

CSE 361 Complexity of Sequential and Parallel Algorithms

Fall 2005 Exam I–Solutions

1. Pick a random sample S of size $\frac{200}{3}\alpha \log_e n$ from A . If S has more ones than zeros, output **Type I**. Otherwise output **Type II**.

Assume that the input is of type I. Let X be the number of ones in S . $E[X] = 50\alpha \log_e n$. Using Chernoff bounds, $Prob.[X \leq 40\alpha \log_e n] \leq \exp(-\frac{1}{25}(50\alpha \log_e n)\frac{1}{2}) = n^{-\alpha}$. Thus with high probability there will be more ones than zeros in S .

We can prove a similar result for type II input also.

2. Sort $a[]$ in time $O(n \log n)$. Then run the following algorithm:

ThreeSum(x)

- (1) for ($i = 1; i \leq n; i++$), ($j = 1; j \leq n; j++$) do
- (2) $k = \text{BinarySearch}(x - a[i] - a[j], 1, n)$;
- (3) if k is nonzero, return (k, i, j) ;
- (4) print “not found” and exit;

3. Note that if A and B are any two sets in sorted order, we can compute $A \cap B$ as well as $A \cup B$ in $O(|A| + |B|)$ time using the merge algorithm discussed in class. Consider a complete binary tree with m leaves where each leaf contains an input set. Let the levels of this tree be $1, 2, \dots, h$ where h corresponds to the leaves.

- (a) **for** $i := (h - 1)$ **to** 1 **step** -1 **do**

for each node in level i compute the intersection of the two sets residing in its children.

Output the set computed at the root.

Note that the total time spent at any level of the tree is proportional to the number of elements in that level. Also, the number of elements at any level i is at most one half of the number of elements at level $i + 1$. Thus the run time of the algorithm is $O(n)$.

- (b) The algorithm is similar to above except that at each node of the tree we perform a union (instead of intersection). There are $O(\log m)$ levels and at each level the time spent is $O(n)$.

4. Let S represent the sum of all the elements in the array. We need to find an element k such that the sum of all the elements less than or equal to k (let's call this sum as LS) is greater than or equal to $S/4$ and the sum of all the elements greater than or equal to k (let's call this sum as US) is greater than or equal to $3S/4$.

Now, select the median (i.e., the $n/2$ th smallest element) M in the array. Let S' represent the sum of all the elements less than or equal to M in the array. Let $S'' (= S - S' + M)$ represent the sum of all the elements greater than or equal to M in the array. If $S' \geq S/4$ and $S'' \geq 3S/4$ then M itself is k and the problem is solved. If $S' < S/4$ then do the following: repeat the problem only in the elements greater than M with LS as $S/4 - S'$ and MS as $3S/4$. If $S'' < 3S/4$ then repeat the problem only in the elements less than M with LS as $S/4$ and US as $(3S/4) - S''$.

Analysis: Time for selecting the $n/2$ th element = $O(n)$ and time for partitioning the array based on $M = O(n)$.

Hence, $T(n) = T(n/2) + O(n)$ which implies $T(n) = O(n)$.

5. Matrices A and B could be divided into k matrices of size $n \times n$, A_1, \dots, A_k and B_1, \dots, B_k respectively. Then $AB =$

$$\begin{pmatrix} A_1 \\ A_2 \\ \dots \\ A_k \end{pmatrix} (B_1 \quad B_2 \quad \dots \quad B_k) = \begin{pmatrix} A_1 B_1 & A_1 B_2 & \dots & A_1 B_k \\ A_2 B_1 & A_2 B_2 & \dots & A_2 B_k \\ \dots & \dots & \dots & \dots \\ A_k B_1 & A_k B_2 & \dots & A_k B_k \end{pmatrix}$$

Thus AB could be found by computing $A_i B_j, 1 \leq i, j \leq k$. Each of $A_i B_j$ can be found by using Strassen's algorithm in $O(n^{\log 7})$ time, thus total time is $O(k^2 n^{\log 7})$.

6. The algorithm has the following steps-

(a) Determine the minimum number in A . Designate this as Min . This takes $O(n)$ time.

(b) **for** $i := 1$ **to** n **do**

$$B[i] = A[i] - Min$$

(c) **for** $i := 1$ **to** n **do**

$$C[i] = B[i] * n^5$$

(d) C contains **Integers** in the range $[1 : n^{10}]$ or $[0 : n^{10} - 1]$. These can be sorted in $O(n)$ time using **Radix-Sort**.

Note:- The elements in C can be considered to be **Integers**, after truncating their real parts, preserving their relative orders.