归结原理

实验课安排调整

周次	课上	课下	重要时间节点
1	Python程序设计基础I	实验1-1	
2	Python程序设计基础II	实验1-2	
3	归结推理	实验2-1	实验1提交
4	知识图谱	实验2-2	
5	Prolog简介	实验2-3	
6	盲目搜索	实验3-1	实验2提交
7	启发式搜索	实验3-2	
8	博弈树搜索	实验4-1	实验3提交
9	高级搜索算法	实验4-2	
10	/		

本次从3, 4, 5周的实验中挑选一个写实验报告

目录

1. 理论课内容回顾

1.1 基本概念

1.2 命题逻辑归结算法

1.3 MGU（最一般合一）算法

1.4 一阶逻辑的归结算法

2. 实验任务

用归结算法求解逻辑推理问题（如Alpine Club问题）

1.1 基本概念

以Alpine Club问题为例

- Tony, Mike, and John belong to the Alpine Club.
- Every member of the Alpine Club who is not a skier is a mountain climber.
- Mountain climbers do not like rain, and anyone who does not like snow is not a skier.
- Mike dislikes whatever Tony likes, and likes whatever Tony dislikes.
- Tony likes rain and snow.
- Is there a member of the Alpine Club who is a mountain climber but not a skier?

1.1 基本概念

- Alpine Club问题形式化为
  - 已知条件（知识库）
    - Facts
      - A(tony)
      - A(mike)
      - A(john)
      - L(tony,rain)
      - L(tony,snow)
    - Rules
      - $\forall x(A(x) \land \neg S(x)) \rightarrow C(x)$
      - $\forall x(C(x) \rightarrow \neg L(x,rain))$
      - $\forall x(\neg L(x,snow) \rightarrow \neg S(x))$
      - $\forall x(L(tony,x) \rightarrow \neg L(mike,x))$
      - $\forall x(\neg L(tony,y) \rightarrow L(mike,y))$
  - 提问
    - $\exists x(A(x) \land C(x) \land \neg S(x))$是否成立

1.1 基本概念

相关概念

- 常量（constant）：任何类型的实体
  - 俱乐部成员：tony，mike，john
  - 天气类型：rain，snow
- 变量（variable）：如x，y这类未知量
- 项（term）：可以理解为谓词/变量的参数项，由递归定义
  - 变量是项（可以看成是0元函数）
  - t1，t2，t3……tn是项，f是n元函数，则f(t1,t2,,,,tn)也是项

Tips: 一阶逻辑中谓词不是项，即不能作为函数/谓词的参数，也就是不存在f(P(x))这种复合方式，但是二阶逻辑中是可以的

1.1 基本概念

相关概念

- 谓词（predicate）：谓词是对其参数（也叫做项，term）的描述
  - 零元谓词：退化为命题，如A(mike)表示mike属于Alpine俱乐部
  - 单元谓词（unary predicate）：只有一个参数，表示参数具备某种属性，如A(x)表示x属于Alpine俱乐部
  - 多元谓词：有多个参数，表示参数之间的关系，如L(x,y)表示x和y具有喜欢关系，即x喜欢y

1.1 基本概念

- 相关概念
  - 事实（fact）：谓词中变量实例化后得到事实
    - S(tony)：tony是skier
    - L(tony, rain)：tony喜欢下雨天
  - 规则（rule）：也叫做公式，通过递归定义
    - t1，t2，t3……tn是项，P是n元谓词，则P(t1,t2,,,,tn)是原子公式
    - t1，t2是项，那么t1=t2是原子公式
    - 如果α和β是公式，那么¬α，α∧β，α∨β，∃α，∀α都是公式

Tips:由于 (α →β) 等价于 (¬α ∨β)，(α↔β) 等价于 (α →β) ∧(β →α)，所以在递归定义中我们没有加入→和↔，它们可以被已有符号替代

1.1 基本概念

- 相关概念
  - 可满足性：
    - 以Alpine俱乐部为例，$\exists x(A(x) \wedge C(x) \wedge \neg S(x))$是否成立就是在问，是否存在一组实例化（一组赋值），使得$A(x) \wedge C(x) \wedge \neg S(x)$成立，这就是一个可满足性问题。对于该可满足性问题，只要能够找到一组赋值（在这里对应{x}的赋值），使得$A(x) \wedge C(x) \wedge \neg S(x)$成立，那么“$A(x) \wedge C(x) \wedge \neg S(x)$”是可满足的
  - 逻辑蕴含和逻辑推论：
    - 逻辑蕴含$S \models \alpha$指对于任意变量赋值，如果S正确，则α也正确
    - 逻辑推论$S \vdash \alpha$指存在一条推理路径，从S出发，推导证明α

1.2 命题逻辑归结算法

- 定理：
  - S |-() 当且仅当 S |= ()， S |= () 当且仅当 S 是不可满足的
  - 通过该定理，我们可得KB |= α 当且仅当 KB ∧¬α 不可满足，于是可以利用该性质来证明KB |= α
- 归结算法：
  - 将α取否定，加入到KB当中
  - 将更新的KB转换为clausal form得到S
  - 反复调用单步归结
    - 如果得到空子句，即S|-()，说明KB ∧¬α 不可满足，算法终止，可得KB |= α
    - 如果一直归结直到不产生新的子句，在这个过程中没有得到空子句，则KB |= α不成立

1.2 命题逻辑归结算法

- 归结算法：
  - Clausal form (便于计算机处理的形式)
    - 每一个子句对应一个元组，元组每一个元素是一个原子公式/原子公式的否定，元素之间的关系是析取关系，表示只要一个原子成立，该子句成立
      - 如子句¬child∨¬male∨boy对应数据结构(¬child,¬male,boy)，空子句()对应False
    - 元组的集合组成子句集S，子句集中每个句子之间是合取关系，表示每一个子句都应该被满足
    - 由于本次实验重点是归结算法，所以问题输入是已经转换过的clausal form
  - 单步归结
    - 从两个子句中分别寻找相同的原子，及其对应的原子否定
    - 去掉该原子并将两个子句合为一个，加入到S子句集合中
    - 例如(¬child,¬female,girl)和(child)合并为(¬female,girl)

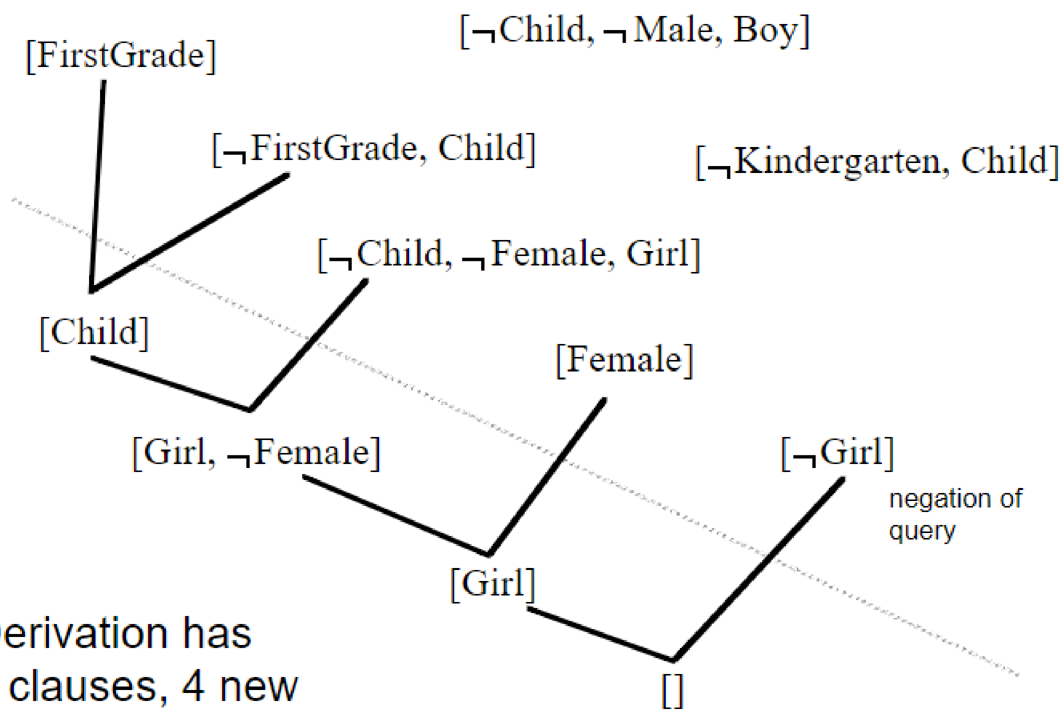
1.2 命题逻辑归结算法

- 例子

Clausal form

FirstGrade		
¬FirstGrade	Child	
¬Child	¬Male	Boy
¬Kindergarten	Child	
¬Child	¬Female	Girl
Female		

Show that KB $\models$ Girl



Derivation has
9 clauses, 4 new

1.3 Most general unifier算法

- 最一般合一算法：
  - 合一（unifier）：
    - 通过变量替换使得两个子句能够被归结（有相同的原子），所以合一也被定义为使得两个原子公式等价的一组变量替换/赋值
    - 由于一阶逻辑中存在变量，所以归结之前需要进行合一，如(P(john),Q(fred),R(x))和(¬P(y),R(susan),R(y))两个子句中，我们无法找到一样的原子及其对应的否定，但是不代表它们不能够归结
    - 通过将y替换为john，我们得到了(P(john),Q(fred),R(x))和(¬P(john),R(susan),R(john))，此时我们两个子句分别存在原子P(john)和它的否定¬P(john)，可以进行归结
  - 最一般合一：指使得两个原子公式等价，最简单的一组变量替换

1.3 Most general unifier算法

- 最一般合一算法：
  - 输入：两个原子公式，它们具有相同的谓词，不同的参数项和“¬”
  - 输出：一组变量替换/赋值
  - 算法流程：
    - $k = 0$; $\sigma_0 = \{\}$; $S_0 = \{f, g\}$
    - 如果S_k中的公式等价，返回σ_k作为最一般合一的结果
      - 否则找出S_k中的不匹配项$D_k = \{e1, e2\}$
    - 如果 e1=V 是变量，e2 = t是一个不包含变量V的项，将“V = t”添加到赋值集合$\sigma_{k+1} = \sigma_k \cup \{V = t\}$；并将$S_k$中的其它V变量也赋值为t，得到 S_{k+1}；$k = k + 1$，转到第二步
      - 否则合一失败

Tips:变量替换是从两个原子公式中找到的，但是最后要施加给整个子句的

1.3 Most general unifier算法

- 例子：
  - P(f(a),g(x)) 和 P(y,y)无法合一
  - P(a,x,h(g(z))) 和 P(z,h(y),h(y))最一般合一为{z=a,x=h(g(a)),y=g(a)}
  - P(x,x) 和 P(y,f(y))无法合一

1.4 一阶逻辑归结算法

- 归结算法：
  - 将α取否定，加入到KB当中
  - 将更新的KB转换为clausal form得到S
  - 反复调用单步归结
    - 如果得到空子句，即S|-()，说明KB ∧¬α 不可满足，算法终止，可得KB |= α
    - 如果一直归结直到不产生新的子句，在这个过程中没有得到空子句，则KB |= α不成立
  - 单步归结
    - 使用MGU算法从两个子句中得到相同的原子，及其对应的原子否定
    - 去掉该原子并将两个子句合为一个，加入到S子句集合中
    - 例如(¬Student(x),HardWorker(x))和(HardWorker(sue))合并为(¬Student(sue))

2. 实验任务

- 编写程序，实现一阶逻辑归结算法，并用于求解给出的三个逻辑推理问题，要求输出按照如下格式：

  1. (P(x),Q(g(x)))
  2. (R(a),Q(z),¬P(a))
  3. R[1a,2c](X=a) = (Q(g(a)),R(a),Q(z))

  … …

  “R” 表示归结步骤.

  “1a” 表示第一个子句(1-th)中的第一个 (a-th)个原子公式，即P(x).

  “2c”表示第二个子句(1-th)中的第三个 (c-th)个原子公式，即¬P(a).

  “1a”和“2c”是冲突的，所以应用最小合一{X = a}.

2. 实验任务

- Graduate Student
  - GradStudent(sue)
  - (¬GradStudent(x), Student(x))
  - (¬Student(x), HardWorker(x))
  - ¬HardWorker(sue)

```
[sysu_hpcedu_302@cpn238 ~/scc22/lsr/mp_linpack/resoluation]$ python main.py
4
GradStudent(sue)
(¬GradStudent(x), Student(x))
(¬Student(x), HardWorker(x))
¬HardWorker(sue)
R[3b,4](x=sue) = ¬Student(sue)
R[1,2a](x=sue) = Student(sue)
R[5,6] = []
```

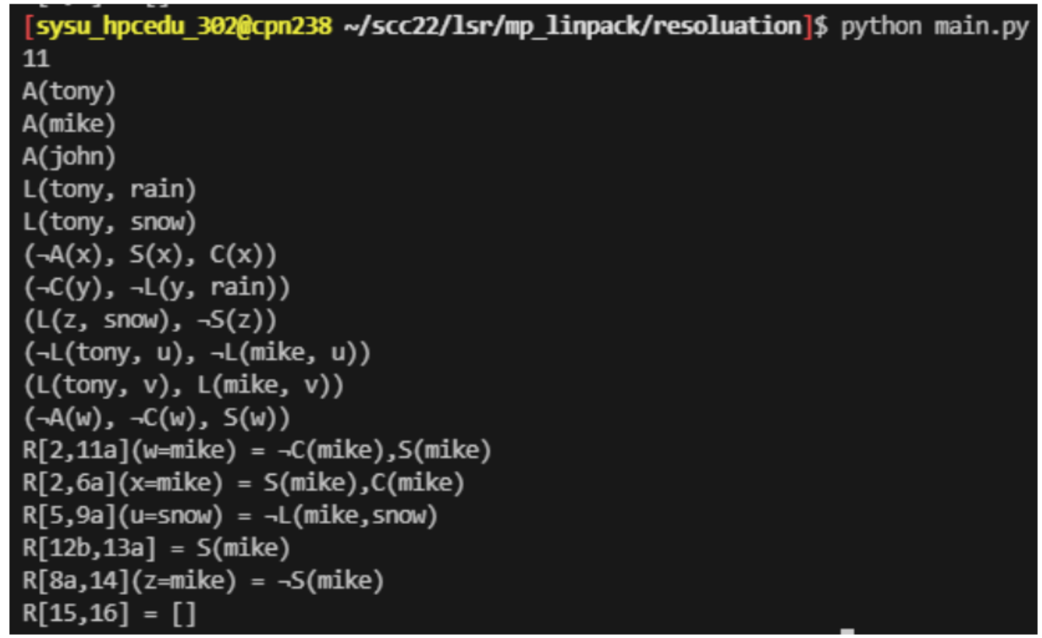
2. 实验任务

- Block World
  - On(aa,bb)
  - On(bb,cc)
  - Green(aa)
  - ¬Green(cc)
  - (¬On(x,y), ¬Green(x), Green(y))

```
[sysu_hpcedu_302@cpn238 ~/scc22/lsr/mp_linpack/resoluation]$ python main.py
5
On(aa,bb)
On(bb,cc)
Green(aa)
¬Green(cc)
(¬On(x,y), ¬Green(x), Green(y))
R[4,5c](y=cc) = ¬On(x,cc),¬Green(x)
R[3,5b](x=aa) = ¬On(aa,y),Green(y)
R[2,6a](x=bb) = ¬Green(bb)
R[1,7a](y=bb) = Green(bb)
R[8,9] = []
```

2. 实验任务

- Aipine Club
  - A(tony)
  - A(mike)
  - A(john)
  - L(tony, rain)
  - L(tony, snow)
  - (¬A(x), S(x), C(x))
  - (¬C(y), ¬L(y, rain))
  - (L(z, snow), ¬S(z))
  - (¬L(tony, u), ¬L(mike, u))
  - (L(tony, v), L(mike, v))
  - (¬A(w), ¬C(w), S(w))



```
[sysu_hpcedu_302@cpn238 ~/scc22/lsr/mp_linpack/resoluation]$ python main.py
11
A(tony)
A(mike)
A(john)
L(tony, rain)
L(tony, snow)
(¬A(x), S(x), C(x))
(¬C(y), ¬L(y, rain))
(L(z, snow), ¬S(z))
(¬L(tony, u), ¬L(mike, u))
(L(tony, v), L(mike, v))
(¬A(w), ¬C(w), S(w))
R[2,11a](w=mike) = ¬C(mike),S(mike)
R[2,6a](x=mike) = S(mike),C(mike)
R[5,9a](u=snow) = ¬L(mike,snow)
R[12b,13a] = S(mike)
R[8a,14](z=mike) = ¬S(mike)
R[15,16] = []
```

实验中需要注意的地方

- 当原始输入为字符串时，如何提取子句中的谓词和项——考虑利用正则表达式来进行提取，或者直接利用字符串截取；
- 在归结时，可以任取两个子句进行归结，这样的话会出现许多的无用子句，如何在输出时忽略无用子句——考虑将归结过程构造成二叉树的形式，根节点为最终的归结结果，然后按层往下遍历，得到每一步的有效归结结果。

关于迟交补交作业

- 在截止日期后一周内提交作业算迟交，每迟交一天扣2分
- 于期中（第九周）和期末（第十九周）设置补交时间，之前未提交的作业均可提交，补交的作业会扣15分
- 第一次迟交或补交的作业不扣分