School of Data and Computer Science, Sun Yat-sen University

E07 FF Planner

17341068 Yangfan Jiang

October 24, 2019

Contents

1 Examples **2**

1.1 Spare Tire 2

1.2 Briefcase World 3

2 Tasks **3**

2.1 8-puzzle 3

2.2 Blocks World 4

3 Codes and Results **6**

3.1 8-puzzle 6

3.2 block-world 8

1 Examples

1.1 Spare Tire

domain_spare_tire.pddl

```
(define (domain spare_tire)
  (:requirements :strips :equality:typing)
  (:types physob location)
  (:predicates  (Tire ?x - physob)
                (at ?x - physob ?y - location))

(:action Remove
             :parameters (?x - physob ?y - location)
             :precondition (At ?x ?y)
             :effect (and (not (At ?x ?y)) (At ?x Ground)))

  (:action PutOn
             :parameters (?x - physob)
             :precondition (and (Tire ?x) (At ?x Ground)
                                (not (At Flat Axle)))
             :effect (and (not (At ?x Ground)) (At ?x Axle)))
  (:action LeaveOvernight
             :effect (and (not (At Spare Ground)) (not (At Spare Axle))
                          (not (At Spare Trunk)) (not (At Flat Ground))
                          (not (At Flat Axle)) (not (At Flat Trunk)) ))
 )
```

spare_tire.pddl

```
(define (problem prob)
 (:domain spare_tire)
 (:objects Flat Spare -physob Axle Trunk Ground - location)
 (:init (Tire Flat)(Tire Spare)(At Flat Axle)(At Spare Trunk))
 (:goal (At Spare Axle))
)
```

```
ai2017@osboxes:~/Desktop/spare_tire$ ff -o domain_spare_tire.pddl -f spare_tire.pddl

ff: parsing domain file
domain 'SPARE_TIRE' defined
 ... done.
ff: parsing problem file
problem 'PROB' defined
 ... done.



Cueing down from goal distance:    3 into depth [1]
                                   2            [1]
                                   1            [1]
                                   0

ff: found legal plan as follows

step    0: REMOVE FLAT AXLE
        1: REMOVE SPARE TRUNK
        2: PUTON SPARE


time spent:    0.00 seconds instantiating 9 easy, 0 hard action templates
               0.00 seconds reachability analysis, yielding 11 facts and 8 actions
               0.00 seconds creating final representation with 10 relevant facts
               0.00 seconds building connectivity graph
               0.00 seconds searching, evaluating 4 states, to a max depth of 1
               0.00 seconds total time
```

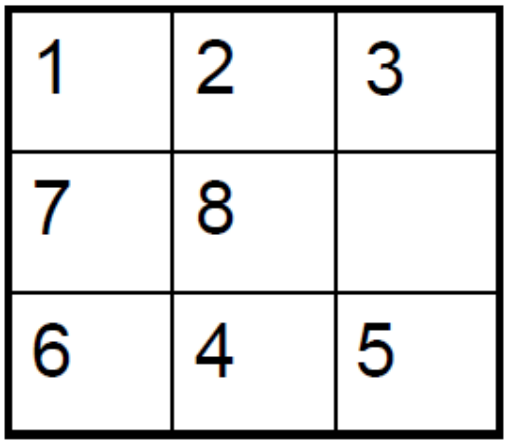
1.2 Briefcase World

Please refer to `pddl.pdf` at page 2. Please pay More attention to the usages of `forall` and `when`.

For more examples, please refer to `ff-domains.tgz` and `benchmarksV1.1.zip`. For more usages of FF planner, please refer to the documentation `pddl.pdf`.

2 Tasks

2.1 8-puzzle



Please complete `domain_puzzle.pddl` and `puzzle.pddl` to solve the 8-puzzle problem.

domain_puzzle.pddl

```
(define (domain puzzle)
  (:requirements :strips :equality:typing)
  (:types num loc)
  (:predicates ())

(:action slide
             :parameters ()
             :precondition ()
             :effect ()
 )
)
```

domain_puzzle.pddl

```
(define (problem prob)
 (:domain puzzle)
 (:objects )
 (:init )
 (:goal ())
)
```

2.2 Blocks World

现有积木若干，积木可以放在桌子上，也可以放在另一块积木上面。有两种操作：

1. $move(x, y)$：把积木x放到积木y上面。前提是积木x和y上面都没有其他积木。
2. $moveToTable(x)$：把积木x放到桌子上，前提是积木x上面无其他积木，且积木x不在桌子上。

Please complete the file `domain_blocks.pddl` to solve the blocks world problem. You should know the usages of `forall` and `when`.

domain_blocks.pddl

```
(define (domain blocks)
  (:requirements :strips :typing:equality
                 :universal-preconditions
                 :conditional-effects)
  (:types physob)
  (:predicates
            (ontable ?x - physob)
            (clear ?x - physob)
            (on ?x ?y - physob))

  (:action move
             :parameters (?x ?y - physob)
             :precondition ()
             :effect ()
             )

  (:action moveToTable
             :parameters (?x - physob)
             :precondition ()
             :effect ( )
 )
```

blocks.pddl

```
(define (problem prob)
 (:domain blocks)
 (:objects A B C D E F - physob)
 (:init (clear A)(on A B)(on B C)(ontable C) (ontable D)
  (ontable F)(on E D)(clear E)(clear F)
)
 (:goal  (and (clear F) (on F A) (on A C) (ontable C)(clear E) (on E B)
         (on B D) (ontable D)) )
 )
```

Please submit a file named E07_YourNumber.pdf, and send it to ai_201901@foxmail.com

3 Codes and Results

3.1 8-puzzle

Codes

domain_puzzle.pddl

```
(define (domain puzzle)
(:requirements :strips :equality:typing)
(:types num loc)
(:predicates  (empty ?x - loc)
              (at ?x - num ?y - loc)
              (near ?x - loc ?y - loc))

(:action slide
            :parameters (?x - num ?z - loc ?y - loc)
            :precondition (and (empty ?y) (at ?x ?z) (near ?z ?y) )
            :effect (and (not (empty ?y)) (empty ?z)
                  (not (at ?x ?z)) (at ?x ?y) )
)
)
```

puzzle.pddl

```
(define (problem prob)
(:domain puzzle)
(:objects n1 n2 n3 n4 n5 n6 n7 n8 - num l1 l2 l3 l4 l5 l6 l7 l8 l9 - loc)
(:init (empty l6)(at n1 l1)(at n2 l2)(at n3 l3)(at n7 l4)(at n8 l5)
       (at n6 l7)(at n4 l8)(at n5 l9)(near l1 l2)(near l1 l4)(near l2 l1)
       (near l2 l3)(near l2 l5)(near l3 l2)(near l3 l6)(near l4 l1)
       (near l4 l5)(near l4 l7)(near l5 l4)(near l5 l2)(near l5 l8)
       (near l5 l6)(near l6 l5)(near l6 l3)(near l6 l9)(near l7 l4)
       (near l7 l8)(near l8 l7)(near l8 l5)(near l8 l9)(near l9 l8)
       (near l9 l6))
```

```
11  (:goal (and (at n1 l1)(at n2 l2)(at n3 l3)(at n4 l4)
12              (at n5 l5)(at n6 l6)(at n7 l7)(at n8 l8)) )
13 )
```

Result

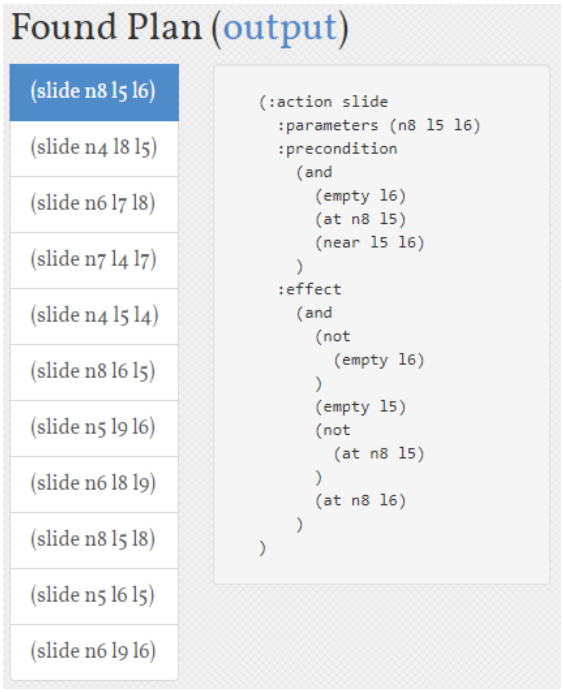


Figure 1: Results

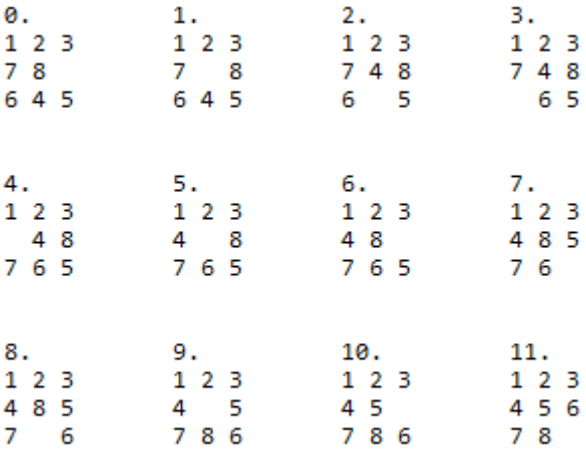


Figure 2: Actions

3.2 block-world

Codes

domain_blocks.pddl

```
(define (domain blocks)
(:requirements :strips :typing:equality
               :universal-preconditions
               :conditional-effects)
(:types physob)
(:predicates
          (ontable ?x - physob)
          (clear ?x - physob)
          (on ?x ?y - physob))

(:action move
           :parameters (?x ?y - physob)
           :precondition (and (clear ?x) (clear ?y) )
           :effect (and
                    (when (ontable ?x)
                      (not (ontable ?x)))
                    (forall (?z - physob)
                      (when (on ?x ?z)
                            (and (not (on ?x ?z))(clear ?z)))
                    )
                    (not (clear ?y))
                    (on ?x ?y)
                  )
  )

(:action moveToTable
           :parameters (?x - physob)
           :precondition (and (not (ontable ?x)) (clear ?x))
           :effect (and (ontable ?x)
```

```
30                          (forall (?z - physob)
31                              (when(on ?x ?z)
32                                  (and (not (on ?x ?z))(clear ?z)))
33                          )
34                         )
35  )
36  )
```

blocks.pddl

```
1   (define (problem prob)
2  (:domain blocks)
3  (:objects A B C D E F - physob)
4  (:init (clear A)(on A B)(on B C)(ontable C) (ontable D)
5   (ontable F)(on E D)(clear E)(clear F)
6 )
7  (:goal  (and ( clear F) (on F A) (on A C) (ontable C)(clear E) (on E B)
8           (on B D) (ontable D)) )
9  )
```

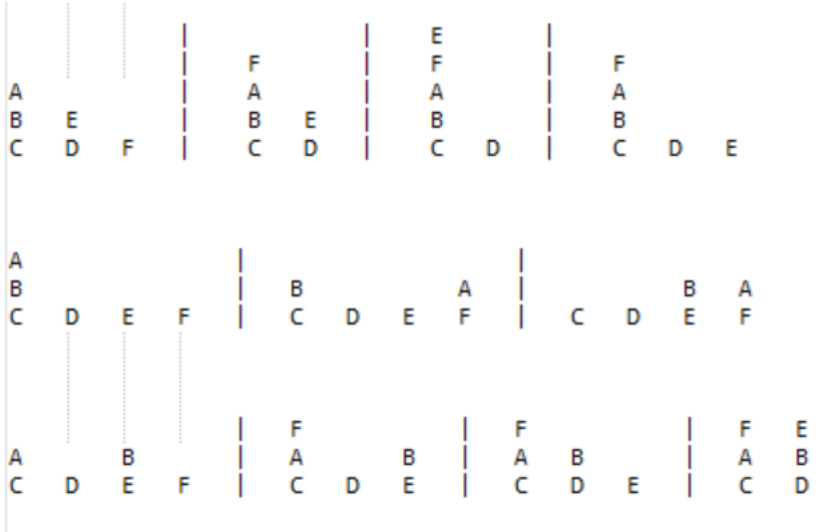


Figure 3: Actions

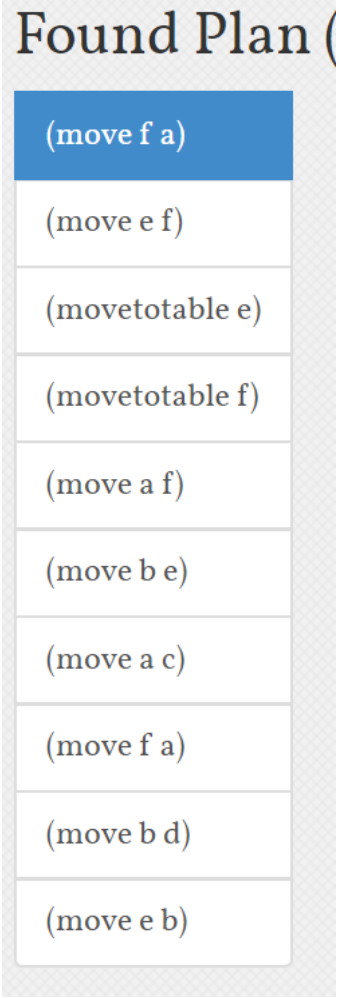


Figure 4: Results