Q1. (15 points)

Consider the problem of adding $n$ numbers. Assume that one processor can add two numbers in time $t_c$. How long will a processor take to add $n$ numbers? (5 points)

Now assume that 16 processor are available for adding $n$ numbers and that it is possible to divide the list of numbers into 16. Furthermore, a processor can pass on the result of an addition (in the form of a single number) to the processor next to it in time $t_w$. How long will it take in the following scenarios?

(a) All 16 processor are connected in a ring. (5 points)
(b) The 16 processors are connected in two rows of 8 processors each, and each processor can communicate with the processors next to it (5 points)

Q2. Consider the three parallel codes in the following figure. For each of them, answer the following questions (20 points).

(a) Is the code SPMD or MPMD? (5 points)
(b) Does the code exploit implicit parallelism or explicit parallelism? (5 points)
(c) Does the code exploit static parallelism or dynamic parallelism? (5 points)
(d) Does the code exploit data parallelism or control parallelism? (5 points)

for ( i = 0 ; i < N ; i ++ )  A[i] = b[i] * b[i+1] ;
for ( i = 0 ; i < N ; i ++ )  c[i] = A[i]  + A[i+1] ;

(a) A sequential code fragment

id = my_process_id ();
p = number_of_processes () ;
for ( i = id ; i < N ; i = i+p )  A[i] = b[i] * b[i+1] ;
barrier () ;
for ( i = id ; i < N ; i = i+p )  c[i] = A[i]  + A[i+1] ;

(b) Equivalent parallel code using library routines

my_process_id();
number_of_processes();
barrier();

A(0:N-1) = b(0:N-1) * b(1:N);
c = A(0:N-1) + A(1:N)

(c) Equivalent code in Fortran 90 using array operations

#pragma parallel
#pragma shared ( A, b, c )
#pragma local ( i )
{
#pragma pfor iterate (i=0; N ; 1)
  for ( i = 0 ; i < N ; i ++ )  A[i] = b[i] * b[i+1] ;
#pragma synchronize
#pragma pfor iterate (i=0; N ; 1)
  for ( i = 0 ; i < N ; i ++ )  c[i] = A[i] + A[i+1] ;
}

(d) Equivalent code using pragmas in SGI Power C

Q3. Consider again the figure shown in Q2 (15 points)
(a) What are the advantages of the code shown in Fig. b over the code shown in Fig. c?
(b) What are the advantages of the code shown in Fig. c over the code shown in Fig. b?
(c) Why is that the code shown in Fig. d is a compromise between the codes shown in Fig. b and Fig. c?

Q4. Write an pseudo SPMD parallel program to compute the maximum of an integer array using each of the following approaches: (15 points)

(a) The library approach (5 points).
(b) The new construct approach (5 points).
(c) The compiler directive approach (5 points).

Q5. Assume the communication network connecting the message passing multiprocessor system for solving the Sieve of Eratosthenes (using the data parallel approach taught in the class) supports concurrent message passing. Propose a faster method of communication which is better than \( \lambda(P-1) \). (20 points)

(a) Describe the faster communication algorithm (7 points)
(b) Describe its time expression (6 points)
(c) Analyze the execution time and speedup on 1, 2, 3, ..., 16 processors of the new Sieve algorithm that will use your communication scheme, assuming \( n = 1,000,000 \) and \( \lambda = 100X \). (7 points)
Q6. Answer the following questions on parallel execution modes of existing parallel computers: (15 points).

(a) What are the advantages of SIMD, MIMD, SPMD, and MPMD?
(b) How do you realize an MIMD parallel application on an SIMD architecture, give a concrete example to explain your answer.
(c) How do you realize an MPMD program on a computer which only support SPMD style? Give a concrete example to explain your answer.

SUBMISSION: WHAT, WHEN & HOW

(1) Provide your answers on in a Word file and email to the TA (saifullah.khalid@mavs.uta.edu).
(2)
(3) (2) This assignment is due on or before March 10, 2018.