CS 473: Fundamental Algorithms, Spring 2011

HW 5

Question 1 is due by **Saturday, 23:59:59, March 5**

Questions 2-3 are due by **Sunday, 23:59:59, March 6**

Question 4 is due by **Monday, 23:59:59, March 7**

This homework contains four problems. **Read the instructions for submitting homework on the course webpage**.

Collaboration Policy: For this homework, Problems 2–4 can be worked in groups of up to three students.

Problem 1 should be answered in Compass as part of the assessment HW5-Online and should be done individually.

1. HW5-Online. (10 pts.)

2. Matching in a tree. (40 pts.)

   Let $G = (V, E)$ be an undirected graph with weights on the edges. For example, for an edge $uv \in E$, we have a weight $w(uv)$ associated with uv. A subset $M \subseteq E$ of edges in G is a ***matching***, if no pair of edges of M share an endpoint; that is, every vertex is contained in at most one edge of M. We are interested in the problem of computing the maximum weight matching in G. In general, this problem can be solved efficiently, but the algorithm is quite complicated. Fortunately for the tree case this is much easier.

   Describe an efficient algorithm, as fast as possible, for computing the maximum weight matching when G is a tree.

3. Broadcasting in packets. (30 pts.)

   You are given n items of information, where the ith item is of size s_i (in bytes). The task is to transmit this information over a network pipe. The pipe allows you to send only packets of fixed size W (which is fortunately bigger than all the given items). The task is to pack the items into the packets as best as possible. Because of noise in the transmission, however, you prefer not to send completely full packets. Formally, if you are using x bytes in a packet, the price of sending it is $D + x^3$, where D is some prespecified parameter. (As a justification, imagine that if the packet is too full, you need more energy to transmit it.) Also, you must pack the items in the same order they are given into the packets (otherwise, the problem becomes much harder!). Naturally, you can put at most W bytes of information into a packet.

   Give an efficient algorithm to compute a (valid) partition of the items into packets that minimizes the total price of the generated packets. How fast is your algorithm?

4. (20 pts.) Implement your algorithm from the previous part in C or C++. The input to your program will be

$$\boxed{\begin{array}{c} n \\ W \\ D \\ s_1 \\ \vdots \\ s_n \end{array}}$$

(Given on the standard input.) The output should be the price of the solution (on its own line), and then for each item the index of the packet it belongs to (again, on its own line). You need to submit a single C (or C++) file that contains your program (which must compile on its own and run efficiently). We will provide several input/output examples on the class webpage.[1]

[1] We will use MOSS to verify that programs are not copied...