CSE 5500 Algorithms

Fall 2018 Exam III (model)– Solutions

1. Let M be the adjacency matrix and let $|V| = n$. Each of the n^2 processors is assigned one entry of M. These n^2 processors then compute the Boolean AND of the n^2 bits of M in $O(1)$ time.

2. Perform a prefix sums computation on $r_1, r_2, \ldots, r_n$ to get $r'_1, r'_2, \ldots, r'_n$. This takes $O(\log n)$ time using $\frac{n}{\log n}$ CREW PRAM processors. Now we compute the outputs as follows: $s_i = r'_{i+k-1} - r'_{i-1}$, for $i = 1, 2, \ldots, (n-k+1)$ and $s_i = r_i$ for $i = (n-k+2), (n-k+3), \ldots, r_n$. This updating can be done in $O(1)$ time using n CREW PRAM processors. Using the slow-down lemma, this updating can also be done in $O(\log n)$ time using $\frac{n}{\log n}$ CREW PRAM processors.

   As a result, the entire algorithm takes $O(\log n)$ time using $\frac{n}{\log n}$ CREW PRAM processors.

3. Without loss of generality assume that $n = 2^k$ for some integer k. Consider a full binary tree of height k (where the root is at level 0 and there are 2^k leaves). At the root we have the input sequence $X = k_1, k_2, \ldots, k_n$ that we are interested in sorting. We find the median M of X in $O(\log n)$ time the total work done being $O(n)$. We partition X into X_1 and X_2 based on M. Specifically, $X_1 = \{q \in X | q < M\}$ and $X_2 = \{q \in X | q > M\}$. We have to output X_1 in sorted order, followed by M, followed by X_2 in sorted order. Partitioning of X can be done in $O(\log n)$ time and $O(n)$ work using prefix computations. X_1 and X_2 form the two children of the root.

   We find the median M_1 of X_1 and partition X_1 based on M_1. These two parts form the children of X_1. Likewise, we find the median M_2 of X_2 and partition X_2 based on M_2. These two parts form the children of X_2, and so on.

   At level i of the tree we have 2^i nodes and each of these nodes has a sequence with no more than $\frac{n}{2^i}$ elements (for $0 \le i \le k$). We have to find the median of each of these sequences and partition each sequence into two based on its median. The total time spent on each node is $O\left(\log\left(\frac{n}{2^i}\right)\right)$, the total work done being $O\left(\frac{n}{2^i}\right)$. Thus the total work done at level i is $O(n)$ the time spent being $O\left(\log\left(\frac{n}{2^i}\right)\right)$. Using the slow-down lemma, all the computations at level i can be completed in $O(\log n)$ time using $\frac{n}{\log n}$ processors (for $0 \le i \le k$).

   As a result, the total run time of the entire algorithm is $O(\log^2 n)$, the total work done being $O(n \log n)$.

4. Let $(p_1, w_1), (p_2, w_2), \ldots, (p_n, w_n); m, P$ be the given instance of the zero-one knapsack problem. Let the objects be $O_1, O_2, \ldots, O_n$. Run ZeroOneK on this instance. If the answer is "yes" then use the following algorithm to find a subset whose total weight is $\le m$ and whose total profit is $\ge P$:

   $S = \{O_1, O_2, \ldots, O_n\}$;
   for $i = 1$ **to** n **do**
       $S' = S - \{O_i\}$; Invoke ZeroOneK on S', m, P;

if the answer is "yes" **then** $S = S'$;

Output S;

In the above algorithm we invoke ZeroOneK n times. If $p(n)$ is the run time of ZeroOneK, then the run time of the above algorithm is $O(np(n))$ which will be a polynomial in n if $p(n)$ is a polynomial in n.

5. We are interested in checking if F has a satisfying assignment in which q variables have the value F and the other $(n - q)$ variables have the value T. Here $0 \leq q \leq c$. The number of assignments in which q variables have the value F and the other $(n - q)$ variables have the value T is $\binom{n}{q}$. For each such assignment we can check if F is satisfiable in $O(|F|)$ time. Therefore, we can check if F has a satisfying assignment in which at most c variables have the value F is $O(\sum_{q=0}^{c} \binom{n}{q}|F|) = O(n^c|F|)$ time, which is a polynomial in n and the input length. Thus the given problem is in $\mathcal{P}$.