CSCI-UA.0480-009

Parallel Computing

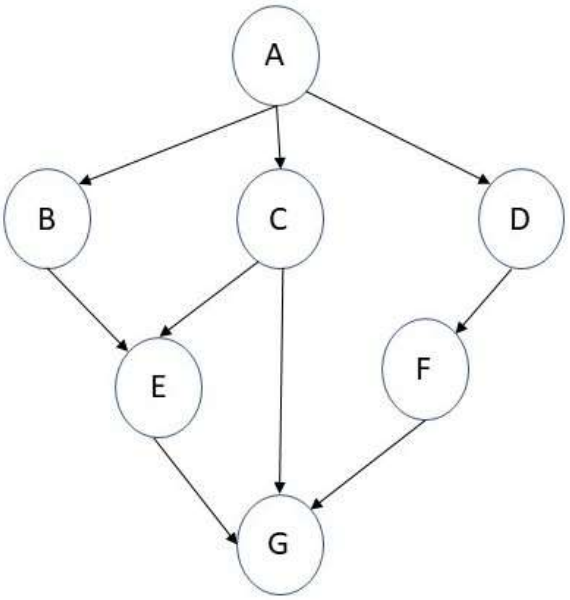
Midterm Exam

Fall 2019 (60 minutes)

Last Name: **First Name:** **NetID:**

- This exam contains **6 questions** with a total of **50 points** in **5 pages.**
- If you must make assumptions to continue solving a problem, state your assumptions clearly.

1. Assume we have the following task graph. An arrow from a task to another means that the first task generates data needed by the second one. Do not make any assumptions regarding cache misses, coherence, etc. The table indicates the time taken by each task. A task cannot be decomposed to smaller tasks.



Task#	Duration
A	10
B	50
C	100
D	100
E	50
F	50
G	10

a. [3 points] What is the smallest number of cores needed to get the best performance? Show the steps that you did to reach your conclusion.

Three cores are needed.
A then B, C, D in parallel, then E and F, then G.

b. [2 points] Based on the number of cores you picked in part (a) above, what is the total run time?

Giving 10+100+50+10= 170

c. [2 points] If we use 4 cores, what will be the efficiency? (You can leave it as a fraction.)

E = Speedup (4) / 4
Speedup (4) = T(seq)/T(4) = 370/170
Efficiency = 370/(170*4) A pretty low one efficiency

d. [2 points] What is the span for the above graph? what is the work?

Span → A,C,E,G or A,D,F,G → both = 170
Work = T (seq) = 370

e. [2 points] What is the parallelism of that graph?

Parallelism = ceil (work/span) = 370/170 = 3

f. [2 points] What does the number you calculated in (e) mean?

I means that we will not gain more performance by using more than three cores.

g. [3 points] What is the speedup when using three cores based on Amadalh's law? [Hint: Assume the purely sequential part are tasks A & G from the 7 tasks, hence F = 2/7].

S(4) = 1/(F + (1-F)/3) = 1/((2/7) + (5/7/3)) = 21/11

h. [2 points] Now, calculate the speedup with three cores when using the graph above

S(3) = 370/170

i. [3 points] Are the numbers you calculated in g and h different? Justify.

Yes, they are different because The graph shows some other dependencies from among the tasks in the parallel part that are not taken into account by Amdahl's law.

2. [3 points] We have studied Amdahl's law in class. Suppose that the fraction of time of the sequential part of your program is 60%. What is the best action to get better performance: reducing the sequential part to 20% and using two cores? Or keeping the sequential at 60% but use four cores? Show all the steps of using Amadahl's law to reach your conclusion.

S(P) = 1/(F + (1-F)/P)
strategy 1: F = 0.2 and P = 2 so speedup = 1/(0.2 + (0.8/2)) = 1/0.6
strategy 2: F = 0.6, P= 4 so Speedup = 1/(0.6 + 0.4/4) = 1/0.7
Strategy 1 is better

3. [2 points] In MPI, sending few long messages are better than sending many short messages. Why?

Sending a message requires a system call and a lot of overhead from the MPI runtime. So sending several messages results on these overheads taking place several times.

4. [4 points] Modify the following piece of code to allow iterations to be execution in parallel to gain some performance.

```
a[0] = 0;
for( i = 1; i < n; i++)
    a[i] = a[i-1] + i;
```

We can see that a[0] = 0, a[1] = a[0]+1 = 1. a[2] = a[1]+2 = 1+2 = 3. a[3]=3+3=6
So a[i] = 1+2+3+…+i = i(1+i)/2 = This makes the new code:

```
for( i = 0; i < n; i++) {
    a[i] = 0;
    for(j = 0; j <= i; j++)  a[i] += j;  or a[i] = (i + i*i)/2; }
```

The outer loop iterations are now totally independent.

5. [12 points] The table below shows three processes in MPI and the operations they do at each time step of the same program. Also assume that the reduction operation is **MPI_SUM**. (Hint: Second argument in MPI_Allreduce is the target variable)

Time	Process 0	Process 1	Process 2
0	x = 2; y = 3;	x = 3; y = 4;	x = 4; y = 5;
1	MPI_Allreduce(&x, &y, …)	MPI_Allreduce(&y, &x, …)	MPI_Allreduce(&x, &y, …)
2	MPI_Allreduce(&y, &x, …)	MPI_Allreduce(&x, &y, …)	MPI_Allreduce(&x, &y, …)

Fill in the following table with values of x and y in each process:
After Step 1:

	Process 0	Process 1	Process 2
x	2	10	4
y	10	4	10

After Step 2:

	Process 0	Process 1	Process 2
x	24	10	4
y	10	24	24

6. [8 points] Circle the correct answer among the choices given. If you circle more than one answer, you will lose the grade of the corresponding question.

1. Which of the following provides more potential performance gain:
a. Multiple threads within a process b. Multiple processes within a thread
c. Multiple threads and no processes d. Multiple processes and no threads

2. Coherence negatively affects performance of MPI programs as the number of processes increases.
a. True b. False c. Depends on MPI implementation

3. MPI does not suffer from race condition
a. True b. False c. Depends on MPI implementation

4. Load imbalance becomes more severe when we have fewer synchronization points in the parallel program.
a. True b. False c. Depends on the language runtime

5. MPI is useful when (choose the most accurate)
a. There are a lot of processes
b. There are a lot of processes exchanging a lot of data
c. There are a lot of dependent processes exchanging very few data
d. There are a lot of independent processes exchanging very few data

6. MPI can have critical sections.
a. True b. False c. Depends on MPI implementation

7. Amdah's law
a. Gives upper bound of the speedup b. Gives lower bound of the speedup

8. I have written my last name, first name, and NetID at the top of the first page of this exam.
a. True b. False c. Depends on L2 cache size!