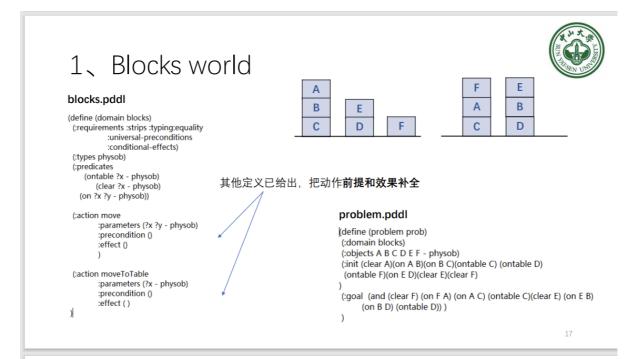
# 中山大学计算机学院 本科生实验报告 (2023 学年春季学期)

# 课程名称: Artificial Intelligence 人工智能

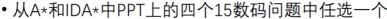
教学班级	专业 (方向)	
学号 2233 6173	姓名 罗弘杰	

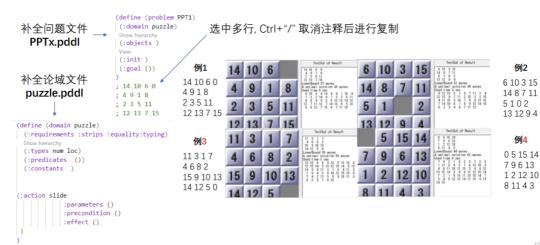
### 实验题目

使用PDDL语言求解一些规划问题



# 2、15puzzle





## 实验内容

规划问题可以使用STRIPS语言以及ADL语言定义和表示:

STRIPS语言使用带有precondition, effect的action来表示规划的改变问题。

ADL原因在其基础之上添加了action的effect的条件效果和全称效果以及对precondition的任意前提条件

PDDL继承了STRIPS和ADL的特性,并在此基础上进行了扩展和改进,成为了一种更通用、更灵活的规划语言

求解规划问题:

规定state状态的表示方式,以及若干可采取的Action使用规定的语言;

对初始状态和最终状态的state讲行定义,;

使用回溯算法的前向反馈算法进行求解,若发现到达最终的目标状态则求解成功,否则回溯到上一步

2. 伪代码

```
function parsePDDL(domainFile, problemFile):
   domain = parseDomain(domainFile) // 解析领域描述文件
   problem = parseProblem(problemFile) // 解析问题描述文件
   return domain, problem
function parseDomain(domainFile):
   domain = {}
   // 从领域描述文件中读取内容,并解析为相应的数据结构
   // 解析物体、谓词、操作等信息
   // 将解析后的信息存储到 domain 数据结构中
   return domain
function parseProblem(problemFile):
   problem = {}
   // 从问题描述文件中读取内容,并解析为相应的数据结构
   // 解析初始状态和目标状态等信息
   // 将解析后的信息存储到 problem 数据结构中
   return problem
// 示例用法
domainFile = "domain.pddl"
problemFile = "problem.pddl"
domain, problem = parsePDDL(domainFile, problemFile)
// 现在可以使用 domain 和 problem 来执行规划
```

### 3. 关键代码展示 (带注释)

任务1 (BLOCK)

论域:若干个积木以及以及他们的相互接触状态 (用on (x, y)表示x 在 y上方),

在桌子上的状态 (ontable (x) 表示x在桌子上),

以及上方无积木块 (用clear (x) 表示 x上方没有积木)

桌子可以放任意多个积木, 每个积木上只能放一个积木在上方;

MOVE (x, y):条件 (clear(x), clear(y), x!= y)

```
效果(not clear(y), on(x, y ), 任意z if on(x, z) then not on(x, z)) movetotable (x):条件(clear(x))  
效果(ontable(x), 任意z if on(x, z) then not on(x, z))
```

### 问题:

linitial: (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)) 需要补充两个动作MOVE和MOVETOTABLE;

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

```
任务2(15 puzzle):
论域:矩阵上不同位置的邻接关系(用neighbor(x,y)表示x与y邻接),
矩阵上位置为空(blank (x)表示x上没有滑块),
矩阵上位置是某个数字(at(tile, pos)表示该滑块在该位置上)
slide (tile, pos1, pos2):条件:(at (tile, pos 1), blank(pos2), neighbour(pos1, pos2)),
效果: (at ?tile pos2), (blank pos1) (not (blank pos2)),
```

```
#puzzle.pddl;
(define (domain puzzle)
  (:requirements :strips :typing:equality
                 :universal-preconditions
                 :conditional-effects)
  (:types num loc)
  (:predicates (at ?tile ?pos)
        (blank ?pos)
        (neighbor ?pos_1 ?pos_2)
  )
  (:action slide
             :parameters (?tile ?pos_1 ?pos_2)
             :precondition (and (at ?tile ?pos_1) (blank ?pos_2) (neighbor ?
pos_1 ?pos_2))
             :effect (and (at ?tile ?pos_2) (blank ?pos_1) (not (blank ?
pos_2)) (not (at ?tile ?pos_1)))
 )
)
```

问题 (PPT4):

矩阵如下:

; 0 5 15 14

;79613

; 1 2 12 10

;81143

代码解释:

初始状态脚本生成器:

由于滑块的邻接关系是固定不变的,只需要改变矩阵上每个滑块的位置就可以了,写了一个脚本来快速生成初始状态:

```
#coding=utf-8

''' ppt4
0 5 15 14
7 9 6 13
1 2 12 10
8 11 4 3
'''

TXT = r"cod\domains\puzzle\ppt4.txt"

TXT_OUT = r"coda\domains\puzzle\ppt4_out.txt"
def input(): #read the file
    with open(TXT, "r") as f:
```

```
return f.read()

r=[]

tmp = input().split("\n")

for i in range(4): #get the initial state
    temp = tmp[i].split(" ")
    r+=['(at tile_%s pos_%d%d)'%(x,i+1,j+1) for j,x in enumerate(temp)]

def output():
    with open(TXT_OUT, "w") as f:
        f.write("\n".join(r))

output()
```

```
#PPT4_Problem.PDDL
(define (problem PPT4)
(:domain puzzle)
 (:objects pos_11 pos_12 pos_13 pos_14
        pos_21 pos_22 pos_23 pos_24
        pos_31 pos_32 pos_33 pos_34
        pos_41 pos_42 pos_43 pos_44
        tile_1 tile_2 tile_3 tile_4 tile_5 tile_6 tile_7 tile_8
        tile_9 tile_10 tile_11 tile_12 tile_13 tile_14 tile_15)
;initial
; 0 5 15 14
; 7 9 6 13
: 1 2 12 10
; 8 11 4 3
(:init (blank pos_11)
        (at tile_5 pos_12)
        (at tile_15 pos_13)
        (at tile_14 pos_14)
        (at tile_7 pos_21)
        (at tile_9 pos_22)
        (at tile_6 pos_23)
        (at tile_13 pos_24)
        (at tile_1 pos_31)
        (at tile_2 pos_32)
        (at tile_12 pos_33)
        (at tile_10 pos_34)
        (at tile_8 pos_41)
        (at tile_11 pos_42)
        (at tile_4 pos_43)
        (at tile_3 pos_44)
         (neighbor pos_11 pos_12) (neighbor pos_12 pos_11)
```

```
(neighbor pos_12 pos_13) (neighbor pos_13 pos_12)
        (neighbor pos_13 pos_14) (neighbor pos_14 pos_13)
        (neighbor pos_21 pos_22) (neighbor pos_22 pos_21)
        (neighbor pos_22 pos_23) (neighbor pos_23 pos_22)
        (neighbor pos_23 pos_24) (neighbor pos_24 pos_23)
        (neighbor pos_31 pos_32) (neighbor pos_32 pos_31)
        (neighbor pos_32 pos_33) (neighbor pos_33 pos_32)
        (neighbor pos_33 pos_34) (neighbor pos_34 pos_33)
        (neighbor pos_41 pos_42) (neighbor pos_42 pos_41)
        (neighbor pos_42 pos_43) (neighbor pos_43 pos_42)
        (neighbor pos_43 pos_44) (neighbor pos_44 pos_43)
        (neighbor pos_11 pos_21) (neighbor pos_21 pos_11)
        (neighbor pos_12 pos_22) (neighbor pos_22 pos_12)
        (neighbor pos_13 pos_23) (neighbor pos_23 pos_13)
        (neighbor pos_14 pos_24) (neighbor pos_24 pos_14)
        (neighbor pos_21 pos_31) (neighbor pos_31 pos_21)
        (neighbor pos_22 pos_32) (neighbor pos_32 pos_22)
        (neighbor pos_23 pos_33) (neighbor pos_33 pos_23)
        (neighbor pos_24 pos_34) (neighbor pos_34 pos_24)
        (neighbor pos_31 pos_41) (neighbor pos_41 pos_31)
        (neighbor pos_32 pos_42) (neighbor pos_42 pos_32)
        (neighbor pos_33 pos_43) (neighbor pos_43 pos_33)
        (neighbor pos_34 pos_44) (neighbor pos_44 pos_34)
 (:goal (and (at tile_1 pos_11)
            (at tile_2 pos_12)
            (at tile_3 pos_13)
            (at tile_4 pos_14)
            (at tile_5 pos_21)
            (at tile_6 pos_22)
            (at tile_7 pos_23)
            (at tile_8 pos_24)
            (at tile_9 pos_31)
            (at tile_10 pos_32)
            (at tile_11 pos_33)
            (at tile_12 pos_34)
            (at tile_13 pos_41)
            (at tile_14 pos_42)
            (at tile_15 pos_43))
)
;goal
; 1 2 3 4
; 5 6 7 8
; 9 10 11 12
; 13 14 15 0
)
```

4. 创新点&优化 (如果有)

设计了一个脚本输出滑块矩阵的初始状态算吗?

# 实验结果及分析

1. 实验结果展示示例(可图可表可文字,尽量可视化)

### 积木块问题

#### 生成求解成功

```
(define (problem prob)
                                                         moveToTable A
      (:domain blocks)
     Show hierarchy
(:objects A B C D E F - physob)
                                                                moveToTable B
                                                                   move E C
      (:init (clear A)(on A B)(on B C)(ontable
       (ontable F)(on E D)(clear E)(clear F)
                                                                            moveToTable F
      (:goal (and (clear F) (on F A) (on A C)
                                                                                move A C
              (on B D) (ontable D))
                                                                                  move F A
                                                        physob
                                                         В
     输出 调试控制台 终端
                                                                                          0.00300: (MUVE € C)
0.00400: (MOVE B D)
0.00500: (MOVE E B)
0.00600: (MOVE F C)
0.00700: (MOVETOTABLE F)
0.00800: (MOVE A C)
0.00900: (MOVE F A)
Metric: 0.0090000000000000001
Makespan: 0.0090000000000000001
States evaluated: undefined
Planner found 1 plan(s) in 3.508secs.
```

validate成功

```
Checking next happening (time 0.005)
Deleting (clear b)
Deleting (on e c)
Adding (on e b)
Adding (clear c)
Checking next happening (time 0.006)
Deleting (clear c)
Deleting (on f a)
Adding (on f c)
Adding (clear a)
Checking next happening (time 0.007)
Deleting (on f c)
Adding (ontable f)
Adding (clear c)
Checking next happening (time 0.008)
Deleting (clear c)
Adding (on a c)
Checking next happening (time 0.009)
Deleting (clear a)
Adding (on f a)
Plan executed successfully - checking goal
Plan valid
Final value: 10
Successful plans:
Value: 10
C:\Users\rogers\AppData\Local\Temp\plan--19068-rsa9WpZt3REC-.pddl 10
```

### 15 puzzle 问题

求解的是ppt4的案例

求解成功:

```
(:a) PPT4.pddl × ··· ¬ Planner output ×
                                                               slide tile_6 pos_12 pos_11
labdata > domains > puzzle > (:a) PPT4.pddl > { } problem
                                                               slide tile_7 pos_22 pos_12
                                                               slide tile 15 pos 23 pos 22
                                                               slide tile_14 pos_13 pos_23
                                                               slide tile_7 pos_12 pos_13
                                                               slide tile_15 pos_22 pos_12
      (:init (blank pos_11)
                                                                slide tile_14 pos_23 pos_22
              (at tile_5 pos_12)
                                                                slide tile_7 pos_13 pos_23
              (at tile_15 pos_13)
 20
                                                                slide tile_13 pos_14 pos_13
              (at tile_14 pos_14)
                                                                slide tile_10 pos_24 pos_14
              (at tile_7 pos_21)
                                                                slide tile_12 pos_34 pos_24
             (at tile 9 pos 22)
                                                                slide tile_3 pos_44 pos_34
             (at tile_6 pos_23)
                                                                slide tile_2 pos_43 pos_44
              (at tile_13 pos_24)
                                                                slide tile_4 pos_42 pos_43
             (at tile 1 pos 31)
    输出 调试控制台 终端
                                                                                                 问题
                                                                              Planner output
0.39700: (SLIDE TILE_12 POS_24 POS_34)
0.39800: (SLIDE TILE_7 POS_23 POS_24)
0.39900: (SLIDE TILE_11 POS_33 POS_23)
0.40000: (SLIDE TILE_8 POS_43 POS_33)
0.40100: (SLIDE TILE_15 POS_44 POS_43)
0.40200: (SLIDE TILE_12 POS_34 POS_44)
0.40300: (SLIDE TILE_8 POS_33 POS_34)
0.40400: (SLIDE TILE_11 POS_23 POS_33)
0.40500: (SLIDE TILE_7 POS_24 POS_23)
0.40600: (SLIDE TILE_8 POS_34 POS_24)
0.40700: (SLIDE TILE_12 POS_44 POS_34)
Metric: 0.40700000000000003
Makespan: 0.40700000000000003
States evaluated: undefined
Planner found 1 plan(s) in 30.612secs
```

#### validate成功:

```
Deleting (at tile_7 pos_24)
Adding (at tile_7 pos_23)
Adding (blank pos_24)
Checking next happening (time 0.406)
Deleting (blank pos_24)
Deleting (at tile 8 pos 34)
Adding (at tile_8 pos_24)
Adding (blank pos_34)
Checking next happening (time 0.407)
Deleting (blank pos_34)
Deleting (at tile_12 pos_44)
Adding (at tile_12 pos_34)
Adding (blank pos_44)
Plan executed successfully - checking goal
Plan valid
Final value: 408
Successful plans:
C:\Users\rogers\AppData\Local\Temp\plan--19068-uMQ8QkpbZDQg-.pddl 408
```

### 参考资料

PS:可以自己设计报告模板,但是内容必须包括上述的几个部分,不需要写实验感想