**CMPE 300 ANALYSIS OF ALGORITHMS** 24.5.2016

###### FINAL

1. Write in pseudocode a CRCW PRAM algorithm that sorts a list L[1:n] *using less than n2 processors*. The complexity must be θ(1). You can use any strategy to resolve concurrent writes (e.g. a combining strategy). Assume that the elements in the list are distinct.

*Hint*: You can use a temporary array of size n.

*Hint*: Think about a semigroup operation algorithm on CRCW PRAM.

Explain which strategy the algorithm uses for concurrent writes; explain in detail that it is a proper strategy for this algorithm. Calculate W(n), S(n), C(n), and E(n) clearly. Is the algorithm cost optimal, cost efficient, time efficient?

1. Consider the following CRCW PRAM program with a priority-write scheme for concurrent writes (assuming the smaller the index, the higher the priority). Notice that in this scheme, when a write conflict occurs, the processor with the highest priority writes.

for 1 ≤ i ≤ n do in parallel

for j=0 to () - 1 do

x = M[i]

y = M[i+2j]

if (i < 5) then

M[i+2j] = x+y

else

M[2j] = x-y

endif

endfor

end in parallel

Consider the case when n=8, p=n=8, and array M is initially given as follows (all the positions after M[8] are 0):

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Position | M[1] | M[2] | M[3] | M[4] | M[5] | M[6] | M[7] | M[8] | M[9] | M[10] | M[11] |
| Value | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 0 | ... |

Trace the execution and show the contents of the M array after the algorithm terminates. (You must show all the execution in a systematic way. No points will be given if only the output is shown.)

1. Consider the problem of guessing a number: The user selects a number between 1 and n and an algorithm tries to guess it by asking “Is the number x?” type questions and getting one of the answers “yes”, “no, less than x”, “no, greater than x” from the user. Prove the lower bound of this problem by using adversary arguments. Explain *clearly*.

*Note*: Be careful of not using a particular algorithm or logic. You must use the adversary technique. That is, state clearly what the adversary does during the execution.

1. Suppose that we have two arrays A[1:n] and B[1:n]. We want to determine whether the two arrays are the same or not; i.e. whether A[i]=B[i], for 1≤i≤n.
2. Write in pseudocode a Monte Carlo algorithm for this task. Explain the properties of the algorithm (bias, complexity, etc.). (Your algorithm must be asymptotically faster than the sequential algorithm. Don not use any repetition algorithm.)
3. Write in pseudocode a Las Vegas algorithm for this task. Explain the properties of the algorithm (complexity, etc.). (Express the complexity with a general formula.)

*(continued on next page)*

*Some equations that you may use:*

*  , for x≠1
*  , for x≠1
* For a binary tree T with L leaf nodes, , where LPL(T) is the leaf path length of T.

*Notes:*

* Where pseudocode is required, the syntax of the pseudocode must be strictly followed. No points will be given if the syntax is not followed or any other language (e.g. C) is used.
* Questions 1,4:25 points
* Time: 2:00 hours
* Close notes and books

Bir kırlangıç bir su birikintisi bir parça gök.

Bir şiirden düşmüş olmalı bunlar.

Böyle diyordu yoldan geçen biri.