**中山大学计算机学院**

**人工智能**

**本科生实验报告**

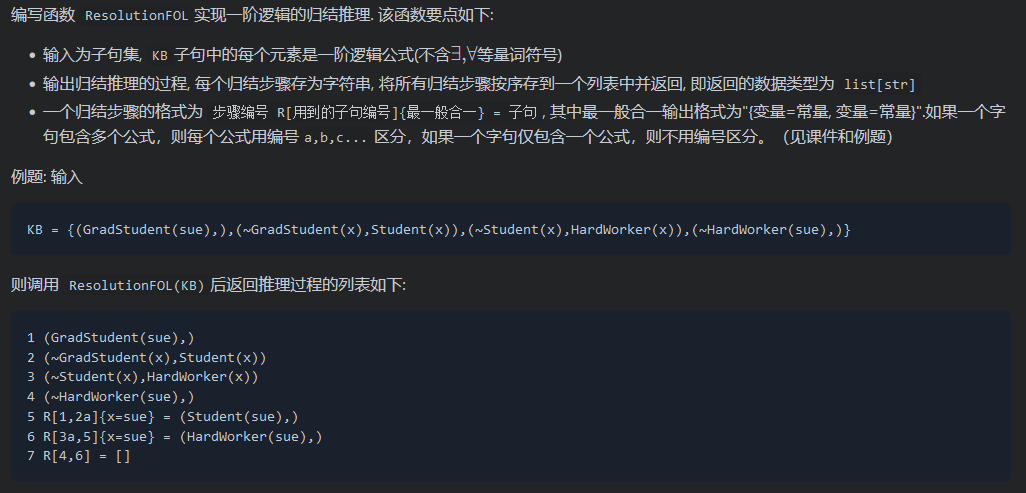
**（2022学年春季学期）**

课程名称：Artificial Intelligence

|  |  |  |  |
| --- | --- | --- | --- |
| 教学班级 | **202320346** | 专业（方向） | **计算机科学与技术** |
| 学号 | **21312450** | 姓名 | **林隽哲** |

# 实验题目

**一阶逻辑的归结推理**

****

**文本

描述已自动生成图形用户界面, 文本, 应用程序

描述已自动生成**

# 实验内容

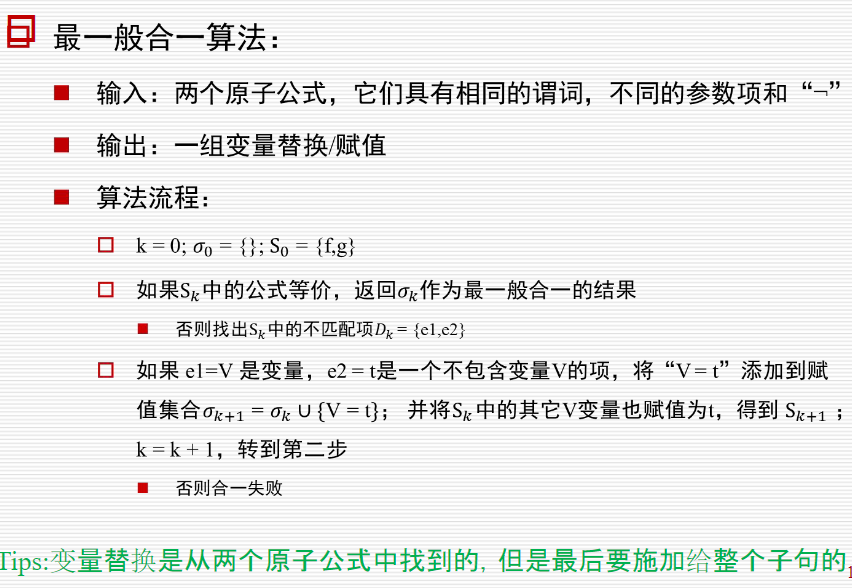
算法原理

课堂中已经证明了定理：**两个子句C1和C2的归结式C是C1和C2的逻辑推论**。由此我们得到了推论：**子句集S={C1, C2, …, Cn}与子句集S1={C, C1, C2, …, Cn}的不可满足性是等价的（其中C是C1和C2的归结式）**。同时在课堂上也证明了归结推理的合理性（**定理：如果S ├ C，那么S ╞ C**）与完备性（**定理：S ├ ()，当且仅当S ╞ ()，当且仅当S不可满足**）。由上述理论支持，可以将一阶逻辑的归结推理过程总结如下：

命题逻辑中，若给定前提集F和命题集R，则：

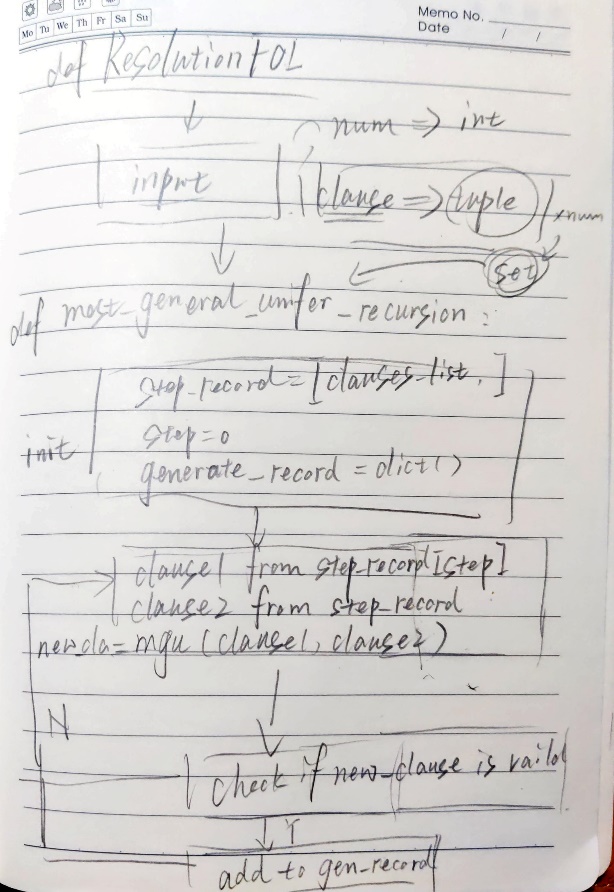
1. 把F转化成子句集表示，得到子句集S0；
2. 把命题R的否定式┐R也转化成子句集表示，并将其加到S0中，得S = S0∪S┐R；
3. 对子句集S反复应用归结推理规则（推导），直至导出含有空子句的扩大子句集为止。即出现归结式为空子句时，表明已找到矛盾，证明过程结束。

在归结过程中，若S中两个子句间有相同互补文字的谓词，但它们的项不同，则必须找出对应的不一致项，并对其进行变量置换，使它们的对应项一致，然后再尝试用它们推导出空子句。通常可以使用最一般合一算法对两个原子公式进行合一，如下：



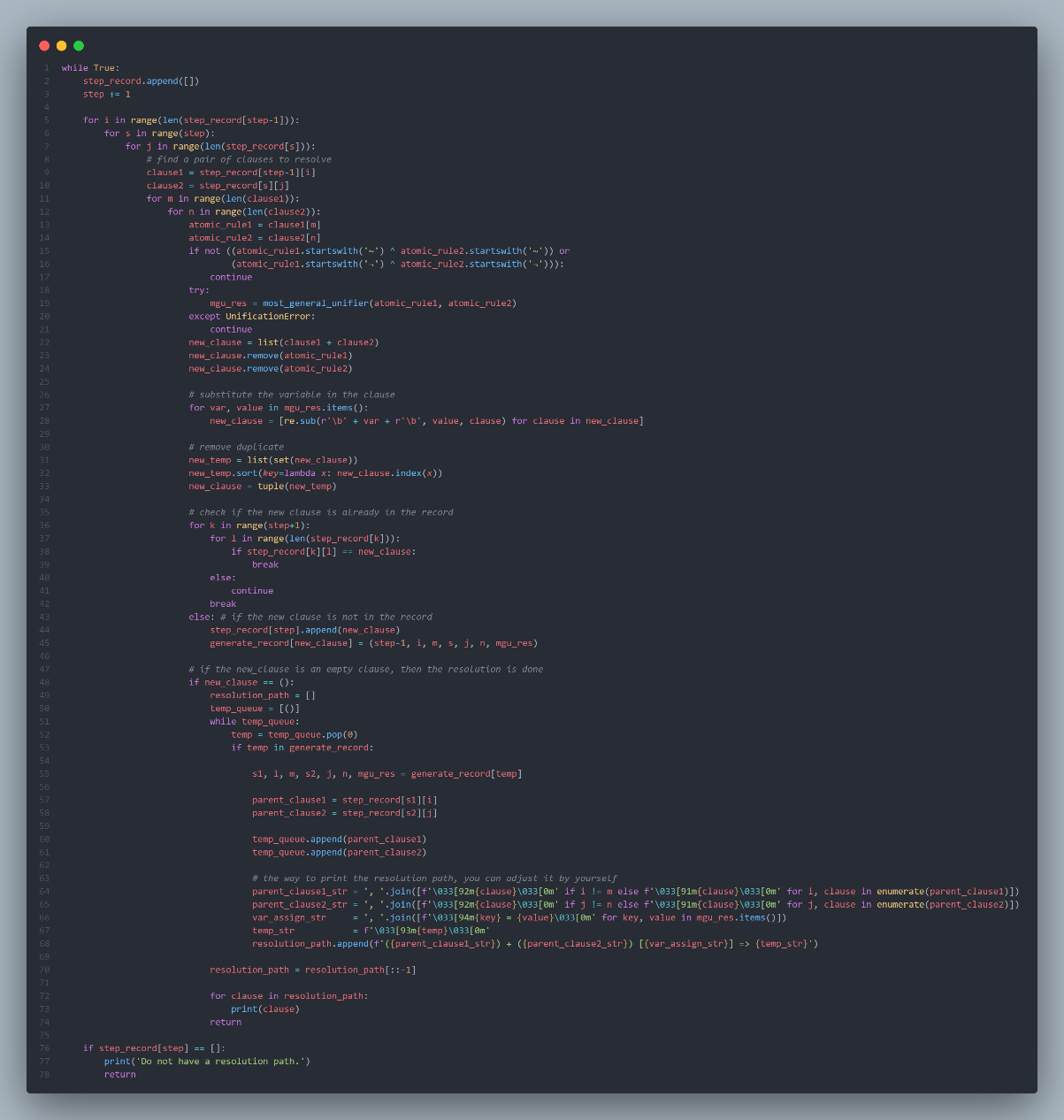
伪代码

这里给出粗略的伪代码实现：



不难看出我使用的是一种分步骤递推的方法进行推理。其中，当step\_record的下一递推为空集时说明推理失败，而当new\_clause为空集时说明找到了一种成功的递推方法，剩下的就是将保存的推理过程打印出来即可。

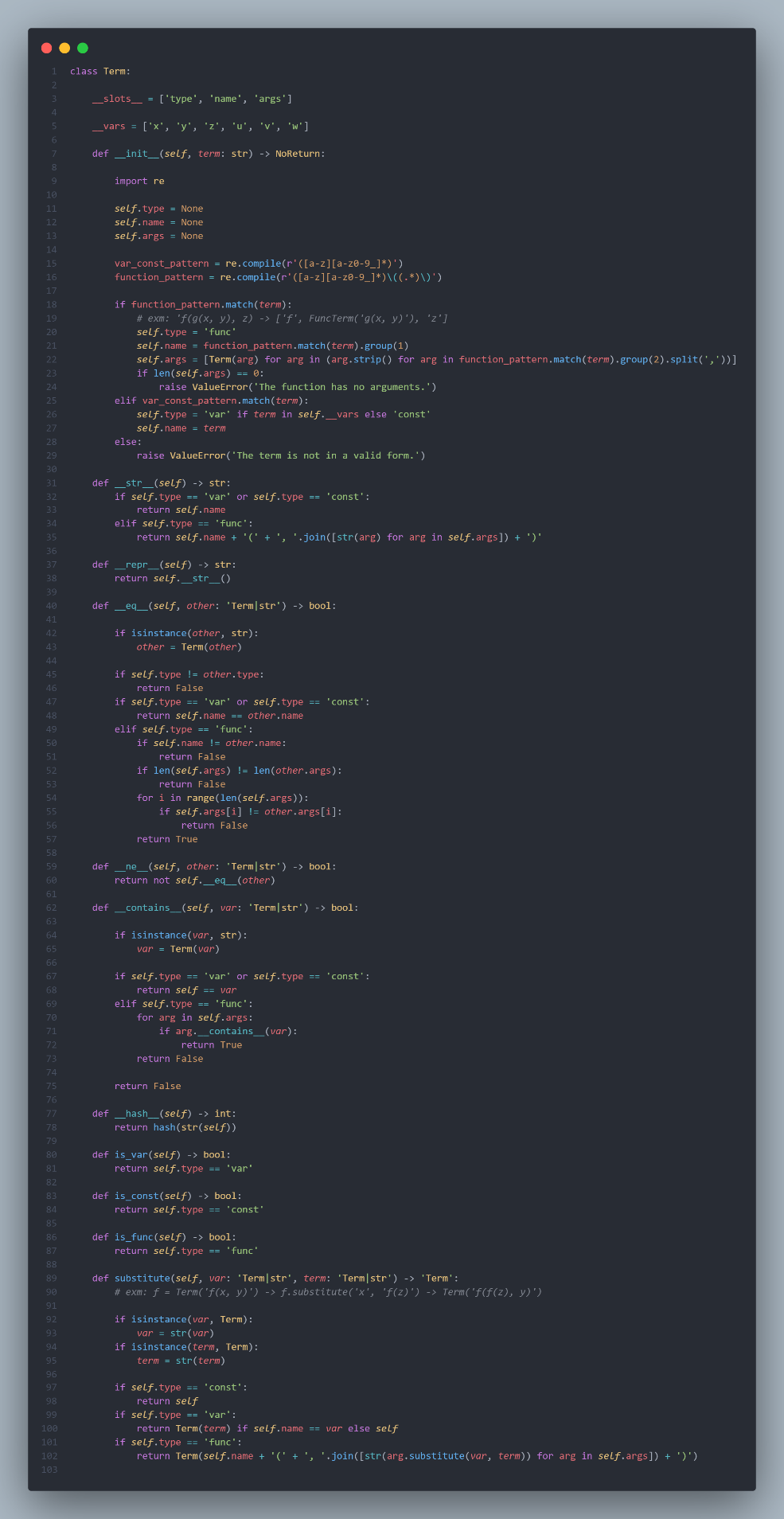
关键代码展示（带注释）



上图是归结过程的核心代码，这里简单的采用了暴力搜索的方式寻找正确的递推路径。相关的实现细节以及说明可见代码内的注释。



上述是最一般合一算法实现的核心代码，为了方便处理变量的赋值以及蕴含关系的判断，我简单的定义了一个Term类，以对谓词中的项进行统一的处理，Term类的具体实现代码如下：



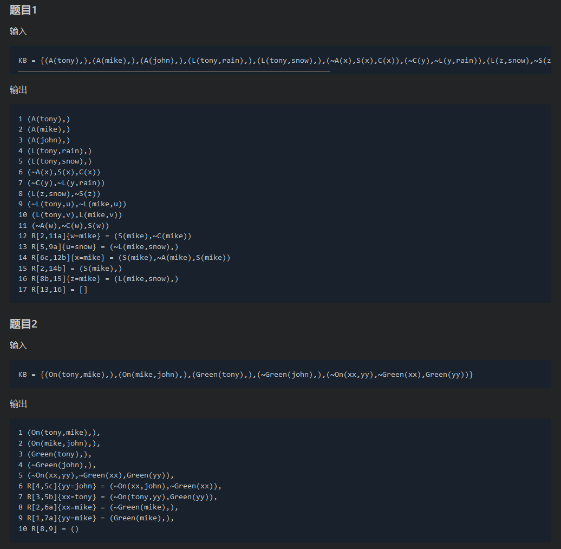
# 实验结果及分析

1. 实验结果展示示例

书写测试函数如下：



运行得到测试结果，并将其与预期输出进行对比：



可见该代码使用了相似的步骤最终正确的完成了一阶逻辑公式的推导。

2. 评测指标展示及分析

由于本程序使用的是暴力搜索的推导方法，因此代码的运行效率会相对较低，且很大程度上取决于子句的长度以及谓词内的项数等因素。

# 参考资料

《Fluent Python》 ­—— Luciano Ramalho 著