Design & Analysis of Parallel Algorithms: Exercise Sheet 2

Please be sure that you have read, understood and adhered to the School and University guidelines on late submission of coursework and academic misconduct, which can be found via the course webpage.
This sheet accounts for 10% of the course final mark. You should submit your work electronically, in pdf, by **4pm on Monday 25th November 2013**, *using the DICE command*

```
submit dapa 2 dapaEx2.pdf
```

If you are **not** *a School of Informatics student then you may submit your work by e-mail to* `mic@inf.ed.ac.uk`. *The best marks will be awarded for simple, correct and clearly argued content. It is important that you use the filename given, and that your document is in pdf.*

1. Design an algorithm for the n processor hypercube which solves the following problem. A string of n characters has been pre-distributed across the hypercube, so that each processor initially stores a single character. We can think of this as a distributed array A, where $A[i]$ is stored by processor i. The goal is to find the largest value k such that $A[i] == A[n-1-i]$ for all $0 \leq i \leq k$. In other words, we are looking for the longest portion from the start of the array which is also present, but reversed, at the end of the array, with our result being the index k of the last character which forms part of this portion. For example, given the string "abcxycba" the answer will be 2, because "abc" is the reverse of "cba" and the index of 'c' is 2. In the extreme case of "aaaaaaaa" the answer will be 7, because the whole string is its own reverse. At the other extreme, for a string such as "abcdefgh", we define the answer to be -1, indicating that there is no matched portion at all. You should make your algorithm as asymptotically fast as possible.

   Analyse the asymptotic run time of your algorithm, assuming the usual model of store-and-forward routing on the hypercube. Comment on your algorithm's cost optimality.

   (60 *marks*)

2. (a) Sketch the sequence of states through which the (non-bitonic) sequence 1, 4, 7, 5, 3, 6, 8, 2 is transformed by an appropriately sized bitonic mergesort network (ie the full network which sorts arbitrary input sequences). Show the output of each column of comparators, but don't draw the network itself.

   (b) Consider a ring connected parallel computer with p processors, where p is a power of 2. Suppose we devise an n item sorting algorithm (where $n = p$) for this machine by mapping wire i of the standard bitonic mergesort network (as labelled in the course overheads) to processor i then implementing the standard bitonic mergesorting algorithm, with appropriate communications and computations taking the place of compare-exchange steps. Assume a simple store-and-forward cost model for communications, where passing a single integer between ring neighbours costs the same unit time as each internal comparison.

[Continued on the next page]

i. Derive the algorithm's asymptotic run time, explaining your expressions carefully.

ii. Now consider a similar algorithm, but with n some multiple of p and each compare-exchange step replaced by a compare-split step in the usual way, and an initial sequential sort of the block of items stored by each processor. Derive the asymptotic run time of the resulting algorithm, explaining your expressions carefully. Discuss the algorithm's cost-optimality.

(40 $marks$)