Trevor Grabham

Kevin Gandham

Question 1

a) What is the average percentage of the time each line is used?

Each channel supports up to 2.5MB per second

In [1]:
```
maximum_rate = 2.5
```

Each channel is on average only sending 0.5MB per second

In [2]:
```
average_rate = 0.5
```

In [3]:
```
average_percent = (average_rate/maximum_rate)*100
print('The average percentage of time that each line is in use is',average_percent,'%')
```

```
The average percentage of time that each line is in use is 20.0 %
```

b) What fraction of the information being sent is overhead?

Each frame sends 45 bytes of header

In [4]:
```
header_size = 45
```

Each frame sends over 1200 bytes total

In [5]:
```
total_size = 1200
```

In [6]:
```
fraction_header = header_size/total_size
print('The fraction of overhead data being sent is', fraction_header)
```

```
The fraction of overhead data being sent is 0.0375
```

c) What is the average input bit rate summed over all signals?

There are 80 channels

In [7]:
```
number_channels = 80
```

The average input bit rate is 0.5MB per second on each channel

In [8]:
```
bit_rate_per_channel = 0.5
```

In [9]:
```
average_bit_rate_total = number_channels * bit_rate_per_channel
print('The average bit rate over all of the channels is',average_bit_rate_total,'MB/s')
```

```
The average bit rate over all of the channels is 40.0 MB/s
```

d) What is the average data transmission rate summed over all signals? (data only)

We know that the data makes up 1155 of the 1200 bytes being send over each frame

In [10]:
```
data_size = 1155
total_size = 1200
```

We therefore know the fraction of the data sent

In [11]:
```
fraction_data = data_size/total_size
print('The fraction of data sent is',fraction_data)
```

```
The fraction of data sent is 0.9625
```

We know what the overall bit rate of all of the channels combined is
and we know what percentage of the bits make up the data.
Therefore, we are able to tell what the data transmission rate is over all the signals

In [12]:
```
average_data_rate_total = average_bit_rate_total*fraction_data
print('The average data transmission rate summed over all signals is',average_data_rate_total,'MB/s')
```

```
The average data transmission rate summed over all signals is 38.5 MB/s
```

e) What percentage of the line's capacity will be utilized?

It depends on whether we are talking about the amount utilized for bit transfer of data transfer

For either of them, the lines total capacity is the same, 50MB/s

In [13]:
```
total_capacity = 50
```

For the total bit transfer, we will use the input bit rate over all of the lines.
We will then be able to calculate the average percent of the lines usage capacity

In [14]:
```
percent_line_capacity = (average_bit_rate_total/total_capacity)*100
print('The average percent of the lines capacity being utilized for bit transfer is',percent_line_capacity,
'%')
```

```
The average percent of the lines capacity being utilized for bit transfer is 80.0 %
```

However, if we are only concerned with the amount that is being used for the data transfer,
we will instead want to use the data transmission rate instead

And we already know what the total data transmission rate is for the entire line.
Therefore, we are able to calculate the average percentage of the lines capacity being utilized

In [15]:
```
percent_line_capacity = (average_data_rate_total/total_capacity)*100
print('The average percent of the lines capacity being utilized for data transfer is',percent_line_capacity,
'%')
```

```
The average percent of the lines capacity being utilized for data transfer is 77.0 %
```

Question 2

a) How many users would the line support assuming each user has a 8 Mbps connection and uses synchronous TDM to access the line? Assume that only 95% of the capacity of the line may be used for user connections.

We know that the total capacity of the line is 180 Mbps

In [16]:
```
total_capacity = 180
```

We also know that each user needs a 8Mbps connection and that the line is divided using synchronous TDM,
so we cannot have the total bandwidth exceed the lines total capacity

In [17]:
```
user_capacity = 8
```

Finally, we know that the line can only use 95% of its total capacity for user connections

In [18]:
```
usable_capacity = total_capacity * 0.95
```

To get the number of users possible, we can divide the usable capacity by the capacity needed per user.
If this number is not an integer, then we will round down as we cannot have partial users and we cannot exceed the usable capacity

In [19]:
```
max_users = usable_capacity//user_capacity                    # integer division to round down
print('The maximum number of users that can be supported by the line is',max_users)
```

```
The maximum number of users that can be supported by the line is 21.0
```

b) Suppose that each user only used their 8 Mbps line 28% of the time, 95% of the capacity of the may be used for user connections and synchronous TDM is being used. How many users can the line support for these assumptions?

Since we are using synchronous TDM, even when the user is not sending any data, the bandwidth that is being reserved for the user cannot be used by other users, so we would still have the same number of users being supported, 21

c) Suppose that each user only used their 8 Mbps line 28% of the time, 92% of the capacity of the may be used for user connections and statistical TDM is being used. How many users can the line support for these assumptions?

Under statistical TDM, we will need to calculate the average amount of input data that each user will generate

In [20]:
```
average_user_input = 8*0.28                # 28% of the time, using 8Mbps
```

We also need to update the usable capacity of the line, as we are only able to use 92% now

In [21]:
```
usable_capacity = total_capacity * 0.92
```

We now want to pick a number of users, such that the average input rate summed over all of the users will be less than the usable capacity of the line. If the number is not an integer, then we will round down

In [22]:
```
max_users  = usable_capacity//average_user_input
print('The maximum number of users that the line can support under these assumptions is', max_users)
```

```
The maximum number of users that the line can support under these assumptions is 73.0
```

The number of users is much larger because on average, the total input rate is only 2.24 Mbps, which is almost four times smaller than the average input for the synchronous method

d) Suppose there are 80 users, using 3 Mbps connections at the same time. Each user uses their connection 41% of the time. Find the probability that at any given time, exactly 18 users are transmitting simultaneously. (Hint: use the binomial distribution).

Using the binomial distribution, we will have the following varaibles

In [23]:
```
p = 0.41
n = 80
k = 18
```

Where p is the probability of each of the users using their line at any given time,
n is the number of users
k is the number of users using the line that we are the same time

The formula for the binomial distribution follows

In [24]:
```
def factorial(n):                          # helper function for the binomial function
    prod = 1
    for i in range(n):
        prod *= (i+1)
    return prod

def binomial(n,p,k):
    n_choose_k = factorial(n)/(factorial(k) * factorial(n-k))
    return (n_choose_k * p**k * (1-p)**(n-k))
```

Therefore, we can now use the binomial distribution to calculate the probability that exactly 18 users are using their lines simultaneously

In [25]:
```
probability = binomial(n,p,k)
print('The probability that exactly 18 lines are in use simultaneously is', probability, 'or', probability*100,'%')
```

```
The probability that exactly 18 lines are in use simultaneously is 0.00023627864472088427 or 0.023627864472088426 %
```

e) Assume that the transmission line in the problem is used within a packet switched network. Would you use Synchronous or Statistical TDM? Why?

For a packet switched network, I would probably use Statistical TDM, because of the variable input rate of the packet
switched network, and because we are working within a packet switched network, we also know that there will be delays
expected along the line already, so this allows us to maximize our throughput, as we don't have to worry as much about
any delays that are introduced into the system

f) Assume that the transmission line in the problem is used within a connection oriented circuit switched network. Would you use Synchronous or Statistical TDM? Why?

For a connection oriented circuit switched network, I would probably use Synchronous TDM. This is because with a
connection oriented circuit switched network, the main advantage is that we will have very little signal delay. If we
were to use Statistical TDM, it would introduce the possibility of a queueing delay into the system, but with a
Synchronous system, we wouldn't have to worry about that. This of course, comes at the cost of the overall throughput
of the system, but with a connection oriented circuit switched network, we were already capped at a lower throughput
anyways

Question 3

a) What does DNS stand for?

DNS stands for Domain Name System

b) State four basic uses for DNS?

1. IP address lookup - looking up IP addresses, given the name of a host
2. Host name lookup - looking up the host name, given a particular IP address
3. Name aliasing - allowing multiple host names to be routed to the same IP address. i.e. google.com and www.google.com
There is also the special case when we can use the same host name for mail and web servers
4. Load distribution - returning a set of IP addresses for larger web servers with a lot of traffic. i.e. google

c) i) After the query for collie.dog.pet.bc.ca. what DNS name server records are stored in the cache (ignore information about root servers)?

If we are ignoring information about the root servers, then after querying for collie.dog.pet.bc.ca, we will have had
to have sent and cached a query to bc.ca, pet.bc.ca, dog.pet.bc.ca, and finally, collie.dog.pet.bc.ca. So we will have
four total records stored on our local DNS server.

ii) Now consider asking for the IP address of siamese.cat.pet.bc.ca. Which DNS server would the first query be sent to? Why?

The first query for siamese.cat.pet.bc.ca will be sent to pet.bc.ca, as we will send the request to the longest string,
matching the query that we are looking for

iii) Now consider asking for the IP address of siamese.cat.pet.ca. When making a query to .pet.bc.ca. what would be the DNS record being requested in the query?

The record being returned on the query to pet.bc.ca would be the record for cat.pet.bc.ca, as this is the most information that is stored on the pet.bc.ca server

iv) Draw an annotated diagram to help you explain how the query for the address of siamese.cat.pet.bc.ca. is executed. Show the Resolver, the local DNS server and all DNS servers queried in your diagram. Show all information travelling between the resolver and the DNS servers in your diagram. Be sure to indicate on your diagram what information is requested and returned in each query. State any assumptions you make. Assume the local DNS server makes iterative queries

As shown in the following diagram, the flow of information would go like such:

1. The resolver sends the query message to the local DNS server for siamese.cat.pet.bc.ca
2. The local DNS server looks and sees that is has a match for the string in its cache and chooses the longest match,
   that being pet.bc.ca and sends the query message to the authoritative DNS server for pet.bc.ca
3. The authoratative server for pet.bc.ca sends back the IP address of the authoritative DNS server for cat.pet.bc.ca
   and this IP address is cached for further use by the local DNS server
4. The local DNS server now sends the query request to the IP address sent back, namely cat.pet.bc.ca
5. cat.pet.bc.ca is directly authoritative over the siamese.cat.pet.bc.ca IP address, and so it sends back this IP
   address to the local DNS server and it is once again, cached by the local DNS server
6. Finally, the local DNS server has recieved the IP address for siamese.cat.pet.bc.ca and it sends this IP address
   back to the resolver, and the querying is complete