一阶逻辑归结求解破案问题

人工智能原理与技术课程

汇报人：*** 指导老师：*** 2022.6.2

目录

content

第一部分

实验概述

实验概述

实验目的

熟悉和掌握归结原理的基本思想和基本方法，通过实验培养学生利用逻辑方法表示知识，并掌握采用机器推理来进行问题求解的基本方法。

实验要求

1. 对所给问题进行知识的逻辑表示，转换为子句，对子句进行归结求解。
2. 选用一种编程语言，在逻辑框架中利归结原理进行求解。
3. 要求界面显示每一步的求解过程。
4. 撰写实验报告，提交源代码（进行注释）、实验报告、汇报PPT。

实验内容

对所给问题进行知识的逻辑表示，转换为子句，对子句进行归结求解。

破案问题：在一栋房子里发生了一件神秘的谋杀案，现在可以肯定以下几点事实：

(a)在这栋房子里仅住有A,B,C三人；
(b)是住在这栋房子里的人杀了A；
(c)谋杀者非常恨受害者；
(d)A所恨的人，C一定不恨；
(e)除了B以外，A恨所有的人；
(f)B恨所有不比A富有的人；
(g)A所恨的人，B也恨；
(h)没有一个人恨所有的人；
(i)杀人嫌疑犯一定不会比受害者富有。

为了推理需要，增加如下常识：

(j)A不等于B。

问：谋杀者是谁？

第二部分

实验设计方案

总体设计思路与总体框架

logicalReasoning_head.h

logicalReasoning_kernel.cpp

总体框架

logicalReasoning_ui.cpp

logicalReasoning_main.cpp

总体设计思路与总体框架

logicalReasoning_head.h

存储整个项目类、结构体、函数、宏定义的声明，作为项目的头文件使用。

内含clauseResolution类的具体实现函数，实现一阶逻辑归结的内部核心算法的求解。还包含部分整个游戏所需工具函数的具体实现。

logicalReasoning_kernel.cpp

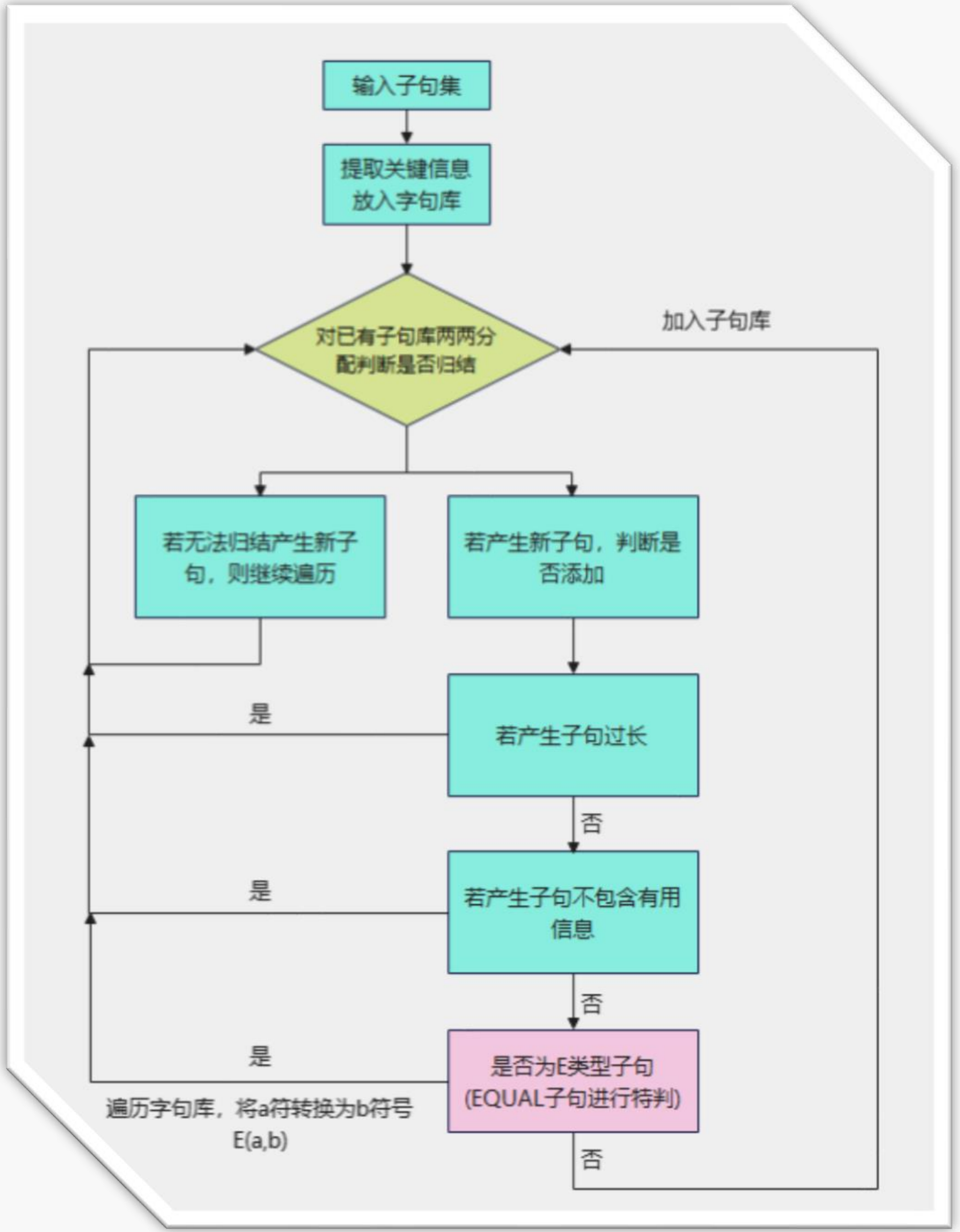
logicalReasoning_ui.cpp

内含clauseUI类的具体实现函数，实现破案问题的可视化界面输入输出。

程序主函数所在地，调用各个类进行对破案问题进行子句归结求解。

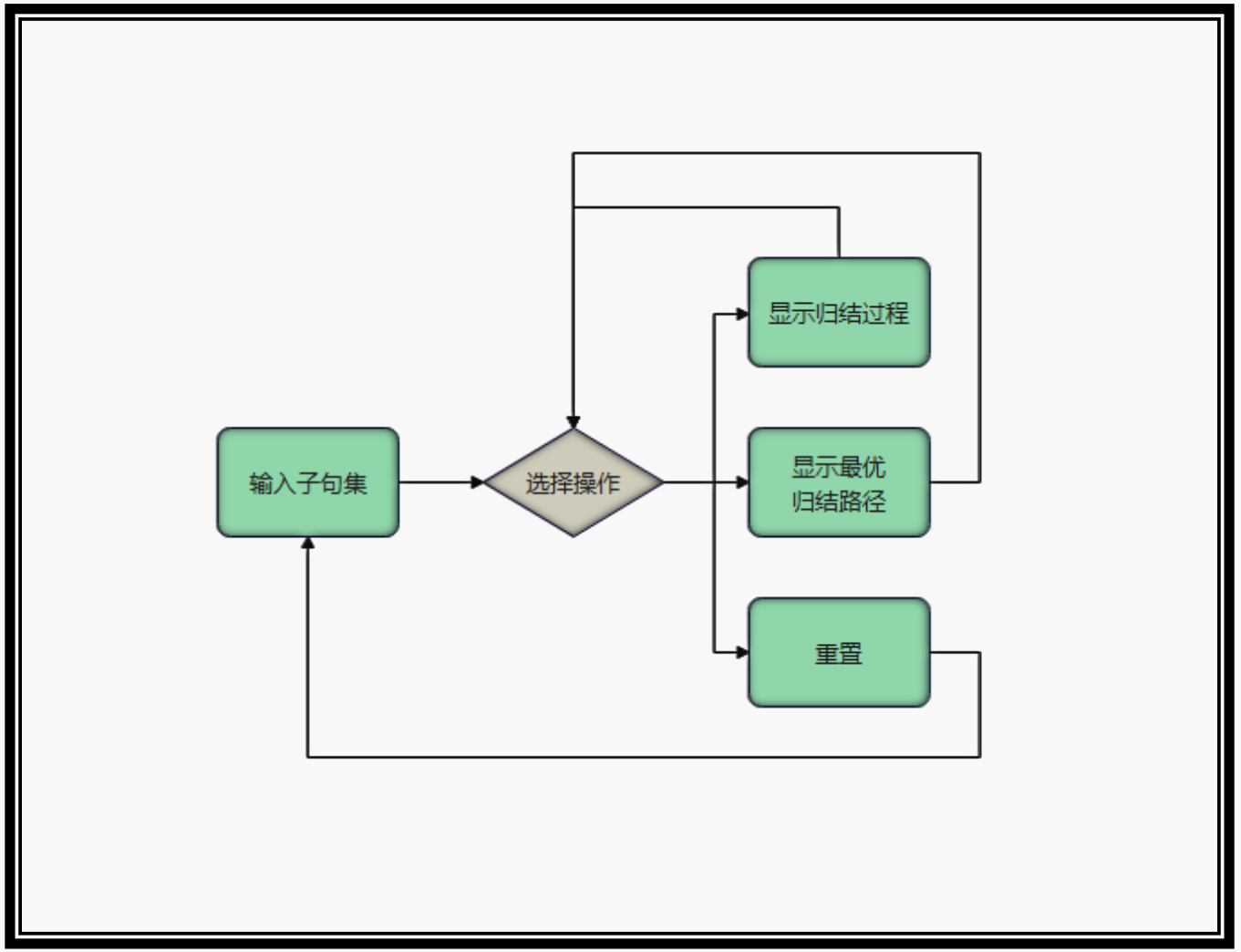
logicalReasoning_main.cpp

总体设计思路与总体框架

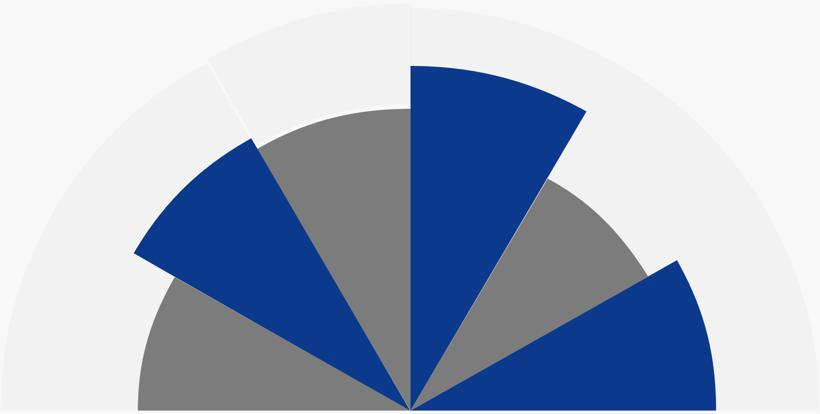


总体框架

总体设计思路与总体框架



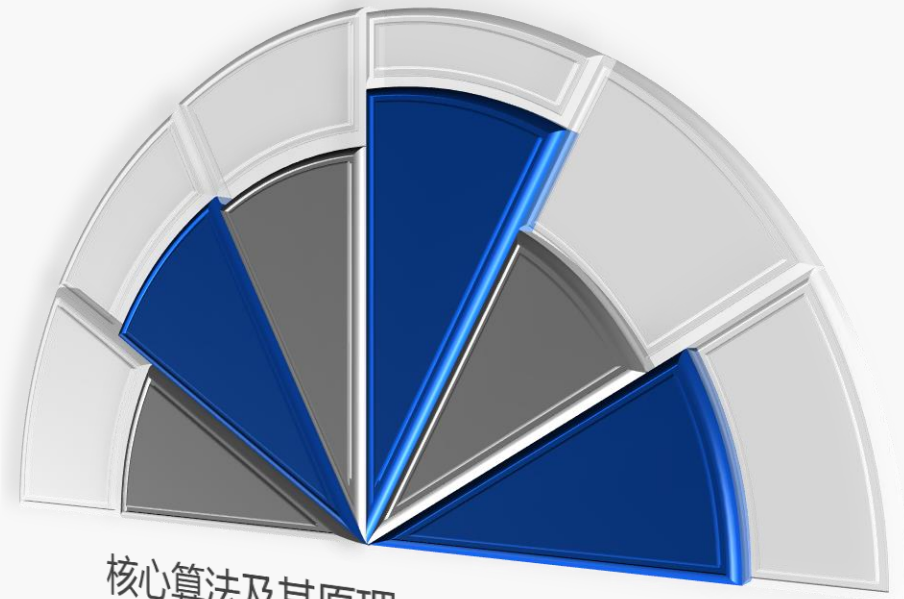
运行流程



核心算法及其原理 — 归结原理

归结过程为：对于两个已经完成变量标准化没有共享变量的两个子句，如果包含互补文字则可对它们进行归结。即如果一个命题文字是另一个命题文字的否定式，则这两个命题文字是互补的；如果一个一阶逻辑文字能和另一个一阶逻辑文字的否定式合一，则这两个一阶逻辑文字是互补的。

$$\frac{l_1 \vee l_2 \vee \ldots \vee l_k \qquad m_1 \vee m_2 \vee \ldots \vee m_n}{SUBST(\theta, l_1 \vee \ldots \vee l_{i-1} \vee l_{i+1} \vee \ldots \vee l_k \vee m_1 \vee \ldots \vee m_{i-1} \vee m_{i+1} \vee \ldots \vee m_n)}$$



算法核心使用归结规则+置换\合一规则，关键在于多次使用以上规则，直至A(Answer)谓词中置换出已知固定值，此值即为问题所需求得的答案。

首先将已知条件用谓词公式表示出来，并转化成子句集。此时需要根据已知条件定义谓词，并将事实用谓词公式表示出来，并将它们化成子句集。此时，关键需要将问题用谓词公式表示出来，使用已定义谓词+A(Answer)谓词对问题进行转化，添加进入子集中。对现有的所有子集进行归结，当得到A谓词的最终结果且内含已知值，归结结束，A谓词所含结果即为所求。

算法实现步骤

找到逻辑0的最终结果

I.读取输入子集，存储至子集库中

II.遍历选择子集库中两个子句进行归结尝试，能进行归结需要两子句中存在以下的表达式

a.谓词种类一致，且一个为非；一个为正

b.谓词中元素若均为常量，则两变量需要一致

c.谓词中元素若均为变量，则满足可归结

d.谓词中一个元素为变量，一个元素为常量或者函数，则可使用合一置换规则进行归结

III.若满足II中条件则进行归结，否则回到II继续执行下一对尝试

a.执行归结，获得置换合一后的新语句

b.判断语句是否过长

c.判断语句是否存在包容归结

若不满足以上条件回到II继续执行；

反之，执行下一语句

IV.判断语句是否为结束语句(逻辑0)。若是，则结束程序，得到结果；反之添加进子集库中，重复执行语句II

模块设计

```cpp
// @function : 子句归结执行内核
class clauseResolution {
protected:

    // 内部参数
    CLAUSES answer;                                  // 最终结果
    int countNum = 0;                                // 用于记录转换个数
    vector<char>expressionType;                      // 该推断问题中表达式的所有种类 --> 'A'-'Z'代替
    vector<CLAUSES>clauseBag;                        // 存储所有转化后的子句集

    // 可视化结果存储
    vector<RECORD>initialList;                       // 初始化数据存储
    vector<RECORD>recordList;                        // 过程数据存储
    vector<RECORD>bestPath;                          // 最优归结路线

    // 内部函数
    void initialRecord(void);                                                                       // 初始逻辑记录
    void processRecord(CLAUSES& c1, CLAUSES& c2, CLAUSES& ans, vector<char>& upData, RECORD& get);  // 归结过程记录
    void traceBackIn(RECORD& c);                                                                    // 寻找最佳归结路线内部递归函数
    void expressionInput(string& original, CLAUSES& c);                                            // 将一个字符串转换成
    bool clausesUpdate(CLAUSES& ans, vector<char>& record, vector<char>& upData);                  // 子句更新
    bool judgeAnswer(CLAUSES& c1);                                                                  // 判断是否归结到终点
    bool equalDeal(CLAUSES& c, vector<CLAUSES>& tempBag);                                          // 等式归结[若为E(f(A),B)则将f(A)替换为B]
    bool containDeal(CLAUSES& c, vector<CLAUSES>& tempBag);                                        // 包容归结
    bool lengthDeal(CLAUSES& c, vector<CLAUSES>& tempBag);                                         // 长度限制
    bool twoClauseResolu(CLAUSES& c1, CLAUSES& c2, CLAUSES& ans, vector<char>& record);             // 两个子句归结
    void processShow(CLAUSES& c1, CLAUSES& c2, CLAUSES& ans, vector<char>& upData);                 // 控制台打印过程
    void initialShow(void);                                                                         // 控制台打印初始过程

public:
    // 外部函数
    void traceBack(void);                            // 寻找最佳归结路线
    void clausInput(vector<string> &originalSentence);  // 将输入子句转化为易计算结构子句
    void resolution(void);                           // 执行子句归结
    void allShow(void);                              // 展示所有归结过程
    void bestShow(void);                             // 展示最优归结过程
    void ansShow(void);                              // 显示最终结果

    void tempShow(void);                             // for test
};
```

01 **clauseResolution模块**

Gonzalez

模块设计

```
// @function : 字句归结UI界面
class clauseUI {
protected:
    // 界面布局结构体定义
    SURFACE backArea;
    SURFACE inputArea;
    SURFACE showallArea;
    SURFACE showbestArea;
    SURFACE restartArea;

    // 界面图片调用定义
    IMAGE background;
    IMAGE input;
    IMAGE inputActive;
    IMAGE showAll;
    IMAGE showAllActive;
    IMAGE showBest;
    IMAGE showBestActive;
    IMAGE restart;
    IMAGE restartActive;

public:
    clauseUI(void);                          // 初始化
    void bgShow(void);                       // 展示背景
    int orderChoose(bool haveIn = false);    // 按键指示

};
```

02 clauseUI模块

模块设计

clauseResolution类

```
void initialRecord(void);
void processRecord(CLAUSES& c1, CLAUSES& c2, CLAUSES& ans, vector<char>& upData, RECORD& get);
void traceBackIn(RECORD& c);
void expressionInput(string& original, CLAUSES& c);
bool clausesUpdate(CLAUSES& ans, vector<char>& record, vector<char>& upData);
bool judgeAnswer(CLAUSES& c1);
bool equalDeal(CLAUSES& c, vector<CLAUSES>& tempBag);
bool containDeal(CLAUSES& c, vector<CLAUSES>& tempBag);
bool lengthDeal(CLAUSES& c, vector<CLAUSES>& tempBag);
bool twoClauseResolu(CLAUSES& c1, CLAUSES& c2, CLAUSES& ans, vector<char>& record);
void processShow(CLAUSES& c1, CLAUSES& c2, CLAUSES& ans, vector<char>& upData);
void initialShow(void);
```

```
void traceBack(void);
void clausInput(vector<string> &originalSentence);
void resolution(void);
void allShow(void);
void bestShow(void);
void ansShow(void);
void tempShow(void);
```

clauseUI类

```
clauseUI(void);                          // 初始化
void bgShow(void);                       // 展示背景
int orderChoose(bool haveIn = false);    // 按键指示
```

01 clauseResolution模块

02 clauseUI模块

创新算法—E关键字逻辑实现

E关键字指的是破案问题中规定的逻辑表示：即E(a,b)表示a和b是相等的。而之所以这个函数要单独拿出来处理，就是因为这种相等关系，使得子句集中的其他字句可以以该函数为推理基础，对文字的参数列表进行替换。

以该问题为例，存在如下两个推理出的子句：

T20:R(f(B),A)

T25:E(f(B),B)

根据这两个子句可知，f(B)与B是等价的，所以应当对T20子句进行一次变量替换得到替换后的子句R(B,A)。经过程序调试发现，如果在程序运行过程中不考虑该置换，则无法归结出结果。因此，当出现E语句时，需要对全局已存在子句进行遍历替换，这是有必要的。由于我们已经执行了E逻辑的功能，若再送入子集库中，显然对最终的结果无任何利处，还将损害程序执行效率。因此，无需将E逻辑语句再送入子集库中.

创新算法—长度限制策略

长度限制策略是指，删除归结结果语句中的部分语句，这些被删除的语句往往长度大于原始子句集中的最大长度，而这些语句包含的内容常常是多余的。

所以我们可以在一开始就对试图进入子集库的子句p进行一次判断，如果其长度大于初始子句集中的最大长度，那么就可认为该子句集是“低效”的，直接拒绝其入子集库请求即可。

这个过程看似较为“鲁莽”，但其实仔细想来，归结的最终目的是要归结出长度为1的A逻辑子句，那么一般的归结过程就应该是新生成的子句长度越来越小，直至生成空子句——而在这一过程中我们的确应当减少对过长子句的考虑，以减少搜索时间。

这一优化策略的确是不完备的，有一定概率会使得归结结果不准确，但正如线性归结策略一样，这种情况发生的概率较小，且该情况不符合绝大多数的归结过程，所以我们可以允许耗费少量的准确性换取更快的推理速度，和更少的推理归结次数。

创新算法—包容归结

包容归结策略的核心就是要清楚所有被知识库中的已有语句包容（即，比该语句更特例）的语句。换言之，该法的核心就是要保证每一次新加入的子句都要包含原子句中所不具有的新信息——所谓新信息就是这条新语句不能由当前子句集中已有的条件推导为永真句。

以破案问题的子句举例，破案问题中存在如下子句：

C2: ~S(x1,A)|H(x1,A)

C3: ~H(A,x2)|~H(C,x2)

T1: ~S(A,A)

我们可以很容易发现C2与C3可以进行子句归结，归结结果为：

p: ~S(A,A)|~H(C,A)

在不使用包容归结策略的前提下，根据最基本的BFS策略，语句p所以应当进入子集库。但当我们仔细查看p子句时，不难发现因为T1为真，所以p的一个文字~S(A,A)也为真，所以p子句其实本质上是一个永真句。

```
T89 : K(B,A)|A(A)    {{c9与T57归结}}
T90 : ~R(B,A)|~R(A,A)    {{c9与T71归结}}
T91 : ~R(B,A)|A(A)    {{c9与T72归结}}
T92 : A(B)|~R(A,A)    {{c9与T73归结}}
T93 : A(B)|A(A)    {{c9与T74归结}}
T94 : ~R(A,A)|H(B,B)    {{c9与T75归结}}
T95 : A(A)|H(B,B)    {{c9与T76归结}}
T96 : A(C)|~R(A,A)    {{c10与T55归结 且 u-->C}}
T97 : A(C)|A(A)    {{c10与T58归结 且 u-->C}}
T98 : H(B,C)|~R(A,A)    {{T10与T59归结}}
T99 : H(B,C)|A(A)    {{T10与T60归结}}
T100 : ~K(A,A)|~R(A,A)    {{T10与T65归结}}
T101 : ~K(A,A)|A(A)    {{T10与T66归结}}
T102 : E(A,B)|~R(A,A)    {{T10与T71归结}}
T103 : E(A,B)|A(A)    {{T10与T72归结}}
T104 : ~K(A,A)|K(A,A)    {{T17与T33归结}}
T105 : E(A,B)|K(A,A)    {{T18与T33归结}}
T106 : K(A,A)    {{T33与T34归结}}
T107 : ~R(A,A)    {{T34与T55归结}}
T108 : A(A)    {{T34与T58归结}}
answer:A
```

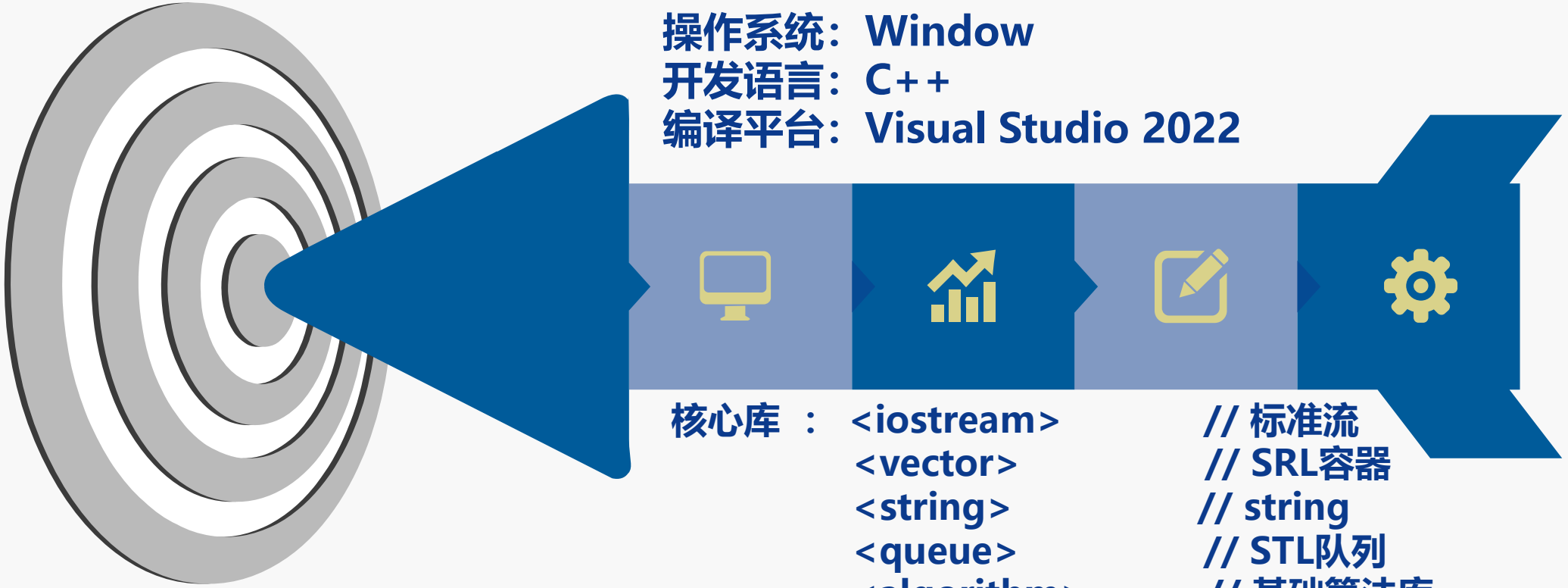
正是因为T1子句的存在，而且T1子句包容了p子句的内容，所以p子句的引入将不会再携带有任何新的信息，在这种情况下，就应该放弃p子句的引入，以使尽可能少的子句进入子集库，从而降低遍历次数。

该功能的实现就是将每一个试图进入子集库的语句进行先进行一次分析，将子集库中目前已存的所有子句与p子句进行比较分析，如果库中存在一个子句的所有文字都在p中有与之相同的文字对应，那么p子句就被该语句包含，则不应该让该语句入库。

第三部分

实验过程

环境说明

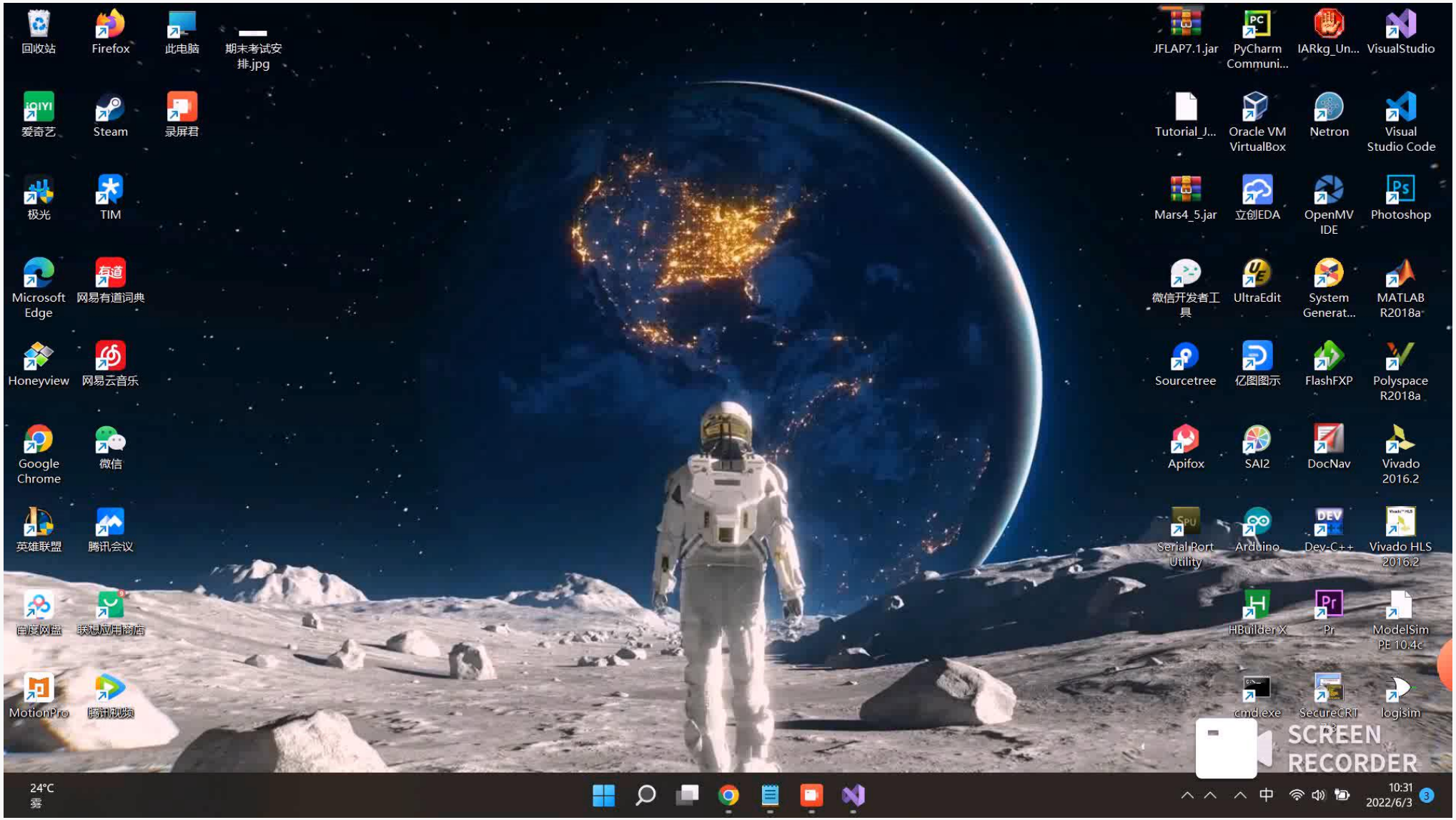


操作系统：Window
开发语言：C++
编译平台：Visual Studio 2022

核心库 ：

```
<iostream>      // 标准流
<vector>        // SRL容器
<string>        // string
<queue>         // STL队列
<algorithm>     // 基础算法库
<iomanip>       // 格式化输入输出
<windows.h>     // windows系统库
<graphics.h>    // EasyX库
<conio.h>       // EasyX库
```

成果展示


回收站
Firefox
此电脑
期末考试安排.jpg
爱奇艺
Steam
录屏君
极光
TIM
Microsoft Edge
网易有道词典
Honeyview
网易云音乐
Google Chrome
微信
英雄联盟
腾讯会议
百度网盘
联想应用商店
MotionPro
腾讯视频
JFLAP7.1.jar
PyCharm Communi...
IARkg_Un...
VisualStudio
Tutorial_J...
Oracle VM VirtualBox
Netron
Visual Studio Code
Mars4_5.jar
立创EDA
OpenMV IDE
Photoshop
微信开发者工具
UltraEdit
System Generat...
MATLAB R2018a
Sourcetree
亿图图示
FlashFXP
Polyspace R2018a
Apifox
SAI2
DocNav
Vivado 2016.2
Serial Port Utility
Arduino
Dev-C++
Vivado HLS 2016.2
HBuilderX
Pr
ModelSim PE 10.4c
cmd.exe
SecureCRT
logisim
SCREEN RECORDER
24°C
雾
中
10:31
2022/6/3

结果展示化简

首先考虑到本次实验的主要功能是实现子集编写的归结逻辑程序对破案问题进行推理，因此重点在于对归结过程的理解上，而非子句集的构造上。因此，实验采用人工的推导的方式生成子句集。

为了表述方便，给出谓词定义如下：

K(a,b):a杀了b

H(a,b):a恨b

E(a,b):a和b是一样的

R(a,b):a比b有钱

然后对题目给定条件使用以上谓词进行表示：

- ① $K(A,A) \vee K(B,A) \vee K(C,A)$
- ② $\forall x$，$K(x,A) \rightarrow H(x,A)$
- ③ $\forall x$，$H(A,x) \rightarrow \sim H(C,x)$
- ④ $\forall x$，$\sim E(x,B) \rightarrow H(A,x)$
- ⑤ $\forall x$，$\sim R(x,A) \rightarrow H(B,x)$
- ⑥ $\forall x$，$H(A,x) \rightarrow H(B,x)$
- ⑦ $\forall x \exists y$，$\sim H(x,y)$
- ⑧ $\forall x$，$K(x,A) \rightarrow \sim R(x,A)$
- ⑨ $\sim E(A,B)$

紧接着，将这9个一阶逻辑子句依次经过消除蕴涵词、否定内移、变量标准化、Skolem化、删除全称量词，将∧分配到∨中这几步操作后，再将结果转化成本程序可识别的子句输入，得到应当输入的内容为：（注:由于字符受限，采用|代替∨，其中f(a)表示除a以外的所有人）

K(A,A)|K(B,A)|K(C,A)

~K(a,A)|H(a,A)

~H(A,b)|~H(C,b)

E(c,B)|H(A,c)

R(d,A)|H(B,d)

~H(A,e)|H(B,e)

~H(g,f(g))

~K(h,A)|~R(h,A)

~E(A,B)

最后需要对问题进行转化，将其用谓词公式进行表示。假定最终结果谓词为A(u),u谋杀了A。因此得到问题谓词G:~K(u,A)|A(u)

结果展示—归结过程

K(A,A)|K(B,A)|K(C,A)
~K(a,A)|H(a,A)
~H(A,b)|~H(C,b)
E(c,B)|H(A,c)
R(d,A)|H(B,d)
~H(A,e)|H(B,e)
~H(g,f(g))
~K(h,A)|~R(h,A)
~E(A,B)
~K(u,A)|A(u)

T22	E(f(B),B)	c7+T7	g-->				
T37	H(C,A)|A(B)|~R(A,A)	c2+T27					
T38	H(C,A)|A(B)|A(A)	c2+T28					
T39	H(C,A)|~R(A,A)|H(B,B)	c2+T29					
T40	H(C,A)|A(A)|H(B,B)	c2+T30					
T41	~R(C,A)|~R(B,A)|~R(A,A)	c8+T24					
T42	~R(C,A)|~R(B,A)|A(A)	c8+T25					
T43	~R(C,A)|A(B)|~R(A,A)	c8+T27					
T44	~R(C,A)|A(B)|A(A)	c8+T28					
T45	~R(C,A)|~R(A,A)|H(B,B)	c8+T29					
T46	~R(C,A)|A(A)|H(B,B)	c8+T30					
T47	A(C)|~R(B,A)|~R(A,A)	c10+T24					
T48	A(C)|~R(B,A)|A(A)	c10+T25					
T49	A(C)|A(B)|~R(A,A)	c10+T27					
T51	A(C)|~R(A,A)|H(B,B)	c10+T29					
T52	A(C)|A(A)|H(B,B)	c10+T30					
T53	K(B,A)|~R(A,A)|~K(A,A)	T2+T17					
T54	K(B,A)|~R(A,A)|E(A,B)	T2+T18					
T55	K(C,A)|~R(A,A)	T2+T26					
T56	K(B,A)|A(A)|~K(A,A)	T3+T17					
T57	K(B,A)|A(A)|E(A,B)	T3+T18					
T58	K(C,A)|A(A)	T3+T26					
T59	H(B,C)|~R(B,A)|~R(A,A)	T11+T24					
T60	H(B,C)|~R(B,A)|A(A)	T11+T25					
T61	H(B,C)|A(B)|~R(A,A)	T11+T27					
T62	H(B,C)|A(B)|A(A)	T11+T28					
T63	H(B,C)|~R(A,A)|H(B,B)	T11+T29					
T64	H(B,C)|A(A)|H(B,B)	T11+T30					
T65	~K(A,A)|~R(B,A)|~R(A,A)	T17+T24					
T66	~K(A,A)|~R(B,A)|A(A)	T17+T25					
T67	~K(A,A)|A(B)|~R(A,A)	T17+T27					
T68	~K(A,A)|A(B)|A(A)	T17+T28					
T69	~K(A,A)|~R(A,A)|H(B,B)	T17+T29					
T70	~K(A,A)|A(A)|H(B,B)	T17+T30					
T71	E(A,B)|~R(B,A)|~R(A,A)	T18+T24					
T72	E(A,B)|~R(B,A)|A(A)	T18+T25					
T73	E(A,B)|A(B)|~R(A,A)	T18+T27					
T74	E(A,B)|A(B)|A(A)	T18+T28					
T75	E(A,B)|~R(A,A)|H(B,B)	T18+T29					
T76	E(A,B)|A(A)|H(B,B)	T18+T30					
T77	K(A,A)|K(B,A)	c1+T34					
T78	H(C,A)|~R(A,A)	c2+T55	a-->				
T79	H(C,A)|A(A)	c2+T58	a-->				
T80	~H(A,A)|~R(B,A)|~R(A,A)	c3+T35	b-->				
T81	~H(A,A)|~R(B,A)|A(A)	c3+T36	b-->				
T82	~H(A,A)|A(B)|~R(A,A)	c3+T37	b-->				
T83	~H(A,A)|A(B)|A(A)	c3+T38	b-->				
T84	~H(A,A)|~R(A,A)|H(B,B)	c3+T39	b-->				
T85	~H(A,A)|A(A)|H(B,B)	c3+T40	b-->				
T86	~R(C,A)|~R(A,A)	c8+T55	h-->				
T87	~R(C,A)|A(A)	c8+T58	h-->				
T88	K(B,A)|~R(A,A)	c9+T54					
T89	K(B,A)|A(A)	c9+T57					
T90	~R(B,A)|~R(A,A)	c9+T71					
T91	~R(B,A)|A(A)	c9+T72					
T92	A(B)|~R(A,A)	c9+T73					
T93	A(B)|A(A)	c9+T74					
T94	~R(A,A)|H(B,B)	c9+T75					
T95	A(A)|H(B,B)	c9+T76					
T96	A(C)|~R(A,A)	c10+T55	u-->				
T97	A(C)|A(A)	c10+T58	u-->				
T98	H(B,C)|~R(A,A)	T10+T59					
T99	H(B,C)|A(A)	T10+T60					
T100	~K(A,A)|~R(A,A)	T10+T65					
T101	~K(A,A)|A(A)	T10+T66					
T102	E(A,B)|~R(A,A)	T10+T71					
T103	E(A,B)|A(A)	T10+T72					
T104	~K(A,A)|K(A,A)	T17+T33					
T105	E(A,B)|K(A,A)	T18+T33					
T106	K(A,A)	T33+T34					
T107	~R(A,A)	T34+T55					
T108	A(A)	T34+T58					

				c1	K(A,A)|K(B,A)|K(C,A)		
				c2	~K(a,A)|H(a,A)		
				c3	~H(A,b)|~H(C,b)		
				c4	E(c,B)|H(A,c)		
				c5	R(d,A)|H(B,d)		
				c6	~H(A,e)|H(B,e)		
				c7	~H(g,f(g))		
			h-->	c8	~K(h,A)|~R(h,A)		
			h-->	c9	~E(A,B)		
			h-->	c10	~K(u,A)|A(u)		
			h-->	T1	K(B,A)|K(C,A)|H(A,A)		
			h-->	T2	K(B,A)|K(C,A)|~R(A,A)		
			h-->	T3	K(B,A)|K(C,A)|A(A)		
				T4	~K(A,A)|~H(C,A)		
				T5	~K(A,A)|H(B,A)		
				T6	~H(C,c)|E(c,B)		
				T7	E(e,B)|H(B,e)		
				T8	E(f(A),B)		
				T9	H(A,A)		
				T10	R(f(B),A)		
				T11	H(B,h)|~K(h,A)		
				T12	~H(A,f(B))		
				T13	K(B,A)|K(C,A)|~H(C,A)		
				T14	K(B,A)|K(C,A)|H(B,A)		
				T15	H(B,A)|K(C,A)|~R(A,A)		
				T16	H(B,A)|K(C,A)|A(A)		
			a-->	T17	~K(C,A)|~K(A,A)		
			a-->	T18	~K(C,A)|E(A,B)		
			b-->	T19	~H(C,A)		

T81	~H(A,A)|~R						
T82	~H(A,A)|A(						
T83	~H(A,A)|A(						
T84	~H(A,A)|~R						
T85	~H(A,A)|A(						
T86	~R(C,A)|~R						
T87	~R(C,A)|A(						
T88	K(B,A)|~R(						
T89	K(B,A)|A(A						
T90	c1+c2	~R(B,A)|~R		a			
T91	c1+c8	~R(B,A)|A(		h			
T92	c1+c10	A(B)|~R(A,		u			
T93	c2+c3	A(B)|A(A)		a			
T94	c2+c6	~R(A,A)|H(		a			
T95	c3+c4	A(A)|H(B,B		b			
T96	c4+c6	A(C)|~R(A,		c			
T97	c4+c7	A(C)|A(A)		g			
T98	c4+c9	H(B,C)|~R(		c			
T99	c5+c7	H(B,C)|A(A		g			
T100	c5+c8	~K(A,A)|~R		d			
T101	c6+c7	~K(A,A)|A(		g			
T102	c1+T4	E(A,B)|~R(A,A)				T10+T71	
T103	c1+T5	E(A,B)|A(A)				T10+T72	
T104	c2+T2	~K(A,A)|K(A,A)		a-->B		T17+T33	
T105	c2+T3	E(A,B)|K(A,A)		a-->B		T18+T33	
T106	c2+T4	K(A,A)		a-->C		T33+T34	
T107	c2+T6	~R(A,A)		a-->C	c-->A	T34+T55	
T108	c3+T9	A(A)		b-->A		T34+T58	

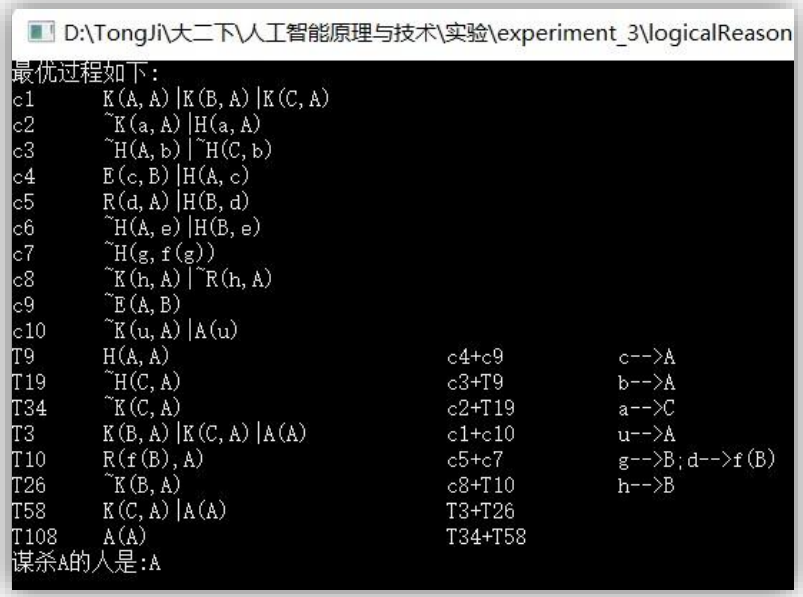
```
D:\TongJi\大二下\人工智能原理与技术\实验\experiment_3\logicalF
T70    ~K(A,A)|A(A)|H(B,B)          T17+T30
T71    E(A,B)|~R(B,A)|~R(A,A)       T18+T24
T72    E(A,B)|~R(B,A)|A(A)          T18+T25
T73    E(A,B)|A(B)|~R(A,A)          T18+T27
T74    E(A,B)|A(B)|A(A)             T18+T28
T75    E(A,B)|~R(A,A)|H(B,B)        T18+T29
T76    E(A,B)|A(A)|H(B,B)           T18+T30
T77    K(A,A)|K(B,A)                c1+T34
T78    H(C,A)|~R(A,A)               c2+T55          a-->C
T79    H(C,A)|A(A)                  c2+T58          a-->C
T80    ~H(A,A)|~R(B,A)|~R(A,A)      c3+T35          b-->A
T81    ~H(A,A)|~R(B,A)|A(A)         c3+T36          b-->A
T82    ~H(A,A)|A(B)|~R(A,A)         c3+T37          b-->A
T83    ~H(A,A)|A(B)|A(A)            c3+T38          b-->A
T84    ~H(A,A)|~R(A,A)|H(B,B)       c3+T39          b-->A
T85    ~H(A,A)|A(A)|H(B,B)          c3+T40          b-->A
T86    ~R(C,A)|~R(A,A)              c8+T55          h-->C
T87    ~R(C,A)|A(A)                 c8+T58          h-->C
T88    K(B,A)|~R(A,A)               c9+T54
T89    K(B,A)|A(A)                  c9+T57
T90    ~R(B,A)|~R(A,A)              c9+T71
T91    ~R(B,A)|A(A)                 c9+T72
T92    A(B)|~R(A,A)                 c9+T73
T93    A(B)|A(A)                    c9+T74
T94    ~R(A,A)|H(B,B)               c9+T75
T95    A(A)|H(B,B)                  c9+T76
T96    A(C)|~R(A,A)                 c10+T55         u-->C
T97    A(C)|A(A)                    c10+T58         u-->C
T98    H(B,C)|~R(A,A)               T10+T59
T99    H(B,C)|A(A)                  T10+T60
T100   ~K(A,A)|~R(A,A)              T10+T65
T101   ~K(A,A)|A(A)                 T10+T66
T102   E(A,B)|~R(A,A)               T10+T71
T103   E(A,B)|A(A)                  T10+T72
T104   ~K(A,A)|K(A,A)               T17+T33
T105   E(A,B)|K(A,A)                T18+T33
T106   K(A,A)                       T33+T34
T107   ~R(A,A)                      T34+T55
T108   A(A)                         T34+T58
谋杀A的人是:A
```

Gonzalez

结果展示—最优归结路径

K(A,A)|K(B,A)|K(C,A)
~K(a,A)|H(a,A)
~H(A,b)|~H(C,b)
E(c,B)|H(A,c)
R(d,A)|H(B,d)
~H(A,e)|H(B,e)
~H(g,f(g))
~K(h,A)|~R(h,A)
~E(A,B)
~K(u,A)|A(u)

由于仍然采用bfs搜索，存在较多的重复步骤，因此，我们可以根据最终结果回溯得到一条最优归结路径，避免了很多不必要的归结过程。可以看出，真正的有用归结只需要8步而已。



最优过程如下:

c1	K(A,A)|K(B,A)|K(C,A)		
c2	~K(a,A)|H(a,A)		
c3	~H(A,b)|~H(C,b)		
c4	E(c,B)|H(A,c)		
c5	R(d,A)|H(B,d)		
c6	~H(A,e)|H(B,e)		
c7	~H(g,f(g))		
c8	~K(h,A)|~R(h,A)		
c9	~E(A,B)		
c10	~K(u,A)|A(u)		
T9	H(A,A)	c4+c9	c-->A
T19	~H(C,A)	c3+T9	b-->A
T34	~K(C,A)	c2+T19	a-->C
T3	K(B,A)|K(C,A)|A(A)	c1+c10	u-->A
T10	R(f(B),A)	c5+c7	g-->B;d-->f(B)
// 过程中执行E(f(B),B)的等价归结			
T26	~K(B,A)	c8+T10	h-->B
T58	K(C,A)|A(A)	T3+T26	
T108	A(A)	T34+T58	

谋杀A的人是:A

第四部分

总结

总结

问题

在实验过程中也遇到较多困难，其中花费较长时间的是对E逻辑的处理上，最开始没有意识到等价的逻辑内涵，导致一直无法运行成功。后续经过手动完成推理过程，发现等价逻辑内涵在归结过程中的应用，才意识到需要在归结过程中对等价的二者进行统一，这样才能推导出最终结果。在采用E等价逻辑后，便能顺利得到正确结果。

体会

经过本次实验，我对于一阶逻辑、归结原理以及置换合一方法有了更深的了解，尤其是对归结原理的具体使用上。使用归结规则可以基于现有的条件推导出正确的结论，起初对于这样的方式感到惊奇，但后续自行推导实现求得最终结论后，也觉得理所当然。通过对其中步骤的亲自实现，对归结内部的原理和细节有了更深的了解，这也激励我之后在这方面进一步学习，做到知其然还要知其所以然。

今后

后续关注如何归结原理的程序实现上进一步优化，找到更优秀的归结方式，能迅速找到对得到最终结果最有效的归结步骤，缩短不必要的归结耗费，提升程序的智能。

与此同时，也可以了解更多基于逻辑的推理方法，学习人工智能领域更多基础的知识，进一步夯实基础。

参考文献

[1]王湘云.一阶谓词逻辑在人工智能知识表示中的应用[J].重庆工学院学报(社会科学版),2007(09):69-71.
[2]徐从富,郝春亮,苏保君,楼俊杰.马尔可夫逻辑网络研究[J].软件学报,2011,22(08):1699-1713.
[3]邹丽. 基于语言真值格蕴涵代数的格值命题逻辑及其归结自动推理研究[D].西南交通大学,2010.
[4]刘叙华.Horn集上的输入半锁归结原理[J].科学通报,1985(16):1201-1202.

感谢聆听

人工智能原理与技术课程

汇报人：*** 指导老师：*** 2022.6.2