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

E07 FF Planner

17341189 姚森舰

2019 年 10 月 19 日

目录

1 Examples **2**
1.1 Spare Tire 2
1.2 Briefcase World 2

2 Tasks **3**
2.1 8-puzzle 3
2.2 Blocks World 4

3 Codes and Results **5**
3.1 8-puzzle 5
3.2 Blocks World 8

1 Examples

1.1 Spare Tire

```
(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)) ))
 )
```

domain_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))
)
```

spare_tire.pddl

1.2 Briefcase World

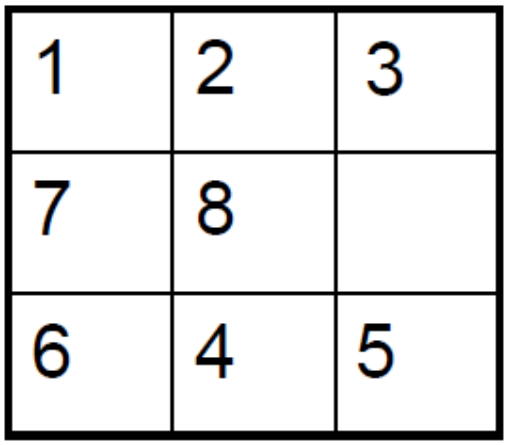
Please refer to `pddl.pdf` at page 2. Please pay More attention to the usages of `forall` and `when`.

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`.

```
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
```

2 Tasks

2.1 8-puzzle



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

```
(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 ())
)
```

domain_puzzle.pddl

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`.

```
(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 ( )
 )
```

domain_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)) )
 )
```

blocks.pddl

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

3 Codes and Results

3.1 8-puzzle

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

    (:action slide
        :parameters (?mov_num - num ?cur_loc - loc ?blank_loc - loc)
        :precondition (and
                        (at ?mov_num ?cur_loc)
                        (at n0 ?blank_loc)
                        (adj ?cur_loc ?blank_loc)
                      )
        :effect (and
                    (at ?mov_num ?blank_loc)
                    (not (at ?mov_num ?cur_loc))
                    (at n0 ?cur_loc)
```

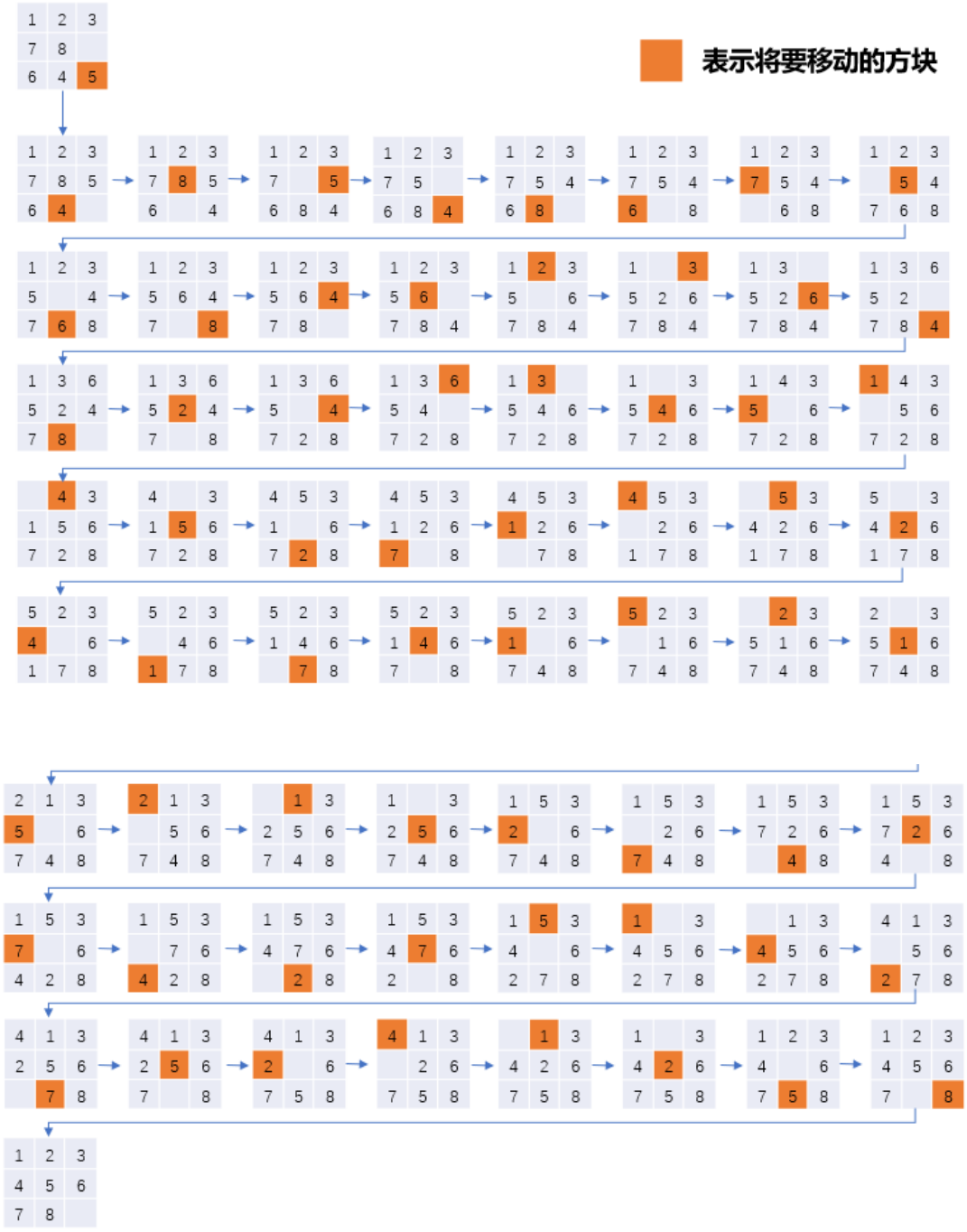
```
                    (not (at n0 ?blank_loc))
        )
    )
)
```

domain_puzzle.pddl

```
(define (problem prob)
    (:domain puzzle)
    (:objects n0 n1 n2 n3 n4 n5 n6 n7 n8 - num
              l1 l2 l3 l4 l5 l6 l7 l8 l9 - loc)

    (:init  (at n1 l1) (at n2 l2) (at n3 l3)
            (at n7 l4) (at n8 l5) (at n0 l6)
            (at n6 l7) (at n4 l8) (at n5 l9)

            (adj l1 l2) (adj l1 l4)
            (adj l2 l1) (adj l2 l3) (adj l2 l5)
            (adj l3 l2) (adj l3 l6)
            (adj l4 l1) (adj l4 l5) (adj l4 l7)
            (adj l5 l4) (adj l5 l2) (adj l5 l6) (adj l5 l8)
            (adj l6 l3) (adj l6 l5) (adj l6 l9)
            (adj l7 l4) (adj l7 l8)
            (adj l8 l7) (adj l8 l9) (adj l8 l5)
            (adj l9 l6) (adj l9 l8)
    )

    (:goal (and
                (at n1 l1) (at n2 l2) (at n3 l3)
                (at n4 l4) (at n5 l5) (at n6 l6)
                (at n7 l7) (at n8 l8) (at n0 l9)
      )
    )
)
```

problem_puzzle.pddl

以下是对 PDF 中的例子的求解结果，步数比较多，截图太长了，画出来更方便检查。后面再找了一个简单一点的例子截图。

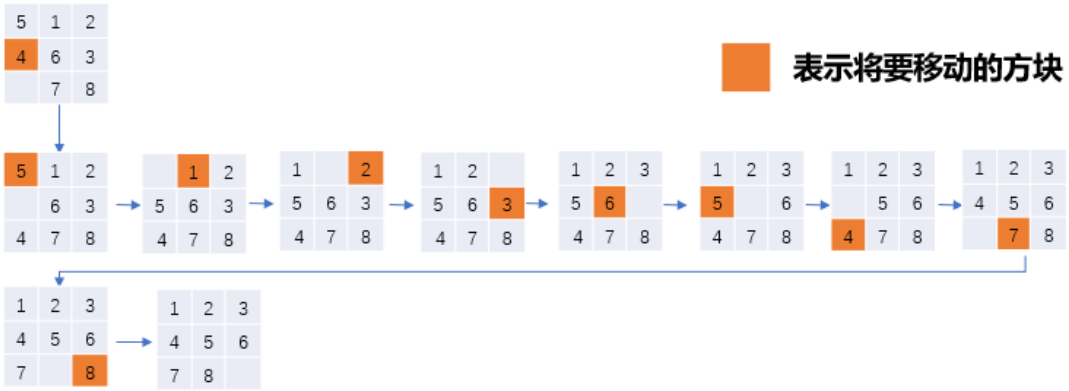


下面是另一个例子:

Found Plan (output)

(slide n4 l4 l7)
(slide n5 l1 l4)
(slide n1 l2 l1)
(slide n2 l3 l2)
(slide n3 l6 l3)
(slide n6 l5 l6)
(slide n5 l4 l5)
(slide n4 l7 l4)
(slide n7 l8 l7)
(slide n8 l9 l8)

```
(:action slide
  :parameters (n4 l4 l7)
  :precondition
    (and
      (at n4 l4)
      (at n0 l7)
      (adj l4 l7)
    )
  :effect
    (and
      (at n4 l7)
      (not
        (at n4 l4)
      )
      (at n0 l4)
      (not
        (at n0 l7)
      )
    )
)
```



3.2 Blocks World

```
(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
                    (
                    forall (?z - physob)
                      (when (on ?x ?z)
                        (and (clear ?z) (not (on ?x ?z)))
                      )
                    )
                    (not (clear ?y))
                    (on ?x ?y)
                )
    )

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

domain_block.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)    )
```

```
    )
)
```

problem_block.pddl

以下是运行结果：

经验证结果正确。这里面需要注意的是如果 x 在 z 上，当 x 从 z 上移走时，其中一个后果是 z 上面就 clear 了，不加这句的话就会出错。