Comprehensive Exam: Algorithms and Concrete Mathematics Autumn 2004

This is a one hour closed-book exam and the point total for all questions is 60.

In questions that ask you to provide an algorithm, please explain the algorithm in words and diagrams, no need to write code or pseudo code. Also, for any algorithm, state and prove its running time. No credit will be given for exponential-time algorithms. Polynomial but slow algorithms will get some partial credit. Amount of credit will depend on how much slower they are compared to what is achievable using the knowledge in the reading list.

For full credit, the answers should be short and precise. Long and convoluted answers will not get full credit even if they are correct.

1. **[14 pts]** Please answer "true" or "false" to each one of the following questions. Correct answers will give you **(2 pts)** each while wrong answers will reduce your score by **(2 pts)** each.

   - $2n = O(n^2)$
   - $n! = o(n^n)$
   - $(\log n)^{10} = \Omega(\sqrt[10]{n})$
   - Any graph with n nodes and m edges has a spanning tree with $n-1$ edges.
   - Assume that T is a minimum cost spanning tree for a weighted graph $G = (V, E)$ with positive weights $w : E \to R^+$. Will T be still a minimum cost spanning tree if instead of w we use $\log w$?
   - You are given a graph $G = (V, E)$ with positive lengths on edges and a shortest path P from $v \in V$ to $u \in V$. Next, lengths are transformed by uniformly adding 1 to the length of each edge. Will P be still a shortest path from v to u ?
   - Let A be an array of n distinct numbers. The goal is to rearrange the numbers so that $\forall i$ where $0 < i < n/2$ we have $A[i] < A[2i]$. It is true that this operation requires $\Omega(n \log n)$ time in the comparison model?.

2. **[15 pts]** Let u and v be two n-bit numbers, where for simplicity n is a power of 2. The traditional multiplication algorithm requires $\Theta(n^2)$ operations. A divide-and conquer based algorithm splits the numbers into two equal parts, computing the product as

$$uv = (a2^{n/2} + b)(c2^{n/2} + d) = ac2^n + (ad + bc)2^{n/2} + bd$$

   Here, a and c are the higher order bits, and b and d are the lower order bits of u and v respectively. Multiplications are done recursively, except multiplication by 2^i for integer i, which is a shift and we assume takes $\Theta(i)$ time. We also assume that addition/subtraction of i-bit numbers takes $\Theta(i)$.

- **[6 pts]** Write a recurrence for the running time of this algorithm as stated. Solve the recurrence and determine the running time.
- **[9 pts]** Note that $ad + bc$ can be computed as $(a + b)(c + d) - ac - bd$. Why is this advantageous? Write and solve a recurrence for the running time of modified algorithm.

3. **[16 pts]** Given a set of items $x_1, x_2, \ldots, x_n$ where item x_i has a non-negative weight $w(x_i)$, the goal is to find a subset of items with maximum total weight under the constraint that if item x_i is chosen, then you are forbidden to chose either x_{i+1} or x_{i-1} (forbidden to choose the "neighbors" of x_i). For example, you can choose x_5, x_8, and x_{10}, but you are not allowed to choose x_5, x_6, x_9, x_{11}. Design an efficient algorithm to solve this problem, prove correctness and analyze running time. For simplicity, your algorithm should just produce the value of the maximum total weight and does not need to list the items that you are choosing to achieve this weight.

4. **[15 pts]** Consider a section of highway with numbered exits. You are given information about n cars that you suspect of speeding. Car i will enter the highway at point s_i and exits at $t_i > s_i$. The goal is to place minimum number of photo-radar units along the highway to ensure that you will catch (photograph) all of the n speeding cars. (Assume that once the photo-radar is placed, you cannot move it.)

   We say that paths of two cars i and j do not intersect (disjoint) if $t_i < s_j$ or $t_j < s_i$. Observe that if there is a set of k of cars that have disjoint paths, then you will need at least k photo-radars.

   Let k^* be the size of the maximum-size set of cars with disjoint paths. The observation above implies that you will need at least k^* photo-radars. Prove that in fact k^* photo-radars is sufficient, i.e. there exists a placement of k^* photo-radars that will photograph all of the cars.