College of Computer & Information Science — Fall 2010
Northeastern University — Handout 9
CS7800: Advanced Algorithms — 12 October 2010

Problem Set 3 (due Monday, October 25)

1. (5 + 5 = 10 points) Minimum-length encoding

This is a continuation of Problem 4 of PS2. Recall that Alice and Bob are using an encoding scheme based on a set S of m *code words* that they both share. Alice wants to send a data string D of length n to Bob. In this exercise, we assume that S is an *arbitrary* set of m code words.

- **(a)** Design an efficient algorithm to determine a minimum-length encoding of a given string D using code words from S. If no such encoding exists, then the algorithm must indicate so. Analyze the worst-case running time of your algorithm.
- **(b)** Of course, the encoding scheme above is probably useful for a given string D only if there is a *unique* minimum-length encoding of D using code words from S. Enhance your algorithm of part (a), or give a new algorithm, which determines whether there exists exactly one minimum-length encoding of D using code words from S. Analyze the worst-case running time of your algorithm.

For both parts, make your algorithm as efficient as you can, in terms of its worst-case running time.

2. (10 points) Matroids

Exercise 16.4-4 of text.

3. (10 points) Resource reservation in video transmission

Consider the following resource reservation problem arising in video transmission. We are given a video in the form of a sequence of n frames. We are also given that frame i requires the reservation of at least s_i units of bandwidth along the transmission link. Since reserving resources separately for each frame may incur a significant overhead, we would like to partition the video into at most k *segments* (where k is usually much smaller than n), and then reserve bandwidth for each segment. Note that each segment is simply a set of contiguous frames and that the segments may be of different lengths.

The amount of bandwidth that we need to reserve for a segment is the maximum, over all frames in the segment, of the bandwidth required for the frame. Formally put, for a given segment S, the bandwidth $B(S)$ required for the segment equals $\max_{i \in S} s_i$.

Given a partition of the video into k segments $S_1, S_2, \ldots, S_k$, we define the *total bandwidth requirement* of the partition as the following summation:

$$\sum_{1 \le i \le k} f(S_i) \cdot B(S_i),$$

where $f(S_i)$ is the number of frames in segment S_i.

Design an efficient algorithm to partition a given n-frame video into k segments such that the total bandwidth requirement of the partition is minimized. Justify the correctness of your algorithm. Analyze the worst-case running time of your algorithm. Make your algorithm as efficient as you can, in terms of its worst-case running time.

4. (5 + 5 = 10 points) Viterbi algorithm for speech processing

Problem 15-7 of text (15-5 in second edition).

5. (5 + 5 = 10 points) Recomputing shortest paths when source changes

Suppose you are given a directed graph $G = (V, E)$ with real weights on edges, a source s, and a number $d[v]$ for each $v \in V$ that supposedly gives the shortest path distance from s to v in G. Let m and n denote the number of edges and vertices, respectively, of G.

(a) Give a linear-time algorithm ($O(m + n)$ time) that determines whether the collection of distances given by $d[v]$, $v \in V$, is indeed correct.

(b) Now suppose you are asked to compute the single-source shortest path distances from a different source s'. We can use the Bellman-Ford algorithm to compute these distances in $O(mn)$ time. Since edge weights may be negative, we cannot directly use Dijkstra's algorithm. We can, however, use the distances $d[v]$ already computed to compute the shortest path distances from s' faster than simply invoking Bellman-Ford, if $d[v]$ is finite for all v.

Show how to compute these new shortest path distances in $O(m \log n)$ time. (*Hint:* Design new edge weights that allow you to use Dijkstra's algorithm.)