

THE AUSTRALIAN NATIONAL UNIVERSITY  
*First Semester Examination – June 2007*

**COMP4300/6430**

**Parallel Systems**

*Study Period: 15 minutes*

*Time Allowed: 3 hours*

*Permitted Materials: Non-Programmable Calculator*

*Exam questions total 100 marks.*

*Answer ALL questions.*

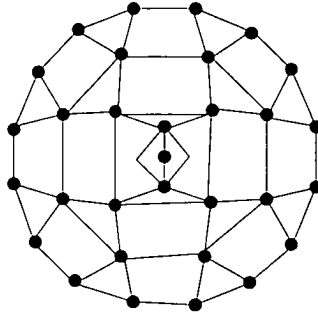
*Write your answers using a black or blue pen.*

*Point form is recommended for 'descriptive' questions.*

*Clarity and conciseness in answers will be highly valued;  
marks may be lost for supplying irrelevant information in  
answers.*

### Question 1 [24 marks]

“SpiderNet” is a new static interconnection proposed for distributed memory parallel computers. It comprises  $d$  concentric circles, where all nodes of a circle are connected in a ring, and each node in a circle is connected to 2 new nodes in the next circle. Each link in SpiderNet is bi-directional and there is cut-through routing. Each node in SpiderNet can only communicate down one channel at any given time. A SpiderNet that starts with the centre node (circle 0) and containing 4 subsequent circles is shown below:



(a) For each of the following terms give **BOTH** a general definition, and then the specific value of this quantity as a function of  $d$  for SpiderNet.

- i) Cost
- ii) Diameter
- iii) Arc connectivity
- iv) Bisection bandwidth

[8 marks]

(b) Design an optimal communication algorithm for a one-to-all broadcast on SpiderNet starting from the centre node. In your answer you should include:

- i) A detailed outline of the algorithm
- ii) The number of steps required as a function of  $d$
- iii) The total communication time as a function of message size

[12 marks]

(c) Outline how your algorithm for one-to-all broadcast on the SpiderNet would change if any node in SpiderNet could be the root node for the broadcast. [4 marks]

## Question 2 [18 marks]

You will be familiar with the following C code from Lab 2 in this course. It performs a binary radix sort for an array (`val`) of  $N$  integers where the maximum integer value is given by `MxInt`:

```
void integer_radix_sort(int val[], int N, int MxInt, int tmp[])
{
    int i,j,ilow,ihigh,nlevel;
    int *swap;
    /* Loop over at most 32 bits or while less than MxInt*/
    for (i=1, nlevel=0; i<MxInt && nlevel<32; i*=2, nlevel++){
        ilow=0;
        ihigh=0;
        /* Identify value of i'th bit in each data item and count zeros */
        for (j=0;j<N;j++)if (((val[j] >> nlevel) & 1) == 0)ihigh++;
        /* For i'th bit, move data items with zero ahead of those with one */
        /* Be sure to preserve relative order from previous bit sorts*/
        for (j=0;j<N;j++){
            if (((val[j] >> nlevel) & 1) == 0){
                tmp[ilow]=val[j];
                ilow++;
            } else {
                tmp[ihigh]=val[j];
                ihigh++;
            }
        }
        /* Swap pointers */
        swap=tmp;
        tmp=val;
        val=swap;
    }

    /* If uneven number of swaps, copy data back to original position*/
    if (nlevel%2 == 1)for (j=0;j<ihigh;j++)tmp[j]=val[j];
}
```

In the above `tmp` is an integer array of size  $N$  that is used as temporary storage. The code compiles and runs correctly on a single processor machine.

- (a) Explain how you would attempt to parallelise this code for a shared memory system using OpenMP. You are free to use additional storage if you believe this is necessary. You should provide pseudo code, i.e. you are not required to write syntactically correct C code or OpenMP pragmas, but you should make your intentions clear.

[12 marks]

- (b) Discuss how you might expect your code to perform as a function of input parameters, number of threads used, and the hardware you are running on.

[6 marks]

### Question 3 [17 marks]

In this question we will use the following notation:

$W(var)value$  : means write  $value$  to shared variable  $var$   
 $R(var)value$  : means read shared variable  $var$ , obtaining  $value$

For processor  $N$ , denoted as  $P_N$ , we represent a series of memory read and write operations as

Processor	First to last event list			
$P_N$	$R(x)0$	$W(x)1$	$W(y)2$	$R(z)2$

where time proceeds from left to right. Thus, in the above processor  $N$  first reads shared variable  $x$  from memory and obtains a value of 0, it then writes 1 to shared variable  $x$ , followed by writing 2 to shared variable  $y$ , and then finally it reads shared variable  $z$  and obtains a value of 2.

- (a) Define sequential memory consistency. [4 marks]
- (b) For each of the following show EITHER how the memory read and write operations can be ordered in time so that they comply with sequential memory consistency, OR state precisely why this is not possible. Before any of the following operations occur, all variables are initialised to 0.

- i) Two processors and one variable

Processor	First to last event list			
$P_0$	$W(x)0$	$W(y)1$	$W(x)1$	$R(x)3$
$P_1$	$R(x)1$	$W(x)2$	$R(x)1$	$W(x)3$

- ii) Three processors and three variables

Processor	First to last event list		
$P_0$	$W(x)1$	$R(y)0$	$R(z)1$
$P_1$	$R(x)1$	$W(y)2$	$R(z)0$
$P_2$	$R(x)0$	$R(y)0$	$W(z)1$

- iii) Three processors and two variables

Processor	First to last event list		
$P_0$	$W(x)1$	$W(y)2$	$R(x)2$
$P_1$	$R(x)1$	$W(x)2$	$R(y)0$
$P_2$	$R(y)2$	$W(y)3$	$R(x)1$

[9 marks]

- (c) If each processor in a multiprocessor system has a store buffer this usually means that the final system will not have a sequentially consistent memory model. Explain why this is the case. [4 marks]

#### Question 4 [10 marks]

The following C code uses MPI and originates from Lab 1 in this course. It compiles and runs to completion on the APAC SGI Altix system used in this course.

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include "mpi.h"

#define MAX_SIZE 8000000
int main( int argc, char **argv )
{
    int      rank, size, i, j, len;
    MPI_Status status;
    int      mbuf[MAX_SIZE];
    double t1,t2,time;

    MPI_Init( &argc, &argv );
    MPI_Comm_rank( MPI_COMM_WORLD, &rank );
    MPI_Comm_size( MPI_COMM_WORLD, &size );

    len=1;
    do {
        for (j=1; j<size; j++){
            for (i=0; i<100; i++){
                t1 = MPI_Wtime();
                if (rank == 0){
                    MPI_Send(mbuf, len, MPI_INTEGER, j, 0, MPI_COMM_WORLD );
                    MPI_Recv(mbuf, len, MPI_INTEGER, j, 1, MPI_COMM_WORLD, &status);
                }
                else if (rank == j){
                    MPI_Recv(mbuf, len, MPI_INTEGER, 0, 0, MPI_COMM_WORLD, &status);
                    MPI_Send(mbuf, len, MPI_INTEGER, 0, 1, MPI_COMM_WORLD);
                }
                t2 = MPI_Wtime()-t1;
                if (i == 0) time = t2;
                time = (t2 < time) ? t2 : time;
            }
            if (rank == j)printf("%10d%5i%15.5e \n",len,j,
                2.0*len*sizeof(int)/time);
        }
        len=len*2;
    }while(len<=MAX_SIZE);

    MPI_Finalize( );
    return 0;
}
```

Recall that the APAC SGI is a NUMA system with a single system image over groups of 32 processors.

- (a) What is the above code trying to measure? [5 marks]
- (b) Discuss what output you might expect to obtain if this program were run in single user mode on the APAC system (i.e. without any other users running on the machine at the same time). In your answer give only the gross features you expect to observe, paying particular attention to asymptotic limits, and what happens for different numbers of processors. (You are not expected to know the intricate details of the APAC SGI architecture.) [5 marks]

### Question 5 [8 marks]

Provide concise definitions (relevant to parallel systems!) for **FOUR** of the following

- i) SPMD Parallel Programming
- ii) Super-linear Speedup
- iii) Non-Blocking Communication Primitives
- iv) Race Condition
- v) Atomic Operation
- vi) Thread Safe Routine

### Question 6 [5 marks]

Differentiate between functional and domain/data decomposition. Which do you think offers the greater potential for parallel speedup? Why?

### Question 7 [5 marks]

Explain what is meant by “one-sided communications”, and outline how this is supported in MPI-2.

### Question 8 [5 marks]

What is the difference between an object and a page based software distributed shared memory system. Which do you think is best? Why?

**Question 9 [4 marks]**

Moore's law is widely quoted in the context of high performance computing. For Grid computing two other "laws" are arguably more relevant, one attributed to Kryder and the other attributed to Foster. What are Kryder's and Foster's laws, and why are they relevant to Grid computing?

**Question 10 [4 marks]**

It is possible for parallel search algorithms to obtain both acceleration anomalies as well as detrimental anomalies. In an acceleration anomaly the observed speedup is substantially more than what might be expected when using  $p$  processors. While in a detrimental anomaly the search takes substantially longer to perform when run in parallel compared to when run sequentially. Suggest potential reasons why these two different situations might occur for parallel search algorithms.

---