Skill Assessment #9

Graded

1 Day, 12 Hours Late

Student

HARRY KIM

Total Points

97.5 / 100 pts

Question 1

Hard disk drive performance

25 / 25 pts

✔ **+ 25 pts** Correct. Solution is correct and explanation is comprehensive

+ 20 pts Good. The solution shows understanding and the explanation is reasonable.

+ 15 pts Fair. The solution shows some understanding but has errors.

+ 10 pts Poor. The solution shows little understanding and has numerous errors.

+ 0 pts Inadequate.

Question 2

SSD performance

25 / 25 pts

✔ **+ 25 pts** Correct. Solution is correct and explanation is comprehensive

+ 20 pts Good. The solution shows understanding and the explanation is reasonable.

+ 15 pts Fair. The solution shows some understanding but has errors.

+ 10 pts Poor. The solution shows little understanding and has numerous errors.

+ 0 pts Inadequate.

Question 3

Transmission Performance

25 / 25 pts

- ✔ **− 0 pts** A and B: Correct. Solution is correct and explanation is comprehensive
- **− 0.5 pts** A and B: Used powers of 2 for network speeds (should use powers of ten here) Answer was otherwise correct.
- **− 1.5 pts** A and B: One correct, other is incorrect due to minor arithmetic/algebra error
- **− 4 pts** A and B: Some work shown but both incorrect
- **− 5 pts** A and B: No work shown
- **− 10 pts** A and B: No work
- **− 1 pt** Handwritten
- ✔ **− 0 pts** C: Points for effort
- ✔ **− 0 pts** D correct
- **− 2 pts** D: Some explanation but incorrect
- **− 10 pts** C: Little or no work
- **− 5 pts** Part D missing

Question 4

Parallel algorithm analysis

22.5 / 25 pts

- **+ 25 pts** Correct. Solution is correct and explanation is comprehensive
- ✔ **+ 22.5 pts** Good. The solution shows understanding and the explanation is reasonable.
- **+ 20 pts** Fair. The solution shows some understanding but has errors or missing key pieces
- **+ 15 pts** Poor. The solution shows little understanding and has numerous errors.
- **+ 0 pts** Inadequate

parallelizing won't help binary search. Ideal number of cores for mergesort is length of array/2;

Question assigned to the following page: 1

Harry Kim
u1226472

Skill Assessment #9

A1.

(a)

Average latency for reading a 1024 byte block of data for drive A:

$$Seek\ delay\ =\ 8\ ms$$

$$average\ rotational\ delay\ =\ \frac{1}{2} * \frac{60\ s}{7{,}500\ RPM} * \frac{1000\ ms}{s}\ =\ 4\ ms$$

$$time\ to\ read\ =\ 1024\ bytes * \frac{1\ megabytes}{1000000\ bytes} * \frac{1\ s}{40\ megabytes} * \frac{1000\ ms}{1\ s} = 0.0256\ ms$$

$$time\ to\ transmit\ data\ =\ 1024\ bytes * \frac{1\ megabytes}{1000000\ bytes} * \frac{1\ s}{120\ megabytes} * \frac{1000\ ms}{1\ s} = 0.0085\overline{3}\ ms$$

$$average\ latency = 8ms\ +\ 4\ ms\ +\ 0.0256\ ms\ +\ 0.0085\overline{3}\ ms\ =\ 12.0341\overline{3}\ ms$$

Average latency for reading a 1024 byte block of data for drive B:

$$Seek\ delay\ =\ 4\ ms$$

$$average\ rotational\ delay\ =\ \frac{1}{2} * \frac{60\ s}{10{,}000\ RPM} * \frac{1000\ ms}{s}\ =\ 3\ ms$$

$$time\ to\ read\ =\ 1024\ bytes * \frac{1\ megabytes}{1000000\ bytes} * \frac{1\ s}{70\ megabytes} * \frac{1000\ ms}{1\ s} = 0.01462857143\ ms$$

$$time\ to\ transmit\ data\ =\ 1024\ bytes * \frac{1\ megabytes}{1000000\ bytes} * \frac{1\ s}{200\ megabytes} * \frac{1000\ ms}{1\ s} = 0.00512\ ms$$

$$average\ latency = 4\ ms\ +\ 3\ ms\ +\ 0.01462857143\ ms\ +\ 0.00512\ ms\ =\ 7.019748571\ ms$$

(b)

Minimum latency for reading a 2048 byte block of data for drive A:

$$Seek\ delay\ =\ 0\ ms$$

$$average\ rotational\ delay = \ 0\ ms$$

$$time\ to\ read\ =\ 2048\ bytes * \frac{1\ megabytes}{1000000\ bytes} * \frac{1\ s}{40\ megabytes} * \frac{1000\ ms}{1\ s} = 0.0512\ ms$$

Questions assigned to the following page: 1 and 2

Harry Kim
u1226472

$$time\ to\ transmit\ data\ =\ 2048\ bytes\ *\ \frac{1\ megabytes}{1000000\ bytes}\ *\ \frac{1\ s}{120\ megabytes}\ *\ \frac{1000\ ms}{1\ s}\ =\ 0.0170\overline{6}\ ms$$

$$minimum\ latency\ =\ 0ms\ +\ 0\ ms\ +\ 0.0512\ ms\ +\ 0.0170\overline{6}\ ms\ =\ 0.0682\overline{6}\ ms$$

Minimum latency for reading a 2048 byte block of data for drive B:

$$Seek\ delay\ =\ 0\ ms$$

$$average\ rotational\ delay\ =\ 0ms$$

$$time\ to\ read\ =\ 2048bytes\ *\ \frac{1\ megabytes}{1000000\ bytes}\ *\ \frac{1\ s}{70\ megabytes}\ *\ \frac{1000\ ms}{1\ s}\ =\ 0.02925714286\ ms$$

$$time\ to\ transmit\ data\ =\ 1024\ bytes\ *\ \frac{1\ megabytes}{1000000\ bytes}\ *\ \frac{1\ s}{200\ megabytes}\ *\ \frac{1000\ ms}{1\ s}\ =\ 0.01024\ ms$$

$$minimum\ latency\ =\ 0\ ms\ +\ 0\ ms\ +\ 0.02925714286\ ms\ +\ 0.01024\ ms\ =\ 0.03949714286\ ms$$

(c)

For both hard drives, the seek time is the dominant factor that determines latency. A possible solution would be to make the disks even smaller to get faster seek times or have multiple read heads that can read multiple tracks on a disk.

A2.

(a)

Average latency for reading a 1024 byte block of data for drive A:

$$time\ to\ read\ =\ 1024\ bytes\ *\ \frac{1\ megabytes}{1000000\ bytes}\ *\ \frac{1\ s}{1200\ megabytes}\ *\ \frac{1000\ ms}{1\ s}\ =\ 8.5\overline{3}\ *\ 10^{-4}\ ms$$

$$time\ to\ transmit\ data\ =\ 1024\ bytes\ *\ \frac{1\ megabytes}{1000000\ bytes}\ *\ \frac{1\ s}{6000\ megabytes}\ *\ \frac{1000\ ms}{1\ s}\ =\ 1.70\overline{6}\ *\ 10^{-4}\ ms$$

$$average\ latency\ =\ 8.5\overline{3}\ *\ 10^{-4}\ ms\ +\ 1.70\overline{6}\ *\ 10^{-4}\ ms\ =\ 0.0010236\ ms$$

Average latency for reading a 1024 byte block of data for drive B:

$$time\ to\ read\ =\ 1024\ bytes\ *\ \frac{1\ megabytes}{1000000\ bytes}\ *\ \frac{1\ s}{200\ megabytes}\ *\ \frac{1000\ ms}{1\ s}\ =\ 0.00512\ ms$$

$$time\ to\ transmit\ data\ =\ 1024\ bytes\ *\ \frac{1\ megabytes}{1000000\ bytes}\ *\ \frac{1\ s}{300\ megabytes}\ *\ \frac{1000\ ms}{1\ s}\ =\ 0.00341\overline{3}\ ms$$

$$average\ latency\ =\ 0.00512\ ms\ +\ 0.00341\overline{3}\ ms\ \ =\ \ 0.0085\overline{3}\ ms$$

(b)

Minimum latency for reading a 2048 byte block of data for drive A:

$$time\ to\ read\ \ =\ \ 2048\ bytes\ *\ \frac{1\ megabytes}{1000000\ bytes}\ *\ \frac{1\ s}{1200\ megabytes}\ *\ \frac{1000\ ms}{1\ s}\ =\ 0.00170\overline{6}\ ms$$

$$time\ to\ transmit\ data\ \ =\ \ 2048\ bytes\ *\ \frac{1\ megabytes}{1000000\ bytes}\ *\ \frac{1\ s}{6000\ megabytes}\ *\ \frac{1000\ ms}{1\ s}\ =\ 3.41\overline{3}\ *\ 10^{-4}\ ms$$

$$Minimum\ latency\ =\ 8.5\overline{3}\ *\ 10^{-4}\ ms\ +\ 1.70\overline{6}\ *\ 10^{-4}\ ms\ \ =\ \ 0.002047\overline{9}\ ms$$

Minimum latency for reading a 2048 byte block of data for drive B:

$$time\ to\ read\ \ =\ \ 2048\ bytes\ *\ \frac{1\ megabytes}{1000000\ bytes}\ *\ \frac{1\ s}{200\ megabytes}\ *\ \frac{1000\ ms}{1\ s}\ =\ 0.01024\ ms$$

$$time\ to\ transmit\ data\ \ =\ \ 2048\ bytes\ *\ \frac{1\ megabytes}{1000000\ bytes}\ *\ \frac{1\ s}{300\ megabytes}\ *\ \frac{1000\ ms}{1\ s}\ =\ 0.00682\overline{6}\ ms$$

$$Minimum\ latency\ =\ 0.01024\ ms\ +\ 0.00682\overline{6}\ ms\ \ =\ \ 0.0170\overline{6}\ ms$$

(c)

Yes because in order to increase memory in an SSD, you must increase the number of bits per cell. When you increase the bits per cell, the latency increases. This is because you’re increasing the precision of which cell is charged (one bit has 2 binary values 0 and 1. Two bits have 4 binary values 00, 01, 10, and 11 and so forth). This is largely insignificant compared to HHDs because the latencies of SSDs are hundreds of times lower than HDDs meaning their throughput of transferring data is greater. SSDs also have no seek latencies or rotational delays which also contributes to a much lower latency than HDDs.

A3.

(a)

$$bits\ "in\ flight"\ \ =\frac{100\ megabits}{1\ s}\ *\ \frac{1000000\ bits}{1\ megabit}\ *\ \frac{1\ s}{2*10^{8}\ m}\ *\ 20\ m\ \ =\ \ 10$$

(b)

$$bits\ "in\ flight"\ \ =\ \ \frac{10\ gigabits}{1\ s}\ \ *\ \ \frac{1000000000\ bits}{1\ megabit}\ *\ \frac{1\ s}{2*10^{8}\ m}\ *\ 2000\ m\ \ =\ \ 100000$$

Questions assigned to the following page: 4 and 3

Harry Kim
u1226472

(c)

Time for cable A:

$$request\ time\ = \frac{10.1\ bits}{1\ byte} * \frac{200\ bytes}{1\ s} * \frac{1\ s}{100000000\ bits} * \frac{1000\ ms}{1\ s} + 0.02\ = 0.0402\ ms$$

$$data\ from\ server\ time\ = \frac{10.1\ bits}{1\ byte} * \frac{100000\ bytes}{1\ s} * \frac{1\ s}{100000000\ bits} * \frac{1000\ ms}{1\ s} + 0.02\ = 10.12\ ms$$

$$total\ time\ = 0.0402\ ms\ + 10.12\ ms\ = 10.1602\ ms$$

Time for cable B:

$$request\ time\ = \frac{10.1\ bits}{1\ byte} * \frac{200\ bytes}{1\ s} * \frac{1\ s}{100000000\ bits} * \frac{1000\ ms}{1\ s} + 0.02\ = 0.020202\ ms$$

$$data\ from\ server\ time\ = \frac{10.1\ bits}{1\ byte} * \frac{100000\ bytes}{1\ s} * \frac{1\ s}{10000000000\ bits} * \frac{1000\ ms}{1\ s} + 0.02\ = 0.121\ ms$$

$$total\ time\ = 0.020202\ ms\ + 0.121\ ms\ = 0.141202\ ms$$

(d)

The bit rate is the dominant factor. Getting a faster transmitter would best decrease the latency of the data block request.

A4.

It is difficult to use a large number of cores to speed up the algorithm because at some point the time needed to partition the work and the time needed to combine all results, everything that has doesn't have anything to do with actually running the algorithm, begins to outweigh the benefits of speeding up the process time if too many cores try to complete a single task.

To find the ideal number of cores, we would need to solve for them using Amdahl's law:

$$speedup\ = \frac{1}{(1 - Fraction\ time\ affected) + \frac{Fraction\ time\ affected}{Amount\ of\ improvement}}$$

Let's say we want a speed factor of 90 times faster with x processors and the fraction time affected was 0.99 for the binary search algorithm. We would simply plug in the values and solve for x.

$$90\ = \frac{1}{(1-0.99) + \frac{0.99}{x}}$$

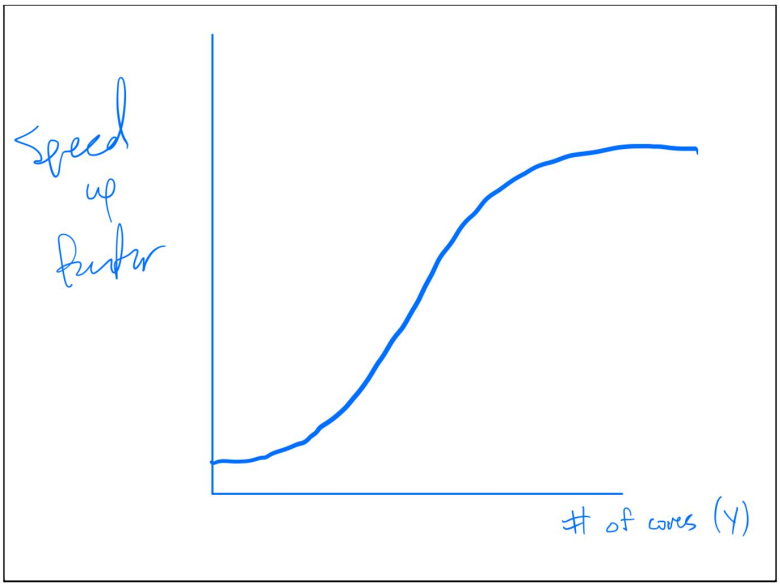
Solving for x would give us 891 cores.

Question assigned to the following page: 4

Harry Kim
u1226472

6.5.1

I would expect the speed up factor to be fairly low since the number of cores is *much* lower than the length. I would expect the speed up factor to increase as more cores are added, but then taper off and reach some limit depending on the amount of code that is parallelizable.



6.5.2

I would expect the speed up factor to be pretty high, but unable to go beyond a certain limit depending on the amount of code that is parallelizable, no matter how many more cores we add. As for trying to obtain the best speed up factor by changing the code, I don't think this would be possible without changing the entire sort method.

The ideal number of cores would be to, again, set up Amdahl's law and solve for the number of cores depending on how much you want your program to speed up.

$$100 = \frac{1}{(1-0.99) + \frac{0.99}{x}}$$