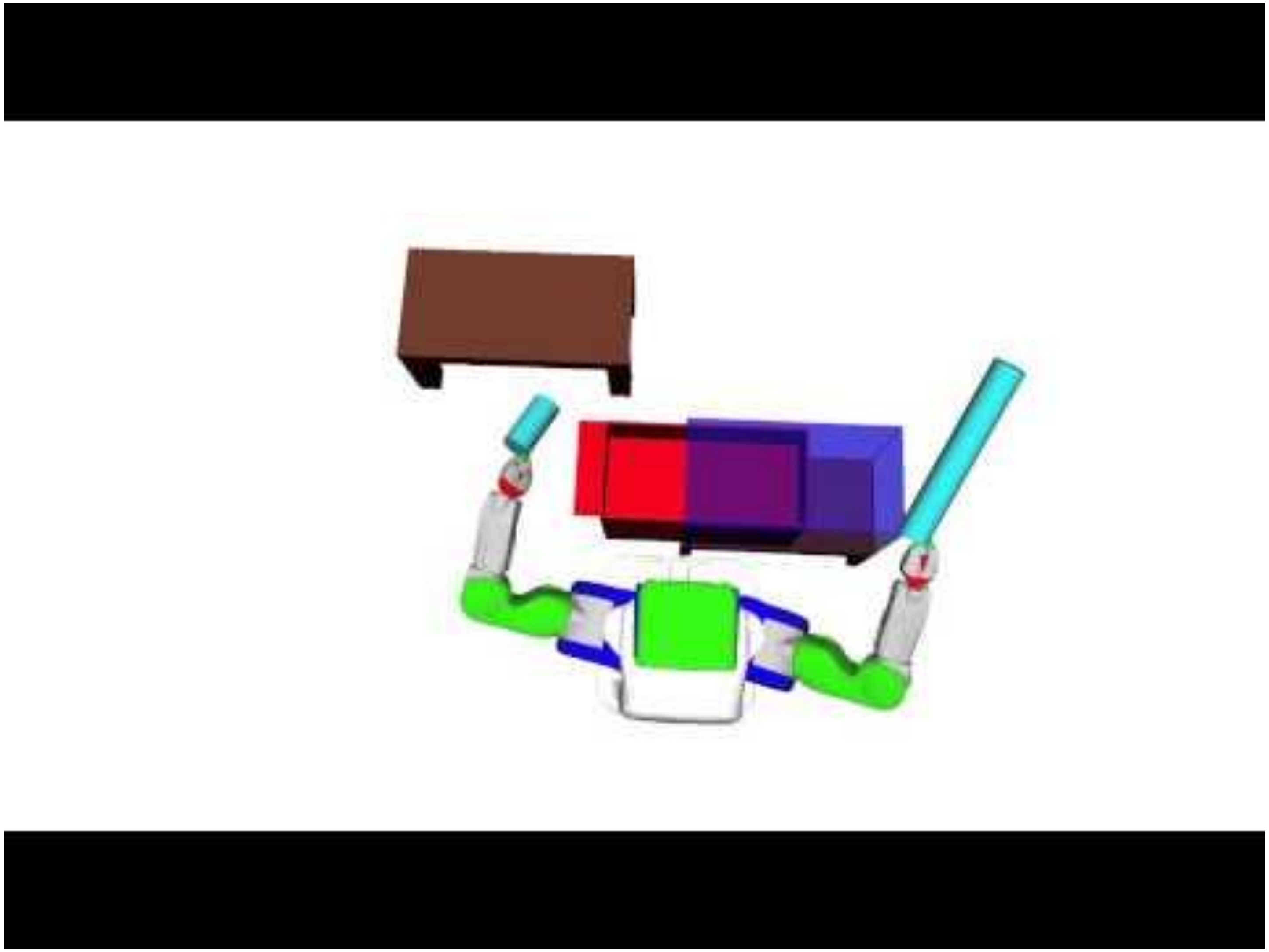
Where does Motion Planning come into play?

1. Action Verification - Collisions. Robot feasibility.
2. Sequence Verification - Order feasibility.
3. Sequence Optimization - Edges have non-unit costs.

Task and Motion Planning



Task and Motion Planning



Task and Motion Planning