循环冗余校验CRC生成和校验程序

第三章实验1报告

赖泽强组

# 1 实验目的

* 熟悉循环冗余校验算法。
* 理论结合实现，提高实践能力。

# 2 实验内容

用三种语言C++,Java,Python实现循环冗余校验码(CRC)的生成与校验，具体要求如下所示。

* 输入信息放在一个文件当中，文件应包含以下信息。
  + 待发送的数据信息二进制比特串（32位）
  + 收发双方预定的生成多项式采用CRC-CCITT=X16+X12+X5+1，对应的二进制比特串（17位）
  + 接收的数据信息二进制比特串（32位）
* 输出应包含以下内容：
* 首先显示待发送的数据信息二进制比特串
* 然后显示收发双方预定的多项式（以二进制比特串显示）。
* 显示生成的CRC-Code，以及带校验和的发送帧
* 显示接收的数据信息二进制比特串，以及计算生成的CRC-Code
* 显示余数，为零表示无错，不为零表示出错

# 3 实验原理

CRC为校验和的一种，是两个字节数据流采用二进制除法（没有进位，使用XOR来代替减法）相除所得到的余数。其中被除数是需要计算校验和的信息数据流的二进制表示；除数是一个长度为(n+1)n+1的预定义（短）的二进制数，通常用多项式的系数来表示。在做除法之前，要在信息数据之后先加上nn个0.

一个简单的例子：

* 被除数（也就是需要被传送的信息）为11010。
* 除数（多项式）为101。

计算过程如表1所示：

|  |  |
| --- | --- |
| 11101  -------------  101 ) 1101000  101....  ---....  111...  101...  ---...  100..  101..  ---..  010.  000.  ---.  100  101  ---  01 | 末尾补3-1=2个0  余数为01 |

表1 CRC示例

# 4 实验环境

## 4.1 开发环境

|  |  |
| --- | --- |
| 语言 | 环境 |
| C++ | macOS 10.14.4  Apple LLVM version 10.0.1 (clang-1001.0.46.3) |
| Java | Windows 10 1903  IntelliJ IDEA 2018.3.3 |
| Python | Windows 10 1903  PyCharm 2018.3.3 |

## 4.2 部署环境

可在Windows，Linux，macOS三平台正确编译运行。

# 5 实验步骤

1. 由赖泽强编写出C++版本的循环冗余校验程序。
2. 由黄天根据C++版本的程序，将其翻译成Java和Python版本。

## 5.1 代码简单流程

1. 读取命令行参数，若指定配置文件，则按指定路径读取配置文件。
2. 使用read\_configuration读取配置文件。
3. 使用crc\_remainder计算CRC-Code。
4. 使用crc\_check校验。
5. 格式化输出。

# 6 实验总结

这个实验主要就是发送端通过特定算法生成校验码，接收端接收并通过校验码判断接收到的信息是否正确。在从c++版本到java版本和python版本时出现了字符覆盖的错误，后通过了解知道用strip()函数进行操作。通过这次试验我们了解了在数据链路层是如何采用冗余编码技术对信息进行检错的一种方式，也对数据链路层数据的传输有了更深刻的认识。