The University of Melbourne
COMP90025 Parallel and Multicore Computing
Semester 2, 2016 Final Examination

Department of Computing and Information Systems
COMP90025 Parallel and Multicore Computing
Reading Time: 15 minutes
Writing Time: 3 hours

Open Book Status: Closed Book

This paper has 3 pages including this page

Identical Examination Papers: none
Common Content: none

Authorized Materials:
No materials are authorized.

Instructions to invigilators:
No papers may be taken from the exam room.

Instructions to students:
All answers are to be written in the script book(s) provided.
Attempt all questions - partial credit is available.
The examination is worth 60% of the subject assessment.

Paper to be held by Baillieu Library: yes

continued on next page

Q.1. **(a)** [**4 marks**] Given a sequential algorithm for a problem that runs in $T(n)$ time, and a parallel algorithm using $p(n)$ processors that runs in $t(n)$ time, define what is meant by the parallel algorithm being optimal.

(b) [**6 marks**] Brent's Principle states that if an algorithm involving a total of x operations can be performed in t time on a PRAM with sufficiently many processors, then it can be performed in time $t + \frac{x-t}{p}$ on a PRAM with p processors.

i. Show how to derive Brent's Principle.

ii. Consider an algorithm that involves $\mathcal{O}(n)$ operations in total and $\mathcal{O}(\log n)$ time. Using Brent's Principle or otherwise, what value of p provides optimality?

Q.2. **(a)** [**5 marks**] Show how to *optimally* compute the prefix sum of n numbers on an EREW PRAM. Make sure to show that your approach is optimal.

(b) [**5 marks**] Explain the *ring termination* algorithm that can handle restarting processes.

Q.3. **(a)** [**3 marks**] Consider a hypercube of size 2^t nodes, where each node contains a number. Write a hypercube algorithm that adds up all of the numbers with the result being stored at all nodes, and taking $\mathcal{O}(t)$ steps.

(b) [**3 marks**] Show by structural induction that a linear array of length 2^t is contained in a hypercube of size 2^t.

(c) [**4 marks**] Define the following terms with respect to graph embeddings:

i. dilation

ii. expansion

iii. load

iv. congestion

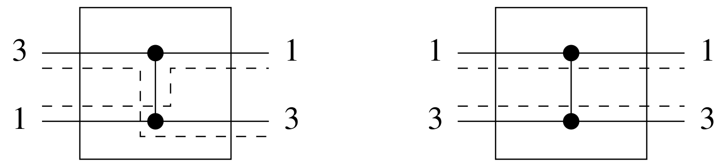
Q.4. **(a)** [**8 marks**] Consider a sorted array of n unique integers. For a given integer x, the search problem is to determine if the array contains x. This can be done sequentially using $\mathcal{O}(\log n)$ time (binary search). Show how search can be done in $\mathcal{O}(\log_p n)$ time using p processors on a CREW PRAM.

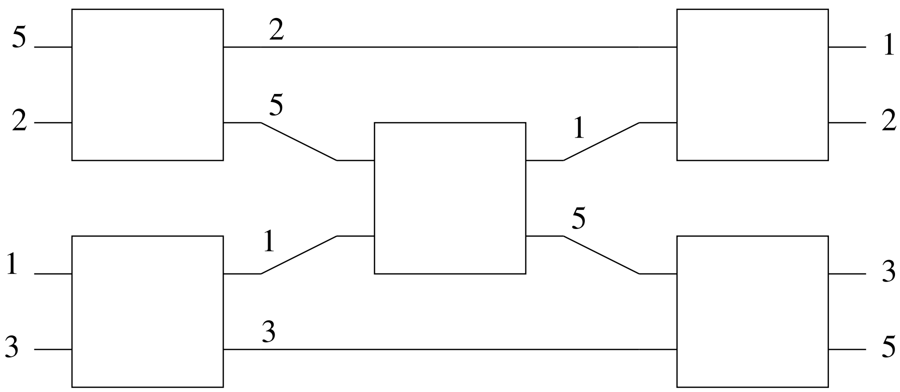
(b) [**8 marks**] Consider a mesh of size $\sqrt{n} \times \sqrt{n}$ nodes, where each node of the mesh contains an integer. The uniqueness problem is to determine whether all of the integers are unique. Consider a parallel algorithm that results in every node of the mesh knowing the result of the uniqueness problem. Show how this can be done in $\mathcal{O}(\sqrt{n} \log n)$ steps.

(c) [**6 marks**] A *comparator* is like a switch, however a comparator always routes the smallest of its input values to the upper output. A comparator is shown in detail in the following figure for both possibilities where the numbers 1 and 3 are provided as example input. The figure also shows a *3 stage sorting network*

 continued on next page

(using comparators) that takes 4 inputs and sorts them, producing the sorted result at the 4 outputs. Explain how to construct an n-input sorting network that uses $\mathcal{O}(\log^2 n)$ stages and $\mathcal{O}(n \log^2 n)$ comparators in total, and draw an example of such a sorting network on 8 inputs.





(d) [**8 marks**] Consider the Sieve of Eratosthenes algorithm, shown below, that returns an array of length n in which the i-th position is set to true if i is a prime and to false otherwise.

```
PRIMES(n):
let A be an array of length n
set all but the first element of A to TRUE
for i from 2 to √n
    begin
        if A[i] is TRUE
        then set all multiples of i up to n to FALSE
    end
```

The PRIMES algorithm above runs in $O(n \log\log n)$ sequential time. Show how to parallelize the PRIMES algorithm using $\mathcal{O}(n)$ processors on a PRAM, that takes $\mathcal{O}(\log\log n)$ time. The following fact may help you:

$$\sum_{p \in prime}^{n} \frac{1}{p} = \mathcal{O}(\log\log n)$$

END OF EXAMINATION