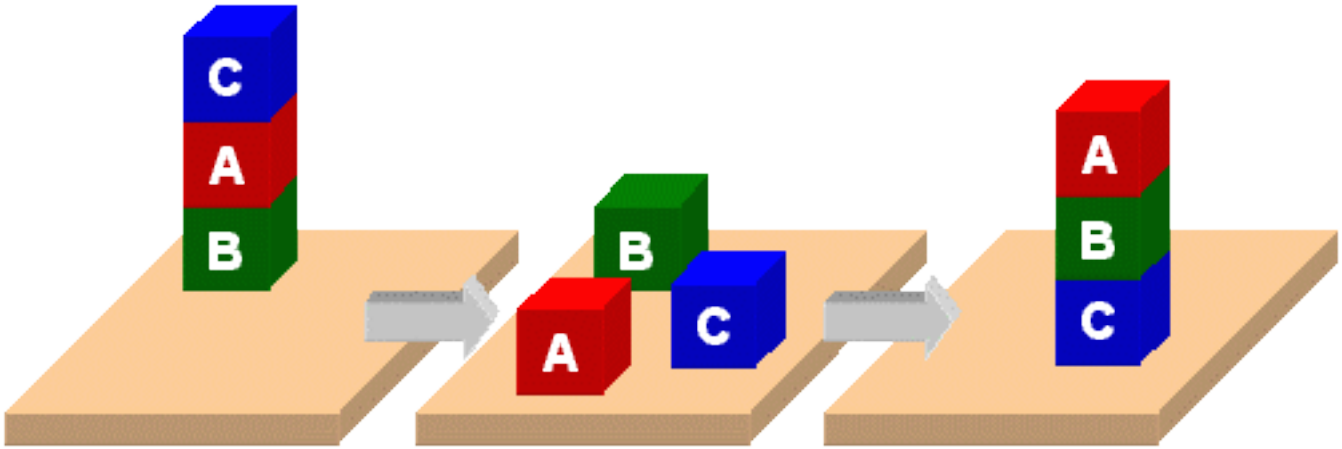
HW5: Planning



PDDL

- Planning Domain Description Language
- Based on STRIPS with various extensions
- Originally defined by Drew McDermott (Yale) and others
- Used in the biennial International Planning Competition (IPC) series
- Many planners use it as a standard input

PDDL Representation

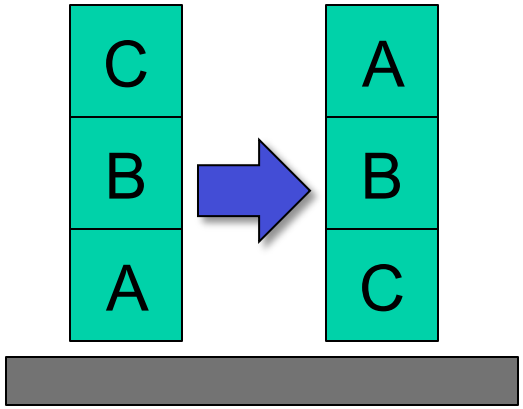
- A task specified via two files: **domain file** and **problem file**
- **Problem file** gives objects, initial state, and goal state
- **Domain file** gives predicates and operators; these may be re-used for different problem files
- **Domain file** corresponds to the transition system, the **problem files** constitute instances in that system

Blocks Word Domain File


```
(define (domain hw5)
  (:requirements :strips)
  (:constants red green blue yellow)
  (:predicates (on ?x ?y) (on-table ?x) (block ?x) … (clean ?x))
  (:action pick-up
     :parameters (?obj1)
     :precondition (and (clear ?obj1) (on-table ?obj1)
                        (arm-empty))
     :effect (and (not (on-table ?obj1))
                  (not (clear ?obj1))
                  (not (arm-empty))
                  (holding ?obj1)))
  … more actions ...)
```

Blocks Word Problem File


```
(define (problem 00)
   (:domain hw5)
   (:objects A B C)
   (:init (arm-empty)
          (block A)
          (color A red)
          (on-table A)
          (block B)
          (on B A)
          (block C)
          (on C B)
          (clear C))
  (:goal (and (on A B) (on B C))))
```



Blackbox planner

- Blackbox planner converts STRIPS-like problems into Boolean satisfiability problems
- Input given in PDDL (domain and problem)
- Solves with a variety of satisfiability engines
- Open source; executables for Linux, Mac, Windows from http://bit.ly/BBpddl
  - Do *blackbox -help* for options
  - Installed on gl as *~finin/pub/blackbox*

Blackbox planner

> **git clone …**

…

> **cd hw5; ls**

domain.pddl p00.pddl …

> **~finin/pub/blackbox -o domain.pddl -f p00.pddl**

blackbox version 43

...

Loading domain file: domain.pddl

Loading fact file: p00.pddl

...

Begin plan

1 (unstack c b)

2 (put-down c)

3 (unstack b a)

4 (stack b c)

5 (pick-up a)

6 (stack a b)

End plan

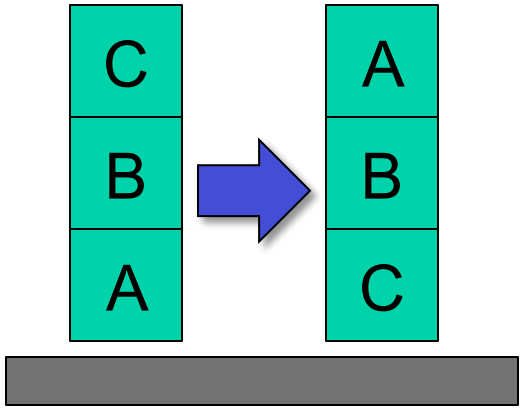
...

Total elapsed time: 0.01 seconds

...

Blocks Word Problem File


```
(define (problem 00)
   (:domain hw5)
   (:objects A B C)
   (:init (arm-empty)
          (block A)
          (color A red)
          (on-table A)
          (block B)
          (on B A)
          (block C)
          (on C B)
          (clear C))
  (:goal (and (on A B) (on B C))))
```



Begin plan
1 (unstack c b)
2 (put-down c)
3 (unstack b a)
4 (stack b c)
5 (pick-up a)
6 (stack a b)
End plan

(1) Extend the domain: new objects

- **Paint sprayers:** Each sprayer can only paint in one color (e.g., red, green, blue, yellow).
- P**aint cans:** A paint can holds only only color of paint.
- **Brushes:** A brush can either be clean or loaded with paint of a particular color.
- **Water bucket:** A water bucket is used to wash brushes.

(2) Extend the domain: new actions

- painting an object a given color with a sprayer
- painting an object a given color with a brush and can
- loading a brush with paint of a given color
- washing a brush in a water bucket to make make it clean

Action preconditions

- To paint an object, it must be on the table and clear
- Painting with a sprayer: just pick it up and spray
- To paint something a color with a brush, it must be loaded with paint of that color.
- To load paint bush with a color, you must be holding brush, brush must be clean and there must be a paint can with that color that is clear. When a brush is loaded with a color it is not clean.
- To wash a brush, making it clean, you must have a water bucket that has nothing on it (i.e., is clear) and you have to be holding the brush

Problem p1.ppd

```
;; There is only one block, A, which is on the table.  A can with
;; red paint is on the table.  There is a clean brush on the
;; table.  Our goal is to have A be red and the arm empty.

(define (problem 1)
  (:domain hw6)
  (:objects ....  )
  (:init (arm-empty)
    ... block A on the table with nothing on it ...
    ... a red paint can on the table with nothing on it ...
    ... a clean brush is on the table with nothing on it ...
                )
  (:goal (and (arm-empty)
              ... A is red …     )))
```