**רשתות תקשורת מחשבים**

**תרגיל תיאורטי 1**

## **מגישים:**

## שם: אבי קצ'ולרו

## ת.ז. 203056585

## מייל: [avi.c33@gmail.com](mailto:avi.c33@gmail.com)

## שם: נעם גוטליב

## ת.ז. 201606951

## מייל: [noam.got@gmail.com](mailto:noam.got@gmail.com)

## שם: מתן סירי

## ת.ז. 304957673

## מייל: [matanse@gmail.com](mailto:matanse@gmail.com)

# שאלה 1

1. There is no 2 bits’ error that can be corrected. Consider the message is ABCD:

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

* If the errors are in , the error can be at
* If the errors are in so we have other options for errors that the parity will not be able to determine which one of them happened.
* In longer messages, it is even harder to determine what the error was.
* It can be formally proofed that the Hamming distance of 2d-parity fulfils so it can correct only less than errors: .

1. Yes. For example, consider the message :

|  |  |  |
| --- | --- | --- |
| 1 | 1 | 0 |
| 1 | 1 | 0 |
| 0 | 0 | 0 |

And after bit-flipping all the message we get :

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 0 | 0 |
| 0 | 0 | 0 |

1. The parity code seems to be correct, so we cannot detect this error.

Yes. For example, consider the message :

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 0 |

And after bit-flipping some of the message we get :

|  |  |  |  |
| --- | --- | --- | --- |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 |

The parity code seems to be correct, so we cannot detect this error

# שאלה 2

* Choose one red station-
* The probability that the chosen station succeeded transmitting the message-
* The probability that any other red station didn’t transmit-
* The probability that no green station succeeded to transmit-
* In total, the probability for one red station to transmit-
* The probability that one green station succeeded to transmit-
* The probability that no red station succeeded to transmit-
* In total, the probability for one green station to transmit- .
* In total, the probability that a frame will be sent successfully in a slot is
* Choose one red station-
* The probability that the chosen station succeeded transmitting the message-
* The probability that any other red station didn’t transmit-
* The probability that no green station succeeded to transmit in an odd or even slot-
* In total, we get

Pure Aloha is better in the case of multiple stations that want to transmit messages, because for every station, there is a good probability that it could transmit if it has a message waiting for transmission, without waiting to its “next slot”.

Slotted Aloha requires more complex architecture since the time-slots are discrete intervals (and not continues as in Pure Aloha). So, in many stations case, it would be harder per station to transmit a message.

# שאלה 3

Given that A and B choose different K's, the larger the difference – the bigger isolation between A and B operation times. So, we will prove that even for the smallest margin (for example KA=0 and KB=1) the re-transmissions do not collide.

|  |  |  |
| --- | --- | --- |
| Time [bit times] | A | B |
| 0 | Begins transmission | Begins transmission |
| 255 | Detects collision, stops transmission and sends J-bits jam signal\* | Detects collision, stops transmission and sends J-bits jam signal\* |
| 255+J | Finishes transmitting jam signal | Finishes transmitting jam signal, waits for 512 bit times |
| (255+J)+255=510+J | B's last bit arrives, detects an idle channel and listens for 96 bit times |
| (510+J)+96=606+J | Begins re-transmission |
| (255+J)+512=767+J | Listens for 96 bit times |
| (606+J)+255=861+J | A's first bit arrives when B is still in listening mode, resets the "listening clock" |

\* Remember J is usually 36-48 bits.

When A's message is fully received at B, B will detect an idle channel, wait 96 more bit times (listening mode) and then re-transmits its message to A successfully.

No collision! Great success!

# שאלה 4

Prop. delay = Cable length / Speed of signal = 250 / 250,000K = 1 us

Round trip time = 2 \* Prop. delay = 2 us

**Minimal frame size = Bandwidth \* Round trip time = 100M \* 2u = 200 bits**

# Link Layer Switches

i.

|  |  |  |
| --- | --- | --- |
| Switch | MAC address | interface |
| B1 | A | 2 |
| B2 | A | 1 |
| E | 2 |
| B3 | A | 2 |
| E | 1 |
| B4 | A | 2 |
| E | 3 |

ii.

Let us draw the ARP table after a message was sent from A to E in the new network configuration (the only change is marked in red color):

|  |  |  |
| --- | --- | --- |
| Switch | MAC address | interface |
| B1 | A | 2 |
| B2 | A | 1 |
| E | 2 |
| B3 | **A** | **4** |
| E | 1 |
| B4 | A | 2 |
| E | 3 |

Note: the notation **Bi:j** stands for interface j in switch Bi.

* **Message from D to A**

Let us write the message's path:

D 🡪 B4:1 🡪B4:2 🡪B3:1🡪B3:4🡪A

Hence, **this message gets to its destination**.

* **Message from C to A**

Let us write the message's path:

C 🡪 B1:3 🡪 B1:2 🡪 B2:3 🡪 B2:1 🡪 no host connected…

Hence, **this message does not get to its destination**. That happens because switch B2 has not learned yet about A's new position (in its table it is still connected via interface 1, when it is actually connected to it via interface 2).

# Bonus - CDMA

Let be the CDMA codes of users 1 and 2 correspondingly, where each code is a vector containing M bits (encoding 1 real bit). We also assume that the norm of each vector equals to M (i.e. the vector's components may be 1 or -1).

Let us define the inner-product that we will use for 2 arbitrary vectors :

( stands for the m-th componenet of )

We assume that the codes are orthonormal, hence:

* For every :
* For every :

When user k sends in time i some bit , it is encoded as a vector (remember that is a scalar). When both users send a bit together we add their vectors:

Hence, the following holds:

Transition (1) is derived from the inner-product definition. Transition (2) holds due to the linearity property of an inner product. Transition (3) holds due to the orthonormality property. □