Lab 2: Basic Network Programming

Duc Viet Le
CS536

September 27, 2016

Problem 1.

Because RTT is perfect, we assume that RTT = Time Out

1. The equation for "$Through\ Put$" as following

$$Through\ Put = \frac{Frame\ Size}{(1+p)\cdot(RTT)}$$

According to the lectures, the through put is equal the frame size divided by the round trip time (i.e. $throughput = (framesize)/RTT$). However, we take into consideration that the probability of failure is p and retransmission always succeeds. Let consider X to be number of time we need to send. We only need to consider till $X = 2$ because resending always succeeds. We have:

$$E(X) = 1 \cdot Pr(X = 1) + 2 \cdot Pr(X = 2) = (1 - p) + 2 \cdot p = (1 + p)$$

Thus, the expected total time is $(1+p)\cdot RTT$ and $throughtput = Frame\ Size/(1+p)\cdot(RTT)$

2. The equation for "$Through\ Put$" as following

$$Through\ Put = \frac{(1-p)\cdot(Frame\ Size)}{RTT}$$

Similarly, define X to be the number of time we need to send. Since resending does not always succeeds. It's not difficult that $Pr(X = k) = p^{(k-1)} \cdot (1 - p)$ (i.e. geometric distribution). Thus, the expected number of time needed to send is:

$$E(X) = \sum_{k=1}^{\infty} k \cdot Pr(x = k) = \sum_{k=1}^{\infty} k \cdot p^{k-1} \cdot (1 - p) = 1/(1 - p)$$

Therefore, the expected total time is $RTT/(1-p)$ and $throughput = (1-p)\cdot(Frame\ Size)/RTT$

Problem 2.

1. Let 4 orthogonal vectors be:

$$u_1 = (1, 2, 0, 0), u_2 = (-2, 1, 0, 0), u_3 = (0, 0, -3, 1), u_4 = (0, 0, 1, 3)$$

For if sender want to send $(1, 1, 1, 0)$ (i.e 1 for first user, 1 for second user ...). He needs to computes:

$$u = (1, 1, 1, -1) \cdot (u_1^T \ u_2^T \ u_3^T \ u_4^T) = (3, -1, -4, -2)$$

He then broadcasts $(3, -1, -4, -2)$. Each user extract his/her bit as following:

- User 1 computes: $(3, -1, -4, -2) \circ u_1 = 1 > 0$. Therefore, received bit is 1
- User 2 computes: $(3, -1, -4, -2) \circ u_2 = 5 > 0$. Therefore, received bit is 1
- User 3 computes: $(3, -1, -4, -2) \circ u_3 = 10 > 0$. Therefore, received bit is 1
- User 3 computes: $(3, -1, -4, -2) \circ u_4 = -10 < 0$. Therefore, received bit is 0

The reason this method works is because the orthogonal property cancels off other users' vectors.

2. What happens to decoding if the code vectors are not orthogonal (although still independent)?
Ans. If vectors they used are not orthogonal, they will not able to extract the hidden bit from the broadcast message because the dot product will not cancel other users' "ingredient" vectors. Thus, the soup vector is mixed up, and the bit extracted might be incorrect.

3. How is this related to an issue we encountered when sending bits using amplitude modulation (AM) of electromagnetic waves modeled as complex sinusoids? what were the two approaches used to address the problem?
Ans. The issue we encountered is Inter-Channel Interference (ICI). It's related because when signal spreading of two or more frequencies are overlapped; the amplitude are distorted . If $e^{-if_1 t}$ will not be able to cancel other $e^{-if_2 t}$ (i.e similar to when using non-orthogonal vectors in CDMA), the extracted bit will be incorrect.
Two approaches to prevent ICI is:

- Put carrier frequencies far apart
- Use sinunoids that are mutually orthogonal (i.e. OFDM).

Problem 3.

Ans. According to Wikipedia, the carrier frequencies used by AM radio stations are 535 to 1700 kHz with bandwidth about 10 KHz. However, Human auditory system is from 20 Hz to 20kHz. Therefore, with current AM radio bandwidth, we are missing 10kHz-20kHz; it explains why AM is more suitable for talk rather than music.
As communication system, FDMA and AM are similar because each carrier has its own channel. However, in FDMA, all users use same frequency channel but each user transmit at

a single frequency while in AM, each user may need their own frequency channel to transmit. If a law mandated that AM radio become digital by transmitting compressed audio data, then all AM receivers in car and home stereos would have to be replaced because AM receivers in car and home stereos are "analog computer" that uses voltage values to represent signal and will not understand digital signal.
Finally, about the audio quality, I think it will increase the audio quality the reason is that digital signal can be recovered after each hop if there is corrupted while analog signal need to reamplified many times. Therefore, I think digital signal is better than analog signal and will improve audio quality.

Programming Part

Problem 1.

I think interleaving will not be problem in this case because now each processes handle each client's request on serperate tcp socket directly.

Problem Bonus.

Using OS X
I tried to run client code on iMac (OS X El Capital) in Hick Undergraduate library. However, the client code does not work using `SendTo()` function. I changed it to `Write()` to write to socket, it works on IMac Machine (It doesn't work the same on linux machine). Here are 4 ping results received using IMac:

```
Request Sent ...
Server Response: terve
Time Elapsed: 1.580000 ms
Server's Port: 10000
Server's IP Address: 125.10.25.101
Request Sent ...
Server Response: terve
Time Elapsed: 1.314000 ms
Server's Port: 10000
Server's IP Address: 125.10.25.101
Request Sent ...
Server Response: terve
Time Elapsed: 1.803000 ms
Server's Port: 10000
Server's IP Address: 125.10.25.101
Server Response: terve
Time Elapsed: 1.203000 ms
Server's Port: 10000
Server's IP Address: 125.10.25.101
```