

PDDL实验



本节安排

- PDDL简介
- 规划器使用和语法
- 实验任务



PDDL简介

- PDDL = **P**lanning **D**omain **D**efinition **L**anguage
- 是对人工智能规划语言进行标准化的一种尝试。PDDL提供了标准化，好处是能使研究**更易于重用**和比较。
- 与针对特定论域的系统相比，这要付出一些表达能力的代价。
- 更多内容：
 - [PDDL Domain - Planning.wiki](https://pddl.github.io/) 包含论域、问题定义、依赖项的例子
 - [STRIPS — AI Planning in PDDL-Descripted World \(triddu33.github.io\)](https://github.com/triddu33/pddl) pddl基本介绍和例子
 - planning.domains 线上规划网站（不稳定）



实验环境

- FF (Fast Forward) 规划器 [链接](#)
 - 采用经典的前向搜索方法
 - 结合启发式算法提高效率
 - 不保证最优性
- 线上使用 <http://editor.planning.domains>
- VSCode插件 (详解)
- Linux可安装FFv2.3.tgz (详见压缩包文档)

PDDL 环境及语法



- Step 1 [安装]：安装pddl插件

The screenshot shows the Visual Studio Code interface with the Extensions view open. The search bar at the top of the Extensions panel contains the text "pddl". Below the search bar, the "PDDL" extension by Jan Dolejsi is highlighted with a red box. To the right of the extension list, the details for the "PDDL" extension are displayed, including its version (v2.25.7), author (Jan Dolejsi), download count (28,556), and a 5-star rating (13 reviews). The extension is described as "Planning Domain Description Language support". Below the description, there are buttons for "Disable", "Uninstall", and a settings gear icon. A message states "This extension is enabled globally." The left sidebar shows the Explorer, Search, Source Control, Run and Debug, and Extensions views. The Extensions view is currently active, and the "PDDL" extension is selected. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, and Help.

EXTENSION: PDDL X

pddl

PDDL 3684ms
Planning Domain Descripti...
Jan Dolejsi

ppd 62
pdd
rbn **Install**

VGUI Syntax High... 4K
Syntax Highlighting and C...
pddstudio **Install**

hyrn-plugin 18
pddcode **Install**

lizard-plugin 429
lizard plugin
pddcode **Install**

PDDL v2.25.7
Jan Dolejsi | 28,556 | ★★★★★ (13)
Planning Domain Description Language support
Disable **Uninstall** ⚙️
This extension is enabled globally.

DETAILS FEATURE CONTRIBUTIONS CHANGELOG RUNTIME STATUS

Planning Domain Description Lang

Build passing

This extension makes VS Code a great place for modeling planning d

This extension brings PDDL to the family of first-class languages with
and empower the expert by following features:

- planning domain modeling



- Step 2 [配置]：Ctrl+shift+p 搜索 pddl: Show overview page
 - 下载VAL（一键自动下载），用于辅助解析和验证

EXTENS... pddl

PDDL 3684ms
Planning Domain Descripti...
Jan Dolejsi

ppd 62
pdd
rbn
[Install](#)

VGUI Syntax High... 4K
Syntax Highlighting and C...
pddstudio
[Install](#)

hyrn-plugin 18
pddcode
[Install](#)

lizard-plugin 429
lizard plugin
pddcode
[Install](#)

AI Planning and PDDL support in VS Code

- Improve usability by installing the [VS Code Icons for PDDL files](#) to quickly distinguish PDDL files. [Install icons e.g. \(a\)](#)
- When processing files using command-line tools, it is simpler to enable file auto-saving. [Enable auto-save](#)
- It is also recommended to go run the [Git: Initialize Repository](#) command to enjoy safety of version control and never miss a working version.
- [Download the latest build of VAL tools to get a PDDL parser, to be able to validate and evaluate plans.](#) [Download](#) plan validation tools
- Enable on-type formatter to save time and make your models readable without wasting time formatting w... [Click to initiate download. You will be able to see what is being downloaded and from where...](#)
- You can switch it off again using the `editor.formatOnType` setting.
- Enable bracket pair guide lines for better readability of your PDDL. See [example](#). [Enable bracket colorization](#) ...for PDDL only
- You can switch it off again using the `editor.guides.bracketPairs` setting. [See bracket pair guides settings...](#)

Did you know that `Ctrl + /` comments out the current line? Press it again to un-comment it.

Getting started

Try [Hello World](#) example
Generate [Nunjucks templated](#) problem file sample
See or [clone PDDL samples](#)

Configuration

Planning engine {} +

- ☒ `http://solver.planning.domains/solve`
- ☐ Planning as a service (preview)

Read [more info about PDDL planners](#)

Output into ☒ Output window ☐ Terminal ☐ Search debugger

PDDL parser

undefined

See [more info about PDDL parsers](#)

Plan Validator

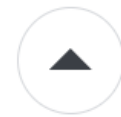
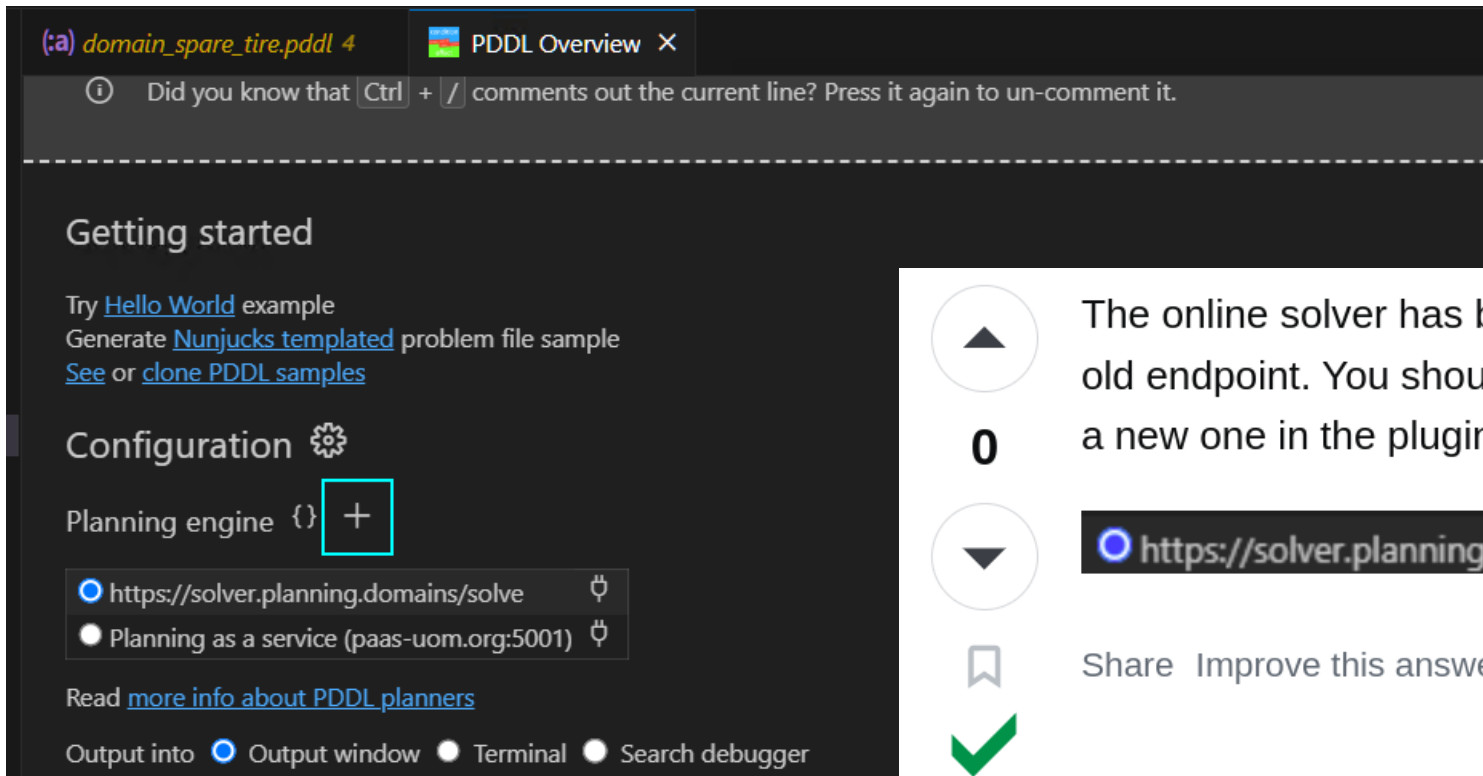
undefined

Clone and compile [VAL from GitHub](#) or...
[Download](#) plan validation tools

☒ Show this overview page once a day when using PDDL



- Step 2 [配置]: Ctrl+shift+p 搜索 pddl: Show overview page
 - Planning engine->Planning-as-a-service->填入下面的网址
 - <https://solver.planning.domains:5001/package>
 - 注: 在pddl文件中不要使用中文注释, 服务器端可能无法正确识别



The online solver has been updated, and it looks like you're using an old endpoint. You should use the "Planning-as-a-Service" planner (add a new one in the plugin settings), and configure it like this:

0



<https://solver.planning.domains:5001/package>



Share Improve this answer Follow

answered Apr 11 at 19:01



haz

650 4 12



<https://solver.planning.domains:5001/package>



• Step 3 [编写]: 编写domain文件 (xxx.pddl)

- domain文件示例:
- 使用括号进行匹配!

```
week 9 PDDL > domains > spare_tire > (:a) spare_tire.pddl > {} Remove
1  (define (domain spare_tire)
2    (:requirements :strips:equality:typing:disjunctive-preconditions:negative-preconditions)
   Show hierarchy
3    (:types physob location)
4    (:predicates (Tire ?x - physob) 1
5                  (At ?x - physob ?y - location) 3 2 2)
6
7    (:constants
8      Ground Axle - location
9      Flat - physob
10   )
11
12   (:action Remove
13     :parameters (?x - physob ?y - location)
14     :precondition (At ?x ?y)
15     :effect (and (not (At ?x ?y)) (At ?x Ground))
16   )
17
18   (:action PutOn
19     :parameters (?x - physob)
20     :precondition (and (Tire ?x) (At ?x Ground) (not (At Flat Axle)))
21     :effect (and (not (At ?x Ground)) (At ?x Axle))
22   )
23   ; (:action LeaveOvernight
24     ; :effect (and (not (At Spare Ground))(not (At Spare Axle))(not (At Spare Trunk))
25     (not (At Flat Ground))(not (At Flat Axle))(not (At Flat Trunk)))
26   ; )
27 )
```

定义 **domain** 论域名

:requirements 依赖项

- strips (strips求解)
- equality (允许使用谓词=)
- typing 使用类型
- disjunctive-preconditions 使前提条件可以带析取 (用析取但没加该依赖项会报错)
- negative-preconditions (报warning, 可不加)
- 动作前提默认可使用合取和否定

定义**类型**, **谓词**tire和at, **动作**。

- 用?x表示变量x, - type_name表示类型
- (action a :parameters() :precondition() :effect())

注意! 前提条件之间可能有合取、析取、否定等关系, 使用括号和 and or not连接。

- (and (···) (···))
- (and (or (···) (not (···))) (not (···)))

常量constants

- 需要的时候可以用:constants在domain文件中定义一些对象, 如图中的Gound, Axle_g和 Flat



- Step 3 [编写]: 编写problem文件 (xxx.pddl)
 - problem文件示例:

```
week 9 PDDL > domains > spare_tire > (:a) problem.pddl > ...
1  (define (problem prob)
2    (:domain spare_tire)
   Show hierarchy
3    (:objects Spare - physob Trunk - location)
4
   View
5    (:init
6      (Tire Flat)
7      (Tire Spare)
8      (At Flat Axle)
9      (At Spare Trunk)
10
11    )
12    (:goal
13      (At Spare Axle)
14    )
15  )
16
```

定义 **problem** 问题名 指定 **domain** 论域

- :objects 定义实体
 - 用 item - name 表示变量
 - 用 a b - name1 c d e - name2 表示 ab类型为 name1 而 cde为name2
- :init 初始化, 列出一组谓词
- :goal 目标, 给出目标结果 (多个合取时用and连接)

注意! 与前提、结果、goal不一样, **init不需要使用**(and()())的格式连接。

*在problem文件里不需要再次声明domain文件中的常量 (constant)



- Step 3 [编写]: 进阶

- effect中可能有条件效果, 可以用(`when (conditions) (effects)`) 表示, 意为当满足条件`conditions`的时候计算谓词2。
 - 需要依赖`conditional-effects`

The semantics of (`when` P E) are as follows: If P is true before the action, then effect E occurs after. P is a *secondary precondition* [?]. The action is feasible even if P is false, but the effect E occurs only if P is true.

- `forall`用于循环, 如(`forall (?z - block) (clear ?z)`) 表示所有块都`clear`。
 - 可以嵌套使用, 但要注意先对哪个变量进行循环
- (`forall (?z - block) (when (clear ?z) (…))`) 对于所有`block`, 如果`clear`则…
- 可参考压缩包中的`pddl.pdf` p3/p10
- 更多内容可参考本课件p3链接。



- Step 4 [求解]: 方式1:通过Testing进行测试。
 - 点击侧边栏上的Testing->刷新->点击Run->选择planner
 - json配置文件已经都填好了, 只需要补充完对应的文件后执行上述操作

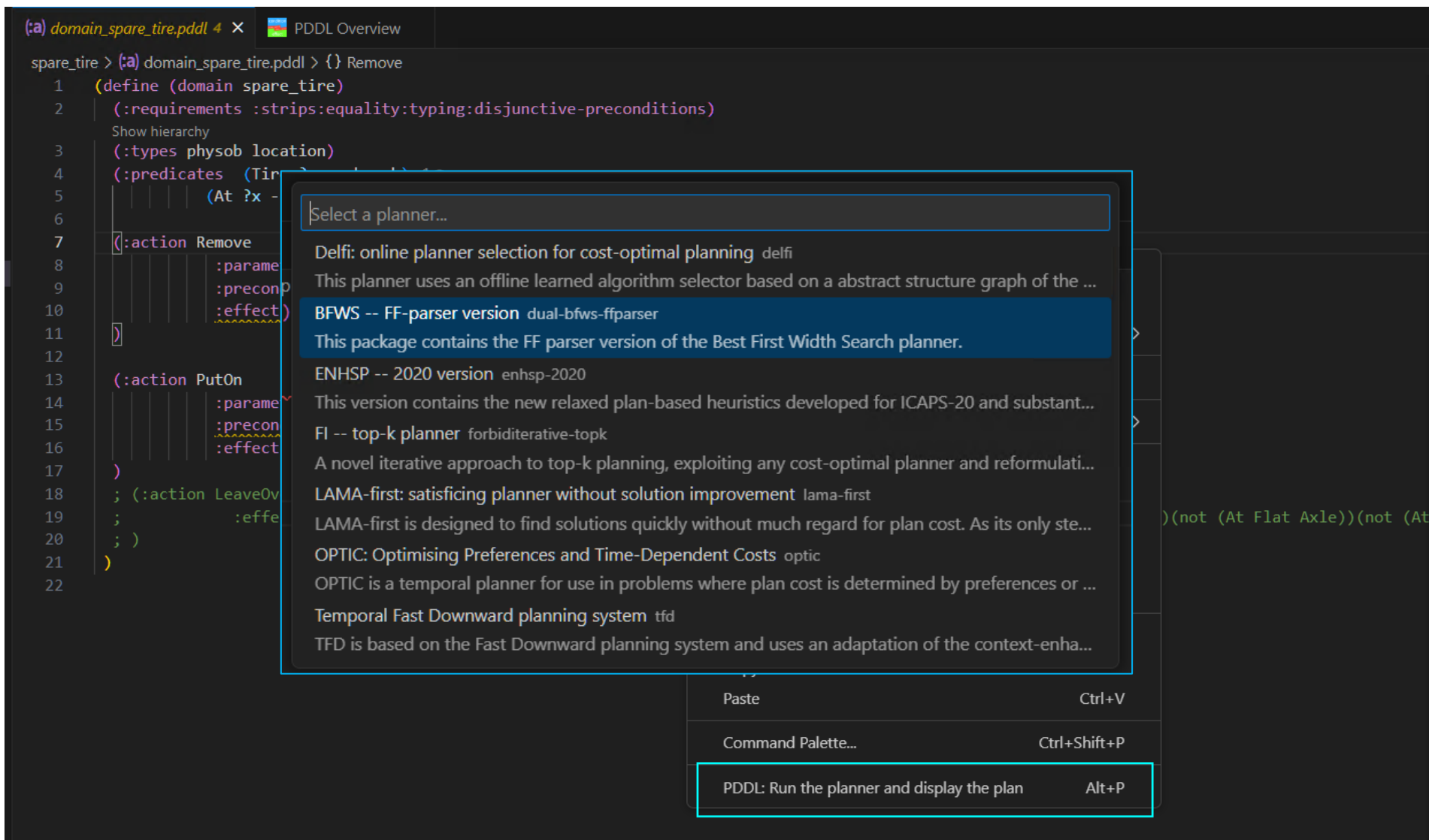
The screenshot shows the testing interface with the following components:

- Left Sidebar:** A tree view showing the project structure. The 'spare_tire' folder is selected, and the 'problem.pddl' file is highlighted. A red box highlights the 'Run' button at the bottom of the sidebar.
- Center Panel:** A 'Select a planner...' dialog box is open, listing several planners. A red box highlights the 'BFWS -- FF-parser version dual-bfws-ffparser' option, which is described as 'This package contains the FF parser version of the Best First Width Search planner.'
- Right Panel:** The 'Planner output' window displays the results of the search. It shows a plan found with a cost of 3, completed in 9.80001e-05 seconds. The plan consists of three actions: (REMOVE SPARE TRUNK), (REMOVE FLAT AXLE), and (PUTON SPARE). The output also shows the metric (0.002) and makespan (0.002).

Arrows point from the text '求解结果' (Solution Result) to the 'Planner output' window and the 'Run' button.



- Step 4 [求解]: 方式2:右击运行
 - 右击domain文件或problem文件->选择PDDL: Run...->进一步选择planner
 - 若同个文件夹下还有多个pddl文件, 会让你进一步进行选择





- 扩展 [验证]: Validate Plan
 - 右上方可以选择导出plan

The screenshot displays the 'PDDL Test Report' interface. On the left, a sidebar shows a file tree with 'domains', 'spare_tire', and 'domain_spare_tire' folders, with 'spare_tire.pddl' selected. The main area is divided into two panes. The top pane, titled 'PDDL Test Report', contains a table with test results:

Test case	sec.	error
domains/spare_tire/domain_spare_tire.ptest.json		
spare_tire.pddl	2.42	

The bottom pane, titled 'Planner output', shows a plan with actions: 'Remove Spare Trunk', 'Remove Flat Axle', and 'PutOn Spare'. A context menu is open over the 'PutOn Spare' action, listing options: 'Generate plan report', 'Export as .plan file...', 'Creates PDDL test with expected test assertion', 'Change width...', 'Change report width...', and 'Select the width of the next exported report (in pixels)'. The 'Export as .plan file...' option is highlighted. The bottom status bar shows 'Planner output' and various icons.

PROBLEMS 8 OUTPUT

total time: 7.49993e-05
Nodes generated during search: 8
Nodes expanded during search: 5
Plan found with cost: 3
Fast-BFS search completed in 7.49993e-05 secs

Plan found:
0.00000: (REMOVE SPARE TRUNK)
0.00100: (REMOVE FLAT AXLE)
0.00200: (PUTON SPARE)
Metric: 0.002
Makespan: 0.002
States evaluated: undefined
☑ spare_tire.pddl (2.422 sec)
Planner found 1 plan(s) in 2.422secs.



- 扩展 [验证]: Validate Plan
 - 右键生成的plan文件, 选择PDDL: Validate plan即可

The screenshot shows the VS Code interface with three panels. The left panel displays the `spare_tire.ptest.json` file with a JSON structure for a test case. The middle panel shows the `problem.plan` file with a sequence of actions: removing a spare trunk, removing a flat axle, and putting on a spare, followed by a makespan and metric. The right panel shows the same `problem.plan` file with a context menu open, highlighting the 'PDDL: Validate plan' option. A blue arrow points from this menu option to the 'Plan valid' message in the bottom panel. The bottom panel shows the output of the validation process, including a detailed log of the plan execution and a final successful result.

```
domains > spare_tire > {} spare_tire.ptest.json > [ ] cases > {} 0
1 {
2   "defaultDomain": "spare_tire.pddl",
3   "defaultOptions": "",
4   "cases": [
5     {
6       "problem": "problem.pddl"
7     }
8   ]
9 }
```

```
domains > spare_tire > problem.plan
1 ;;!domain: spare_tire
2 ;;!problem: spare_tire_problem
3
4 0.00000: (REMOVE SPARE TRUNK)
5 0.00100: (REMOVE FLAT AXLE)
6 0.00200: (PUTON SPARE)
7
8 ; Makespan: 0.002
9 ; Metric: 0.002
```

```
domains > spare_tire > problem.plan
1 ;;!domain: spare_tire
2 ;;!problem: spare_tire_problem
3
4 0.00000: (REMOVE SPARE TRUNK)
5 0.00100: (REMOVE FLAT AXLE)
6 0.00200: (PUTON SPARE)
7
```

Plan Validation details
-----|
Checking next happening (time 0)
Deleting (at spare trunk)
Adding (at spare ground)
Checking next happening (time 0.001)
Deleting (at flat axle)
Adding (at flat ground)
Checking next happening (time 0.002)
Deleting (at spare ground)
Adding (at spare axle)
Plan executed successfully - checking goal
Plan valid
Final value: 3
Successful plans:
Value: 3
C:\Users\ARCSIN~1\AppData\Local\Temp\plan--23952-x2BWBFri9Hms-.pddl 3

Screen Reader Optimized Ln 7, Col 1 Spaces: 4 UTF-8 LF PDDL Plan

实验任务



1、Blocks world

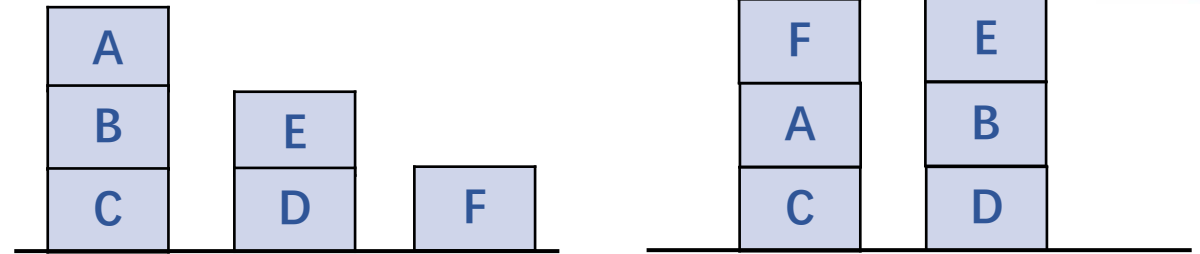
blocks.pddl

```
(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 ()
  )
)
```

其他定义已给出，把动作前提和效果补全

problem.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)))
)
```





2、15puzzle

- 从A*和IDA*中PPT上的四个15数码问题中任选一个

补全问题文件
PPTx.pddl

补全论域文件
puzzle.pddl

```
(define (problem PPT1)
  (:domain puzzle)
  Show hierarchy
  (:objects )
  View
  (:init )
  (:goal ())
)
; 14 10 6 0
; 4 9 1 8
; 2 3 5 11
; 12 13 7 15
```

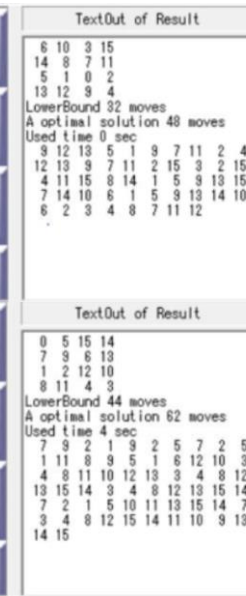
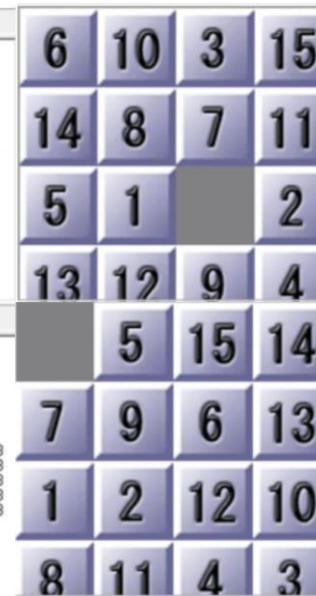
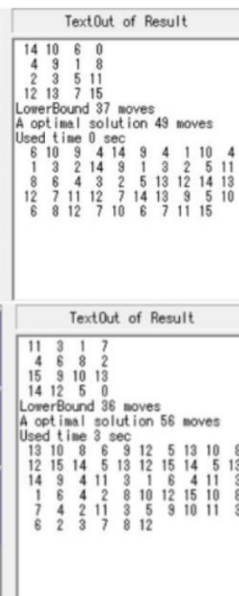
选中多行, Ctrl+“/” 取消注释后进行复制

例1

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

例3

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



例2

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

例4

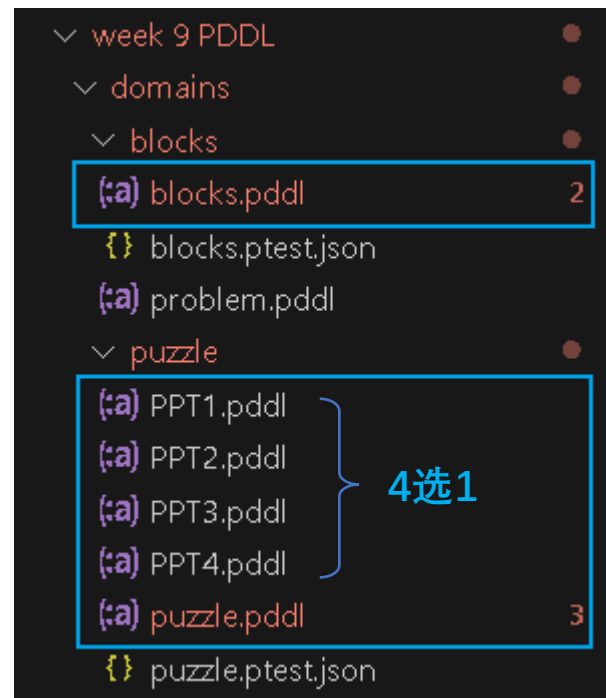
0 5 15 14
7 9 6 13
1 2 12 10
8 11 4 3

```
(define (domain puzzle)
  (:requirements :strips :equality:typing)
  Show hierarchy
  (:types num loc)
  (:predicates ())
  (:constants )
)
(:action slide
  | | | | |
  :parameters ()
  :precondition ()
  :effect ()
)
)
```



实验内容详解

- 建议复习理论课件Planning24 (10页始)
- 积木世界:
 - 补全动作(blocks.pddl)
 - 你可能需要forall和when
- 15数码
 - Planning24, 21页始
 - 不需要考虑解的最优性!!
 - 补充完整domain文件(puzzle.pddl)和一个PPTx.pddl文件, 只允许有slide这一个动作, 其他自行定义
 - 表示空格位置的两种方法:
 - 直接像PPT那样和block一起编码, 但是否需要在action中引入常量?
 - 将“某位置是否为空格”单独作为谓词, 但要怎么修改动作的前提结果和知识库?
 - 最好先判断是否在空格, 再判断邻居 (遍历的顺序)
 - 目标状态
 - 如何表示? 是否一定要写全十六个位置的状态? 15个可以吗? 为什么?



(以上提问的回答不要求写到报告里, 但想写可以写写)



实验要求

- 本次算法原理部分和伪代码/流程图部分可简写。主要是写清论域和问题的定义。可参考理论课件，和课件有出入的要描述清楚。
 - 补全要求的pddl文件，放在code文件夹中一并提交。
 - 在实验结果中展示使用规划器求解的结果（不要求对解进行Validate）
-
- 提交命名：E5_学号
 - 截止时间：4月29号23:59（一周）

可能有用（15puzzle） ↓

```
r=[]  
for i in range(4):  
    tmp=input().split(' ')  
    r+=['(at block%s position%d)'%(x,j+i*4+1) for j,x in enumerate(tmp)]
```



- 附加题（无加分，有兴趣的同学可以试一下）
 - 有兴趣的同学可以考虑玩二阶魔方问题：
 - [Online 2x2 Rubik's Cube Solver \(rubiks-cube-solver.com\)](http://rubiks-cube-solver.com)
 - 仅实现U（顶层往左转）,R（右列往上转）,F（正面顺时针转）即可求解，其他动作（URF连续做两次，或者反向拧）可选。
 - 根据这些动作，8个角里面有1个的位置是固定的，是哪一个呢？
 - 定义：可以按色块位置（6*4）和颜色（6）来判断。也可以定义8个角？
 - 随机打乱魔方试试吧！
 - 其他的问题也可以尝试用pddl解决