

Interactive Application Showcasing Planning Techniques

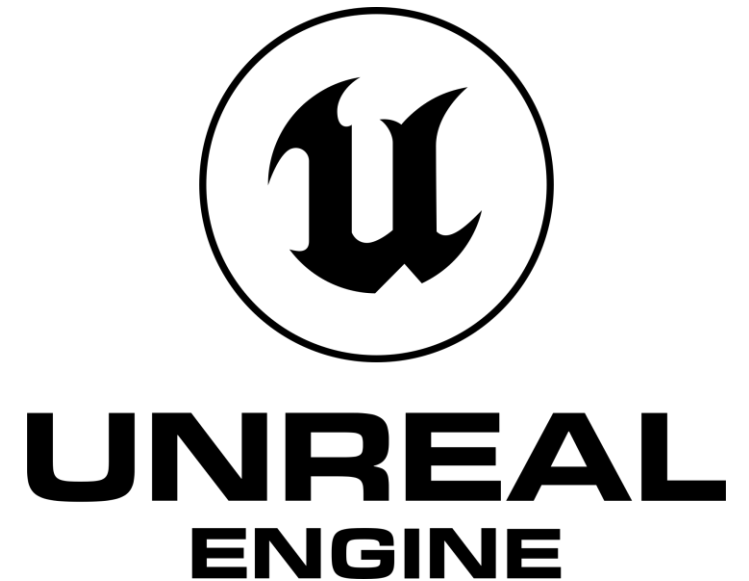
Elia Hänggi

11. September 2023

Introduction



<https://www.fast-downward.org/>

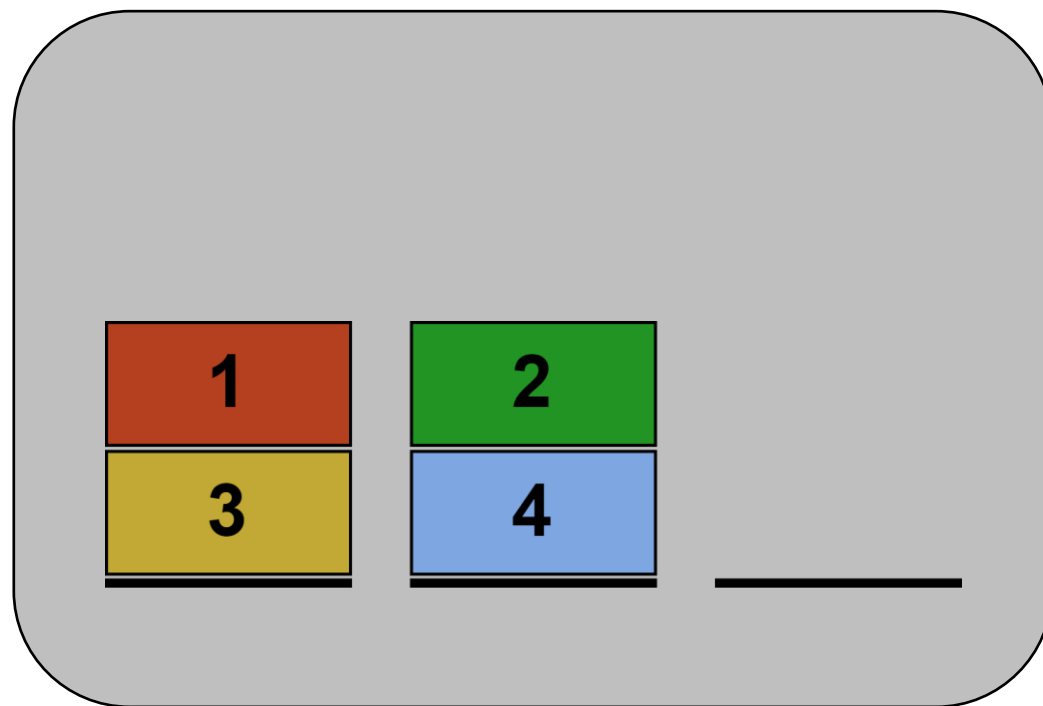
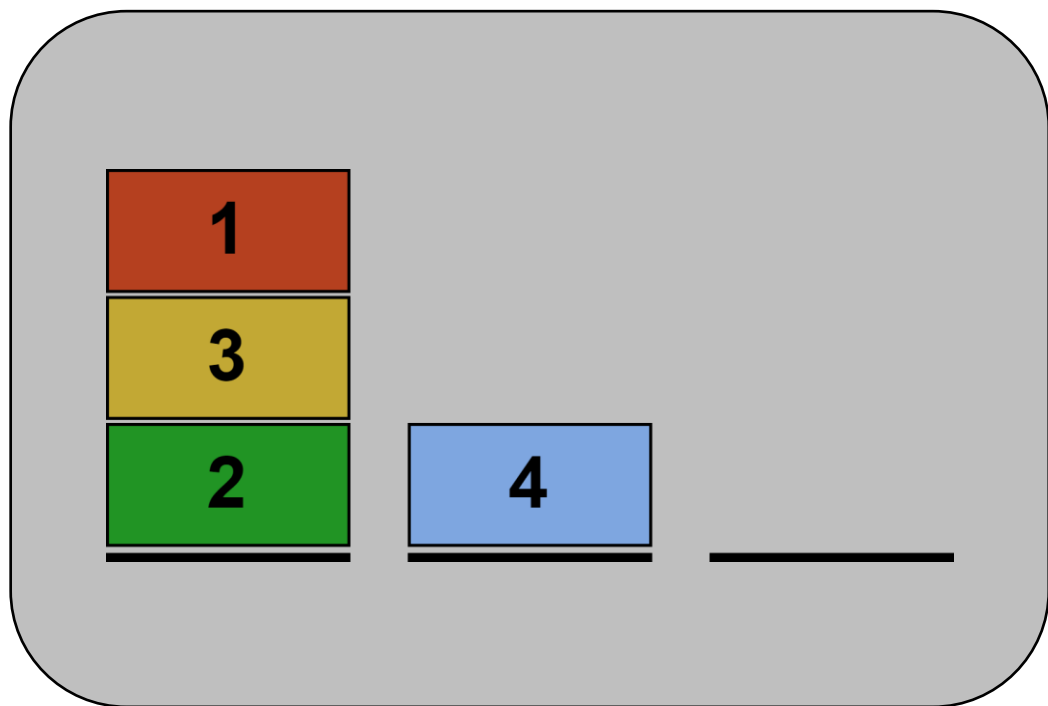


<https://www.unrealengine.com/en-US/>

Blocksworld



Blocksworld



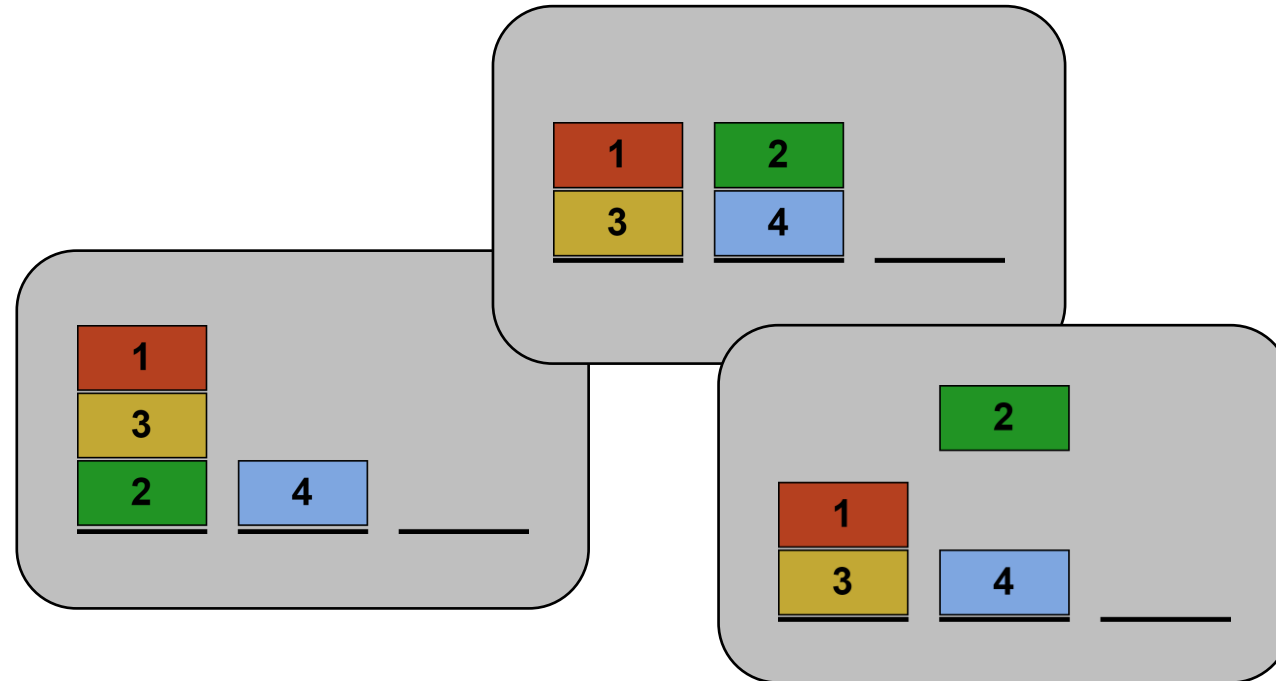
X

Save



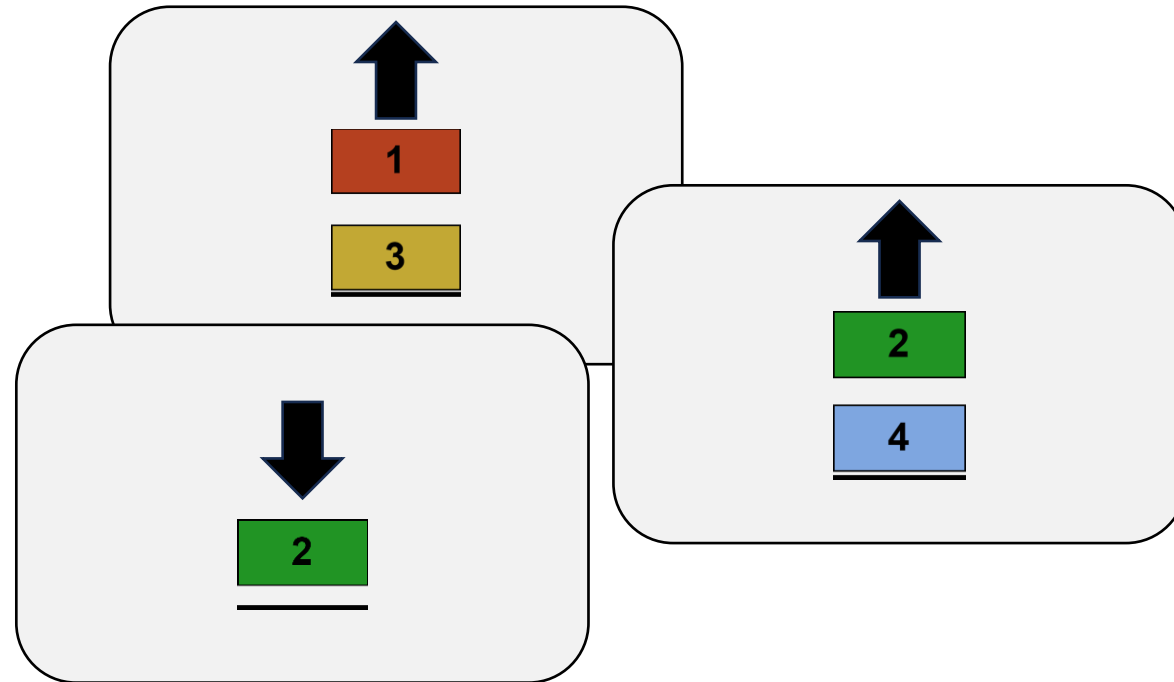
State Space

- Set of States



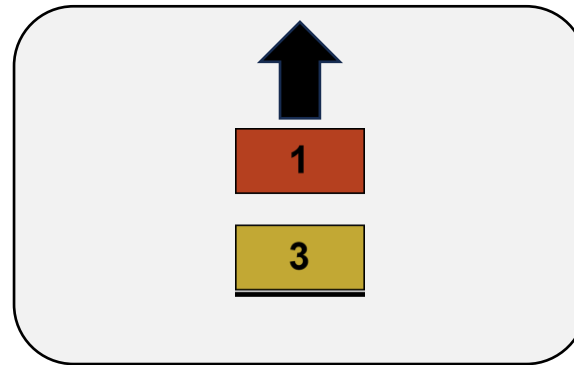
State Space

- Set of States
- Set of Actions

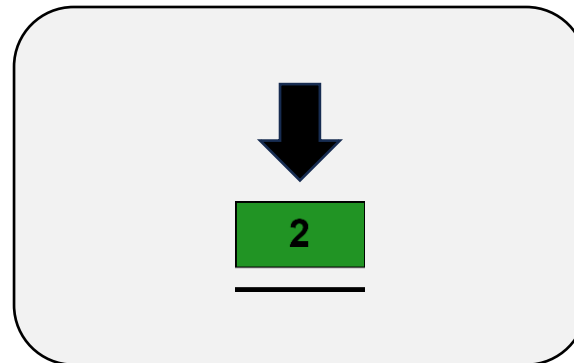


State Space

- Set of States
- Set of Action
- Action costs



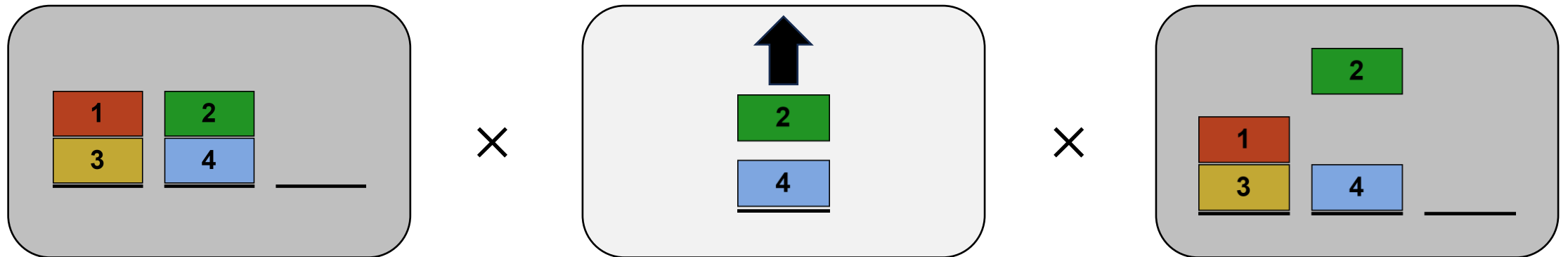
→ 1



→ 1

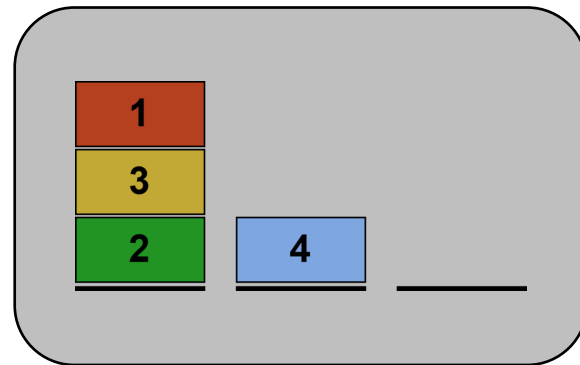
State Space

- Set of States
- Set of Actions
- Action costs
- Transition relation



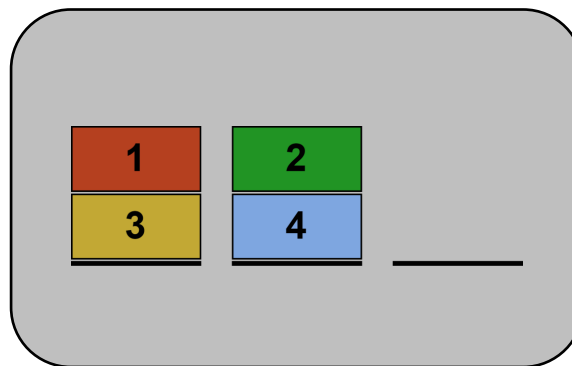
State Space

- Set of States
- Set of Actions
- Action costs
- Transition relation
- Initial state

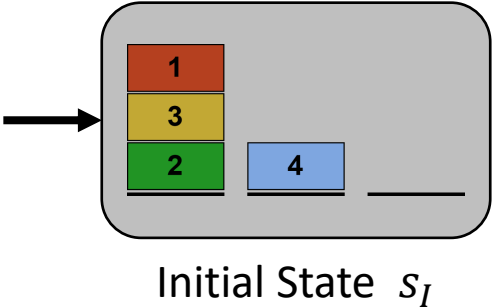


State Space

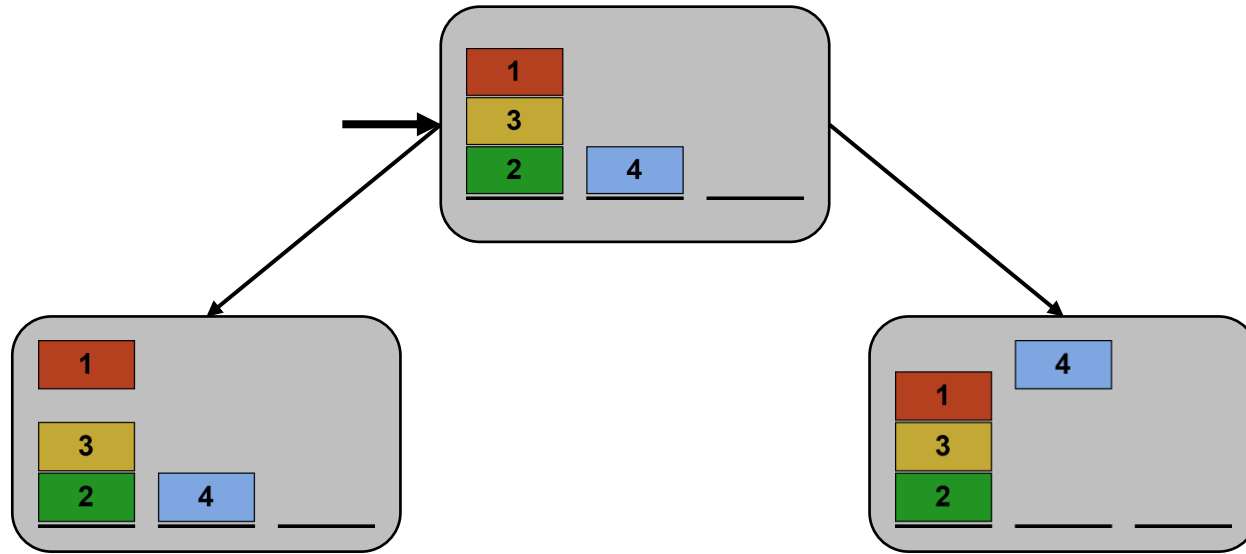
- Set of States
- Set of Actions
- Action costs
- Transition relation
- Initial state
- Set of goal states



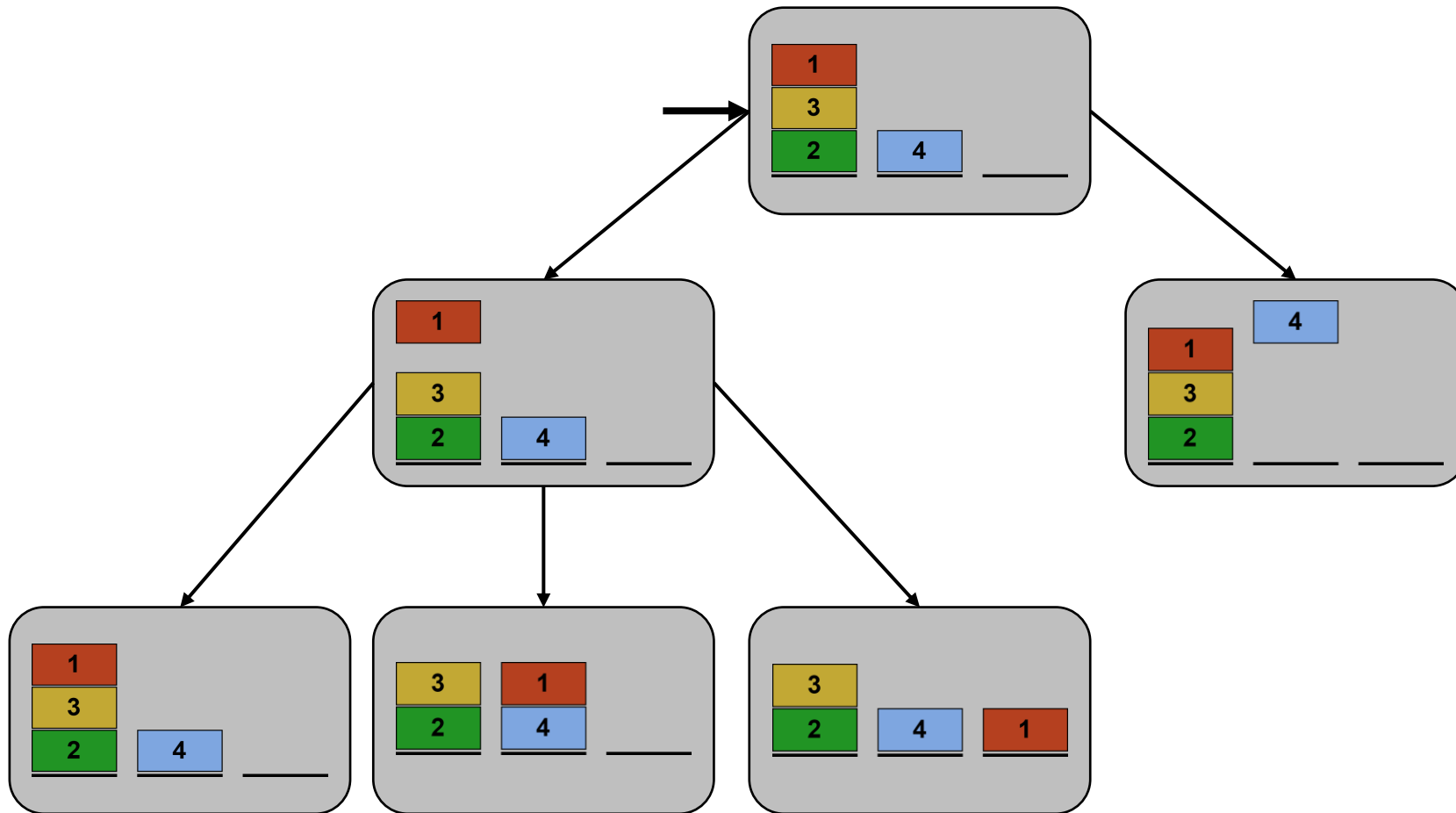
State Space Search



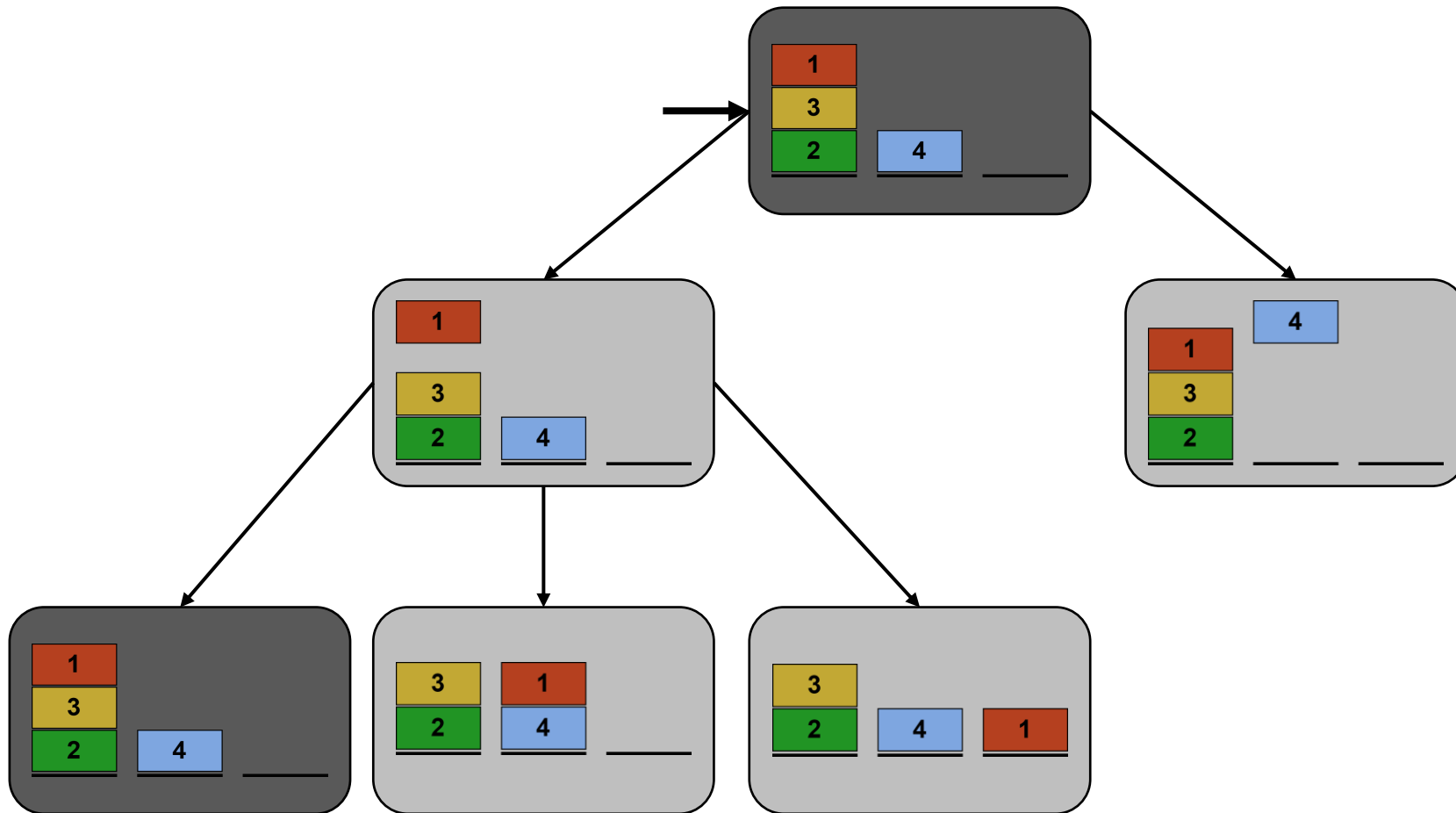
State Space Search



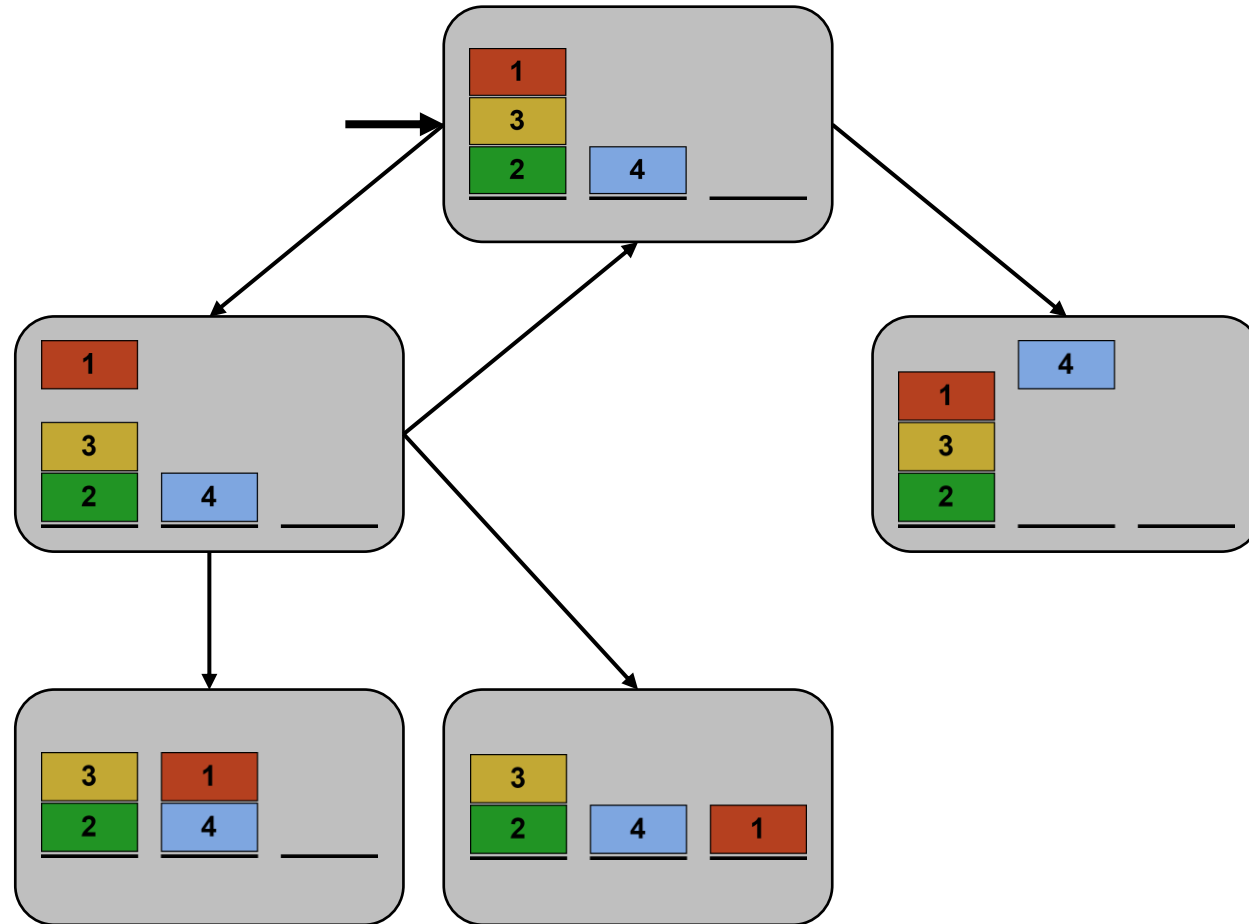
State Space Search



State Space Search



State Space Search



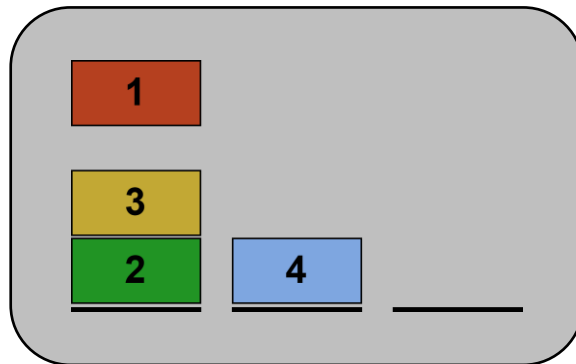
Heuristic h

- Estimated distance from state s to goal

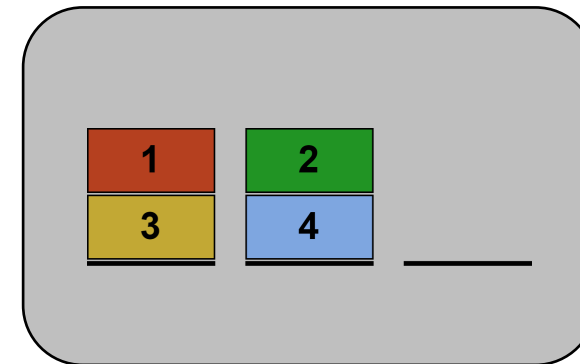
Heuristic h

- Estimated distance from state s to goal

State s



Goal state



Heuristic h

- Estimated distance from state s to goal

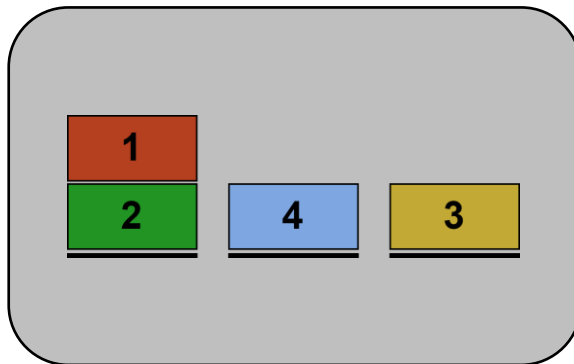


$$h_{Blocks}(s) = 2 \cdot 2 + 1 = 5$$

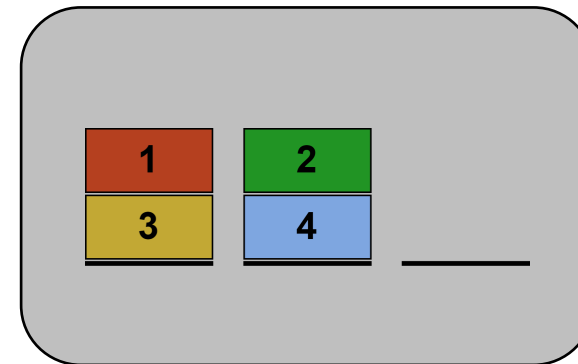
Heuristic h

- Estimated distance from state s to goal

State s



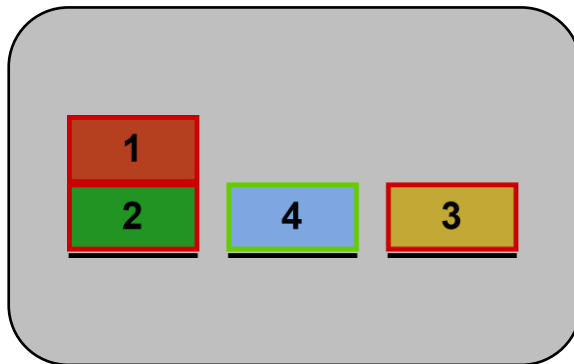
Goal state



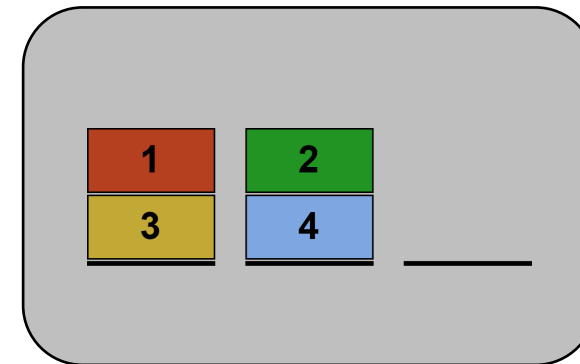
Heuristic h

- Estimated distance from state s to goal

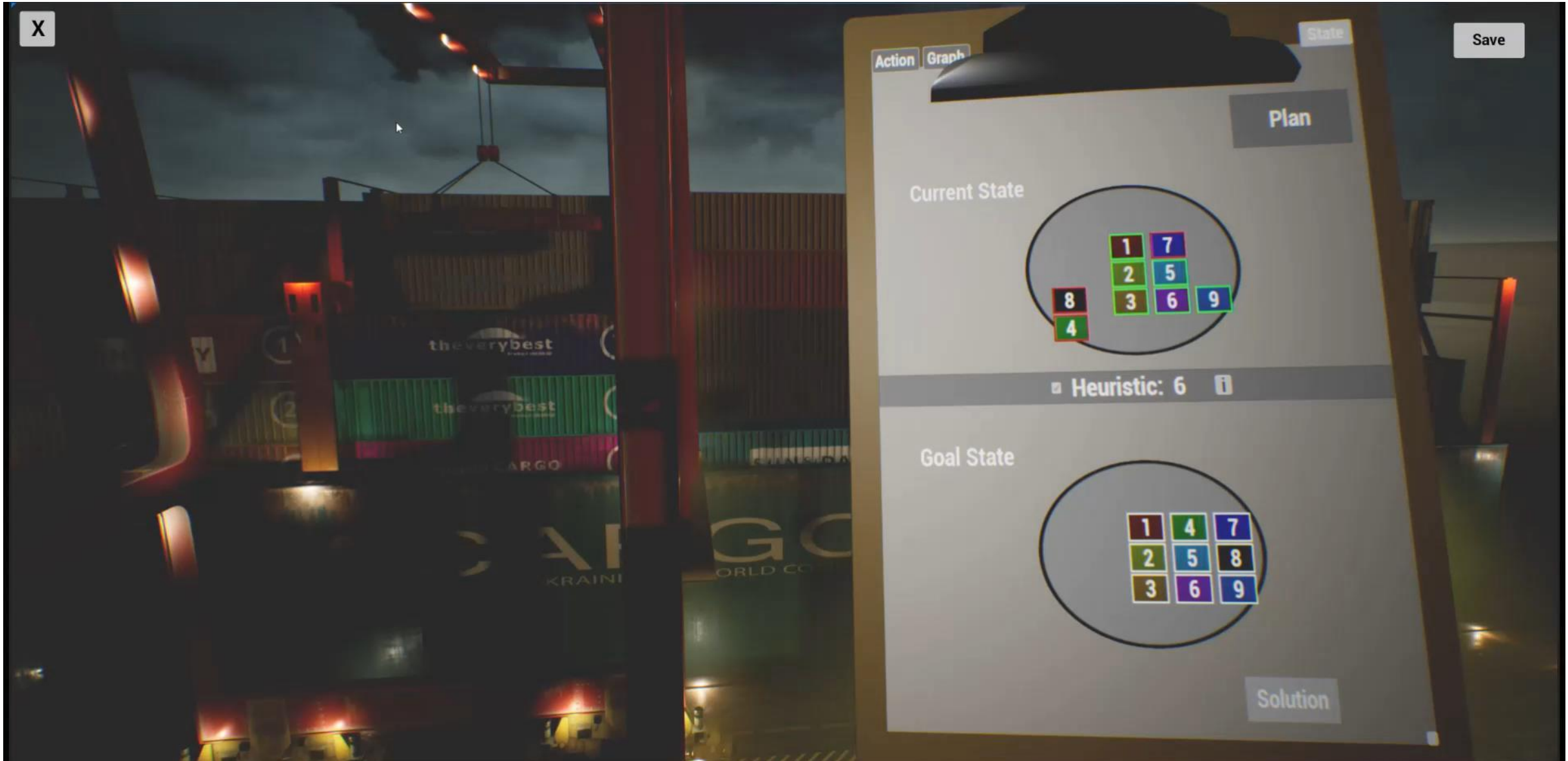
State s



Goal state



$$h_{Blocks}(s) = 3 \cdot 2 = 6$$



X

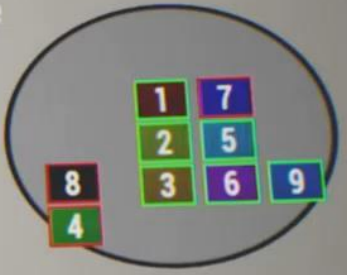
Save

Action Graph

State

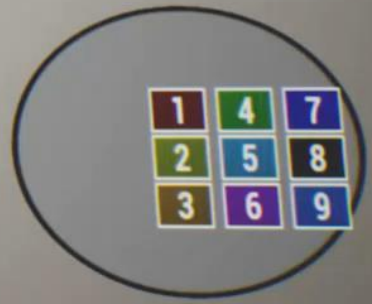
Plan

Current State



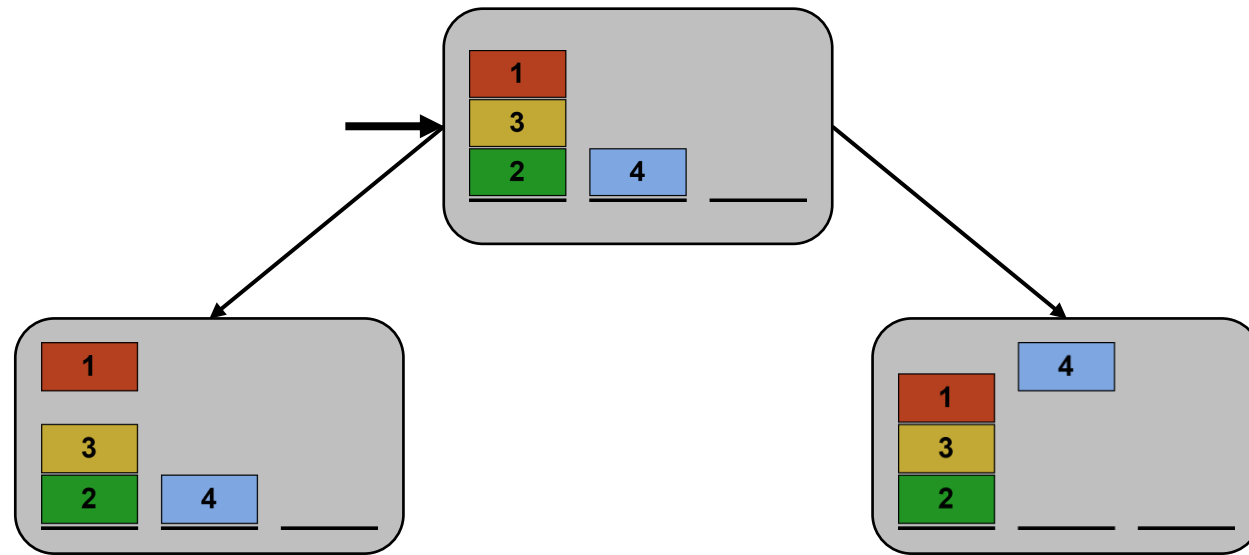
Heuristic: 6

Goal State



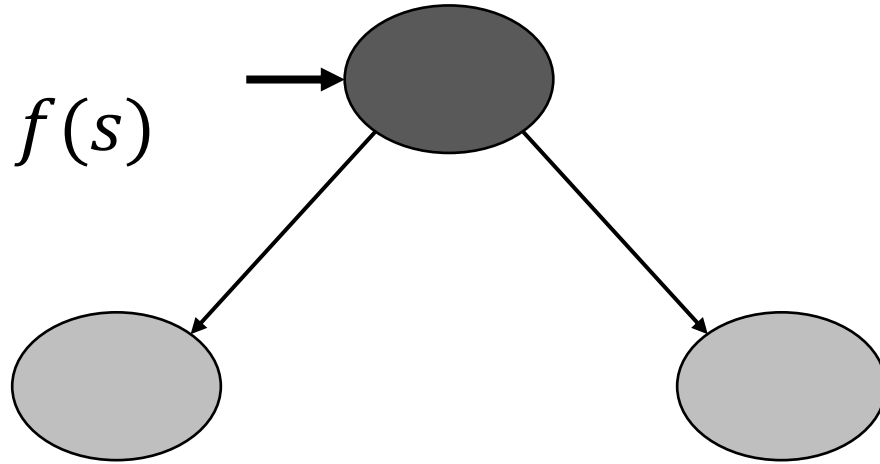
Solution

Best-first search



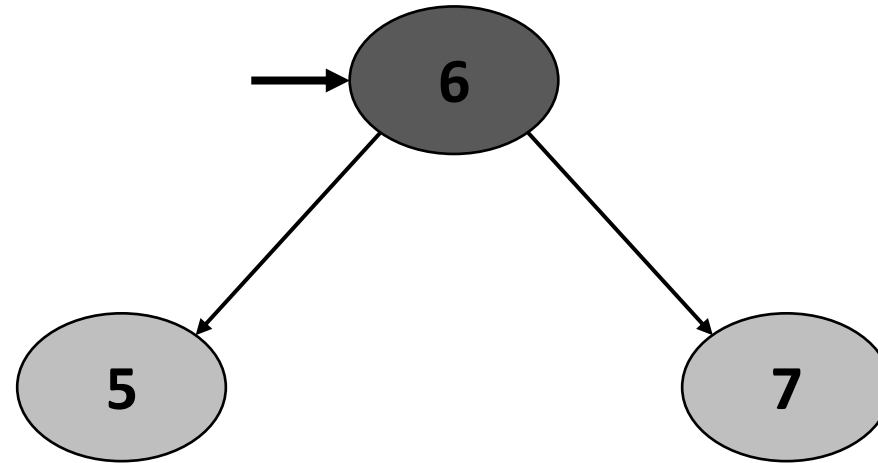
Best-first search

- Evaluation function $f(s)$



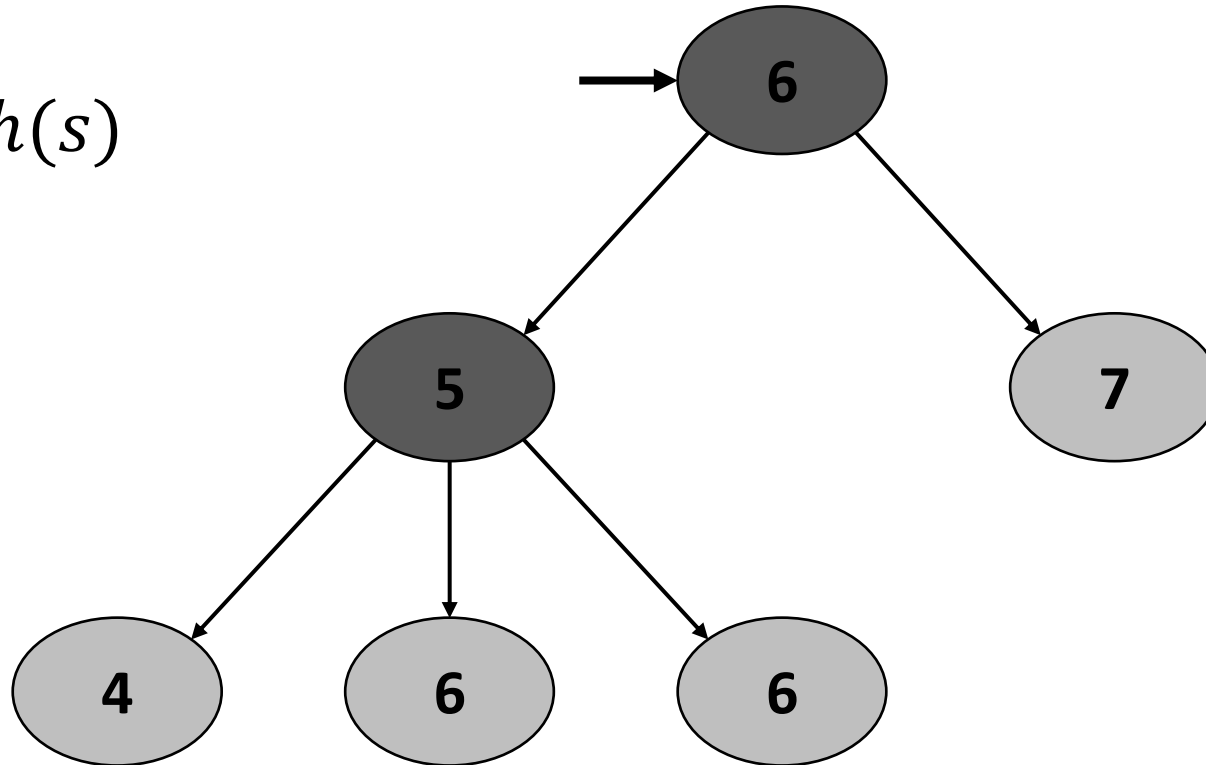
Best-first search

- $f(s) = h(s)$



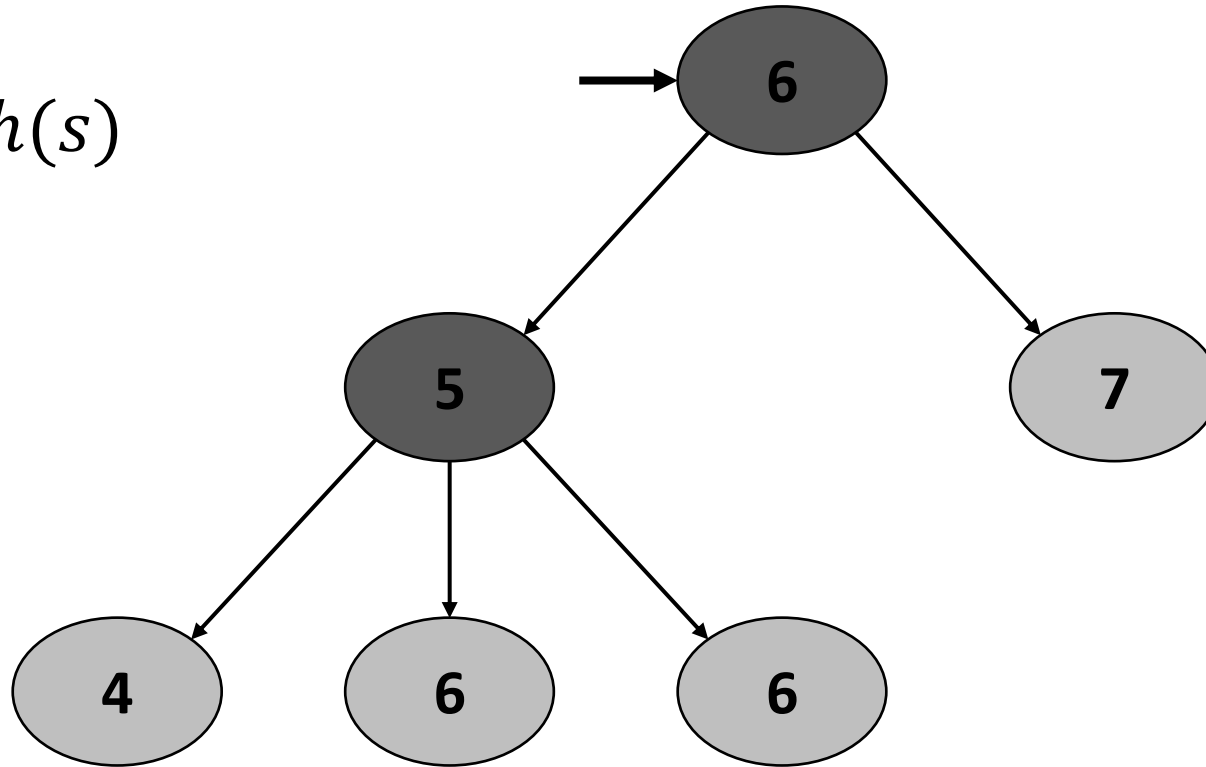
Best-first search

- $f(s) = h(s)$



Best-first search

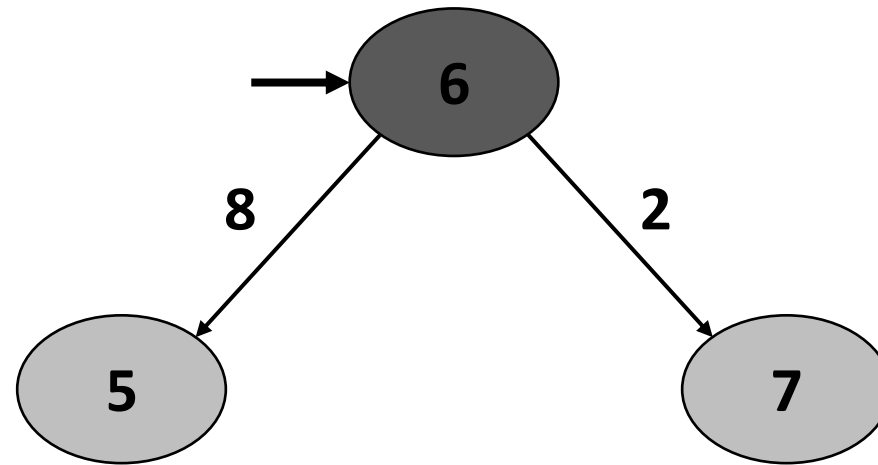
- $f(s) = h(s)$



 Greedy-best-first search

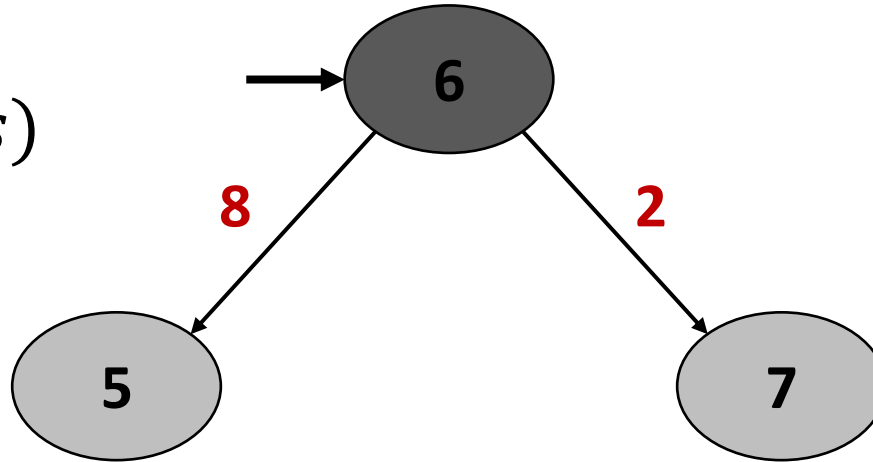
Best-first search

- $f(s) = h(s)$



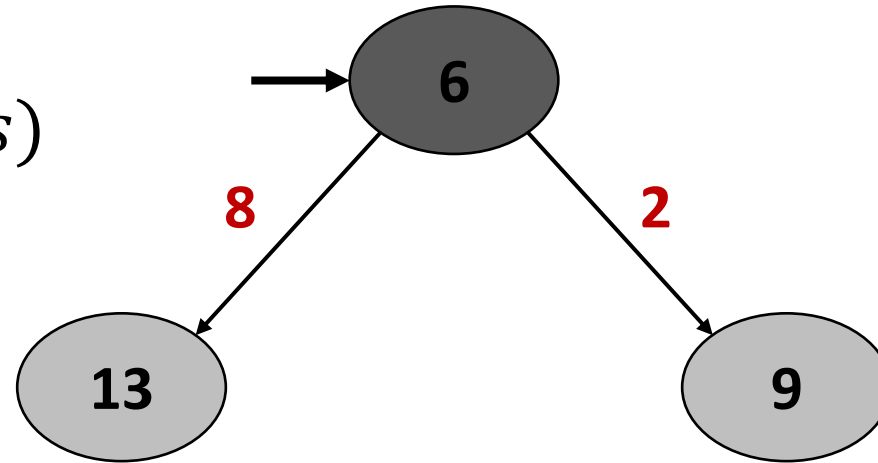
Best-first search

- $f(s) = g(s) + h(s)$



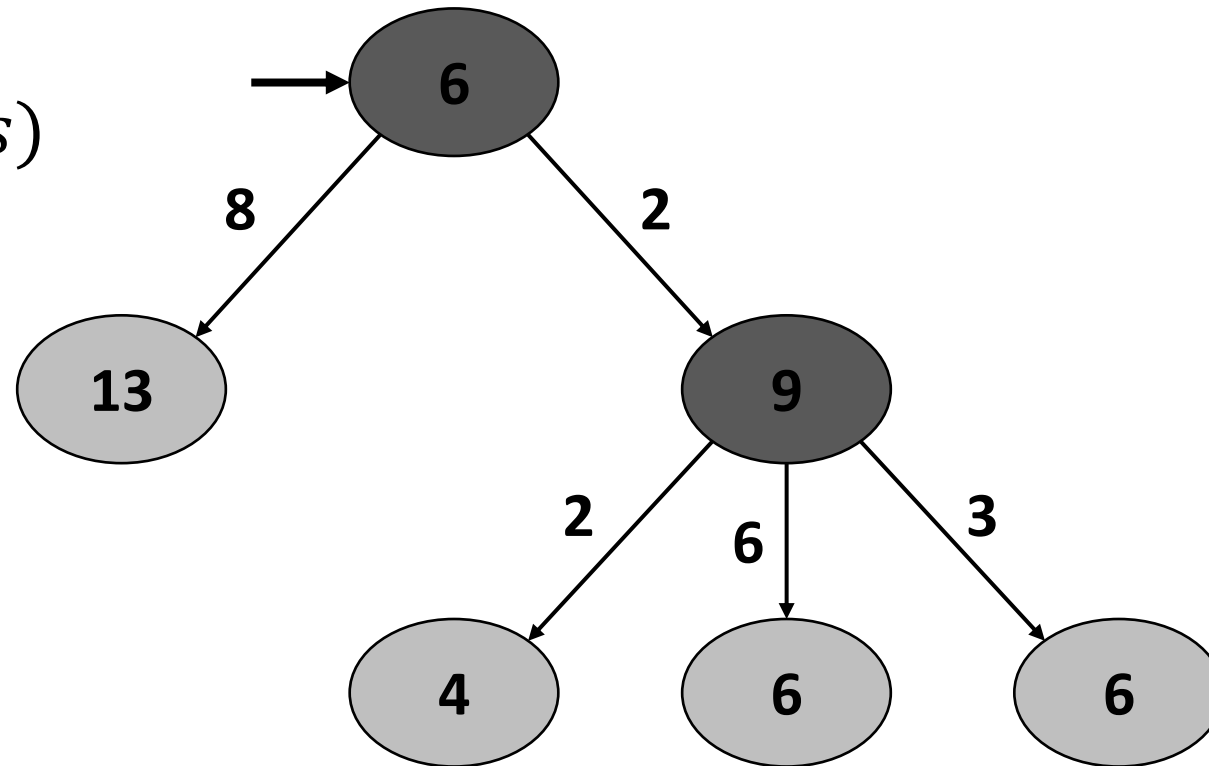
Best-first search

- $f(s) = g(s) + h(s)$



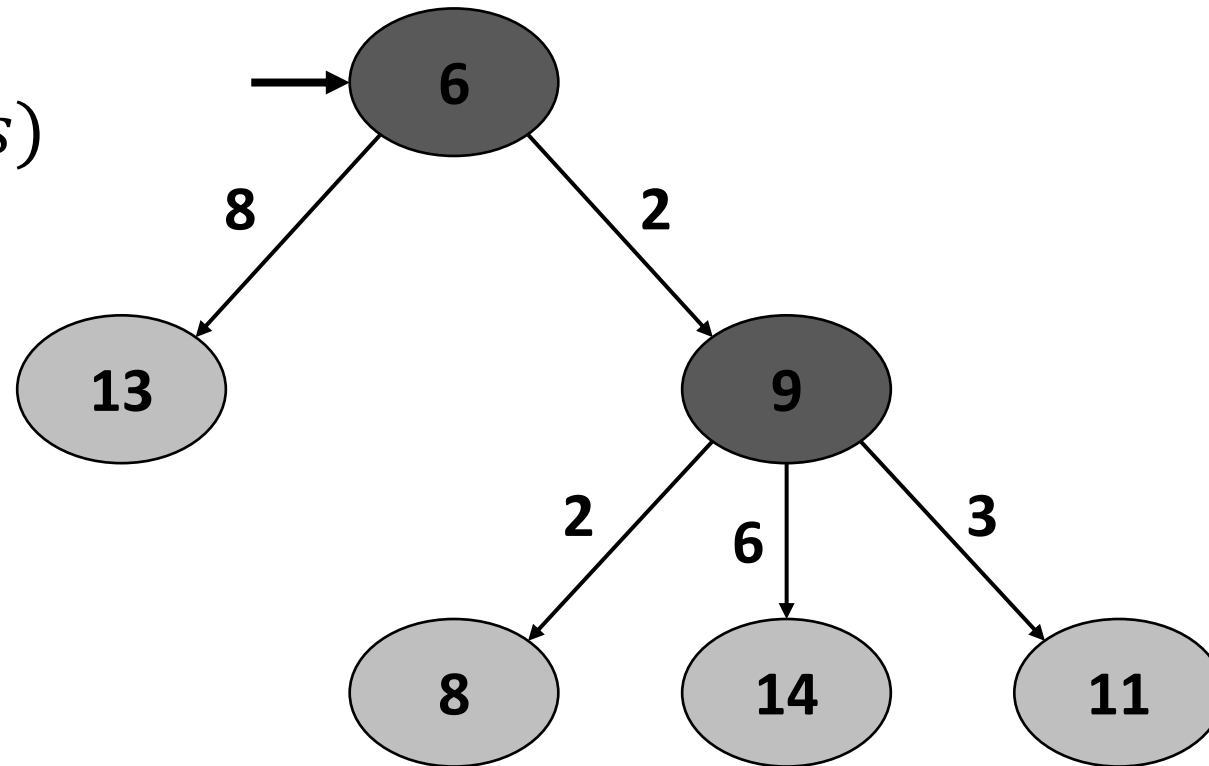
Best-first search

- $f(s) = g(s) + h(s)$



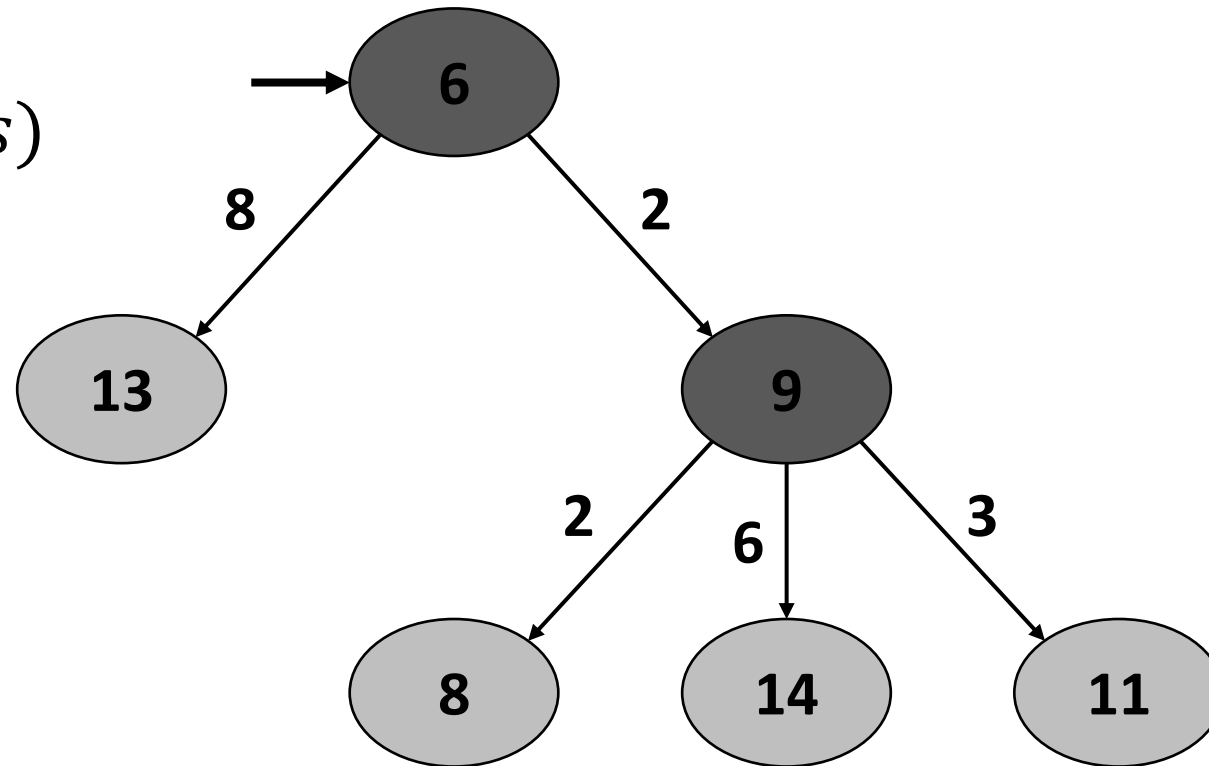
Best-first search


- $f(s) = g(s) + h(s)$



Best-first search

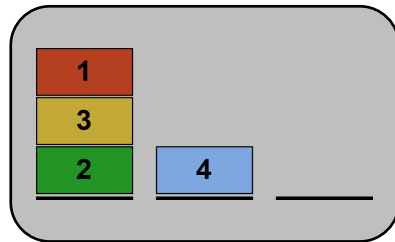
- $f(s) = g(s) + h(s)$



 A*



Fast Downward in the application



Current state s

```
(define (problem BLOCKS)
  (:domain BLOCKS)
  (:objects stackland stack2 stack1 stack3 c0 c1 c2 c3 - container h0 h1
            h2 h3 n0 n1 n2 n3 n4 - height)

  (:INIT (craneempty) (SUCC h0 h1) (SUCC h1 h2) (SUCC h2 h3) (SUCC n0 n1)
         (SUCC n1 n2) (SUCC n2 n3) (SUCC n3 n4)
         (on-height stackland h0) (on-height stack1 h0) (on-height stack2 h0)
         (on-height stack3 h0)
         (clear c0) (on c0 c3) (on-height c0 n2) (on c3 c7) (on-height c3 n1)
         (clear c2) (on c2 c1) (on-height c2 n2) (on c1 c6) (on-height c1 n1)
         (clear stack3)
         (clear stackland))

  (:goal (and (on c0 c1) (on c1 stack1)
             (on c2 c3) (on c3 stack2))))
```

PDDL problem file

```
(define (domain BLOCKS)
  (:requirements :strips)
  (:types block height)
  (:predicates (on ?x ?y - container)
              (clear ?x - container)
              (craneempty)
              (holding ?x - container)
              (on-height ?x - container ?y - height)
              (SUCC ?hx ?hy - height))

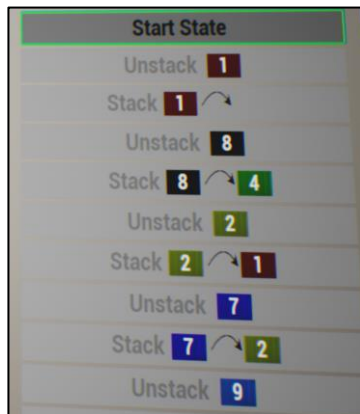
  (:action stack
    (parameters (?x ?y - container ?hx ?hy - height)
     :precondition (and (holding ?x) (clear ?y)
                       (on-height ?y ?hy) (SUCC ?hy ?hx)))
    :effect
    (and (not (holding ?x))
         (not (clear ?y))
         (clear ?x)
         (handempty)
         (on ?x ?y)
         (on-height ?x ?hx)))

  (:action unstack
    (parameters (?x ?y - container ?hx - height)
     :precondition (and (on ?x ?y) (clear ?x)
                       (craneempty) (on-height ?x ?hx)))
    :effect
    (and (holding ?x)
         (clear ?y)
         (not (clear ?x))
         (not (craneempty))
         (not (on ?x ?y))
         (not (on-height ?x ?hx))))
```

PDDL domain file

A^*, h_{Blocks}

**FAST
DOWNWARD**



Plan in application

```
(stack c7 landstack h1 h0)
(unstack c8 c5 n3)
(stack c8 c7 h2 h1)
(unstack c0 c3 n3)
(stack c0 c5 n3 n2)
(unstack c3 c2 n2)
(stack c3 c8 h3 h2)
(unstack c1 c6 n2)
(stack c1 c2 n2 n1)
(unstack c0 c5 n3)
(stack c0 c1 n3 n2)
(unstack c6 stack2 n1)
(stack c6 c0 n4 n3)
(unstack c5 c4 n2)
(stack c5 stack2 n1 n0)
(unstack c4 stack3 n1)
(stack c4 c5 n2 n1)
(unstack c3 c8 h3)
(stack c3 c4 n3 n2)
(unstack c8 c7 h2)
(stack c8 stack3 n1 n0)
(unstack c7 landstack h1)
(stack c7 c8 n2 n1)
(unstack c6 c0 n4)
(stack c6 c7 n3 n2)
```

Plan



Conclusion

- Explains planning techniques
 - State space
 - Heuristics
 - Concepts of A*
- Interface to a planning system
- Example of a planning domain