

Introduction to Computer Algorithms CS470
Final Examination
Monday, May 2, 2005
11:30am – 2:00pm

Student name

Student number

There are three problems. For full credit solve all of them. You can use three letter-size pages of your own notes, written on any of the two sides of the pages; no other material is permitted (e.g., no books).

For every algorithm that you design, precede the algorithm with an outline in English describing how the algorithm works.

Problem 1: [30 points]

Consider the following recursive sorting algorithm for an array A of length n , that is initially invoked as $\text{Sort}(A,1,n)$

```
Sort( A, i, j )
1  if i+1 = j
2      if A[i]>A[j]
3          swap A[i] with A[j]
4  if i+1 < j
5      d = ceiling( 2/3 * (j-i+1) )
6      Sort( A, i, i+d-1 )
7      Sort( A, j-d+1, j )
8      Sort( A, i, i+d-1 )
```

Argue formally and in detail why the algorithm is correct using mathematical induction. Why is the length of an “overlap” relative to the length a “leftover” important? Formulate a recurrence on the worst-case running time of the algorithm and solve the recurrence.

Solution:

Problem 2: [30 points]

You are trying to schedule jobs on a computer to maximize utility. Specifically, there are n jobs identified with numbers $\{1, 2, \dots, n\}$. Each job i has a starting time s_i and a finishing time f_i such that s_i and f_i are integers $0 < s_i < f_i$. Only jobs whose running times do not overlap can be scheduled together on the computer. Each job i has utility u_i that is an integer. The running time and utility is given as a triple (s_i, f_i, u_i) . Given n and a list of triples, your goal is to find a subset $S \subseteq \{1, 2, \dots, n\}$ of jobs that maximizes utility and can be scheduled together i.e., maximizes the value of $\sum_{i \in S} u_i$ across subsets S such that for every distinct $i, j \in S$ intervals $[s_i, f_i]$ and $[s_j, f_j]$ are disjoint (note that these are closed, not open, intervals).

Example

When the input is

6

(1,4,20)

(5,8,30)

(2,7,40)

(10,12,20)

(13,15,20)

(11,15,50)

then the answer is

1,2,6

Prove why your algorithm is correct and determine its worst-case running time.

Solution:

Problem 3: [40 points]

Imagine that you are working on controlling the spreading of a disease. The disease can be passed from a person to a person when they meet. You would like to find out who and when can get ill given a center of the disease. Formally, there are n people denoted by $\{1, 2, \dots, n\}$. Let L be a list of meetings between pairs of people and when the meetings occur i.e., any entry on the list is a triple (p_1, p_2, t) of natural numbers such that $1 \leq p_1, p_2 \leq n$, that means that p_1 meets p_2 at time t . Let $center$ denote the first person who becomes ill, and $time$ denote the time when that person becomes ill. A person p becomes ill when it is not ill and meets an ill person q at a time after q has become ill. Your goal is to design an algorithm that given n , a list L , and $center$ and $time$, determines, for each person, the earliest time when the person becomes ill.

Example

When the input is

4,
(1,2,1)
(1,3,2)
(3,4,6)
(1,3,4)
1,
0

then the answer is

1, 0
2, 1
3, 2
4, 6

but when the time person 1 becomes ill is 3 instead of 0, then the answer is

1, 3
2, infinity
3, 4
4, 6

Prove why your algorithm is correct and determine its worst-case running time.

Solution: