SABANCI UNIVERSITY

CS406/531: Parallel Computing,
Spring 2015, Midterm II
Duration: 110 minutes

Q0 (10pts): What is the cost of transferring an m-word message from one processor to another on a network (assuming the simplified cost model)? Please explain each term (one sentence for each), just a simple formula won't get any credit.

Q1 (5pts): What is a **message tag** in MPI? For which cases, would it be important/useful?

Q2 (15pts): Please answer the following

a. What is the maximum speedup of the computation given that only %60 of the computation can be executed in parallel? Clearly explain. There will be no credit for a numeric value.
b. If you have an algorithm whose sequential complexity is O(log N) and **p** processors, what is the maximum speedup you can get? Is it possible to achieve greater speedup and, if so, under what circumstance might this happen?
c. Given a sequential algorithm requiring **n** operations on **n** numbers, you implemented a parallel algorithm requiring **(n/p + 2log p**) time, what is the speedup (S) and efficiency (E) of your algorithm.
d. (Continue from part (c)): What is the **n** value if you want to have 80% efficiency with 4 and 16 processors?
e. (Continue from part (c)): Give a formula of **n** (in terms of **p**) to have 80% efficiency for every **p** value.

Q3 (5pts): An operation must be _____ to be used in a parallel reduction?

a. Commutative: (e.g., a + b = b + a)
b. Associative: (e.g., a + (b + c) = (a + b) + c)
c. Distributive: (e.g., a (b + c) = ab + ac)

Please choose the suitable answer(s) above and explain clearly.

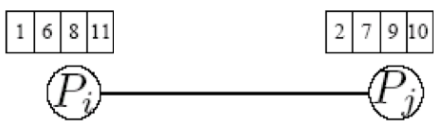
Q4 (10pts): You are told that when the below code is executed with a single block having 32 threads you have a bank conflict:

```
__global__ void conflict() {
   __shared__ double values[33];
   values[threadIdx.x] = threadIdx.x;
}
```

Please write why and state how many conflicts occur on a bank? Change one word of this code and make it work without any bank conflict.

Q5 (15pts): On parallel sorting:

a. Processors P_i and P_j has the following numbers. They want to do a **compare-split** (not compare-exchange), please draw what happens after each step



b. The mapping of bitonic sort algorithm to a hypercube (with 2^d processors and one integer per processor) is given below

```
1.  procedure BITONIC_SORT(label, d)
2.  begin
3.      for i := 0 to d − 1 do
4.          for j := i downto 0 do
5.              if (i + 1)^st bit of label ≠ j^th bit of label then
6.                  comp_exchange_max(j);
7.              else
8.                  comp_exchange_min(j);
9.  end BITONIC_SORT
```

What is the parallel execution time? Is this algorithm cost optimal w.r.t. its serial counterpart (show your math)? Is this algorithm cost optimal with respect to the best sequential sorting algorithm?

c. Why Bubble sort (a simple sorting algorithm that repeatedly steps through the list to be sorted, compares each pair of adjacent items and swaps them if they are in the wrong order) is hard to parallelize? Describe a more parallelizable version of Bubble sort and write a simple pseudocode.

Q6 (10pts): Explain the advantages and drawbacks of the Cannon's matrix multiplication algorithm? Are there better algorithms to solve the drawbacks? How do they solve them?

Q7 (15pts): Parallelize the following codes using OpenMP pragmas. Be sure to explicitly specify the "schedule" options that should be used, even if you want to use the default options. For each please rewrite the code. Assume that N is large (in the tens of thousands or more). You must explicitly list all variables within the range of a parallel pragma that are private using the private() directive.

a.

```
for (i=0;i<N;i++){
    for (j=0;j<N;j++){
        A[i,j] = max(A[i,j],B[i,j]);
    }
}
```

b. Given the data structure for a linked list as shown below, parallelize the for loop.

```
typedef struct element
{
  int value;
  struct element *next;
} Element;

Element *D[N]; // D[] is an array of pointers to linked lists
of varying length
int C[N];


for (i=0;i<N;i++){
    C[i] = computeAverageValueOfAllListElements(D[i]);
}
```

c. For the same D array above:

```
Element *pointer_to_max = NULL;
for (i=0;i<N;i++){
    Element *tmpPtr = findListElementWithMaxValue(D[i]);
    if (!pointer_to_max ||
        (tmpPtr && tmpPtr->value > pointer_to_max->value))
       pointer_to_max = tmpPtr;
}
```

Q8 (10pts): On Apache Hadoop and Spark:

a. What is a mapper, combiner, and reducer in Hadoop? Please explain each with their differences and responsibilities within the MapReduce paradigm.
b. What is Apache Spark, why it can be better than Hadoop + MapReduce? For what kind of algorithms it can be worse than Hadoop + MapReduce?

Q9 (20pts): The connected components of an undirected graph G = (V,E), are the maximal disjoint sets $C_1, C_2, \cdots, C_k$ such that $V = C_1 \cup C_2 \cup ... \cup C_k$, and u, v $\in C_i$ if and only if u is reachable from v and v is reachable from u.

a. Formulate a high-level parallel algorithm for finding the connected components of an undirected graph.
b. Formulate the parallel runtime of the algorithm (your runtime can depend on several graph parameters).
c. Discuss the implementation of the algorithm on a distributed system with MPI. What will be the potential bottlenecks and potential solutions?
d. Discuss the implementation of the algorithm on a GPU. What can be the potential bottlenecks and potential solutions?