CS425A: Computer Networks
Assignment 2

Surendra Kumar Ahirwar
211083

Question 1 (20 points)

Write a program (in any language) that generates an n-bit frame for transmission from a k-bit data block D and a $(n-k+1)$ bit CRC pattern P. Compile and run the program with at least two sets of inputs to confirm that this program is generating CRC patterns correctly.

Now, modify the program that performs the following steps:

1. Generates a message of $k = 10$ bits.
2. Uses the previous code with $P = 110101$ to generate the corresponding 15-bit frame T for transmission.
3. Generates transmission errors at any bit positions of T.
4. Applies CRC to the received frame (i.e., frame T after introducing errors) to determine if the frame should be accepted or discarded.

Attach your code with your report, and also put comments stating the instructions of how to run it.

Solution: The file is named `ass_2 Q1.cpp`. The program only uses the STL (Standard Template Library) of C++ and can be compiled using `g++` compiler using the command `g++ ass_2 Q1.cpp` which produces an executable.

The program asks if the generation of CRC patterns is to be verified (i.e., the first statement), or introduce some errors in a 15-bit frame test its validity (i.e., the latter parts). Enter 1 for the first statement and 2 for the latter part of the question.

If 2 is entered, the program automatically produces a message of length 10 bits, calculates its FCS with respect to P, introduces some errors to its bits (independently and with some fixed error probability), and tests the new frame for validity.

However, on entering 1, the user is given a choice on whether they want to give input to the values of n and k. If not, the program continues with the default values $n = 10$ and $k = 6$.

The program can be modified easily to generate errors for frames (or messages or polynomials) of different sizes.

Question 2 (10 points)

In the Go-back-N ARQ mechanism using k-bit sequence numbers, why is the window size limited to $2^k - 1$ and not 2^k?

Solution: In the Go-Back-N Automatic Repeat reQuest (ARQ) mechanism, the window size is limited to $2^k - 1$, where k represents the number of bits in the sequence number field. This limitation is due to the need to differentiate between different sequence numbers while ensuring that the sequence number space doesn't wrap around too quickly.

The window size is limited to $2^k - 1$ because:

1. **Sequence Number Space**: With k bits for sequence numbers, there are 2^k possible sequence numbers. However, these sequence numbers need to be utilized to uniquely identify each frame in the window. If the window size were to be 2^k, the sequence number space would be exhausted after transmitting 2^k frames, and the sender would have to start reusing sequence numbers.

2. **Avoiding Confusion**: If the window size were exactly 2^k, the sender and receiver might confuse old frames with new ones. For example, if the window size were 4 (2^2) and the sequence numbers were 00, 01, 10, and 11, once frame 11 is sent, the next sequence number would wrap around to 00, potentially causing confusion if frame 11 were to be delayed or lost and then later retransmitted.

3. **Acknowledgment Handling**: With a window size of $2^k - 1$, the receiver can distinguish between new frames and retransmitted frames using the sequence numbers. If the window size were 2^k, it would be more difficult for the receiver to distinguish between the original transmission of a frame and its retransmission, leading to potential errors in acknowledgment handling.

Suppose the window size is $n = 2^k$ and the sender sends a window containing the frames in the order $(0, 1, \ldots, n-1)$ (without loss of generality). The receiver receives all of these successfully, sends an acknowledgement signal, and expects to receive the next window also containing the frames in the same order.

Let the acknowledgment signal be lost due to noise or some other reason, so that the sender runs out of time. Thus, it sends frames corresponding to the previous window again. However, since the receiver was expecting frames of the next window, it accepts them into the next window. Now, the receiver has a duplicate of the first window, which was not intended by the sender. Hence, we run into an error.

In the case of window size $n = (2^k - 1)$ however, this problem does not occur. Thus, the window size is limited to $2^k - 1$.

By limiting the window size to $2^k - 1$, the Go-Back-N ARQ mechanism ensures efficient and reliable communication while utilizing the available sequence number space effectively.

Question 3 (10 points)

What is the maximum window size that can be used in the Selective-Reject ARQ mechanism that uses k-bit sequence numbers? Explain your answer.

Solution: Using k-bit sequence numbers, there can be at most 2^k unique frame numbers. Let the window size be w. Assuming a scenario where there's an overlap between consecutive windows. Without loss of generality, it considers the first window, f_1, containing frame numbers from 0 to $w-1$, and the next window, f_2, containing frame numbers from w to 0 and possibly some additional frames. This overlap introduces ambiguity regarding which window an overlapping frame belongs to.

Ambiguity in Overlapping Frames: The sender repeats the first window (f_1) due to a timeout, sending frame 0 again. However, since the receiver was expecting frames from the next window (f_2), it accepts frame 0 into its buffer, assuming that the preceding frames of window f_2 were lost. This ambiguity arises because it's unclear which window the repeated frame 0 belongs to.

Avoiding Overlap: To avoid this ambiguity, the solution argues that consecutive windows must be disjoint, meaning they contain distinct frames with no overlap. This ensures that each frame uniquely belongs to its respective window.

Maximum Window Size Calculation: Given that two consecutive windows contain $2w$ distinct frames and the total number of frames with k-bit sequence numbers is 2^k, the solution sets up the inequality $2w \leq 2^k$ to find the maximum window size (w) that prevents overlap. Solving the inequality yields $w \leq 2^{k-1}$, indicating that the window size must be less than or equal to half of the sequence number space (2^k), ensuring no overlap between consecutive windows.

Therefore, the maximum window size that can be used in the Selective-Reject ARQ mechanism with k-bit sequence numbers is determined to be 2^{k-1}.

Question 4 (10 points)

A channel has a data rate of 4 kbps and a propagation delay of 20 ms. For what range of frame sizes does stop-and-wait give an efficiency of at least 50%?

Solution: We need the efficiency (U) to be at least 50%. We find a constraint on a using the efficiency relation for stop-and-wait flow control which is given by $U = \frac{1}{1+2a}$:

$$U \geq 50\% = \frac{1}{2}$$

$$\frac{1}{1+2a} \geq \frac{1}{2} > 0$$

$$1 + 2a \leq 2$$

$$2a \leq 1$$

$$a \leq \frac{1}{2}$$

Now, given this constraint on a, we find a constraint on the transmission time t_{frame} using the definition of $a = \frac{t_{prop}}{t_{frame}}$ and the fact that we are given $t_{prop} = 20$ ms. Essentially, we want 20 ms$/t_{frame} \leq \frac{1}{2} \Rightarrow 40$ ms. $\leq t_{frame}$.

Finally, let S_f be the size of a frame (in bits). Then we have,

$$40ms. \leq t_{frame} = \frac{S_f}{4kbps}$$

$$\Rightarrow 40 \times 10^{-3} \leq \frac{S_f}{4000bps}$$

$$160bits \leq S_f$$

Thus, we need the size of the frame to be at least 160 bits in order for the efficiency to be at least 50% in stop-and-wait.

Solution (5):

(a) ∵ error is $10^{-3} = p$

probability of no error in particular bit

$$1 - p = 0.999$$

Let E_i be the event that i^{th} bit flipped

$$\Rightarrow P(E_i) = p \Leftrightarrow P(\bar{E}_i) = 1 - p$$

Let $P(\bar{E}_1 \cap \bar{E}_2 \cap \bar{E}_3 \cap \bar{E}_4)$

i.e there no error in $1^{st}, 2^{nd}, 3^{rd}, 4^{th}$ bit

$$= \prod_{i=1}^{i=4} P(\bar{E}_i)$$

$$= \prod_{i=1}^{i=4} (1-p) = (1-p)^4$$

$$\approx 1 - 4 \times 10^{-3}$$

$$\boxed{= 0.996}$$

(b) using first part.

probability that frame contain at least one error, let S be event of this.

$$P(S) = 1 - P(\bar{S})$$

$$= 1 - 0.996$$

$$\boxed{= 0.004}$$

(C) when addition of a parity bit

Frame = 5 bit

- we can not deetect error if there is even no. of inversions by parity check

<u>cases</u> with 2 errors & 4 errors

it can be check by 5 bit parity check.

<u>case : 2 errors</u>

no. of pairs of 2 bits

$= \binom{5}{2}$

$\therefore$ each bit has same probability of being erroneous.

<u>then</u> probabilty j & k being erroneous.

$$P(\mathcal{E}_j \cap \mathcal{E}_k)\big|_{j=k} = P(\mathcal{E}_j) \cdot P(\mathcal{E}_k)$$

$$= p \times p = p^2$$

but for exactly 2 errors.

we ensure that there is no error in any other bit

$$P\left(\bigcap_{t \notin (j,k)} \bar{\mathcal{E}}_t\right) = \prod_{t \notin (j,k)} P(\bar{\mathcal{E}}_t)$$

$$= (1-p)^3$$

$$P(2 \text{ error}) = \prod_{j \neq k} p^2 \times (1-p)^3$$

$$= \binom{5}{2} \cdot p^2 \cdot (1-p)^3$$

$$= 9.97 \times 10^{-6}$$

Case exactly 4 error

$$P(4 \text{ errors}) = \binom{5}{4} \cdot P^4 \times (1-P)^1$$

$$= 5 \times P^4 \times (1-P).$$

$$= 4.995 \cdot 10^{-12}$$

probability of error not detected.

$$P(2 \text{ error}) + P(4 \text{ error})$$

$$= 9.97 \times 10^{-6}$$

Q.6 For $P = 110011$ and $M = 1110001 1$
Find CRC

```
             1 0 1 1 0 1 1 0
P= 110011 ) 1 1 1 0 0 0 1 1 0 0 0 0 0
            1 1 0 0 1 1
            -----------
            0 0 1 0 1 1 1
              0 0 0 0 0 0
              -----------
              1 0 1 1 1 1
              1 1 0 0 1 1
              -----------
                1 1 1 0 0 0
                1 1 0 0 1 1
                -----------
                0 1 0 1 1 0
                0 0 0 0 0 0
                -----------
                  1 0 1 1 0 0
                  1 1 0 0 1 1
                  -----------
                    1 1 1 1 1 0
                    1 1 0 0 1 1
                    -----------
                    0 1 1 0 1 0
                    0 0 0 0 0 0
                    -----------
               R =    1 1 0 1 0
```

So,

CRC is
$= 11010$

Q.7

Solution

(a) $M = 100100111011 \Leftrightarrow x^{10}+x^7+x^4+x^3+x+1$

$P(x) = x^4+x+1$

then. $\frac{x^4 \times M(x)}{P(x)}$

$$P(x) = x^4+x+1 \;\Big)\; \begin{array}{l} x^{10}+x^6 \\ \hline x^{14}+x^{11}+x^8+x^7+x^5+x^4 \\ x^{14}+x^{11}+x^{10} \\ \hline x^{10}+x^8+x^7 \\ x^{10}+x^7+x^6 \\ \hline x^8+x^6+x^5+x^4 \\ x^8+x^5+x^4 \\ \hline x^6 \\ x^6+x^3+x^2 \\ \hline R(x) = x^3+x^2 \end{array}$$

remainder = $\boxed{x^3+x^2}$

$= 1100$ in binary

encoding = $MR = 100100111011100$

(b) error pattern 10001000000000000

by XOR we received is 000110110111100

W = 000110110111100 by flipping 1 & 5 bit of MR

P = 10011

```
         11001110
10011 ) 000110110111100
           10011
            10000
            10011
             0011111
               10011
               011001
                10011
                 10100
                 10011
R.                1110
```

$R = 1110 = x^3 + x^2 + x$

which is non zero

So error will be detected

(c)

error pattern 10011000000000000

XOR $w = (100100110111100) \oplus 10011000000000000$

W = 000010110111100

```
            1010100
P(x)
=10011 ) 000010110111100
             10011
              10111
              10011
                10011
                10011
                R = 0
```

∴ R = 0000

$R(x) = 0$

error cannot detected