CS 350: Homework #1

Due 3PM, Tuesday, February 2

Kristel Tan (ktan@bu.edu)

January 24, 2016

Question 1

Part (a)

$T_{total} = 2 + 8 + 2 + 6 + 1 = 19ms$
$P_{CPU} = T_{CPU} \ / \ T_{total} = (2 + 2 + 1) \ / \ 19 = 5/19 \approx 26\%$
$P_{disk} = T_{disk} \ / \ T_{total} = 8/19 \approx 42\%$
$P_{network} = T_{network} \ / \ T_{total} = 6/19 \approx 32\%$

Part (b)

Capacity of web server $= 1000ms \ / \ 19ms \approx 53$ requests per sec

Part (c)

To solve this problem, we can simulate the utilizations of each process throughout the system until we identify a pattern...

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
C	A	A	B	B							A	A							A	B	B	A	A
D			A	A	A	A	A	A	A	A	B	B	B	B	B	B	B	B					
N													A	A	A	A	A	A				B	B

	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46
C					B	B	B		A	A							A	B	B	A	A		
D	A	A	A	A	A	A	A	A	B	B	B	B	B	B	B	B						A	A
N	B	B	B	B							A	A	A	A	A	A				B	B	B	B

	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69
C			B	B	B		A	A							A	B	B	A	A				
D	A	A	A	A	A	A	B	B	B	B	B	B	B	B						A	A	A	A
N	B	B							A	A	A	A	A	A				B	B	B	B	B	B

The first pattern is evident between 19 seconds and 40 seconds. Hence, $T_{total} = 21ms$. The utilizations of the CPU, disk, and network are as follows:

$P_{CPU} = T_{CPU}$ / T_{total} = 10/21 ≈ 48%
$P_{disk} = T_{disk}$ / T_{total} = 16/21 ≈ 76%
$P_{network} = T_{network}$ / T_{total} = 12/21 = 57%

Part (d)

Capacity of web server = # processes completed in 21ms / 21ms = 1/21 ≈ 0.05

Part (e)

Similar to when N = 2, we can solve this problem by simulating the utilizations of each process throughout the system until we identify a pattern...

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
C	A	A	B	B	C	C					A	A							A	B	B	A	A
D			A	A	A	A	A	A	A	A	B	B	B	B	B	B	B	B	C	C	C	C	C
N													A	A	A	A	A	A				B	B

	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46
C					B	C	C	B	B			A	A	C	C	C				A	B	B	A
D	C	C	C	A	A	A	A	A	A	A	A	B	B	B	B	B	B	B	B	C	C	C	C
N	B	B	B	B				C	C	C	C	C	C	A	A	A	A	A	A				B

	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69
C	A					B	C	C	B	B			A	A	C	C	C				A	B	B
D	C	C	C	C	A	A	A	A	A	A	A	A	B	B	B	B	B	B	B	B	C	C	C
N	B	B	B	B	B				C	C	C	C	C	C	A	A	A	A	A	A			

	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92
C	A	A					B	C	C	B	B			A	A	C						A	B
D	C	C	C	C	C	A	A	A	A	A	A	A	A	B	B	B	B	B	B	B	B		
N	B	B	B	B	B	B				C	C	C	C	C	C	A	A	A	A	A	A		

The first pattern is evident between 19 seconds and 43 seconds. Hence, T_{total} = 24ms. The utilizations of the CPU, disk, and network are as follows:

$P_{CPU} = T_{CPU}$ / T_{total} = 15/24 ≈ 63%

$P_{disk} = T_{disk} / T_{total} = 24/24 = 100\%$
$P_{network} = T_{network} / T_{total} = 18/24 = 75\%$

Capacity of web server = # process completed in 24ms / 24ms = 3/24 ≈ 0.13 requests per ms

Part (f)

The maximum capacity of the web server is reached when the utilization of either the CPU, disk, or network reaches 100%. As seen in part (e) of this question, when using the "first in first out" method, the disk already reaches 100% utilization at $N = 3$ processes.

Therefore, the maximum capacity of the web server is 3/24 or ≈ 0.13 requests per ms.

Part (g)

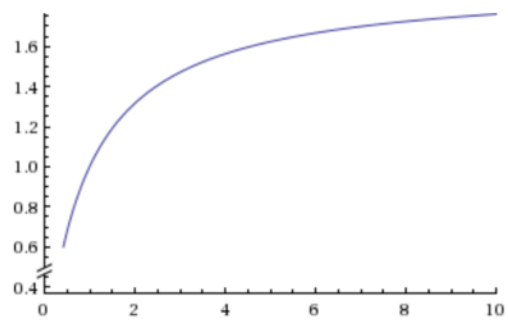
The bottleneck that limits the capacity of the web server is the disk.

Question 2

Part (a)

For MPL = 1, we know that the CPU's utilization is 48%. We can apply Amdahl's Law to this problem, where p = percentage of execution time of the CPU and r = speedup in latency of the execution of the enhancement.

$$speedup = \frac{1}{(1-p)+(p/s)} = \frac{1}{(1-0.48)+0.48/r} = \frac{1}{0.52+0.48/r}$$



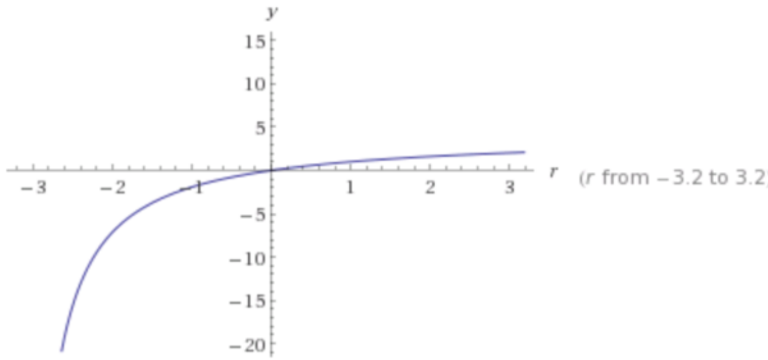
Part (b)

The theoretical limit for the speedup of the web server, assuming that the CPU can be infinitely fast (*i.e.* $r \longrightarrow \infty$) is 2.

Part (c)

For MPL = 1, we know that the disk's utilization is 76%. We can apply Amdahl's Law to this problem, where p = percentage of execution time of the disk and r = speedup in latency of the execution of the enhancement.

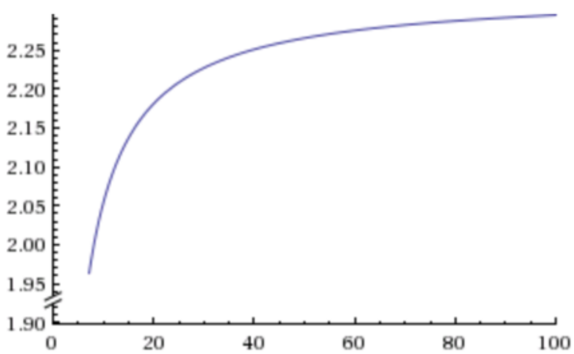
$$speedup = \frac{1}{(1-p)+(p/s)} = \frac{1}{(1-0.76)+0.76/r} = \frac{1}{0.24+0.76/r}$$



The theoretical limit for the speedup of the web server, assuming that the disk can be infinitely fast (*i.e.* $r \longrightarrow \infty$) is 5.

For MPL = 1, we know that the network's utilization is 57%. We can apply Amdahl's Law to this problem, where p = percentage of execution time of the network and r = speedup in latency of the execution of the enhancement.

$$speedup = \frac{1}{(1-p)+(p/s)} = \frac{1}{(1-0.57)+0.57/r} = \frac{1}{0.43+0.57/r}$$



The theoretical limit for the speedup of the web server, assuming that the network can be infinitely fast (*i.e.* $r \longrightarrow \infty$) is 3.

Part (d)

One way we can equalize the utilization of all the resources is by setting it terms of the least utilized resource. In the case that MPL = 1, that resource would be the CPU, which is at 10/21. Then, the following minimum speedup calculations will hold:

Minimum speedup for disk: $\frac{16}{21r} = \frac{10}{21} \rightarrow r = \frac{8}{5}\ or\ 1.6$

Minimum speedup for network: $\frac{12}{21r} = \frac{10}{21} \rightarrow r = \frac{6}{5}\ or\ 1.2$

Part (e)

$$speedup = \frac{old\,time}{new\,time}$$

Based off of the ratios we got in part (f) and the formula we know for speedup, we can look for a new pattern again, updating step 2 to have the process wait 5 seconds for the disk instead of 8 and step 4 to have the process wait 5 seconds for the network instead of 6.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
C	A	A	B	B				A	A				B	B	A	A	A			B	B	B	A
D			A	A	A	A	A	B	B	B	B	B						A	A	A	A	A	B
N										A	A	A	A	A	B	B	B	B	B				

	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46
C	A				B	B	A	A	A			B	B	B	A	A				B	B	A	A
D	B	B	B	B						A	A	A	A	A	B	B	B	B	B				
N		A	A	A	A	A	B	B	B	B	B						A	A	A	A	A	B	B

The first pattern is evident between 13 seconds and 28 seconds. Hence, T_{total} = 15 ms.

Capacity of web server = # process completed in 30ms / 30ms = 2/15 ≈ 0.13 requests per ms

Question 3

Part (a)

Implementation 1 — main memory latency cut by 50%
Implementation 2 — cache latency cut by 20%

average memory access time = hit time + (miss rate × miss penalty)

speedup with main memory optimization: $\dfrac{0.95+(0.05*8)}{0.95+\dfrac{0.05*8}{2}}=\dfrac{0.95+0.4}{0.95+0.2}=\dfrac{1.35}{1.15}\approx 1.17$

speedup with cache optimization: $\dfrac{0.95+(0.05*8)}{(0.95*.8)+(0.05*8)}=\dfrac{0.95+0.4}{0.76+0.4}=\dfrac{1.35}{1.16}\approx 1.16$

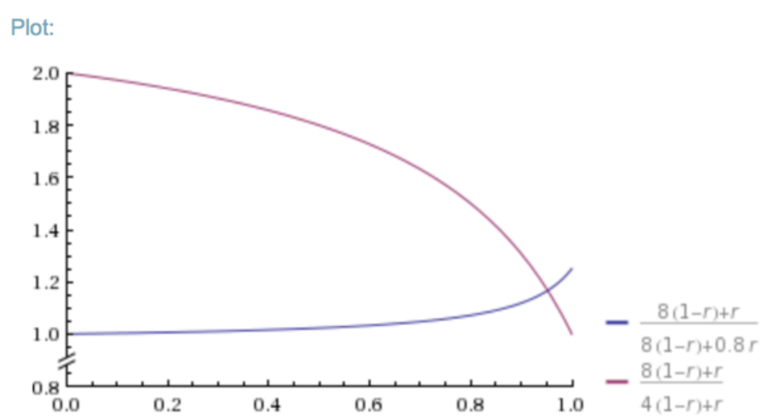
Part (b)

We can generalize the enhanced speedup for each optimization with the following equations and find their intersection point to answer this problem:

speedup with main memory optimization: $\dfrac{r+((1-r)*8)}{r+\dfrac{((1-r)*8)}{2}}=\dfrac{r+((1-r)*8)}{r+((1-r)*4)}$

speedup with cache optimization: $\dfrac{r+((1-r)*8)}{(r*.8)+((1-r)*8)}$

As we can see, the two graphs intersect when the cache hit rate is about 95%. Therefore for cache hit



rates below 95%, I would select the main memory optimization and for rates above 95%, I would select the cache optimization.

Part (c)

The speedup that is achieved if both optimizations are adopted is found by the following function where the cache hit rate is always at its highest possible rate:

$$f(h) = \left\{ \begin{array}{c} \dfrac{h+((1-h)*8)}{h+((1-h)*4)}, 0 \le h \le 0.95 \\ \dfrac{h+((1-h)*8)}{(h*.8)+((1-h)*8)}, 0.95 < h \le 1 \end{array} \right\}$$

Question 4

Part (a)

Let
u = percentage of useless packets = 150 / 1500 = 10% = 0.1
P = fraction of packets that are lost
H = total number of hops in path
n = network throughput = 10 Mbps

f(H) = ((1 − P) * n * (1 − u)) * H = ((1 − P) * 10 * 0.9) * H = (9 − 9P) * H

Part (b)

effective throughput = f(H) = (9 − 9P) * H = (9 − 9(0.01)) * 1 = 8.91 Mbps

Part (c)

We can use the equation that we got from part (a) to solve this problem...

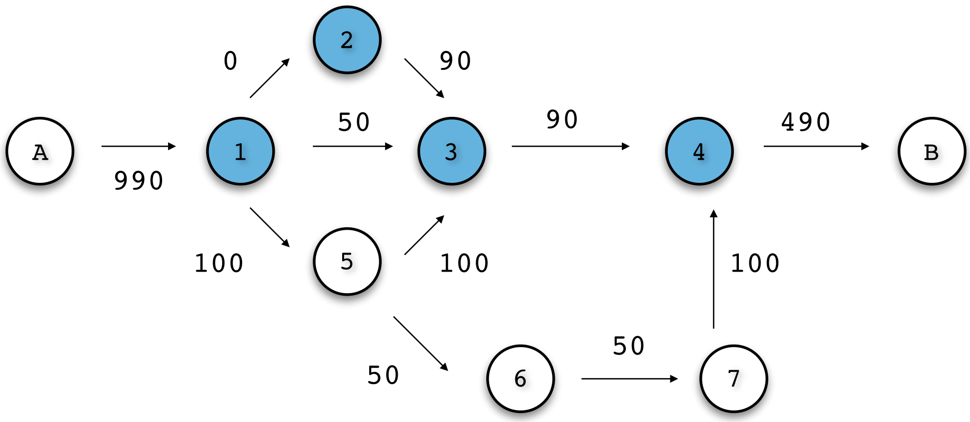
6 Mbps > $(9 - 9\text{P})L$
9P < $3L$
P < 1/3*L

Question 5

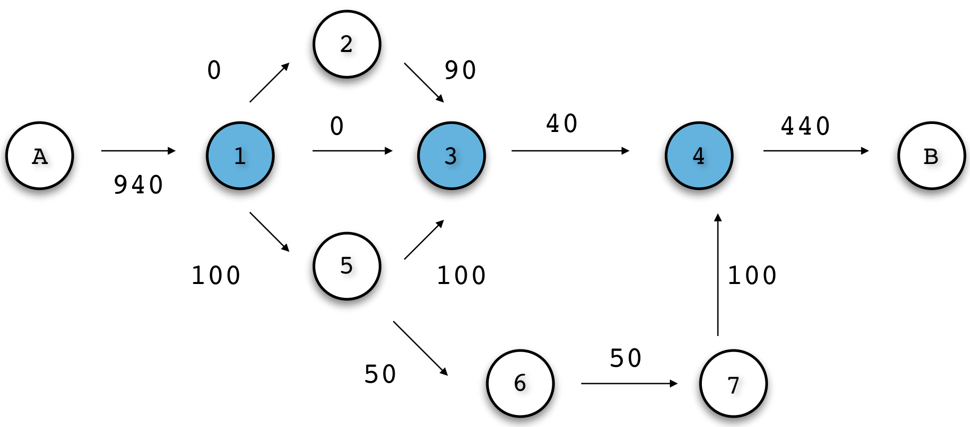
Part (a)

Residual graph #1: A —> 1 —> 2 —> 3 —> 4 —> B
capacity = 10

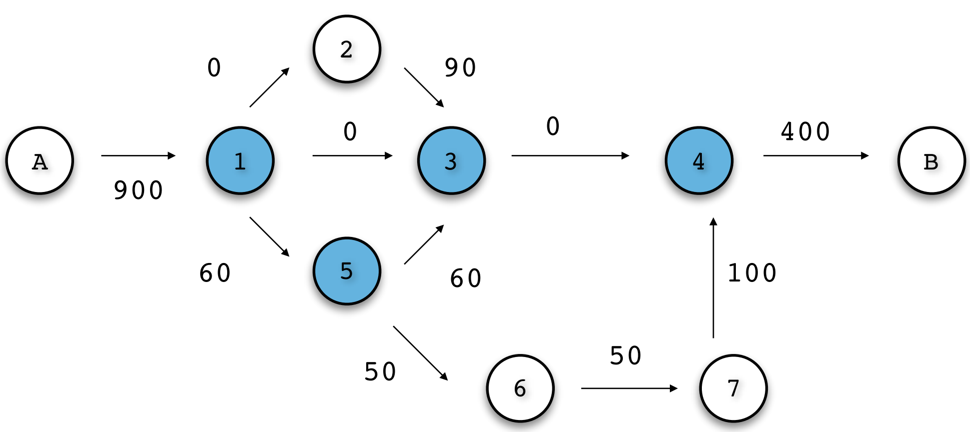
[graph on next page]



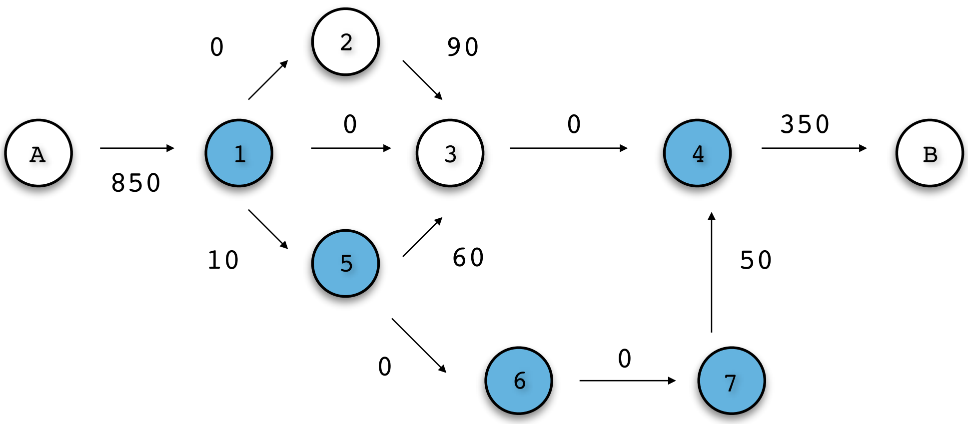
Residual graph #2: A —> 1 —> 3 —> 4 —> B
capacity = 50



Residual graph #3: A —> 1 —> 5 —> 3 —> 4 —> B
capacity = 40



Residual graph #4: A —> 1 —> 5 —> 6 —> 7 —> 4 —> B capacity = 50

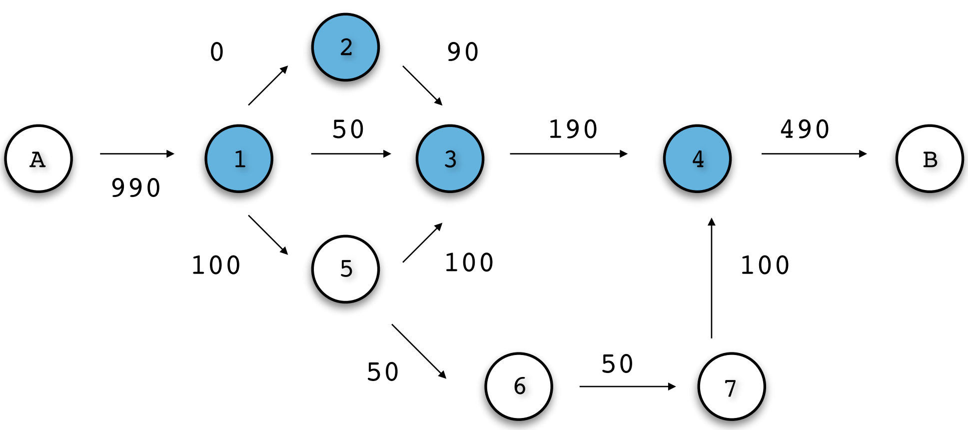


Max network capacity = 10 + 50 + 40 + 50 = 150

Part (b)

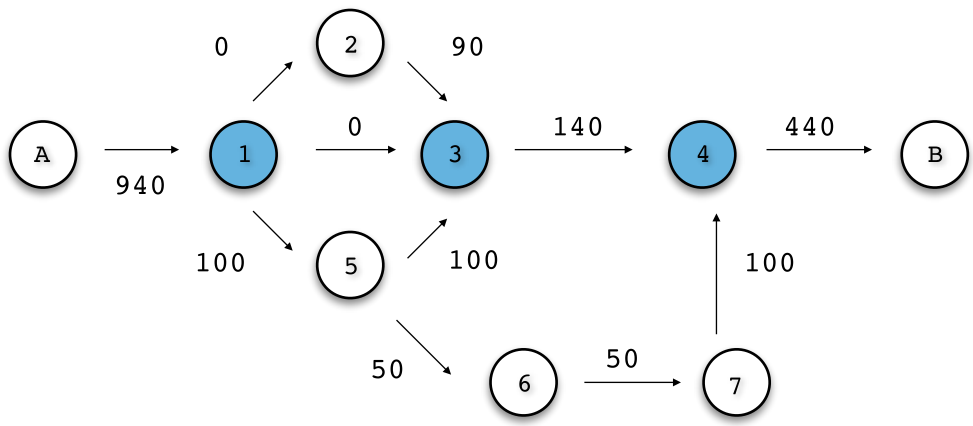
The most plausible link to upgrade the capacity of is the link with the node that has the most incoming connections. In our network, 3 has the most incoming connections, which is 3. Therefore, we can upgrade the capacity of link 3 —> 4 by doubling it from 100 to 200. When this change is applied to the network, we will see the capacity of the network increase as shown in the following residual graphs:

Residual graph #1: A —> 1 —> 2 —> 3 —> 4 —> B
capacity = 10

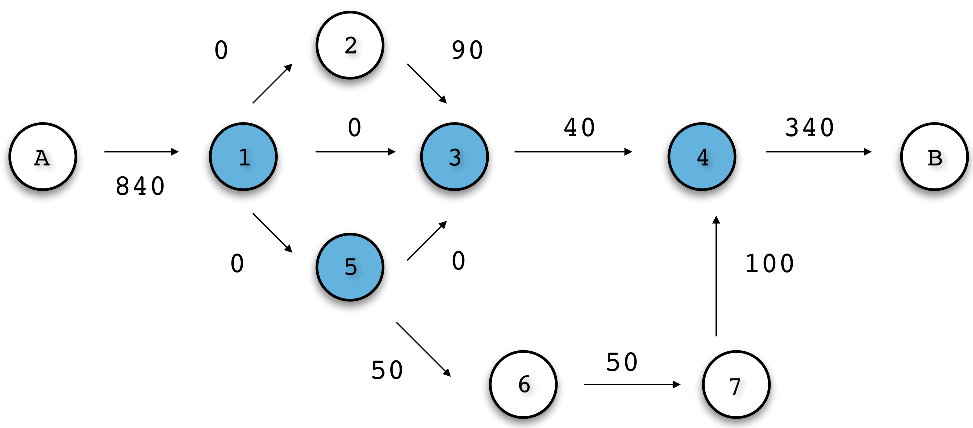


Residual graph #2: A —> 1 —> 3 —> 4 —> B
capacity = 50

[graph on next page]



Residual graph #3: A —> 1 —> 5 —> 3 —> 4 —> B
capacity = 100



The new max network capacity = 10 + 50 + 100 = 160

Part (c)

The network capacity will be increased by the difference of the new maximum capacity after upgrading the link and the maximum capacity before upgrading the link. For example, we can see from parts (a) and (b) that after doubling the capacity of link 3 —> 4, the max network capacity increased by a unit of 160 - 150 = 10. This amount that the network capacity increases by will differ depending on which link you decide to upgrade and by how much.

Part (d)

If I wanted to inflict the most performance degradation in a cyber warfare, I would try to take down the link that has the node with the most incoming connections. This node creates many different paths for data to travel; so by removing it, we've essentially decreased the number of possible paths in the graph. Logically speaking, if there are less paths to travel by, the probability that the maximum capacity will be reached sooner is much higher.