人工智能导论实验报告

实验一：斑马问题

一、问题重述

> **斑马问题：** 5 个不同国家（英国、西班牙、日本、意大利、挪威）且工作各不相同（油漆工、摄影师、外交官、小提琴家、医生）的人分别住在一条街上的 5 所房子里，
> 每所房子的颜色不同（红色、白色、蓝色、黄色、绿色），每个人都有自己养的不同宠物（狗、蜗牛、斑马、马、狐狸），喜欢喝不同的饮料（矿泉水、牛奶、茶、橘子汁、咖啡）。
>
> 根据以下提示，你能告诉我哪所房子里的人养斑马，哪所房子里的人喜欢喝矿泉水吗?
>
> 1.英国人住在红色的房子里
>
> 2.西班牙人养了一条狗
>
> 3.日本人是一个油漆工
>
> 4.意大利人喜欢喝茶
>
> 5.挪威人住在左边的第一个房子里
>
> 6.绿房子在白房子的右边
>
> 7.摄影师养了一只蜗牛
>
> 8.外交官住在黄房子里
>
> 9.中间那个房子的人喜欢喝牛奶
>
> 10.喜欢喝咖啡的人住在绿房子里
>
> 11.挪威人住在蓝色的房子旁边
>
> 12.小提琴家喜欢喝橘子汁
>
> 13.养狐狸的人所住的房子与医生的房子相邻
>
> 14.养马的人所住的房子与外交官的房子相邻

在问题中一共包括5个对象（房屋），每个房屋都有5个特征，依次为`(国家，工作，饮料，宠物，颜色)`五个元素；如果元素不可知则用`var()`代替填空，总体的集合用`self.units`表示。其中的关系可以用`membero`、`eq`和自定义的`right`、`left`、`nextby`表示房子和房子之间的位置关系。

二、设计思想

（1）python包的导入

由于接下来需要使用集合中合并交一类的操作，所以我们需要导入外界包，使用`import`语句。

```
from kanren import run, eq, membero, var, conde# kanren一个描述性Python逻辑
from kanren.core import lall       # lall包用于定义规则
```

（2）自定义位置功能性函数

```
def left(x,y,units):
    sets=zip(units,units[1:])
    return(membero((x,y),sets))

def right(x,y,units):
    return(left(y,x,units))

def nextby(x,y,units):
    return conde([left(x,y,units)],[right(x,y,units)])
```

将units进行左一位的切片，再与原units进行zip打包操作，求出所有的左邻关系。关于右邻关系，x和y用`left`函数取反即可。对于`nextby`函数需要返回`left`和`right`的并集。

（3）构建智能体类对象

```
self.units = var()                  # 单个unit变量指代一座房子的信息(国家，工作，饮料，宠物，颜色)
```

初始化`units`变量，并且在初始化时赋值为`var()`，即赋值为空。

```
(eq, (var(), var(), var(), var(), var()), self.units),
# self.units共包含五个unit成员，即每一个unit对应的var都指代一座房子(国家，工作，饮料，宠物，颜色)
```

`agent`中还定义了`rules_zebraproblem`和`solutions`，分别用来定义规则和存储结果

在智能体中，有许多定义规则的函数，我们使用`kanren`包中的`lall`函数定义规则，下面我们一一列举。

1.membero

```
(membero,('西班牙人',var(),var(),'狗',var()),self.units)
```

该语句表示养狗的西班牙人的房子包含在`units`集合中

2.eq

```
(eq,(('挪威人',var(),var(),var(),var()),var(),var(),var(),var()),self.units)
```

用相等关系表示，挪威人住在左边第一个的房子里。

3.自定义的right、left、nextby函数

在自定义函数的模块已经介绍，不再赘述

（4）实现智能体的逻辑关系

```
(membero,(var(), var(), var(), '斑马', var()), self.units),
(membero,(var(), var(), '矿泉水', var(), var()), self.units),
(membero,('英国人',var(),var(),var(),'红色'),self.units),
(membero,('西班牙人',var(),var(),'狗',var()),self.units),
(membero,('日本人','油漆工',var(),var(),var()),self.units),
(membero,('意大利人',var(),'茶',var(),var()),self.units),
(eq,(('挪威人',var(),var(),var(),var()),var(),var(),var(),var()),self.units),
(right,(var(),var(),var(),var(),'绿色'),(var(),var(),var(),var(),'白
色'),self.units),
(membero,(var(),'摄影师',var(),'蜗牛',var()),self.units),
(membero,(var(),'外交官',var(),var(),'黄色'),self.units),
(eq,(var(),var(),(var(),var(),'牛奶',var(),var()),var(),var()),self.units),
(membero,(var(),var(),'咖啡',var(),'绿色'),self.units),
(nextby,('挪威人',var(),var(),var(),var()),(var(),var(),var(),var(),'蓝
色'),self.units),
(membero,(var(),'小提琴家','橘子汁',var(),var()),self.units),
(nextby,(var(),var(),var(),'狐狸',var()),(var(),'医
生',var(),var(),var()),self.units),
(nextby,(var(),var(),var(),'马',var()),(var(),'外交
官',var(),var(),var()),self.units)
```

（5）规则求解器

```
def solve(self):
        """
        规则求解器(请勿修改此函数).
        return: 斑马规则求解器给出的答案，共包含五条匹配信息，解唯一.
        """

        self.define_rules()
        self.solutions = run(0, self.units, self.rules_zebraproblem)
        return self.solutions
```

`slove`会根据已有的逻辑关系推导出五个房屋与五个特征之间的对应关系。

四、实验结果

本地测试

将代码放在`vscode`上运行，得到以下输出：

```
绿色房子里的人养斑马
黄色房子里的人喜欢喝矿泉水
('挪威人', '外交官', '矿泉水', '狐狸', '黄色')
('意大利人', '医生', '茶', '马', '蓝色')
('英国人', '摄影师', '牛奶', '蜗牛', '红色')
('西班牙人', '小提琴家', '橘子汁', '狗', '白色')
('日本人', '油漆工', '咖啡', '斑马', '绿色')
(pytorch39) PS D:\Desktop\rubbish3>
```

平台在线测试

我们在平台端进行在线测试得到以下结果：

五、总结

1、本实验进行了 `kanren` 包的使用，利用其强大的逻辑推理能力来完成斑马问题的推理

2、问题的关键之处就是理解 `kanren` 语句中的逻辑关系，学会编写逻辑关系的代码，能够读懂 `kanren` 逻辑语句。

3、该逻辑语句的实现的核心就是在于对于自定义函数的理解与编写，我们需要利用切片与zip来实现三个函数功能——left，right，next。

4、我们可以发现，该问题实际上可以通过穷举法来实现，方法是先生成结果，然后与条件一一匹配， 但是这样做虽然思路简单，但是所消耗的时间很长，这个方法不太好，于是我们使用了 `kanren` 逻 辑关系库来进一步实现

。