COSC 4P14 – Assignment 1 – 6005011 / 6276463

SAWYER FENWICK, DANIEL FIGUEROA, Brock University, Canada

ACM Reference Format:
Sawyer Fenwick, Daniel Figueroa. 2021. COSC 4P14 – Assignment 1 – 6005011 / 6276463. *ACM Comput. Surv.* 0, 0, Article 1 (October 2021), 11 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

1.1 Suppose users share a 8 Mbps link. Also suppose each user transmits continuously at 3 Mbps when transmitting, but each user transmits only 15 percent of the time. (Refer the discussion of statistical multiplexing).

1.1.1 When circuit switching is used, how many users can be supported?

Since circuit switched networks require that the resources needed along a path are reserved for each user for the duration of their session we would only be able to support 2 users ($2 \cdot 3\text{Mbps} = 6\text{Mbps} < 8\text{Mbps}$). 3 users would require a 9Mbps link (or higher).

1.1.2 For the remainder of this problem, suppose packet switching is used. Why will there be essentially no queuing delay before the link if two or fewer users transmit at the same time? Why will there be a queuing delay if three users transmit at the same time?

There would be essentially no queuing delay with 2 users transmitting at the same time because the link is capable of handling 8Mbps and 2 users only requires 6Mbps. With 3 users transmitting at the same time we would begin to see a queuing delay because the 3 users are trying to use more Mbps than the link can support. It can only support one user at 1/3 capacity. Since packet switched networks share bandwidth all users could experience a delay.

1.1.3 Find the probability that a given user is transmitting.

$\text{p} = 0.15$

1.1.4 Suppose now there are three users. Find the probability that at any given time, all three users are transmitting simultaneously. Find the fraction of time during which the queue grows.

$$p_3 = \binom{3}{3} 0.15^3 0.85^0 = 0.00375, \text{ fraction of time} = 0.00375$$

Author's address: Sawyer Fenwick, Daniel Figueroa, Brock University, 1812 Sir Isaac Brock Way, St. Catharines, ON, L2S 3A1, Canada.

2021. 0360-0300/2021/10-ART1 $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1.2 Assume that we know the bottleneck link along the path from the server to the client is the first link with rate Rs bits/sec. Suppose we send a pair of packets back to back from the server to the client, and there is no other traffic on this path. Assume each packet of size L bits, and both links have the same propagation delay dprop.

1.2.1 What is the packet inter-arrival time at the destination? That is, how much time elapses from when the last bit of the first packet arrives until the last bit of the second packet arrives?

Time for first packet to arrive at client $T_1 = \frac{L}{R_s} + \frac{L}{R_c} + d_{prop}$

Time for second packet to arrive at client $T_2 = \frac{L}{R_s} + \frac{L}{R_s} + \frac{L}{R_c} + d_{prop}$

Inter-arrival time at destination $= T_1 - T_2 = \frac{L}{R_s}$

1.2.2 Now assume that the second link is the bottleneck link (i.e., Rc<Rs). Is it possible that the second packet queues at the input queue of the second link? Explain. Now suppose that the server sends the second packet T seconds after sending the first packet. How large must T be to ensure no queuing before the second link? Explain.

It is possible that the second packet will queue at the input queue of the second link since $R_c < R_s$.

Time for first packet to arrive at client $T_1 = \frac{L}{R_s} + \frac{L}{R_c} + d_{prop}$

Time for the second packet to arrive at input queue $T_2 = \frac{L}{R_s} + \frac{L}{R_s} + d_{prop}$

Since $T_2 < T_1$ the second packet will arrive at the input queue.

To ensure that the packet does not experience delay at the input queue T must be:

$$\frac{L}{R_s} + \frac{L}{R_c} + d_{prop} < \frac{L}{R_s} + \frac{L}{R_s} + d_{prop}$$

$$\frac{L}{R_s} + \frac{L}{R_c} + d_{prop} + T \geq \frac{L}{R_s} + \frac{L}{R_s} + d_{prop}$$

$$T \geq \frac{L}{R_c} - \frac{L}{R_s}$$

2 APPLICATION LAYER

2.1 Consider the following string of ASCII characters that were captured by Wireshark when the browser sent an HTTP GET message (i.e., this is the actual content of an HTTP GET message). The characters <cr><lf> are carriage return and line-feed characters (that is, the italicized character string <cr> in the text below represents the single carriage-return character that was contained at that point in the HTTP header). Answer the following questions, indicating where in the HTTP GET message below you find the answer.

```
GET /1667/index.html HTTP/1.1<cr><lf>Host: imgs.xkcd.com
<cr><lf>User-Agent: Mozilla/5.0 (
Windows;U; Windows NT 5.1; en-US; rv:1.7.2) Gec
ko/20040804 Netscape/7.2 (ax) <cr><lf>Accept:ex
t/xml, application/xml, application/xhtml+xml, text
/html;q=0.9, text/plain;q=0.8, image/png,*/*;q=0.5
<cr><lf>Accept-Language: en-us, en;q=0.5<cr><lf>Accept-
Encoding: zip, deflate<cr><lf>Accept-Charset: ISO
-8859-1, utf-8;q=0.7,*;q=0.7<cr><lf>Keep-Alive: 300<cr>
<lf>Connection:keep-alive<cr><lf><cr><lf>
```

2.1.1 What is the URL of the document requested by the browser?

Host: imgs.xkcd.com + GET /1667/index.html = imgs.xkcd.com/1667/index.html

2.1.2 What version of HTTP is the browser running?

HTTP/1.1 = HTTP Version 1.1

2.1.3 Does the browser request a non-persistent or a persistent connection?

Connection:keep-alive therefore the browser is requesting persistent connection

2.1.4 What is the IP address of the host on which the browser is running?

This information is not contained in the HTPP GET message.

2.1.5 What type of browser initiates this message? Why is the browser type needed in an HTTP request message?

User-Agent: Mozilla/5.0 = Mozilla/5.0. The browser type is needed in an HTTP request because the server needs to know what type of object to send back to that type of browser.

2.2 Suppose within your Web browser you click on a link to obtain a Web page. The IP address for the associated URL is not cached in your local host, so a DNS lookup is necessary to obtain the IP address. Suppose that n DNS servers are visited before your host receives the IP address from DNS; the successive visits incur an RTT of RTT1,. . .,RTTn. Further suppose that the Web page associated with the link contains exactly one object, consisting of a small amount of HTML text. Let RTT 0 denote the RTT between the local host and the server containing the object.

2.2.1 Assuming zero transmission time of the object, how much time elapses from when the client clicks on the link until the client receives the object?

RTT_0 to connect to the server, RTT_0 to retrieve the object, RTT_1+...+RTT_n to find the IP Address in n DNS servers = $2RTT_0 + RTT_1 + RTT_2 + ... + RTT_n$

2.2.2 Suppose the HTML file references eight very small objects on the same server. Neglecting transmission times, how much time elapses with Non-persistent HTTP with no parallel TCP connections?

Non-persistent HTTP response time = $2RTT_0$ + file transmission time
File transmission time = $2 \cdot 8RTT_0$ = $16RTT_0$
= $2RRT_0$ +$16RTT_0$ + DNS Time = $18RTT_0 + RTT_1$ +...+RTT_n

2.2.3 Suppose the HTML file references eight very small objects on the same server. Neglecting transmission times, how much time elapses with Non-persistent HTTP with the browser configured for 5 parallel connections?

$2RTT_0$ for the response time of non-persistent HTTP, $2RTT_0$ to retrieve the first 5 objects given the 5 parallel TCP connections, an additional $2RTT_0$ for the remaining 3 objects to be retrieved through the 5 possible parallel TCP connections culminating in $6RTT_0$ appended by the $RTT_1 + RTT_2$ +...+ RTT_n to find the IP Address in n DNS servers.

The total RTT is: $6RTT_0 + RTT_1 + RTT_2 + ... + RTT_n$

2.2.4 Suppose the HTML file references eight very small objects on the same server. Neglecting transmission times, how much time elapses with Persistent HTTP?

. Persistent HTTP: $2RTT_0$ for inital connection and request then 1 RTT per object
$2RTT_0 + 8RTT_0$ = $10RTT_0$ plus DNS time = $10RTT_0 + RTT_1$+...+RTT_n

2.3 Consider distributing a file of F=15 Gbits to N peers. The server has an upload rate of us=30 Mbps, and each peer has a download rate of di=2 Mbps and an upload rate of u. For N=10, 100, and 1,000 and u=300 Kbps, 700 Kbps, and 2 Mbps, prepare a chart giving the minimum distribution time for each of the combinations of N and u for both client-server distribution and P2P distribution.

CLIENT-SERVER: $D_{c-s} \geq \max\{NF/u_s, F/d_{min}\}$

N = 10
$D_{c-s} \geq \max\{10 \cdot 15360/30, 15360/2\} \geq \max\{5120, 7680\} = 7680$
N = 100
$D_{c-s} \geq \max\{100 \cdot 15360/30, 15360/2\} \geq \max\{51200, 7680\} = 51200$
N = 1000
$D_{c-s} \geq \max\{1000 \cdot 15360/30, 15360/2\} \geq \max\{512000, 7680\} = 512000$

P2P: $D_{P2P} \geq \max\{F/u_s, F/d_{min}, NF/u_s\ \Sigma u_i\}$

u = 300kbps = 300/1024Mbps, N = 10, 100, 1000
$D_{P2P} \geq \max\{15360/30, 15360/2, 10{\cdot}15360/30{+}10{\cdot}300/1024\} = \max\{512, 7680, 4665\} = 7680$
$D_{P2P} \geq \max\{15360/30, 15360/2, 100 \cdot 15360/30{+}100 \cdot 300/1024\} = \max\{512, 7680, 25904\} = 25904$
$D_{P2P} \geq \max\{15360/30, 15360/2, 1000{\cdot}15360/30{+}1000{\cdot}300/1024\} = \max\{512, 7680, 47559\} = 47559$

u = 700kbps = 700/1024Mbps, N = 10, 100, 1000
$D_{P2P} \geq \max\{15360/30, 15360/2, 10{\cdot}15360/30{+}10{\cdot}700/1024\} = \max\{512, 7680, 4170\} = 7680$
$D_{P2P} \geq \max\{15360/30, 15360/2, 100 \cdot 15360/30{+}100 \cdot 700/1024\} = \max\{512, 7680, 15616\} = 15616$
$D_{P2P} \geq \max\{15360/30, 15360/2, 1000{\cdot}15360/30{+}1000{\cdot}700/1024\} = \max\{512, 7680, 21525\} = 21525$

u = 2Mbps, N = 10, 100, 1000
$D_{P2P} \geq \max\{15360/30, 15360/2, 10 \cdot 15360/30{+}10 \cdot 2\} = \max\{512, 7680, 3072\} = 7680$
$D_{P2P} \geq \max\{15360/30, 15360/2, 100 \cdot 15360/30{+}100 \cdot 2\} = \max\{512, 7680, 6678\} = 7680$
$D_{P2P} \geq \max\{15360/30, 15360/2, 1000 \cdot 15360/30{+}1000 \cdot 2\} = \max\{512, 7680, 7567\} = 7680$

	N			
		10	100	1000
u	300kbps	7680	51200	512000
	700kbps	7680	51200	51200
	2Mbps	7680	51200	51200

Fig. 1. CLIENT-SERVER

	N			
		10	100	1000
u	300kbps	7680	25904	47559
	700kbps	7680	15616	21525
	2Mbps	7680	7680	7680

Fig. 2. P2P

3 TRANSPORT LAYER

3.1 Suppose Client A initiates a Telnet session with Server S. At about the same time, Client also initiates a Telnet session with Server S. Provide possible source and destination port numbers for the following situations

The port for client A will be 1024.
The port for client B will be 1025.
The default port for Telnet is 23.

3.1.1 The segments sent from A to S

.

Source Port	Destination Port
1024	23

3.1.2 The segments sent from B to S

.

Source Port	Destination Port
1025	23

3.1.3 The segments sent from S to A

.

Source Port	Destination Port
23	1024

3.1.4 The segments sent from S to B

.

Source Port	Destination Port
23	1025

3.1.5 If A and B are different hosts, is it possible that the source port number in the segments from A to S is the same as that from B to S?

. Yes it is entirely possible that the source port number of both segments can be the same. This is because different hosts can have the same port numbers yet remain distinct due to the fact the two segments have differing IP addresses.

3.1.6 How about if they are the same host?

If they are the same host, this would mean that the IP addresses for A and B would be the same in addition to their port numbers being the same. This is not possible because a host will not provide duplicate port numbers as it will only assign ports that are free and not the same occupied one.

3.2 UDP and TCP use 1s complement for their checksums

3.2.1 Suppose you have the following 2 bytes: 01011100 and 01100101. What is the 1s complement of the sum of these 2 bytes?

$$\begin{array}{r} 0101\ 1100 \\ 0110\ 0101 \\ \hline 1100\ 0001 \end{array} \longrightarrow \text{1's complement: } 0011\ 1110$$

3.2.2 Suppose you have the following 2 bytes: 11011010 and 01100101. What is the 1s complement of the sum of these 2 bytes?

$$\begin{array}{r} 1101\ 1010 \\ 0110\ 0101 \\ \hline 0011\ 1111 \\ 0000\ 0001 \\ \hline 0100\ 0000 \end{array} \longrightarrow \text{1's complement: } 1011\ 1111$$

3.2.3 For the bytes in part (a), give an example where one bit is flipped in each of the 2 bytes and yet the 1s complement doesn't change.

01011100 $\longrightarrow$ 01011101 and 01100101 $\longrightarrow$ 01100100

3.3 Compare GBN, SR, and TCP (no delayed ACK). Assume that the timeout values for all three protocols are 2 sufficiently long such that 5 consecutive data segments and their corresponding ACKs can be received (if not lost in the channel) by the receiving host (Host B) and the sending host (Host A) respectively. Suppose Host A sends 5 data segments to Host B, and the 2nd segment (sent from A) is lost. In the end, all 5 data segments have been correctly received by Host B

3.3.1 How many segments has Host A sent in total and how many ACKs has Host B sent in total? What are their sequence numbers? Answer this question for all three protocols.

GBN
A sends 9 segments. They are (in order): 1,2,3,4,5,2,3,4,5
B sends 8 acknowledgements. They are (in order): 1,1,1,1,2,3,4,5

SR
A sends 6 segments. They are (in order): 1,2,3,4,5,2
B sends 5 acknowledgements. They are (in order): 1,3,4,5,2

TCP
A sends 6 segments. They are (in order): 1,2,3,4,5,2
B sends 5 acknowledgements. They are (in order): 2,2,2,2,6

3.3.2 If the timeout values for all three protocol are much longer than 5 RTT, then which protocol successfully delivers all five data segments in shortest time interval?

TCP would be fastest since it uses fast retransmit it resends the lost packet after 3 duplicate ACKs while the other two wait.

3.4 Consider that only a single TCP (Reno) connection uses one 10Mbps link which does not buffer any data. Suppose that this link is the only congested link between the sending and receiving hosts. Assume that the TCP sender has a huge file to send to the receiver, and the receiver's receive buffer is much larger than the congestion window. We also make the following assumptions: each TCP segment size is 1,500 bytes; the two-way propagation delay of this connection is 150 msec; and this TCP connection is always in congestion avoidance phase, that is, ignore slow start.

3.4.1 What is the maximum window size (in segments) that this TCP connection can achieve?

The maximum window size can be calculated via the formula: $W = \frac{RTT}{\frac{MSS}{Bandwidth}}$

W represents the window size, RTT represents Round Trip Time and MSS represents Maximum Segment Size.

Unto which, W is unknown and what we are searching for, RTT is 150 milliseconds, MSS is 1500 bytes and bandwidth is 10 megabits per second.
The formula can be rearranged as: $W = \frac{RTT \cdot Bandwidth}{MSS}$

$$W = \frac{150msec \cdot 10Mbps}{1500bytes}$$

All the units are to be converted to kilobytes. Thus 10Mbps becomes 1250 Kilobytes per second. 1500 bytes become 1.5 kilobytes. Additionally, 150 milliseconds is 0.15 seconds.

$$W = \frac{0.15seconds \cdot 1250Kbps}{1.5kilobytes}$$

$$W = \frac{187.5kb}{1.5kb}$$

$$W = 125$$

Thus, the maximum window size in segments that the TCP connection can achieve is 125 segments.

3.4.2 What is the average window size (in segments) and average throughput (in bps) of this TCP connection?

The average window size of this TCP connection is $\frac{3}{4} \cdot W$ since the window size fluctuates between $\frac{W}{2}$ and W generally so the midpoint is used as the average.
Thus: $\frac{3}{4} \cdot W = 93.75$

Average Throughput is calculated by the product of $AvgWindowSize \cdot \frac{MSS}{RTT}$

Regarding conversion, MSS is 1500 bytes which becomes 12000 bits. RTT us 150 milliseconds which becomes 0.15 seconds.

$$AvgWindowSize \cdot \frac{MSS}{RTT} = 93.75 \cdot \frac{12000}{0.15} = 7500000bps$$

3.4.3 How long would it take for this TCP connection to reach its maximum window again after recovering from a packet loss?

The formula for calculating this is as follows: $\frac{W}{2} \cdot RTT$

$\frac{W}{2} \cdot RTT = \frac{93.75}{2} \cdot 0.15s = 7.03125$ seconds to reach maximum window again after recovering from a packet loss.

4 PROJECT - PART I

4.1 Undergraduate Students must propose, design, and implement a network application. The following topics are suggestions for students to pick: ▪ Network Game; ▪ Data Streaming service; ▪ Network File System; ▪ Real-time data analysis; ▪ Distributed computing system. For all projects, there are few objectives that must be addressed:

4.1.1 For these applications, the students must specify and detail the paradigm, such as Client/server and P2P, they follow for their designs.

The paradigm is a surprisingly simple one with the server maintaining a connection between 2 hosts and sending data back and forth with those two. More precisely, host A connects to the server and host B connects to the server but the hosts themselves are not connected to the other as the server intends to act as the courier of data for each endpoint with respect to the game of Battleship.

4.1.2 The students must focus on the network aspects, but they need to design and implement the bare minimum in their end points. For instance, if implementing a Network Game, the end points should run a game that coherently justifies the use of the network.

Given that the Network game in question is that of Battleship, the end points for network are to be the desktops/laptops of the players for the game as those systems will interact back and forth with the server. That server is additionally an endpoint as well due to its nature of communication back and forth with multiple computers as well. These endpoints are needed as the players have their own boards and said boards need to be updated with the actions of the opposing player. The server is then used to receive the information of the players and send it to the other and facilitate a turn based progression of the game. The server is additionally used to pair players together beyond just the first two as this game can be played by multiple pairs of players hence the need for the aforementioned endpoints for the purposes of our implementation.

4.1.3 The applications need to follow a well-defined protocol. The students need to specify an application protocol that must be followed so that the end points (peers or clients and servers) can effectively communicated.

The application protocol the Battleship game will adhere to is that of DNS. Utilized due to the fact it is a client/server network communication protocol. The clients which are effectively the players send requests to the server which will be that of actions and the server responds accordingly to the clients whether they missed, hit, got hit etc.

4.1.4 The students must justify the use of a selected Transport Layer protocol for the supporting the communication among the application parts. This decision will necessarily reflect on the implementation: Socket types used in their codes.

The transport protocol utilized for this project was that of TCP. This transport protocol was used due to the desire to ensure the results of the moves of each player is made known and not risk it being lost. Additionally, the socket types used in our code was that of TCP sockets for the same reason of reliability.

4.2 How to Use

The implementation of our Battleship game was done in Java utilizing JDK 17 to be precise. We have provided a project file to be used with Intellij, and the raw client/server Java files, as well as sample output from a game we played.

4.2.1 Setting up the Server

. To set up a game of Battleship, the server will need to be set up initially. This can be done by running the file "BattleshipServer.java" either through the command "java BattleshipServer.java" while in the same directory in command line or simply executing the code in an IDE.
The server effectively acts on a first come, first serve basis and as such **the server will wait for the first two clients to connect** and only then will a game of Battleship begin as it requires exactly two players. Were a third client to connect to the server, that client will await until a fourth client to connect to initiate a game and this applies for every additional pair of clients until the server reaches a certain threshold.

4.2.2 Setting up the Clients

. At least two clients will need to connect to the server to start a game of Battleship proper and therefore, the "BattleshipClient" class must be executed. This can be done by entering the command "java BattleshipClient.java" while in the same directory within command line. Since another client is needed for a game to be underway, another client must be instantiated through repeating the above steps in a separate command line window. Once at least two clients are connected and set up, the game will begin.

4.2.3 Game Startup

. Once the game begins, the clients will be displayed with a list of possible commands for the game of Battleship that the client can enter. The valid commands are strictly numbers and are as follows:

- Entering **1** will denote the client as ready to play a game of Battleship. If they did not set up a board prior, the client will be prompted to do so. Otherwise, they will await until the other client has also entered 1 and set up their board. After both players are ready, the game will begin.
- Entering **2** will direct the player to an interface that will assist in placing their ships on the board through the entering of strings such as "a0-a4" or "f2-f3" (This is explained in more detail in 4.2.4). The purpose of this is to allow a client to prepare their board while they await another user to connect. If this is completed, when the client enters a game, they will not again be prompted to set their ships.
- Entering **3** will display strings to the client consisting of all the pertinent rules of the game, game progression and the conditions for victory in Battleship.
- Entering **0** will denote that the client desires to quit and terminate the client program in turn disconnecting themselves from the server.

4.2.4 Setting Ships on the Board

. The setup of the ships on the board are done through the entering of the start and end tiles of where the client desires to put the ship. Thus the possibles rows are that of **A to J** and possible columns are that of **0 to 9** and as a result, entering combinations of letters outside the range of A to J and numbers outside the range of 0 to 9 will result in invalid input. Additionally the entering of anything that isn't a string consisting of a character in the 0 index and a number in the 1 index will result in invalid input.

Valid input consists of entering a range of coordinates that match the length of the ship you wish to place. For example, a Carrier ship is 5 tiles long, valid input in this case would be that of **a0-a4** which would you mean the range of which your ship is to be placed starts at the tile a0 and ends at a4. It is valid as the range of the coordinates matches the length of the ship (The coordinates can also be entered in reversed provided the range is valid).

Were the range lesser (a0-a3) or greater (a0-a5), then this will be not be accepted as valid input. Any and all invalid input will result in a prompt to the user to enter valid input and this process will repeat until valid input is finally entered.

The program will display the size of the ships prior to placing them so their is no confusion as to the size of each piece, a ship of size 4 must have coordinates with a range of 4 such as (b2-b5). A ship of size 3 must have coordinates with a range of 3 such as (d1-d3) and so on.

4.2.5 Gameplay and Conditions for Victory

. Battleship is a strategy game where each player takes turns stating coordinates they wish to hit in an effort to destroy all the ships of the opposing player (In this case it is the other client in the game) in order to achieve victory. The placements of ships is not known to the opposing player and as a result, players will need to guess and use their intuition to fire on certain tiles in order to sink all the ships of the opposing player.

When a player is prompted to make their move, they will enter the tile they wish to hit. Valid input of a tile to hit consists of a string of 2 length consisting of a character and a number in which the character and number are within the range of the board (**A to J** for characters and **0 to 9** for numbers). Therefore, an example of valid input is that of **b6** and once this is entered, the game will respond with the result of your proposed hit whether it connected (i.e. hit) or missed a ship. Once a player has hit all parts of a ship, they will have sunken it and the game will notify the player of which ship they have sunken upon doing so. The player is also to be told when their own ships are sunk in a similar manner.

Once a player has won the game, the game is then over. Both clients are then disconnected from the server. However, the server remains active and awaits the connection of more clients. Once active, the server remains online until you terminate it.