The University of Melbourne
COMP90025 Parallel and Multicore Computing
Semester 2, 2020 Final Examination

School of Computing and Information Systems

COMP90025 Parallel and Multicore Computing

Reading Time: 15 minutes

Writing Time: 2 hours

Open Book Status: Open Book - Online.

Authorized Materials: You are permitted to make use of material available on the Web in general, including course notes, recordings, materials on LMS, and research literature and websites.

This paper has 4 pages including this page

Identical Examination Papers: The examination paper is available as either a PDF or Word document. The documents have identical content and differ only in minor formatting ways.

Common Content: none

Instructions to students:

• The total marks for this paper is 40.
• The examination is worth 40% of the subject assessment.
• Attempt all questions - partial credit is available.
• You are required to submit a single document that contains all of your answers. Acceptable document formats include PDF, Word, or a plain text file. Please include your student number at the top of the first page of the document, prepare your answers in the same order as the questions given in the exam paper, and number your answers with the same numbering as in the exam paper.
• You may type your answers, include diagrams/images if you wish, and you may draw and scan images to include in the document. You may prepare all of your answers on paper, using handwritten answers if you wish, and simply scan all of your answers into a single PDF document and submit that way if you wish. Make sure that all scanned material and diagrams/images are clearly legible.
• You must adhere to the Academic Integrity Declaration. In particular, in each of your answers, cite all materials that you sourced information from.
• Unless you have a special time allowance, you have an examination time of 2 hours and 15 minutes to complete and upload your answers document, for it to be considered "on time". After the examination time, you have an additional late submission period of 30 minutes to submit your answers document – submissions in this period will be considered "late". The late submission period cannot be used as additional exam time. The Subject Coordinator may request evidence of technical issues or a late penalty may apply. After this period you will not be able to upload your answers document.
• You may submit multiple times (unlimited), but only the final (last) submission will be looked at for assessment. If the final submission is within the late period then your submission will be considered late, as above. You will not be able to revert to an earlier submission.

Paper to be held by Baillieu Library: yes

continued on next page

Q.1. (a) [**6 marks**] Define *Amdahl's Law* and *Gustafson's Law*. In **your own words**, discuss the differences between the two laws and explain how each law is useful. Consider a cluster like the SNOWY partition on Spartan, that you used for your project work, and the problem of *multiple sequence alignment* that you solved in Project 2B. Discuss how/where each of the laws applies (or not) in this case.

(b) [**4 marks**] The *granularity* of a task in a parallel program can be defined as the task's total *computation time* divided by the task's total *communication time*. Discuss the consequences of granularity in general, and with respect to *parallelism* and *speedup*.

Q.2. (a) [**2 marks**] Suppose you have an unsorted array of n elements where each element in the array is either a 0 or a 1. Briefly explain an EREW PRAM *optimal* method for sorting the array.

(b) [**3 marks**] Suppose you have an array of size n, where exactly 2 of the elements in the array have a value, and all other elements are empty, i.e. have value $\emptyset$. Show how to compact the array such that the 2 values are moved to the first two elements of the array (in the same order they appear originally), using a CREW PRAM, in *constant time*, using any number of processors that you wish.

(c) [**5 marks**] Show how to sort an array of n integers using n processors on a CREW PRAM in $O(\log n \log\log n)$ time. Make sure to show how to do all aspects of your solution.

Q.3. (a) [**3 marks**] Explain how two $n \times n$ matrices can be multiplied by n^3 processors in time $O(\log n)$ on an EREW PRAM. Explain how it can be done in *constant time* using an appropriate PRAM model.

(b) [**4 marks**] Consider an array of $n = 2^k$ elements on processor 1 of an n processor message passing machine where processors are numbered $1, 2, \ldots, n$. A scatter operation requires processor $i > 1$ to receive element i. Assume that sending a message requires a startup cost of time t_s plus time t_m per element. E.g. sending a message with 5 elements takes time $t_s + 5t_m$ (the message is completely received after this time). Each processor can be sending at most 1 message at a time. A processor can not send and receive messages at the same time. Assume that there are no other delays in the system and that any number of processors can be sending a message at the same time without performance penalty.

Method 1 for scattering the array requires processor 1 to send each element, sequentially one after the other.

Method 2 for scattering the array uses a scatter tree, where half of the elements are first sent to processor $(n/2+1)$, and recursively the elements are scattered in parallel.

If $t_m = a\,t_s$, then for what values of $a > 0$ will Method 2 provide a speed up greater than unity over Method 1? What value of a will give a speed up of approximately $\sqrt{n}$?

continued on next page

(**c**) [**3 marks**] For a hypercube with $n = 2^t$ nodes, $t \geq 0$, the number of nodes at distance $0 \leq i \leq t$ is given by $\binom{t}{i}$ (t choose i). Prove the theorem by induction on t. (HINT: you may use the additive identity $\binom{x}{y} = \binom{x-1}{y} + \binom{x-1}{y-1}$.)

Q.4. (**a**) [**4 marks**] Consider the OpenMP code that follows.

- Which loops are parallelized?
- Guess how many threads there are likely to be if it is run on a machine with C cores. What extra information would you need to know to be sure of that answer?
- What one-line change could increase the amount of parallelism?
- In general, what is the disadvantage of increasing the parallelism too much?

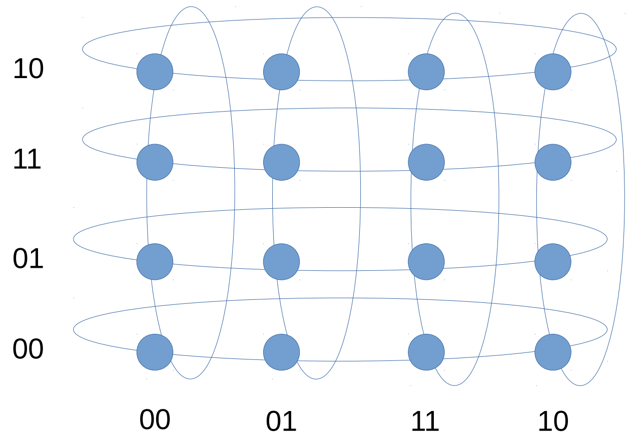
```
#pragma omp parallel private(i, j, k)
{
    #pragma omp for
    for(i = 0; i < M; i++ ) {
        for(j = 0; j < N; j++) {
            D[i][j] = 0;
            for(k = 0; k < P; k++){
                D[i][j] += A[i][k] * B[k][j];
            }
        }
    }
}
```

(**b**) [**2 marks**] Given the following CUDA code, identify three errors which either make the code invalid or fail to use parallelism. For each, explain why the current code is wrong, and how you would fix it.

```
__device__ square (float *A, float *B)
{
       int i;
       for (i = 0; i < 10; i++)
           B[i] = A[i] * A[i];
}

int main ()
{
    float A[10] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
    float B[10];

    square<<<5, 1>>> (A, B)

    return 0;
}
```

continued on next page

(**c**) [**4 marks**] Imagine you manage a supercomputer with a 256-node hypercube interconnect, with nodes numbered 00000000 to 11111111 such that any pair of nodes whose numbers differ by a single bit are connected by a link. Other topologies can be implemented within a hypercube by ignoring some of the links. For example, a 4×4 mesh torus can be implemented in a 16-node hypercube as:



You are asked to allocate multiple jobs to this computer simultaneously, each with its own topology. Which of the following combinations are possible? Explain how the nodes would be arranged, and justify the claim that all of the required links are present in the supercomputer.

i. 256-node ring, alone

ii. 16×16-node mesh, alone

iii. 128-node 2-D mesh, 72-node 2-D mesh, 56-node 2-D mesh

iv. 128-node 2-D mesh, 70-node 2-D mesh, 55-node 2-D mesh

END OF EXAMINATION