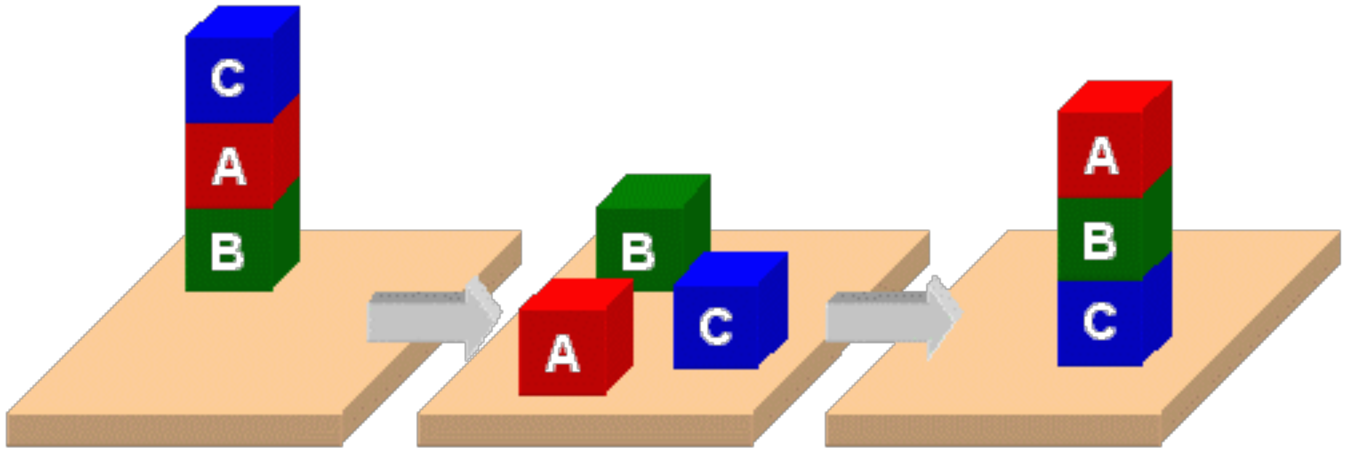
HW: Planning



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

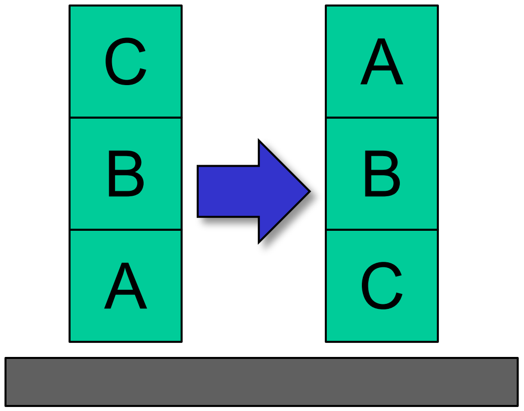
- A task specified via two files: **domain file** and **problem file**
- **Problem file** gives objects, initial state, and goal state
- **Domain file** gives predicates and operators; these may be re-used for different problem files
- **Domain file** corresponds to the transition system, the **problem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

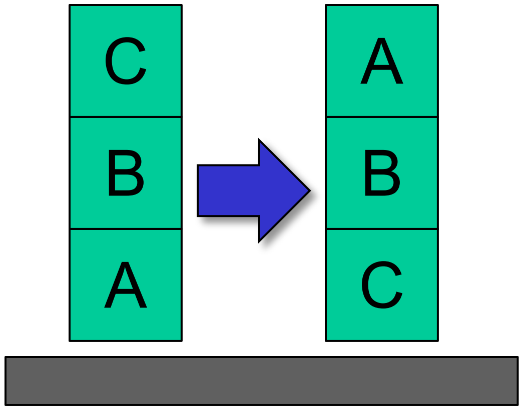
Blocks Word Problem File

```
(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))))
```



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 cans:** A paint can holds only only color of paint. It can also be open (i.e., no lid) or not open (i.e., it’s lid is on)
- **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 brush and can
- loading a brush with paint of a given color
- washing a brush in a water bucket to make make it clean
- Removing the lid of a paint can
- Replacing the lid of a paint can

Action preconditions

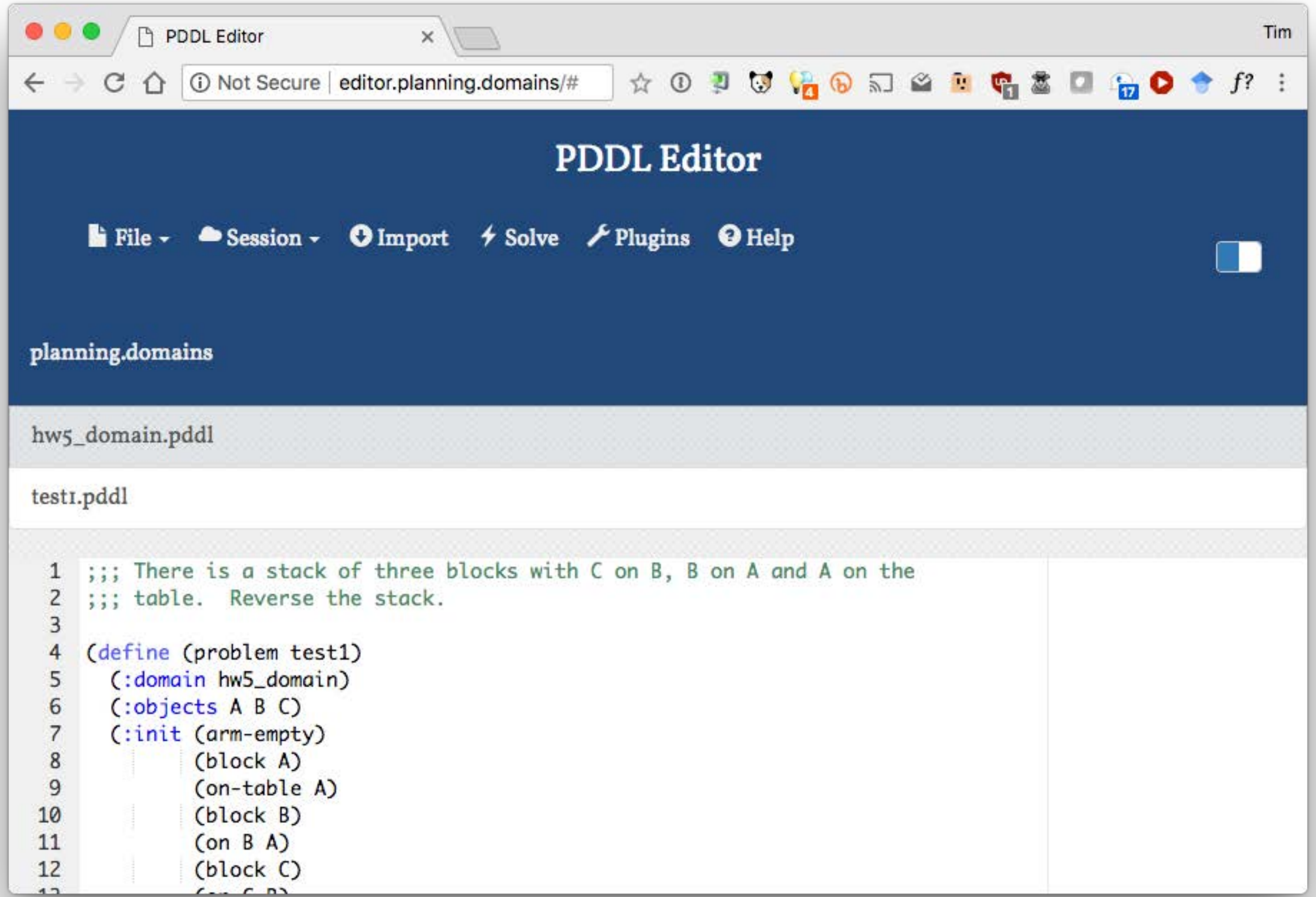
- To paint an object, it must be on the table and clear
- To paint something a color with a brush, it must be loaded with paint of that color
- To load paint bush with a color, you must be holding brush, it must be clean & there must be a paint can with that color that is clear & open. When a brush is loaded with a color it's not clean.
- To wash brush, making it clean, you must have a water bucket with nothing on it (i.e., is clear) and you must be holding brush
- To make paint-can open, it has to be not open and clear and on the table
- To make paint-can not open, it has to be open and clear and on the table

Problem p0.ppd

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

(define (problem p0)
  (:domain hw5_domain)
  (:objects a brush1)
  (:init (arm-empty)
         (block a) (on-table a) (clear a)
         (brush brush1) (on-table brush1)
         (clear brush1) (loaded brush1 red))
  (:goal (and (color a red) (arm-empty))))
```

http://planning.domains/



```
;;; There is a stack of three blocks with C on B, B on A and A on the
;;; table.  Reverse the stack.

(define (problem test1)
  (:domain hw5_domain)
  (:objects A B C)
  (:init (arm-empty)
         (block A)
         (on-table A)
         (block B)
         (on B A)
         (block C)
```

P4

```
;; Block A is on the table, B is on A and C on B.  On the table are a water
;; bucket, cans of red, green and blue paint stacked on each other and a clean
;; brush.  The goal is to make A red, B green and C blue and to have A on B, B
;; on C and C on the table, the cans closed and the brush clean and arm empty.

(define (problem p4)
  (:domain hw5_domain)
  (:objects A B C can1 can2 can3 brush1 wb1)
  (:init (arm-empty)
     (block a) (on-table a)
     (block b) (on b a)
     (block c) (on c b) (clear c)
     (water-bucket wb1) (on-table wb1)(clear wb1)
     (paint-can can1 red) (on-table can1) (not (open can1))
     (paint-can can2 green) (on can2 can1) (not (open can2))
     (paint-can can3 blue) (on can3 can2) (clear can3) (not (open can3))
     (brush brush1)(clean brush1)(on-table brush1)(clear brush1))
  (:goal (and (arm-empty)
 (on a b) (on b c) (on-table c) (clear a)
        (color a red) (color b green) (color c blue)
        (not (open can1)) (not (open can2))
        (not (open can3)) (clean brush1))))
```

http://planning.domains/

Fín