CSE 4502/5717 Big Data Analytics. Fall 2022

Model Exam IV Solutions

1. Here is an algorithm:

   repeat
   Flip an n-sided coin to get i;
   Flip an n-sided coin to get j;
   if $i \neq j$ **and** $A[i] = A[j]$ **then output** $A[i]$ and **quit**;
   forever

   Analysis: Call one execution of the **repeat** loop as a basic step. Probability of success in one basic step is $\frac{\frac{n}{2}\left(\frac{n}{10}-1\right)}{n^2} \approx \frac{1}{20}$. This means that the probability of failure in one basic step is $\leq \frac{19}{20}$. Probability of failure in k basic steps is $\leq \left(\frac{19}{20}\right)^k$.

   We want the above probability to be $\leq n^{-\alpha}$. This will happen if $k \geq \frac{\alpha \log n}{\log(20/19)}$. This in turn means that the run time of the above algorithm is $\widetilde{O}(\log n)$.

2. Let $C = AB$. By definition, $C_{i,j} = \bigvee_{k=1}^{n} A_{i,k} \wedge B_{k,j}$, for $1 \leq i, j \leq n$. Let the processors be labelled $P_{i,j}^k$, $1 \leq i, j, k \leq n$. We can assign n processors to compute each element in the product as follows:

   ```
   1)   for 1 ≤ i, j ≤ n in parallel do
   2)       /* Processors P^1_{i,j}, P^2_{i,j}, ..., P^n_{i,j} compute C_{i,j}. */
   3)       for 1 ≤ k ≤ n in parallel do
   4)           Processor P^k_{i,j} computes c^k_{i,j} = A_{i,k} ∧ B_{k,j};
   5)           Processors P^1_{i,j}, P^2_{i,j}, ..., P^n_{i,j} compute C_{i,j} = c^1_{i,j} ∨ c^2_{i,j} ∨ ··· ∨ c^n_{i,j};
   ```

   In the above algorithm, step 4 takes one unit of time. In step 5 we have to compute the Boolean OR of n bits. This can be done in $O(1)$ time using n common CRCW PRAM processors, as was shown in class. Thus, the whole algorithm takes $O(1)$ time.

3. Have an output buffer of size $\frac{BD}{C}$ for each value in the range $[1, C]$. Bring BD elements at a time from the disks into the main memory. Distribute these keys to the buffers based on the key values. Repeat this process. When any buffer is full, write these $\frac{BD}{C}$ elements into the disks. One possibility is to write them in $\frac{D}{C}$ disks (a block each). In the disks, we will grow C runs in separate regions. After one read pass through the data, X has been sorted into C runs in the disks. Note that the number of write passes is $O(C)$.

   Now we have to write the runs contiguously in the disks. This can be done in one more pass through the data.

4. Construct a suffix tree Q for S in $O(n)$ time. Followed by this, perform an in-order traversal of Q to label every internal node u of Q with an integer $c[u]$ such that $c[u]$ is the number of leaves in the subtree rooted at u.

Now, perform one more traversal through Q to mark every node whose string depth is $\geq k$. In one additional traversal through Q identify the node u that is marked and whose $c[u]$ is the largest. Finally, output any substring of the path label of u whose length is k.

Clearly, the total run time of the algorithm is $O(n)$.

5. Consider a complete binary tree with k leaves where each leaf has one of the input polynomials. Perform a computation up the tree as follows. Each internal node multiplies the two children polynomials and sends the result to its parent. When the root completes its operation we get the product of the k ploynomials. There are $\log k$ levels in the tree and the time spent at each level is $O(n \log n)$. Thus the run time of the algorithm is $O(n \log n \log k)$.

6. The loss function is $L(w_1, w_2) = (w_2 - 6)^2 + (w_1 - 2)^2 + (w_1 + w_2 - 5)^2 + (w_1 + 2w_2 - 10)^2$ $= 3w_1^2 + 6w_2^2 + 6w_1w_2 - 34w_1 - 62w_2 + 165$. We want to have: $\frac{\partial L}{\partial w_1} = 0$ and $\frac{\partial L}{\partial w_2} = 0$.
$\frac{\partial L}{\partial w_1} = 0$ implies that $3w_1 + 3w_2 = 17$ and $\frac{\partial L}{\partial w_2} = 0$ implies that $3w_1 + 6w_2 = 31$. Solving these two equations, we get: $w_1 = 1$ and $w_2 = \frac{14}{3}$.