Exercise 1

问题解决与规划的区别和相似之处

区别：问题解决聚焦于消除现有障碍或达成特定结果，而规划侧重于制定未来行动路径（如长期目标或资源分配）。问题解决强调即时性和灵活性（应对突发问题），规划则注重长期性和有序性（分阶段推进）。

相似性：两者均需明确目标，并通过分解任务实现（如问题拆解或步骤设计）；均依赖反馈机制，需根据执行情况修正方案或规划；均需根据现有资源，并分析可行性。

Exercise 2

Given the action schemas and initial state from Figure 11.1, what are all the applicable concrete instances of $Fly(p, from, to)$ in the state described by

$$At(P_1, JFK) \land At(P_2, SFO) \land Plane(P_1) \land Plane(P_2) \land Airport(JFK) \land Airport(SFO)\ ?$$

$$\begin{aligned} &Action(Fly(p, from, to),\\ &\quad \text{PRECOND: } At(p, from) \land Plane(p) \land Airport(from) \land Airport(to)\\ &\quad \text{EFFECT: } \neg At(p, from) \land At(p, to)) \end{aligned}$$

Fly(P1, JFK, SFO), Fly(P2, SFO, JFK)都是可行的动作

(若接受飞机不动，则可行的动作还有 Fly(P1, JFK, JFK) , Fly(P2, SFO, SFO))

Exercise 3

The monkey-and-bananas problem is faced by a monkey in a laboratory with some bananas hanging out of reach from the ceiling. A box is available that will enable the monkey to reach the bananas if he climbs on it. Initially, the monkey is at A, the bananas at B, and the box at C. The monkey and box have height Low, but if the monkey climbs onto the box he will have height $High$, the same as the bananas. The actions available to the monkey include Go from one place to another, $Push$ an object from one place to another, $ClimbUp$ onto or $ClimbDown$ from an object, and $Grasp$ or $Ungrasp$ an object. The result of a $Grasp$ is that the monkey holds the object if the monkey and object are in the same place at the same height.

1. Write down the initial state description.
2. Write the six action schemas.
3. Suppose the monkey wants to fool the scientists, who are off to tea, by grabbing the bananas, but leaving the box in its original place. Write this as a general goal (i.e., not assuming that the box is necessarily at C) in the language of situation calculus. Can this goal be solved by a classical planning system?
4. Your schema for pushing is probably incorrect, because if the object is too heavy, its position will remain the same when the $Push$ schema is applied. Fix your action schema to account for heavy objects.

1. 语义解释：At(obj, A): obj 在 A 处

Height(obj, heit): obj 的高度为 heit

Holds(monkey, obj): monkey 拿着 obj

初始状态：

Init(At(monkey, A)∧At(bananas, B)∧At(box, C)∧Height(monkey, Low)∧Height(box, Low∧Height(bananas, High) ∧ ¬ Hold(monkey, bananas) ∧ ¬ Hold(monkey, box))

2.

Action(Go(from, to),

PRECOND: At(monkey, from)∧Height(monkey, Low)

EFFECT: ¬At(monkey, from)∧At(monkey, to)

)

Action(Push(obj, from, to),

PRECOND: At(monkey, from)∧At(obj, from)∧Height(monkey, Low)∧ Height(Object, Low)

EFFECT: At(monkey, to)∧At(obj, to)∧ ¬At(monkey, from) ∧ ¬At(obj, from)

)

Action(ClimbUp(obj),

PRECOND: At(monkey, place)∧At(obj, place)∧Height(monkey, Low)∧Height(obj, Low)

EFFECT: Height(monkey, High)

)

Action(ClimbDown(obj),

PRECOND: At(monkey, place)∧At(obj, place)∧Height(monkey, High)∧Height(obj, Low)

EFFECT: Height(monkey, Low)

)

Action(Grasp(obj),

PRECOND: At(monkey, place)∧At(obj, place)∧Height(monkey, h)∧Height(obj, h)∧¬Hold(monkey, obj)

EFFECT: Hold(monkey, obj)

)

Action(Ungrasp(obj),

PRECOND: Hold(monkey, obj)

EFFECT: ¬Hold(monkey, obj)

)

3. Goal(Hold(monkey, bananas)∧At(box, InitialPosition)）其中 InitialPosition 为盒子

的初始位置。

此问题可以由经典规划解决，假设 InitialPos 为盒子的初始位置，可规划动作序列为：

Go(A, InitialPos)，Push(box, InitialPos, B)，ClimbUp(box)，Grasp(bananas)，

ClimbDown(box)，Push(box, B, InitialPos)

4. 增加一个属性 Heavy(obj): obj 是沉重的

修改 Push 的动作模式

Action(Push(obj, from, to),

PRECOND: At(monkey, from)∧At(obj, from)∧Height(monkey, Low)∧

Height(Object, Low)∧¬Heavy(obj)

EFFECT: At(monkey, to)∧At(obj, to)∧ ¬At(monkey, from) ∧ ¬At(obj, from)

)

Exercise 4

（1）使用 PDDL 定义一组基元动作：

前进 Forward (t)，左转 TurnLeft(t)，右转 TurnRight(t)

装货 Load(p,t)，卸货 Unload(p,t)

其中 p 是货物，t 是卡车；动作模式具体定义略

（2）定义一些谓词：

Truck(t): t 是卡车

Package(p): p 是包裹

At(obj, [x,y])：obj 在坐标[x, y]上

Destination（p，[x′, y′]): 包裹 p 的目的地为[x′, y′]

（3）定义高层动作 HLA：

Nagivate(t，[x，y]）：将卡车 t 开到坐标[x, y]

Deliver(t，p)：将包裹 p 使用卡车 t 的运送到 p 的目的地

Refinement(Deliver (t, p),
PRECOND:Truck(t) ∧ Package(p) ∧ At(p, [x, y]) ∧ Destination(p, [x ′ , y′])
STEPS: [Navigate(t, [x, y]), Load(p, t), Navigate(t, [x ′ , y′]), Unload(p, t)])
Refinement(Navigate(t, [x, y]),
PRECOND:Truck(t) ∧ At(t, [x, y])
STEPS: [])
Refinement(Navigate(t, [x, y]),
PRECOND:Truck(t)
STEPS: [Forward(t), Navigate(t, [x, y])])
Refinement(Navigate(t, [x, y]),
PRECOND:Truck(t)
STEPS: [TurnLeft(t), Navigate(t, [x, y])])
Refinement(Navigate(t, [x, y]),
PRECOND:Truck(t)
STEPS: [TurnRight(t), Navigate(t, [x, y])])

（4）编码知识：

卡车一次只能携带一个包裹的知识；

卡车需要在目的地放下包裹；

两辆或以上卡车不能同时在一个坐标格中；

前方格中有其他卡车时，卡车不能前进。