ECE 563 Second Midterm Spring 2018

Name (printed):

Signed:

By signing you are stating that you have neither given nor received help from others during this exam, and that you understand that if you have given or received help from others you will receive a grade of zero on the exam.

In any problem needing calculations, answers only need to be computed to two digits.

Question 1.

(a) (6 points) Ted analyzes his program and finds that the fraction of a sequential execution that is parallel is .8. Using Amdahl's law, his speedup should be 2.5. When he runs the program his speedup is less than this. Briefly say why this might be the case.

Amdahl's Law does not take communication **and other parallel overheads *into account*, and thus the actual program execution time will be higher than $\sigma(n) + \phi(n)/P$. Thus the speedup will be less. (The italicized part of the answer was sufficient).**

(b) (6 points) According to Amdahl's Law, what would the speedup be for Ted's program on 8 processors?

f = 1 - 0.8
$\Psi = 1 / (.2 + (1-.2)/\pi)$
$= 1 / (.1 + .2)$
$= 1 / .3$
$= .33$

(c) (6 points) Using the speedup computed in (b), what is the efficiency of Ted's program on 8 processes?

E = 3.3 / 8 = .41

Question 2.

(a) **(6 points)** Ted's manager tells Ted that they can increase the problem size for the program of Question 1, and that the parallel work will grow faster than the sequential work. Will the speedup increase or decrease?

Increase. **Ignoring communication, $\psi = (\sigma(n) + \phi(n)) / (\sigma(n) + \phi(n)/P)$. Increasing n, where $\phi(n) > \sigma(n)$ leads to a larger ψ. (The italicized part of the answer was sufficient).**

b) **(6 points)** With the larger problem size, the *fraction of a parallel execution* that is sequential is .2. What is the predicted speedup on 8 processors?

Using Gustafsen-Barsis law (because we have the fraction of a parallel execution) we have:

$\Psi = 8 + (1\text{-}8) * 2$
$= 8 - 7*.2$
$= 6.6$

(c) **(6 points)** Again, when Ted runs the program, the actual speedup is less than predicted. Briefly say what could cause this to happen.

G-B, like Amdahl's Law, does not take into account parallel overhead such as communication.

(d) **(6 points)** Ted's manager has left over money and wants to buy a 1024 node machine. What is the fraction of parallel execution that can be sequential and still achieve a speedup of 512 on this machine?

Again, using G-B Law, we have
$512 = P + (1\text{-}P) * s$
$= 1024 + (1\text{-}1024) * s$
$s = 512 / 1023$
$= .50$

Question 3.

(a) (6 points) Ted has written two programs. The first has speedups of 1, 1.6, 2.3, 2.9 and 3.4 on 1, 2, 4, 8 and 16 processors, respectively. Compute the values of *e*, the experimentally determined serial fraction of the program, in the table below. You only need to compute *e* to 2 digts.

P	2	4	8	16
Speedup	1.6	2.3	2.9	3.4
e	**~0.25**	**~0.25**	**~0.25**	**~0.25**

(b) (6 points) The second program has speedups of 1, 1.8, 2.6, 2.9 and 3.6 on 1, 2, 4, 8 and 16 processors, respectively. Values of *e* are shown below. Explain, in terms of *e,* which program will be most scalable.

P	2	4	8	16
Speedup	1.8	2.6	3.3	3.6
e	0.11	0.18	0.20	0.23

***e* is increasing for the second program, and will dominate with large numbers of processors, leading to worse speedups than with the first. *So the second program will be faster with large numbers of processors.* (The italicized part of the answer was sufficient).**

Question 4.

Ted has written two programs that both solve the same problem, and that will be run on up to 1,000,000 processors. The two programs' complexity can be described by the following, where a communication operation takes 1 unit of time, and 1 unit of work takes 1 unit of time.

Program 1.
$W = n^2$, where n is the problem size
Communication overhead grows as $\log_2 P$

Program 2.
$W = n$, where n is the problem size
Communication overhead grows as P^2

(a) (6 points) Derive the isoefficiency relationship for Program 1.

$T_P = n^2 / P + \log_2 P$
$T_O = PT_P - T1 = P^* n^2 / P + P * \log_2 P - n^2$
$= n^2 + P * \log_2 P - n^2$
$= P * \log_2 P$

W must grow proportional to T_O, therefore n^2 must grow proportional to $P * \log_2 P$

(b) (6 points) Derive the isoefficiency relationship for Program 2.

$T_P = n / P + P^2$
$T_O = PT_P - T1 = P^* n / P + P * P^2 - n$
$= n + P^3 - n$
$= P^3$

W must grow proportional to T_O, therefore n must grow proportional to P^3

(c) (6 points) Explain, in terms of any or all of the isoefficiency relationship, the amount of work, and the communication overhead, which program would be fastest to run on 8 processors with n=100.

Time to run Program 1:
$TP = 100^2/8 + 3$
$= 1250 + 3$
$= 1253$

Time to run Program 2:
$TP = 100/8 + 3$
$= 12.5 + 64$
$= 76.5$

Program 2 is fastest.

(d) (6 points) Explain, in terms of any or all of the isoefficiency relationship, the amount of work, and the communication overhead, which program would be fastest if $P=10^6$ and $n=10^6$.

Time to run Program 1:
$TP = 10^{12}/10^6 + 20$
$= 10^6 + 3$

Time to run Program 2:
$TP = 10^6 / 10^6 + 10^{12}$
$= 10^{12} + 1$

Program 1 is fastest.

The machine we are running on gets new processors that have vector units. These decrease the amount of work needed when running as a parallel program by a factor of 16 since they can perform 16 operations at once on each core (they are not used in the serial program).

a) (6 points) Assuming the only thing that changes in the machine is the addition of vector units. Does the program run faster or slower?

Faster. Up to 16 operations can be done in parallel.

b) (6 points) Given $T_p = (n^4)/16*P + P^2$ what is the isoefficiency relationship?

There are two ways to compute this. The first assumes that the sequential processor executes 16P operations at a time. I counted either as correct. Then:

$$
\begin{aligned}
T_O = PT_P - T1 &= P*n^4/16*P + P*P^2 - n^4/16 \\
&= n^4/16 + P^3 - n^4/16 \\
&= P^3
\end{aligned}
$$

The second assumes that the sequential processor executes 1 operation at a time:

$$
\begin{aligned}
T_O = PT_P - T1 &= P*n^4/16*P + P*P^2 - n^4 \\
&= n^4/16 + P^3 - n^4 \\
&= -15n^4/16 + P^3
\end{aligned}
$$

To find the isoefficiency formula we set $W = T_O$

$n^4 = -15n^4/16 + P^3$

$31*n^4/16 = P^3$

$n^4 = P^3*16/31$

c) (6 points) Given your answer to (b), does the program running on the machine with vector units need to increase its data faster, slower or the same to maintain isoefficiency than when running on a machine without vector units?

For the first way above, the answer would be "the same".
For the second answer above, the answer could be either "the same" or "slower", depending on whether you count the fraction 16/31 or just fold it into the constants.

In general, I took you answer to (b) to determine the correctness of (c)