同济大学
TONGJI UNIVERSITY

离散数学

课程实验报告

命题逻辑推理

姓　　名：　　马小龙

学　　号：　　2353814

学　　院：计算机科学与技术学院（软件学院）

专　　业：　　软件工程

指导教师：　　李冰

二〇二四年十一月二十九日

目录

一 实验目的

本实验旨在通过具体的命题逻辑推理问题，帮助学生深入理解命题逻辑的基本概念、规律和应用，培养逻辑推理能力，并将理论知识与实践应用相结合。

1.通过实验，学生将深入理解命题逻辑的核心概念，以及命题公式的变形与推导，掌握命题逻辑的基本框架。

2.通过解决具体的命题逻辑推理问题，学生将锻炼分析复杂逻辑关系的能力，提升对命题之间逻辑联系的敏感性与理解力。

3.学生将学习如何将一个问题转化为逻辑表达式，借助逻辑表达式进行命题逻辑推理，为处理更复杂的逻辑问题奠定基础。

4.通过命题逻辑的推导与验证，学生将提高抽象思维能力和系统分析能力，能够将逻辑推理方法应用于其他学科或实际问题中。

5.通过编写程序，学生将实践命题逻辑在计算机科学中的应用，理解其在程序验证、算法设计和人工智能等领域中的实际作用。

通过本实验，学生不仅能够熟练掌握命题逻辑推理的基本工具和技巧，还能有效将理论知识应用于实际问题，进一步提升逻辑思维、分析和编程能力。实验的最终目标是使学生能够运用命题逻辑方法解决日常生活和学术研究中的逻辑推理问题，同时为深入学习其他形式逻辑体系打下坚实基础。

二 实验内容

根据下面命题，用命题逻辑推理方法确定谁是作案者，并给出推理过程。

（1）营业员 A 或 B 偷了手表；

（2）若 A 作案，则作案不在营业时间；

（3）若 B 提供的证据正确，则货柜未上锁；

（4）若 B 提供的证据不正确，则作案发生在营业时间；

（5）货柜上了锁。

三 实验环境

编程开发语言：C++

文 件 编 码：UTF-8

集成开发环境：Clion 2024

工 具 集：MinGW 11.0 w64

四 实验原理

4.1 合取

二元命题联结词。将两个命题 P、Q 联结起来，构成一个新的命题 P∧Q, 读作 P、Q 的合取, 也可读作 P 与 Q。这个新命题的真值与构成它的命题 P、Q 的真值间的关系为只有当两个命题变项 P 为真,Q 为真时，P∧Q 为真, 而 P、Q 只要有一方为假，则 P∧Q 为假。

P	Q	P∧Q
0	0	0
0	1	0
1	0	0
1	1	1

表 4.1 合取运算真值表

4.2 析取

二元命题联结词。将两个命题 P、Q 联结起来，构成一个新的命题 P∨Q, 读作 P、Q 的析取, 也可读作 P 或 Q。这个新命题的真值与构成它的命题 P、Q 的真值间的关系为只有当两个命题变项 P 为假,Q 为假时，P∨Q 为假, 而 P、Q 只要有一为真则 P∨Q 为真。

P	Q	P∨Q
0	0	0
0	1	1
1	0	1
1	1	1

表 4.2 析取运算真值表

4.3 条件

二元命题联结词。将两个命题 P、Q 联结起来，构成一个新的命题 P→Q, 读作 P 条件 Q, 也可读作如果 P，那么 Q。这个命题的真值与构成它的命题 P、Q 的真值间的关系为只有当两个命题变项 P 为真, Q 为假时，P→Q 才为假, 其余均为真。

P	Q	P→Q
0	0	1
0	1	1
1	0	0
1	1	1

表 4.3 条件运算真值表

4.4 双向条件

二元命题联结词。将两个命题 P、Q 联结起来，构成一个新的命题 P←→Q, 读作 P 双条件于 Q，也称 P、Q 等价。这个新命题的真值与构成它的命题 P、Q 的真值间的关系为当两个命题变项P 为真, Q为真时 P←→Q 为真, 其余均为假。

P	Q	P→Q
0	0	1
0	1	1
1	0	0
1	1	1

表 4.4 双向条件运算真值表

4.5 命题逻辑推理

命题逻辑推理是通过应用命题逻辑的规则，从已知的命题（前提）推导出新的命题（结论）的过程。命题逻辑是一种形式化的推理方法，主要研究命题及其之间的逻辑关系。推理过程通过逻辑联结词（如“与”、“或”、“非”等）将简单命题组合成复杂命题，然后应用推理规则进行分析。常见的推理规则包括假言推理（Modus Ponens）、否定假言推理（Modus Tollens）、选言推理（Disjunctive Syllogism）等。这些规则确保了推理过程的正确性和有效性。命题逻辑推理广泛应用于数学、计算机科学和人工智能等领域。

五 实验过程

5.1 实验思路

1.符号化上面的命题，将它们作为条件，得出一个复合命题：

第一步：用命题变元表示：

A:营业员 A 偷了手表

B:营业员 B 偷了手表

C:作案不在营业时间

D:B 提供的证据正确

E:货柜未上锁

第二步：符号化命题：

命题 1: 营业员 A 或 B 偷了手表\

符号化: A ∨ B

命题 2: 若 A 作案，则作案不在营业时间

符号化: A → C

命题 3: 若 B 提供的证据正确，则货柜未上锁\

符号化: D → E

命题 4: 若 B 提供的证据不正确，则作案发生在营业时间\

符号化: ¬D → ¬C

命题 5: 货柜上了锁

符号化: ¬E

第三步：复合命题表示

(A ∨ B) ∧ (A → C) ∧ (D → E) ∧ (¬D → ¬C) ∧ ¬E

改为 C++语言表示的符号化逻辑表达式

(A || B) && (!A || C) && (!D || E) && (D || !C) && !E

2. 程序通过遍历所有可能的赋值情况为逻辑表达式，找到逻辑表达式的成真赋值，此时取出命题 A，和命题 B 的真值，根据真值即可判断作案者身份。

5.2 程序框架设计

程序的框架大致分为几个部分：

1. 命题变元表示：将自然语言中的命题用符号表示出来。在命题逻辑中，我们使用字母（如 A、B、C 等）来代表具体的陈述、不能再细分的命题。命题变元是一种简化和抽象化的方式，便于进行后续的推理和逻辑运算。

2. 符号化命题：将自然语言中的命题通过逻辑符号表达出来。使用一系列的逻辑联接词连接命题变元来描述称述之间的逻辑关系，形成符号化命题。

3. 构建复合命题：将多个命题通过逻辑联接词（本题主要是“与”）结合。复合命题将包含全部的逻辑关系，使我们能够从中推导出最终结论。复合命题中的联结词不能直接被编译器识别，我们还需要将复合命题改写为符合 C++语言的逻辑表达式。

4.推理：程序通过检查所有可能的命题变元赋值，找到满足所有条件的特定赋值组合。根据找到的真值，输出逻辑推理结果。

5.3 核心算法实现

核心的推理过程是通过穷举法来遍历所有可能的命题变元的真值组合，并通过逻辑运算符检查每个组合是否满足复合命题的条件。具体思路如下：

1.程序使用整数 x 来表示所有可能的命题变元真值组合。每个整数 x 在二进制中有 5 位（A、B、C、D、E），其中每一位对应一个命题变元的真值（0 表示假，1 表示真）。

2.程序通过位操作提取每一位的值，确定每个命题变元的真假。具体的操作如下：

A = (x >> 4) & 1;：右移 4 位后与 1 进行按位与运算，得到 A 的值（x 的第 5 位）。

B = (x >> 3) & 1;：右移 3 位后与 1 进行按位与运算，得到 B 的值（x 的第 4 位）。

C = (x >> 2) & 1;：右移 2 位后与 1 进行按位与运算，得到 C 的值（x 的第 3 位）。

D = (x >> 1) & 1;：右移 1 位后与 1 进行按位与运算，得到 D 的值（x 的第 2 位）。

E = x & 1;：与 1 进行按位与运算，得到 E 的值（x 的第 1 位）。

3.命题推理与判断条件

程序通过逐个检查每个可能的命题真值组合（x 从 0 到 31，共 32 种组合），并根据复合命题的逻辑条件进行判断。对于每一种组合，程序会检查以下条件是否成立：

(A || B)：营业员 A 或 B 偷了手表。

(!A || C)：若 A 偷了手表，则作案不在营业时间（即如果 A 为真，则 C 必须为真）。

(!D || E)：若 B 提供的证据正确，则货柜未上锁（即如果 D 为真，则 E 必须为真）。

(D || !C)：若 B 提供的证据不正确，则作案发生在营业时间（即如果 D 为假，则 C 必须为真）。

!E：货柜上了锁（即 E 必须为假）。

当找到符合所有条件的真值组合时，程序输出推理结果：

4.根据真值组合判断哪位营业员偷了手表：如果 A 为真，则表示营业员 A 偷了手表；否则，表示营业员 B 偷了手表。

如果没有符合条件的解，程序会输出：“没有找到符合条件的解。”

六 实验结果测试

```
****************************************
**                                    **
**          欢迎进入命题逻辑推理系统      **
**                                    **
****************************************


**问题描述**
命题1 营业员 A 或 B 偷了手表；
命题2 若 A 作案，则作案不在营业时间；
命题3 若 B 提供的证据正确，则货柜未上锁；
命题4 若 B 提供的证据不正确，则作案发生在营业时间；
命题5 货柜上了锁。

**命题变元表示**
A：营业员 A 偷了手表
B：营业员 B 偷了手表
C：作案不在营业时间
D：B 提供的证据正确
E：货柜未上锁

**符号化命题表示**
命题1：营业员 A 或 B 偷了手表
符号化：A ∨ B
命题2：若 A 作案，则作案不在营业时间
符号化：A → C
命题3：若 B 提供的证据正确，则货柜未上锁
符号化：D → E
命题4：若 B 提供的证据不正确，则作案发生在营业时间
符号化：¬D → ¬C
命题5：货柜上了锁
符号化：¬E

**复合命题表示**
复合命题：(A ∨ B) ∧ (A → C) ∧ (D → E) ∧ (¬D → ¬C) ∧ ¬E

**C++语言代码表示**
C++语言代码：(A || B) && (!A || C) && (!D || E) && (D || !C) && !E

A=0, B=1
推理结果是：是营业员 B 偷了手表

欢迎下次再次使用！

进程已结束，退出代码为 0
```

程序通过检查所有可能的命题变元真值组合，找到满足所有条件的特定组合。命题逻辑推理结果为 A = 0，B = 1。

因此，是营业员 B 偷了手表。

七 实验心得

通过本次命题逻辑推理实验，我深入理解了命题逻辑的基本原理，掌握了如何将日常问题转化为形式化的逻辑表达，并通过程序实现逻辑推理过程。

在实验中，我首先将问题中的命题转化为符号化表示，这是进行命题逻辑推理的较为关键的步骤。通过使用命题变元（如 A、B、C、D、E）和逻辑运算符（如 ∨、→、¬），我能够将自然语言中的推理问题转化为更为精确、严谨的逻辑形式。这一过程帮助我加深了对命题逻辑符号的理解，也体会到了逻辑推理的抽象性和严谨性。

通过本次实验，我更加熟悉了命题逻辑中的基本逻辑运算符，如“与”(∧)、或(∨)、“非”(¬)、“蕴含”(→)等，这些运算符用于构建更复杂的命题并实现推理。尤其是在符号化命题中，如何根据给定条件将多个命题组合成复合命题并进行推理，是学习逻辑推理的核心。我还通过实验了解了如何使用这些基本的逻辑规则判断命题的真假，并进行推理。

在实验中，我使用了穷举法对所有可能的命题真值组合进行验证。通过对每一个命题变元的真值组合进行遍历，结合位运算符进行逻辑判断，最终找出符合条件的解。这种方法虽然计算量较大，但它通过系统化的方式保证了能够找出所有可能的解，并且保证了推理结果的正确性。当然，穷举法在面对更复杂的问题时可能效率较低，尤其是命题数量较多时，计算量会急剧增加，此时，如何有效组织和优化推理过程极为关键。因此，未来我希望进一步研究其他更高效的推理方法，以提升推理的效率。

本次命题逻辑推理实验不仅加深了我对命题逻辑的理解，还提升了我编程能力和逻辑思维能力。通过符号化命题、复合命题构建、逻辑推理等环节的实践，我体会到逻辑严谨性和程序设计的结合。同时，我也意识到在复杂推理问题中，如何选择合适的算法进行推理是非常关键的。未来我将继续学习并探索更多的逻辑推理方法和应用，以期进一步提高我的思维能力和问题解决能力。

附录 · 源码

```cpp
#include <iostream>
/*************************************************************
* Function Name:   main
* Function:
* Input Parameter: 命题逻辑推理
* Returned Value:  0
*************************************************************/
int main()
{
    //标语
    std::cout << ("*****************************************\n")
              << ("**                                     **\n")
              << ("**         欢迎进入命题逻辑推理系统      **\n")
              << ("**                                     **\n")
              << ("*****************************************\n\n\n");
    //问题描述
    std::cout << "**问题描述**\n"
              << "命题 1 营业员 A 或 B 偷了手表; \n"
              << "命题 2 若 A 作案, 则作案不在营业时间; \n"
              << "命题 3 若 B 提供的证据正确, 则货柜未上锁; \n"
              << "命题 4 若 B 提供的证据不正确, 则作案发生在营业时间; \n"
              << "命题 5 货柜上了锁。\n\n";
    //命题变元表示
    std::cout << "**命题变元表示**\n"
              << "A: 营业员 A 偷了手表\n"
              << "B: 营业员 B 偷了手表\n"
              << "C: 作案不在营业时间\n"
              << "D: B 提供的证据正确\n"
              << "E: 货柜未上锁\n\n";
    std::cout << "**符号化命题表示**\n";
    // 输出每个命题的符号化表达式
    std::cout << "命题 1: 营业员 A 或 B 偷了手表\n"
            << "符号化: A ∨ B\n"; // A 或 B 偷了手表
    std::cout << "命题 2: 若 A 作案, 则作案不在营业时间\n"
            << "符号化: A → C\n"; // 若 A 作案, 则作案不在营业时间
    std::cout << "命题 3: 若 B 提供的证据正确, 则货柜未上锁\n"
            << "符号化: D → E\n"; // 若 B 提供的证据正确, 则货柜未上锁
    std::cout << "命题 4: 若 B 提供的证据不正确, 则作案发生在营业时间\n"
            << "符号化: ¬D → ¬C\n"; // 若 B 提供的证据不正确, 则作案发生在营业时
间
```

```
    std::cout << "命题5: 货柜上了锁\n"
            << "符号化: ¬E\n\n"; // 货柜上了锁
    std::cout << "**复合命题表示**\n";
    std::cout << "复合命题: (A ∨ B) ∧ (A → C) ∧ (D → E) ∧ (¬D → ¬C) ∧
¬E\n\n";
    std::cout<<"**C++语言代码表示**\n";
    std::cout<<"C++语言代码: (A || B) && (!A || C) && (!D || E) && (D || !C)
&& !E\n\n";
    //命题推理过程
    // x 从 0 到 31, 对应二进制 00000 到 11111
    for (int x = 0; x <= 31; x++) {
        // 通过位操作提取每一位
        // >>: 右移运算符;  &: 按位与运算符
        const bool A = (x >> 4) & 1;
        const bool B = (x >> 3) & 1;
        const bool C = (x >> 2) & 1;
        const bool D = (x >> 1) & 1;
        const bool E = x & 1;
        // 判断逻辑条件
        if ((A || B) && (!A || C) && (!D || E) && (D || !C) && !E) {
            std::cout << "A=" << A << ", B=" << B << "\n";
            std::cout << "推理结果是: 是营业员" << (A ?" A ": " B ") << "偷了手表
\n\n";
            std::cout << "欢迎下次再次使用!\n";
            return 0;
        }
    }
    std::cout << "没有找到符合条件的解。\n";
    return 0;
}
```