

---

## UNIVERSITI SAINS MALAYSIA

First Semester Examination  
2014/2015 Academic Session

December 2014/January 2015

### **CST331 – Principles of Parallel & Distributed Programming** *[Prinsip Pengaturcaraan Selari & Teragih]*

Duration : 2 hours  
*[Masa : 2 jam]*

---

#### **INSTRUCTIONS TO CANDIDATE:**

##### ***[ARAHAN KEPADA CALON:]***

- Please ensure that this examination paper contains **FOUR** questions in **NINE** printed pages before you begin the examination.

*[Sila pastikan bahawa kertas peperiksaan ini mengandungi **EMPAT** soalan di dalam **SEMBILAN** muka surat yang bercetak sebelum anda memulakan peperiksaan ini.]*

- Answer **ALL** questions.

*[Jawab **SEMUA** soalan.]*

- You may answer the questions either in English or in Bahasa Malaysia.

*[Anda dibenarkan menjawab soalan sama ada dalam bahasa Inggeris atau bahasa Malaysia.]*

- In the event of any discrepancies, the English version shall be used.

*[Sekiranya terdapat sebarang percanggahan pada soalan peperiksaan, versi bahasa Inggeris hendaklah diguna pakai.]*

1. (a) What are the differences of parallel computing and distributed computing?  
(6/100)
- (b) Name the **three (3)** modifications made to the von Neumann machine to increase the CPU performance.  
(3/100)
- (c) Define Flynn's Taxonomy.  
(3/100)
- (d) What is the use of Flynn's Taxonomy?  
(1/100)
- (e) State the **four (4)** schemes of Flynn's Taxonomy.  
(4/100)
- (f) What are the advantages of private cache and shared cache?  
(4/100)
- (g) Give **two (2)** sources of overhead along with explanation.  
(4/100)
2. (a) Differentiate how coordination is being handled in Shared Memory platform and Distributed Memory platform.  
(4/100)
- (b) What are the phases involves in designing parallel program?  
(4/100)
- (c) Give any **two (2)** points of checklists for the first phase of designing a parallel program.  
(4/100)
- (d) What is the important issue to be identified in the second phase of designing a parallel program?  
(3/100)
- (e) Describe the relationship of process and threads. You may accompany your answer with illustration.  
(4/100)

- (f) Differentiate process and threads in terms of address space and interactions. (6/100)
- (g) Define race condition, critical region and mutual exclusion. (6/100)
3. (a) List all synchronization objects used in OpenMP and pthreads to resolve race condition. (5/100)
- (b) What are the **three (3)** components of OpenMP API? (3/100)
- (c) Name and describe **two (2)** worksharing constructs in OpenMP. (4/100)
- (d) Given two sequential codes as below, answer the following questions:

Code A	Code B
<pre> #define CHUNKSIZE 100 #define N 1000  main () {     int i, chunk;     float a[N], b[N], c[N];      for (i=0; i &lt; N; i++)         a[i] = b[i] = i * 1.0;     chunk = CHUNKSIZE;      for (i=0; i &lt; N; i++)         c[i] = a[i] + b[i]; } </pre>	<pre> #define N 1000  main () {     int i;     float a[N], b[N], c[N], d[N];      for (i=0; i &lt; N; i++) {         a[i] = i * 1.5;         b[i] = i + 22.35;     }      for (i=0; i &lt; N; i++)         c[i] = a[i] + b[i];      for (i=0; i &lt; N; i++)         d[i] = a[i] * b[i]; } </pre>

- (i) Determine the type of parallelism of Code A and Code B. (2/100)
- (ii) Justify your answer in 3(d)(ii). (6/100)
- (iii) What are the two OpenMP work-sharing constructs that are suitable to be used for Code A and Code B, respectively? (2/100)

- (e) Write the complete parallel version in OpenMP of Code A and Code B using the appropriate OpenMP directives from the following list:

```
#pragma omp for schedule(dynamic,chunk) nowait
#pragma omp parallel shared(a,b,c,chunk) private(i)
#pragma omp parallel shared(a,b,c,d) private(i)
#pragma omp sections nowait
#pragma omp section
#include omp.h
```

(10/100)

- (f) Define data scoping of shared and private variables in OpenMP.

(2/100)

4. (a) Define Collective Communication and Point to Point Communication in MPI.

(4/100)

- (b) Write a complete parallel version in MPI of the following sequential code using MPI\_Scatter.

```
#include <stdio.h>
#define SIZE 4

main(int argc, char *argv[]) {
int numtasks, rank, sendcount, recvcount, source;
float sendbuf[SIZE][SIZE] = {
    {1.0, 2.0, 3.0, 4.0},
    {5.0, 6.0, 7.0, 8.0},
    {9.0, 10.0, 11.0, 12.0},
    {13.0, 14.0, 15.0, 16.0} };
float recvbuf[SIZE];

if (numtasks == SIZE) {
    source = 1;
    sendcount = SIZE;
    recvcount = SIZE;

    printf("rank= %d Results: %f %f %f %f\n",rank,recvbuf[0],
        recvbuf[1],recvbuf[2],recvbuf[3]);
    }
else
    printf("Must specify %d processors. Terminating.\n",SIZE);
}
```

Hint: you may want to use the following statement somewhere in your code  
MPI\_Scatter(sendbuf,sendcount,MPI\_FLOAT,recvbuf,recvcount,MPI\_FLOAT,  
source,MPI\_COMM\_WORLD);

(4/100)

(c) Describe the role played by the following CUDA calls:

- (i) `cudaMalloc()`
- (ii) `cudaFree()`
- (iii) `cudaMemcpy()`

(4/100)

**KERTAS SOALAN DALAM VERSI BAHASA MALAYSIA**

[CST331]

- 6 -

1. (a) Apakah perbezaan antara perkomputeran selari dengan perkomputeran teragih?  
(6/100)
- (b) Namakan **tiga (3)** pengubahsuaian yang dibuat kepada mesin von Neumann untuk meningkatkan prestasi CPU.  
(3/100)
- (c) Takrifkan Taksonomi Flynn.  
(3/100)
- (d) Apakah kegunaan Taksonomi Flynn?  
(1/100)
- (e) Nyatakan **empat (4)** skema dalam Taksonomi Flynn.  
(4/100)
- (f) Apakah kelebihan memori singgahan tersendiri dan memori singgahan terkongsi?  
(4/100)
- (g) Berikan **dua (2)** punca overhed berserta huraian.  
(4/100)
  
2. (a) Bezakan bagaimana koordinasi dilaksanakan di dalam platform memori terkongsi dan memori teragih.  
(4/100)
- (b) Apakah fasa-fasa yang terlibat dalam pembentukan program selari?  
(4/100)
- (c) Berikan mana-mana **dua (2)** fakta dari senarai semak fasa pertama.  
(4/100)
- (d) Apakah isu penting yang perlu dikenalpasti dalam fasa pertama?  
(3/100)

- (e) Huraikan hubungan antara proses dan bebenang. Anda boleh menggunakan ilustrasi untuk menyokong jawapan anda. (4/100)
- (f) Bezakan antara proses dan bebenang dari segi alamat dan interaksi. (6/100)
- (g) Takrifkan perebutan sumber, bahagian kritikal dan saling pengecualian. (6/100)
3. (a) Senaraikan semua objek penyelarasan yang digunakan dalam OpenMP dan pthreads untuk menyelesaikan perebutan sumber. (5/100)
- (b) Apakah komponen-komponen API OpenMP? (3/100)
- (c) Nama dan huraikan **dua (2)** binaan perkongsian kerja dalam OpenMP. (4/100)
- (d) Diberikan dua kod atur cara berjujukan seperti di bawah, jawab soalan-soalan berikut:

Kod A	Kod B
<pre> #define CHUNKSIZE 100 #define N      1000  main () {  int i, chunk; float a[N], b[N], c[N];  for (i=0; i &lt; N; i++)     a[i] = b[i] = i * 1.0;     chunk = CHUNKSIZE;  for (i=0; i &lt; N; i++)     c[i] = a[i] + b[i];  } </pre>	<pre> #define N      1000  main () {  int i; float a[N], b[N], c[N], d[N];  for (i=0; i &lt; N; i++) {     a[i] = i * 1.5;     b[i] = i + 22.35; }  for (i=0; i &lt; N; i++)     c[i] = a[i] + b[i];  for (i=0; i &lt; N; i++)     d[i] = a[i] * b[i];  } </pre>

(i) Tentukan jenis penyelarian untuk Kod A dan Kod B.

(2/100)

(ii) Jelaskan jawapan anda dalam 3(d)(ii).

(6/100)

(iii) Apakah **dua (2)** binaan perkongsian kerja yang sesuai untuk Kod A dan Kod B?

(2/100)

(e) Tulis atur cara selari dalam OpenMP untuk Kod A dan Kod B menggunakan arahan-arahan OpenMP bersesuaian dari senarai berikut:

```
#pragma omp for schedule(dynamic, chunk) nowait
#pragma omp parallel shared(a, b, c, chunk) private(i)
#pragma omp parallel shared(a, b, c, d) private(i)
#pragma omp sections nowait
#pragma omp section
#include omp.h
```

(10/100)

(f) Takrifkan skop data pemboleh ubah terkongsi dan tersendiri OpenMP.

(2/100)

4. (a) Takrifkan Komunikasi Berkelompok dan Komunikasi Titik ke Titik dalam MPI.

(4/100)



- (b) Tulis atur cara selari yang lengkap dalam MPI untuk kod atur cara berjujukan berikut menggunakan MPI\_Scatter.

```

#include <stdio.h>
#define SIZE 4

main(int argc, char *argv[]) {
int numtasks, rank, sendcount, recvcount, source;
float sendbuf[SIZE][SIZE] = {
    {1.0, 2.0, 3.0, 4.0},
    {5.0, 6.0, 7.0, 8.0},
    {9.0, 10.0, 11.0, 12.0},
    {13.0, 14.0, 15.0, 16.0} };
float recvbuf[SIZE];

if (numtasks == SIZE) {
    source = 1;
    sendcount = SIZE;
    recvcount = SIZE;

    printf("rank= %d Results: %f %f %f %f\n",rank,recvbuf[0],
        recvbuf[1],recvbuf[2],recvbuf[3]);
    }
else
    printf("Must specify %d processors. Terminating.\n",SIZE);
}

```

Petunjuk: anda mungkin ingin gunakan kenyataan di bawah di dalam atur cara anda

```

MPI_Scatter(sendbuf,sendcount,MPI_FLOAT,recvbuf,recvcount,MPI_FLOAT,
source, MPI_COMM_WORLD);

```

(4/100)

- (c) Huraikan peranan yang dimainkan oleh panggilan CUDA berikut:

- (i) cudaMalloc()
- (ii) cudaFree()
- (iii) cudaMemcpy()

(4/100)