Problem Session 2

1. Hamming error correcting code is used for data transmission of size 22 bits. What is the minimum number of redundancy bits to correct (i) a single bit error (ii) at most two bit errors.
`Solution:`
(i) $2^r \geq 1 + (22 + r)$

(ii) $2^r \geq 1 + (22 + r) + \binom{(22+r)}{2}$

2. Consider the two protocols $P1$ and $P2$ which are a modified version of classical Stop-Wait protocol. For both $P1$ and $P2$, there are NO drops and NO delayed Acks/Nacks.

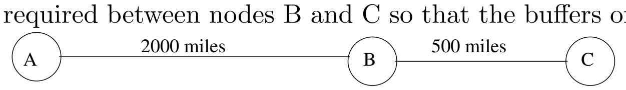
    - $P1$ operates with time-out and NACK. NO acks.

    - $P2$ operates with time-out and ACK. NO nacks.

    Between $P1$ and $P2$, which gives a better utilization. Is it necessary to include sequence numbers for both $P1$ and $P2$.
    `Solution:`
    **Case 1:** Channel is less noisy. If the channel is less noisy, then we get less number of NACKs compared to ACKs. In $P1$, ACK is modeled using time-out, i.e., the sender sends $F0$ and waits for an acknowledgement; if the sender does not receive NACK within time-out, it assumes, $F0$ is without error and sends $F1$. In case of $P2$, we get ACK 1. Since, the value of time-out is usually more than $t_{trans} + 2t_{prop} + t_{proc}$ and the sender can transmit the next frame as soon as it receives ACK 1, $P2$ performs better than $P1$ in this case.
    **Case 2:** Channel is noisy. We can expect more NACKs compared to ACKs and hence, with respect to $P2$, the time-out will be more. Thus, $P1$ performs better than $P2$.
    If the number of NACKs and ACKs are same, then both performs equally better.
    Since there are no delayed ACKs/drops, no need for sequence numbers. There will not be any synchronization issue.

3. A sliding window protocol is designed between the nodes $A$ and $B$ with SWS=32 at $A$. The link propagation is 100 micro.sec. What would be the link data transfer speed? Does it depend on RWS.
`Solution:` Suppose $l$ represents the link speed and $s$ represents the frame size, then $l$ can be computed from the following expression.
$\frac{100 \times 10^{-6} \times l}{s} = 32$. Based on $s$, the $l$ can be computed.
$l$ does not depend on RWS, however, it depends on DB product and the frame size.

4. A channel has a data rate of 4 kbps and a propagation delay of 20ms. For what range of frame sizes does stop and wait give an efficiency of at least 50%.
`Solution:` Assuming $t_{proc} = 0$ and there are no NACKs and time outs, $U = \frac{t_{trans}}{t_{trans} + 2t_{prop}} \geq 0.5$.
$t_{prop} = 20 \times 10^{-3}$; $t_{trans} = \frac{s}{4 \times 10^3}$.

5. In Figure given below, frames are generated at node A and sent to node C through node B. Determine the minimum transmission rate required between nodes B and C so that the buffers of node B are not flooded, based on the following:


(a) The data rate between A and B is 100 kbps. The propagation delay is 10 micro sec per mile for both lines. All data frames are 1000 bits long; ACK frames are separate frames of negligible length.

(b) Between A and B, a sliding window protocol with a window size of 3 is used. Between B and C, stop and wait is used.

(c) **Note:** In order **not** to flood the buffers of B, the average number of frames entering and leaving B must be the same over a long interval.

`Solution:`
To avoid flooding at $B$, the effective bandwidth of the link(A,B) must be same as the effective bandwidth of the link(B,C).
Utilization(A,B) $\times$ BW(A,B) = Utilization(B,C) $\times$ BW(B,C)
$\frac{3t_{trans}}{3t_{trans}+2t_{prop}} \times 100 \times 10^3 = \frac{t_{trans}}{t_{trans}+2t_{prop}} \times y$
Thus, we get $y = 75$ Kbps.
The protocol also works fine for $y = 150$ Kbps, $y = 200$ Kbps and $y = 300$ Kbps. `Assumption:` In the above example, it is assumed that simultaneous reading (from AB channel) and writing (into BC channel) is possible at $B$. Also, while $B$ is transmitting a frame to $C$, the part of the buffer of $B$ can be used to store frames from $A$. Further, there are no NACKs/delayed ACK/NACK / frame drops.

6. There is a point to point link between nodes $A$ and $B$ and it is proposed to implement one of the following protocols: Which protocol is suitable if the protocol strategy is Sliding Window Go-back-n

- $P1$: SWS: Infinite and RWS: Finite
- $P2$: SWS: Finite and RWS: Infinite

`Solution:` $P1$: If we assume, $F_0, \ldots, F_\infty$ are the frames with $\{0, \ldots, \infty\}$ are the sequence numbers, then this scheme does not work always. For example, during Iteration 1, the sender sends $F_0, \ldots, F_\infty$, the receiver acknowledges with ACK $\infty$, assuming it is delayed, and due to which the sender retransmits the previous set of infinite frames. The receiver is unaware of this scenario and accepts the next set thinking it is actually the next set of infinite frames. Supposing, we use $\{0, \ldots, \infty\} \cup \{-1, -2, \ldots, -\infty\}$, on receiving the first set of infinite frames, the receiver sends ACK $-1$ expecting $F_{-1}, F_{-2}, \ldots, F_{-\infty}$. Since there is no overlap between two windows, this modified scheme works fine. `Assumption:` It is assumed that 'infinite' in the question refers to the set $\{0, \ldots, \infty\}$. Otherwise, even the modified scheme faces the same issue as before.

$P2$: If the number of sequence numbers is at least 1+SWS, then this scheme always works fine. Although, RWS is infinite, it can see and hold a finite number of outstanding frames. For selective-reject, If the number of sequence numbers is at least twice the SWS, then this scheme always works fine.