PRINCE OF SONGKLA UNIVERSITY
FACULTY OF ENGINEERING

Midterm Examination: Semester I
Date: 15 October 2016
Subject Number: 242-530
Subject Title: Parallel and Distributed Computing

Academic Year: 2016
Time: 13.30 - 16.30 (3 hours)
Room: A401

Exam Duration: 3 hours

This paper has 18 pages, 10 questions and 150 marks (30%).

Authorised Materials:

- Writing instruments (e.g. pens, pencils).
- An A4 Sheet and a dictionary are permitted.

Instructions to Students:

- Scan all the questions before answering so that you can manage your time better.
- Answers **must** be written in **English**.
- Write your name and ID on every page.
- Any unreadable parts will be considered wrong.

When drawing diagrams or coding, use good layout, and short comments; marks will not be deducted for minor syntax errors.

Cheating in this examination

Lowest punishment: Failed in this subject and courses dropped for next semester.

Highest punishment: Expelled.

NO	Time (Min)	Marks	Collected	NO	Time (Min)	Marks	Collected
1	60	44		6	10	8	
2	20	22		7	15	10	
3	10	8		8	15	10	
4	10	12		9	10	9	
5	10	12		10	20	15	
Total	**180**	**150**			**30%**		

Question 1

(44 marks; 40minutes)

Answer the following questions.

a) What are the differences between *communication* and *synchronization*? (2 marks)

Communication	Synchronization

b) Compare Parallel Vector Processor (PVP) and Symmetric Multiprocessors (SMP). (6 marks)

Parallel Vector Processor	Symmetric Multiprocessors

c) Compare the advantages and disadvantages of Shared Memory Model and Distributed Memory Model. (6 marks)

Shared Memory Model	Distributed Memory Model

d) Compare OpenMP and MPI. (6 marks)

OpenMP	MPI

Name______________________________ ID________________

e) Which are the significant constraints in building a faster serial computer. (4 marks)

f) Give 4 reasons for using parallel computing. (4 marks)

i) Explain how to do *automatic parallelization*, tell when to choose which method and inform the tradeoffs. (4 marks)

Name____ID______________________

j) What and why do we need to identify which is the first step in developing parallel software when we are in the process of understanding the problem and the program? (6 marks)

k) Explain the effects of each parameter in the Amdahl's law (6 marks)

Name____ID____________________

Question 2 (22 marks; 20 minutes)

Answer the following questions.

Questions a) - d) are about *Cache Coherence*:

a) What does *Cache Coherent* mean? (2 marks)

b) Each processor in a shared memory system views the memory through its cache, how can the system ensure that different processors do not have different values for the same memory location especially when a processor has an obsolete image of an address location stored in its cache? (2 marks)

c) Giving two interconnection media types: Shared Medium and Switched Medium. Which interconnection media type does enable multiple messages to be sent simultaneously and allow scaling of network to accommodate increase in processors? (2 marks)

d) Explain why the cache controller cannot simply *snoop* the shared memory in case of *Distributed Shared Memory.* (2 marks)

Questions e) - f) are about *Uniform Memory Access (UMA)* and *Non-Uniform Memory Access (NUMA)*:

e) Which kind of memory access does require identical processors? (2 marks)

f) Which kind of memory access does not provide equal access time to all processors? (2 marks

Name____ ID______________________

Questions g) - i) are about *Synchronization*:

g) Explain the *Synchronization Problem* associated with shared data. (2 marks)

h) What is *Mutual Exclusion*? (2 marks)

i) Explain what a *Barrier Mechanism* does. (2 marks)

Questions j) - i) are about *Switched Network Topologies*:

j) In which cases does the Tree Switched Network Topology perform well and badly? (2 marks)

k) What are factors that contribute to *parallel overhead*? (2 marks)

Question 4 (12 marks; 10 minutes)

Explain the advantages and disadvantages of the following ways to program parallel computers:

a) Extend compilers

b) Extend languages

c) Add parallel language layer on top of sequential language

d) Define totally new parallel language and compiler system

Name____ID_______________________

Question 5 (12 marks; 10 minutes)

Compare the following parallel programming models:

a) Shared Memory Model

b) Threads Model

c) Message Passing Model

d) Data Parallel Model

Name______________________________ID__________________

Question 6 (8 marks; 10 minutes)

Explain how to implement Cache Coherence in case of Distributed Shared Memory.

Name____ID________________________

Question 7 (10 marks; 15 minutes)

Tell whether the following equations are parallelizable or non-parallelizable. Also show how to ***decompose*** the parts of the equations.

a)

```
do i=1,n
   a(i) = a(i-1) + b(i)
enddo
```

b)

```
w = a[0] * b[0];
for (i=1; i<N; i++) {
  c[i] = w;
  w = a[i] * b[i];
  d[i] = 2 * w;
}
```

c) $F(x) = c*M(x) + N(x)/n - O(x)$

d) $F(i+1) = F(i)/i + G(i) * H(i)$

e) $G(m,n) = m! / n!$

Name____ ID____________________

Question 8 (10 marks; 15 minutes)

Find the best way to parallelize the following code fragment, explain how by using pictures or diagrams where it is possible.

```
For i = 1 to 5
        For j=1 to 500,000
          For k=1 to 5
              $F(i, j, k) = (A(i)^k \% B(j))$
```

When A and B are functions that produce a 1D array and F is a function that produces a 3D array.

Name________________________ID________________

Question 9 (9 marks; 10 minutes)

From the following OpenMP code fragments, 1) explain how the code will be processed, 2) check if there is something wrong with the code, and 3) correct it or suggest better code fragment.

a)

```
if (z > max)
#pragma omp critical
  max = z;
```

b)

```
#pragma omp parallel private(i,j)
for (i = 0; i < NUMBER; i++) {
  sum_a += F(i);
  sum_b += G(i);
  if (sum_a == sum_b) {
#pragma omp single
    printf ("Exiting (%d)\n", i);
    break;
  }
}
```

c)

```
#pragma omp parallel sections
 {
   P();
   Q();
   R();
 }
```

Name____ID____________________________

Question 10 (15 marks; 20 minutes)

From the following MPI program, explain what it does and how it partitions the tasks to processes.

```
# include <stdlib.h>
# include <stdio.h>
# include <math.h>
# include <time.h>
# include "mpi.h"

int main ( int argc, char *argv[] );
int prime_number ( int n, int id, int p );
void timestamp ( void );

/**************************************************************/
int main ( int argc, char *argv[] )
{
 int i;
 int id;
 int ierr;
 int master = 0;
 int n;
 int n_factor;
 int n_hi;
 int n_lo;
 int p;
 int primes;
 int primes_part;
 double wtime;

 n_lo = 1;
 n_hi = 131072;
 n_factor = 2;

/*  Initialize MPI. */
 ierr = MPI_Init ( &argc, &argv );
```

Name____ID____________________

```
/*  Get the number of processes. */
 ierr = MPI_Comm_size ( MPI_COMM_WORLD, &p );
/*  Determine this processes's rank. */
 ierr = MPI_Comm_rank ( MPI_COMM_WORLD, &id );

 if ( id == master )
 {
  timestamp ( );
  printf ( "\n" );
  printf ( "'PRIME_MPI\n" );
  printf ( "  C/MPI version\n" );
  printf ( "\n" );
  printf ( "  An MPI example program to count the number of primes.\n" );
  printf ( "  The number of processes is %d\n", p );
  printf ( "\n" );
  printf ( "         N      Pi         Time\n" );
  printf ( "\n" );
 }

n = n_lo;

while ( n <= n_hi )
{
 if ( id == master )
 {
  wtime = MPI_Wtime ( );
 }
 ierr = MPI_Bcast ( &n, 1, MPI_INT, master, MPI_COMM_WORLD );

 primes_part = prime_number ( n, id, p );

 ierr = MPI_Reduce ( &primes_part, &primes, 1, MPI_INT, MPI_SUM,
master,
   MPI_COMM_WORLD );
```

Name____ ID____________________

```
      if ( id == master )
      {
        wtime = MPI_Wtime ( ) - wtime;
        printf ( "  %8d  %8d  %14f\n", n, primes, wtime );
      }
      n = n * n_factor;
    }

  /*  Terminate MPI. */
    ierr = MPI_Finalize ( );
  /*  Terminate. */

   if ( id == master )
   {
     printf ( "\n");
     printf ( "PRIME_MPI - Master process:\n");
     printf ( "  Normal end of execution.\n");
     printf ( "\n" );
     timestamp ( );
   }

   return 0;
 }
/*****************************************************************/
int prime_number ( int n, int id, int p )
{
  int i;
  int j;
  int prime;
  int total;

  total = 0;

  for ( i = 2 + id; i <= n; i = i + p )
```

Name____ID_________________________

```
  {
   prime = 1;
   for ( j = 2; j < i; j++ )
   {
    if ( ( i % j ) == 0 )
    {
     prime = 0;
     break;
    }
   }
   total = total + prime;
  }
  return total;
}
/*************************************************************/
void timestamp ( void )
{
# define TIME_SIZE 40

  static char time_buffer[TIME_SIZE];
  const struct tm *tm;
  size_t len;
  time_t now;

  now = time ( NULL );
  tm = localtime ( &now );

  len = strftime ( time_buffer, TIME_SIZE, "%d %B %Y %I:%M:%S %p", tm
);

  printf ( "%s\n", time_buffer );

  return;
# undef TIME_SIZE
}
```

Name____ID________________________

----End of Examination----

Pichaya Tandayya Lecturer

Name____ID________________________