人工智能实验报告 第3周

姓名:刘卓逸 学号:21307303

一.实验题目

命题逻辑的归结推理、最一般合一算法、一阶逻辑的归结推理

二.实验内容

1.算法原理

(1)命题逻辑的归结推理

见下文 **一阶逻辑的归结推理**

命题可以视为没有参数的一阶逻辑，可用一阶逻辑的命题推理代替该命题逻辑的归结推理

(2)最一般合一算法

```
function MGU: (f1:str,f2:str) -> ( ans:dict(str,str) )
f1、f2:输入两个原子公式
ans:最一般合一的结果，若无法归一化则为空字典

分离谓词名字与内层参数
判断谓词名是否相同，不相同显然不能归结
将内层参数的项分离
while 归一化未完成
    找出不同的项
    while 两项被函数包括
        分离外层函数与内层项
        若外层函数名相同，则无法归一化，退出
    若两个项的最内层都是常量，则无法归一化，退出
    两个项的最内层分别设为一个变量A与一个不包含前面变量的项B
        若无法这样设则无法归一化，退出
    若A出现在字典ans的键，则无法归一化，退出
    字典ans中加入用项B来代替项A
    将 字典ans、两个原子公式 中的所有A都用B来代替
归一化成功，返回字典ans
```

(3)一阶逻辑的归结推理

判断两个子句是否可归结

```
function check (f1:tuple(str),f2:tuple(str)) ->
(key:str,chg:dict(str,str),from:int)
```

```
f1、f2:输入的两个子句
key:能归结的话用来归结的项，无法归结的话为空串
dict:归结项最一般合一的结果
from: 0->归结项在f1中取反;1->归结项在f2中取反;2->无法归结

遍历f1中的所有项ai
    遍历f2中的所有项bi
        若ai与bi谓词名相同且互相取反
            尝试进行归一化
            归一化成功则表示可以归结，返回相关结果
遍历完了仍无法归结，返回 无法归结
```

将两个可归结的子句归结

```
function fusion: ( f1:tuple(str),f2:tuple(str),key:str,chg:dict(str,str) ) -> (
ans:tuple(str) )
    根据字典对key进行变换
    遍历f1中的所有子句x
        根据字典对x进行变换
        若x不是归结项key且x没有出现在ans中
            则往ans中加入x
    遍历f2中的所有子句y
        根据字典对y进行变换
        若y不是归结项的取反~key且y没有出现在ans中
            则往ans中加入y
    将ans转化为tuple并返回
```

一阶逻辑的归结推理:

```
funxtion ResolutionFOL: ( KB:set(tuple(str)) ) -> ( ans:list(str) )
KB:子句集
ans:归结推理过程
现将初始的子句全部加入处理池
遍历处理池中的所有子句i
    遍历处理池中的所有子句j
        判断i与j是否可归结
            若可归结则求出归结结果k
            不可归结则继续枚举(i,j)
        判断k的命题数是否比i、j中较多的那个少
            如果少，则可以加入到处理池末尾，并记录推理过程"(i,j)->k"
            否则当作无事发生
        若命题k是空子句
            给推理过程重新分配编号，去掉无用推理，返回最终推理过程
            推理过程重新编号具体过程见下文
归结不出空子句，返回推理全过程，debug
```

推理过程重新编号

```
function order: (table:list,fa:list(tuple(int)),origin:int,target:int) ->
(ans:list)
table:原始推理过程
fa:每条子句是从哪两条子句归结而来
origin:原始子句有几条
target:最终结果空子句在原始推理过程的位置，由于空子句一定在最末，所以也等于原始推理过程的
大小
ans:去掉无用推理且重新编号后的推理过程
将推理过程视为一张图，根据若子句x由y、z推理而来，这x指向y、z分别建一条单向边
用宽度搜索从target开始遍历图，能走到的子句都打上标记(代表都是有用推理)
将origin以下的子句(原始条件)全部打上标记
按顺序遍历原始推理过程
    若该子句被打上过标记，则分配新编号并加入到ans末尾
返回ans
```

2.关键代码展示

(1)命题逻辑的归结推理（只是从一阶逻辑的归结推理修改了输出格式）

```
def ResolutionProp(KB):
    ans=[""]
    a=list(KB)
    ai=len(a)
    ais=len(a)
    fa=[(0,0) for i in range(ai+1)]
    for i in range(ais):
        ans.append(str(a[i]))
    i=0
    while i<ai:
        j=0
        while j<ai:
            (key,chg,fr)=check(a[i],a[j])
            if (fr!=2):
                aadd=()
                if fr:
                    aadd=fusion(a[i],a[j],key,chg)
                else:
                    aadd=fusion(a[j],a[i],key,chg)
                #print(a[i],a[j],'=>',fr,key,chg,aadd)
                if (aadd in a or len(aadd)>=max(len(a[i]),len(a[j]))):
                    j+=1
                    continue
                a.append(aadd)
                ai+=1
                s=': '+str(aadd)
                ans.append(s)
                fa.append((i+1,j+1))
                if aadd==():
                    return order(ans,fa,ais,ai)
            j+=1
```

```
        i+=1
    return order(ans,fa,ai,ai)
```

(2)最一般合一算法

1.判断是否是变量

```
def isvariate(f:str):
    if f in ['xx','yy','zz','uu','vv','ww']:
        return True
    return False
```

2.找出不匹配的项

```
def diff(f1:list,f2:list):
    for i in range(len(f1)):
        if f1[i]!=f2[i]:
            return (f1[i],f2[i])
```

3.分离谓词的外层谓词名与内层项目

```
def peel(f:str):
    nl=0
    while nl<len(f) and (f[nl].isalpha() or f[nl]=='~'):
        nl+=1
    name=f[0:nl]
    if nl<len(f):
        f=f[nl+1:len(f)]
        f=f[0:len(f)-1]
    else:
        f=""
    return (name,f)
```

4.最一般合一算法

```
def MGU(f1:str, f2:str): #归一化
    (f1name,f1in)=peel(f1) #拆出最外层的谓词
    (f2name,f2in)=peel(f2)
    if (f1name != f2name): #最外层谓词不同那不考虑
        return {}
    f1item=f1in.split(',') #分离项
    f2item=f2in.split(',')
    ans={}
    while (f1item != f2item): #没有归一完成
```

```
        (f1x,f2x)=diff(f1item,f2item) #找出不同项
        while (f1x[-1]==')' and f2x[-1]==')'): #两个的外层还有函数，剥开函数
            (n1,f1x)=peel(f1x)
            (n2,f2x)=peel(f2x)
            if (n1!=n2): #若外层函数名不同那肯定不能归一直接结束
                return {}
        if (isvariate(f1x) and (not f1x in f2x)): #f1是变量f2是项的情况
            if f1x in ans: #f1已经被替换过（这可能吗，匹配完不都是直接消失的吗）
                return {}
            ans[f1x]=f2x #写入字典
            f1item=[x.replace(f1x,f2x) for x in f1item] #两个原子命题与字典进行全替
换
            f2item=[x.replace(f1x,f2x) for x in f2item]
            ans=dict((k,v.replace(f1x,f2x)) for k,v in ans.items())
        elif (isvariate(f2x) and (not f2x in f1x)): #f2是变量f1是项的情况
            if f2x in ans:
                return {}
            ans[f2x]=f1x
            f1item=[x.replace(f2x,f1x) for x in f1item]
            f2item=[x.replace(f2x,f1x) for x in f2item]
            ans=dict((k,v.replace(f2x,f1x)) for k,v in ans.items())
        else: #两个常量的情况，寄
            return{}
    return ans
```

(3)一阶逻辑的归结推理

检查两个子句是否可归结

```
def check(a:tuple,b:tuple):
    for ai in a:
        (ainame,aitem)=peel(ai)
        for bi in b:
            (biname,bitem)=peel(bi)
            if (ainame=='~'+biname): #谓词相同且互补
                if (ai=='~'+bi): #整条式子完全相同，不需要进行归一化
                    return (bi,{},0)
                chg=MGU(ai[1:len(ai)],bi) #进行归一化，返回归一化所需变量替换
                if (chg!={}): #如果归一化成功
                    return (bi,chg,0) #可以归结
            if ('~'+ainame==biname):
                if ('~'+ai==bi):
                    return (ai,{},1)
                chg=MGU(ai,bi[1:len(bi)])
                if (chg!={}):
                    return (ai,chg,1)
    return ("",{},2) #扫完了，无法归结
```

将字符串按照字典的规则进行子串替换

```
def chgs(a:str,chg:dict): #将字符串a按照chg进行所有的变量替换
    for (key,val) in chg.items():
        a=a.replace(key,val)
    return a
```

将可归结的子句归结

```
def fusion(a:tuple,b:tuple,key:str,chg:dict):
    key=chgs(key,chg)
    ans=[]
    for j in a:
        i=chgs(j,chg)
        if i!=key and not i in ans:
            ans.append(i)
    for j in b:
        i=chgs(j,chg)
        if i!='~'+key and not i in ans:
            ans.append(i)
    return tuple(ans)
```

推理过程的过滤与重编号

```
def order(table:list,fa:list,origin:int,x:int): #table过程 fa推理来源 origin原始条
件 x最终结果
    temp=[i<=origin for i in range(x+1)]
    realnum=[0 for i in range(x+1)]
    temp[x]=True
    bfs=[x,]
    bi=0
    while (bi<len(bfs)):
        k=bfs[bi]
        if k<=origin:
            bi+=1
            continue
        if (not temp[fa[k][0]]):
            bfs.append(fa[k][0])
            temp[fa[k][0]]=True
        if (not temp[fa[k][1]]):
            bfs.append(fa[k][1])
            temp[fa[k][1]]=True
        bi+=1
    ex=[]
    for i in range(1,x+1):
        if temp[i]:
            if fa[i][0]>0 or fa[i][1]>0:
                ex.append(str(len(ex)+1)+' R['+str(realnum[fa[i]
[0]])+','+str(realnum[fa[i][1]])+']: '+table[i])
```

```
            else:
                ex.append(str(len(ex)+1)+' '+table[i])
            realnum[i]=len(ex)
    return ex
```

一阶逻辑的归结推理

```
def ResolutionFOL(KB):
    ans=[""]
    a=list(KB)
    ai=len(a)
    ais=len(a)
    fa=[(0,0) for i in range(ai+1)]
    for i in range(ais):
        ans.append(str(a[i]))
    i=0
    while i<ai:
        j=0
        while j<ai:
            (key,chg,fr)=check(a[i],a[j])
            if (fr!=2):
                aadd=()
                if fr:
                    aadd=fusion(a[i],a[j],key,chg)
                else:
                    aadd=fusion(a[j],a[i],key,chg)
                #print(a[i],a[j],'=>',fr,key,chg,aadd)
                if (aadd in a or len(aadd)>=max(len(a[i]),len(a[j]))):
                    j+=1
                    continue
                a.append(aadd)
                ai+=1
                s='{'
                for (key,value) in chg.items():
                    s+=key+'='+value+','
                s=s[:len(s)-(0 if (s[len(s)-1]=='{') else 1 )]+'}: '+str(aadd)
                ans.append(s)
                fa.append((i+1,j+1))
                if aadd==():
                    return order(ans,fa,ais,ai)
            j+=1
        i+=1
    return order(ans,fa,ai,ai)
```

三.实验结果及分析

1.实验结果展示示例

测试用代码

```
if __name__ == '__main__':
    print("----test1----")
    KB1 = {('FirstGrade',), ('~FirstGrade', 'Child'), ('~Child',)}
    result1 = ResolutionProp(KB1)
    for r in result1:
        print(r)
    print("----test2----")
    print(MGU('P(xx,a)', 'P(b,yy)'))
    print(MGU('P(a,xx,f(g(yy)))', 'P(zz,f(zz),f(uu))'))
    print("----test3----")
    KB2 = {('On(a,b)',), ('On(b,c)',), ('Green(a)',), ('~Green(c)',),
('~On(xx,yy)', '~Green(xx)', 'Green(yy)')}
    result2 = ResolutionFOL(KB2)
    for r in result2:
        print(r)
    print("----test3 Pro----")
    KB3={('A(tony)',),('A(mike)',),('A(john)',),('L(tony,rain)',),
('L(tony,snow)',),('~A(xx)','S(xx)','C(xx)'),('~C(yy)','~L(yy,rain)'),
('L(zz,snow)','~S(zz)'),('~L(tony,uu)','~L(mike,uu)'),('L(tony,vv)','L(mike,vv)'),
('~A(ww)','~C(ww)','S(ww)')}
    result3 = ResolutionFOL(KB3)
    for r in result3:
        print(r)
    print("----test3 Alter----无法证明")
    KB4={('F(xx)',), ('~F(fgo)','C(yy)'), ('~C(pcr)','D(ygo)'),
('~F(csgo)','C(zz)'),('D(duel)',)}
    result4 = ResolutionFOL(KB4)
    for r in result4:
        print(r)
```

输出结果

```
PS C:\Users\DonaLdZY> python -u
"c:\Users\DonaLdZY\Documents\Lesson\2023AI\Reports\hw3\hw3_21307303_liuzhuoyi.py"
----test1----
1 ('FirstGrade',)
2 ('~FirstGrade', 'Child')
3 ('~Child',)
4 R[1,2]: ('Child',)
5 R[3,4]: ()
----test2----
{'xx': 'b', 'yy': 'a'}
{'zz': 'a', 'xx': 'f(a)', 'uu': 'g(yy)'}
----test3----
1 ('~On(xx,yy)', '~Green(xx)', 'Green(yy)')
2 ('Green(a)',)
3 ('On(a,b)',)
4 ('On(b,c)',)
5 ('~Green(c)',)
6 R[1,3]{xx=a,yy=b}: ('~Green(a)', 'Green(b)')
7 R[1,5]{yy=c}: ('~On(xx,c)', '~Green(xx)')
```

```
8 R[2,6]{}: ('Green(b)',)
9 R[4,7]{xx=b}: ('~Green(b)',)
10 R[8,9]{}: ()
----test3 Pro----
1 ('~A(ww)', '~C(ww)', 'S(ww)')
2 ('A(john)',)
3 ('~A(xx)', 'S(xx)', 'C(xx)')
4 ('L(tony,vv)', 'L(mike,vv)')
5 ('L(zz,snow)', '~S(zz)')
6 ('A(mike)',)
7 ('L(tony,rain)',)
8 ('~C(yy)', '~L(yy,rain)')
9 ('L(tony,snow)',)
10 ('A(tony)',)
11 ('~L(tony,uu)', '~L(mike,uu)')
12 R[1,3]{ww=xx}: ('~A(xx)', 'S(xx)')
13 R[6,12]{xx=mike}: ('S(mike)',)
14 R[9,11]{uu=snow}: ('~L(mike,snow)',)
15 R[13,5]{zz=mike}: ('L(mike,snow)',)
16 R[14,15]{}: ()
----test3 Alter----无法证明
1 ('F(xx)',)
2 ('D(duel)',)
3 ('~F(csgo)', 'C(zz)')
4 ('~F(fgo)', 'C(yy)')
5 ('~C(pcr)', 'D(ygo)')
6 R[1,3]{xx=csgo}: ('C(zz)',)
7 R[1,4]{xx=fgo}: ('C(yy)',)
8 R[5,6]{zz=pcr}: ('D(ygo)',)
```

推理过程正确，且无多余推理过程

总之按照实验要求实现了一阶逻辑的归结推理