THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

EXPERIMENT NO. 3

Aim :- Implementation of CRC (Cyclic Redundancy Code) for Error Detection.

Theory :-

CRC or Cyclic Redundancy Check is a method of detecting accidental changes / errors in communication channel.

CRC uses generator polynomial is of the form like x^3+x+1 which is available on both sender and receiver side. Another example is x^2+1 that represents key 101.

Let n be number of bits in data to be sent from sender side.
Let k be number of bits in the key obtained from generator polynomial.

Sender Side :

1. The binary data is first augmented by adding k-1 zeroes in the end of the data.
2. Use modulo-2 binary division to divide the key and store remainder of division.
3. Append the remainder at the end of the data to form encoded data and send the same.

Receiver Side :

Perform modulo-2 division again and if the remainder is 0, then there are no errors.
For modulo-2 binary division, instead of subtraction, we use XOR here.
In each step, a copy of divisor is XORed with k bits of dividend. The result of the XOR operation (remainder) is (n-1) bits, which is used for next step after 1 extra bit is pulled down to make it n bits long. When there are no bits left to pull down, we have a result. The (n-1) bit remainder which is appended at sender side.

For example, data word to be sent - 100100
Key - 1101

Sender Side,
Adding n-1 = 4-1 = 3 zeroes at the end of data and taking modulo-2 division,

```
          111101
      ------------
1101 ) 100100000
       - 1101
       -------
         1000
         - 1101
         -------
           1010
           - 1101
           -------
           011010
```

```
  1101
- 0110
  0000
- 1100
  1101
-  001
```

Therefore, remainder is 001 and hence the encoded data sent is 100100001.

Now, for the Receiver Side,
Code word received at receiver side : 1001000

```
          111101
      ____________
1101 ) 100100001
       - 1101
         1000
       - 1101
          1010
        - 1101
           1110
         - 1101
            1101
          - 1101
            0000
```

Therefore, the remainder is all zeroes. Hence the data received has no error.

Conclusion :-

Thus, the cyclic redundancy check (CRC) is a technique which is used to detect the errors in the digital data.

22/8/23

```
#CRC code generation

def CRC(data, poly):


    data.extend([0] * (len(poly) - 1))
    for i in range(len(data) - len(poly) + 1):
        if data[i] == 1:
            for j in range(len(poly)):
                data[i + j] = data[i + j] ^ poly[j]

    crc = data[-len(poly) + 1:]

    return crc



data = list(map(int, input("Enter the data bits:\n")))
dup_data=data.copy()
poly = list(map(int, input("Enter the generator polynomial in binary
form:\n")))
crc_code = CRC(data,poly)
dup_data.extend(crc_code)
print("CRC code:{}".format(''.join(map(str, dup_data)) ))
# error checking
def check():
    data1 = list(map(int,input("Enter the data word received by you:
\n")))
    data1_dup=data1.copy()

    poly1 = list(map(int,input("Enter your generator polynomial in
binary form:\n")))

    for i in range(len(data1) - len(poly1) + 1):
        if data1[i] == 1:
            for j in range(len(poly1)):
                data1[i + j] = data1[i + j] ^ poly1[j]
    crc = data1[-len(poly) + 1:]

    count = 0
    for i in range(len(poly1)-1):
        if crc[i]!=0:
            count+=1
    if count==0:
        print("The codeword received has no error")
        print("The correct data bits are =",(data1_dup[0:-
len(poly1)+1]))
    else:
        print("The received code word is wrong\n")

check()
'''
python -u "C:/Users/Rishab/OneDrive/Desktop/CN Experiments/import
crc.py"
Enter the data bits:
110101
Enter the generator polynomial in binary form:
101
CRC code:11010111
```

```
Enter the data word received by you:
11010110
Enter your generator polynomial in binary form:
101
The received code word is wrong
python -u "C:/Users/Rishab/OneDrive/Desktop/CN Experiments/import
crc.py"
Enter the data bits:
1101011
Enter the generator polynomial in binary form:
101
CRC code:110101110
Enter the data word received by you:
110101110
Enter your generator polynomial in binary form:
101
The codeword received has no error
The correct data bits are = [1, 1, 0, 1, 0, 1, 1]
'''
```