Massachusetts Institute of Technology
6.046J/18.410J: Introduction to Algorithms
Professors Michel Goemans and Piotr Indyk

Handout 19
March 21, 2002

Problem Set 4

This problem set is due *before* class on *April 4, 2002*.

Both exercises and problems should be solved, but *only the problems* should be turned in. Exercises are intended to help you master the course material and will be useful in solving the assessed problems. You are responsible for material covered by the exercises. Each problem is to be done on a separate sheet (or sheets) of paper. Mark the top of each sheet with your name, 6.046J/18.410J, the problem number, your TA and recitation section, the date, and the names of any students with whom you collaborated.

Exercise 4-1. Do Exercise 15.4-4 in CLRS on page 356.

Exercise 4-2. Do Exercise 15.5-2 in CLRS on page 363.

Exercise 4-3. Do Exercise 16.2-2 in CLRS on page 384.

Problem 4-1. Planning a Party

Do Problem 15-4 in CLRS on page 367.

Problem 4-2. Class Photo

There are n students of heights $h_1 \ldots h_n$ standing in a single row for their class photograph. In order for the photograph to look nice, we want to make sure that the heights of nearby students don't differ by too much. Specifically, we say that the row is *unbalanced* if there exists a block of k adjacent students in which the tallest student in the block is greater than or equal to twice the height of the shortest student in the block.

(a) Give an $O(n \log k)$ algorithm to determine if the row is unbalanced.

(b) (Completely optional, not graded) Give an $O(n)$ algorithm to determine if the row is unbalanced.

Problem 4-3. Study Groups

In order to help students with their homework assignments, the 6.046 TAs have decided to split them up into study groups. There are $n \geq 2$ students $s(1) \ldots s(n)$ in the class (assume this list of students is already ordered alphabetically for simplicity), and the TAs want to form exactly k study groups, where $1 \leq k \leq n/2$. Each study group should contain at least two students. To simplify the splitting process, the TAs will line the students up in a row in alphabetical order by name, and cut the row in $k - 1$ places to form the k study groups.

The students, in an effort to form groups with their friends, have provided the TAs with a list of m pairs of friends. We denote these by pairs of indices into the list of students $(a_1, b_1), (a_2, b_2), \ldots, (a_m, b_m)$, such that for $1 \leq i \leq m$, student $s(a_i)$ is a friend of student $s(b_i)$. A pair of friends in this list is said to be *respected* if both friends end up in the same study group.

Unfortunately, your TAs are so busy creating fiendishly difficult questions for the second quiz that they have no time to figure out the optimal way to partition the students. Help them out by giving a dynamic programming algorithm which determines the $k-1$ locations where the row of students should be cut, such that the number of respected friendships is maximized. Try to make your algorithm as efficient as possible. The best we found was $O(n^2k)$ time.