

## Homework 2: Dynamic Programming

Assigned: September 19, 2007

Due Date: October 3, 2007

Whenever you are asked to give an algorithm for a problem, I expect you to do all of the following:

1. give a clear, concise description of your algorithm;
2. prove that the algorithm outputs an optimal solution;
3. give the asymptotic time complexity of the algorithm;
4. prove that the time complexity is as claimed.

You must submit your homework with a signed cover sheet attached to the front.

Note that the point values may not add up to 100; your score will be total points earned divided by total number of *required* points.

## Core Problems

1. (10 points) Let's return to the babysitting problem in Homework 1. In that problem, Rena's sitters charged a flat fee (say, \$20) for their services, regardless of how long they worked. Now, however, all the sitters have joined the International Brotherhood of Baby Wranglers (a local of the UAW), and union rules say they must charge \$20/hour. Hence, the cost of a sitter is now proportional to the length of time he or she works.

As in the previous problem, each sitter  $b_i \in B$  will work the interval  $[s_i, f_i]$  (precisely; no partial intervals are permitted, so sitters may overlap). Give an efficient algorithm to select babysitters to cover the period  $[S, F]$  that minimizes the total cost (i.e. total length of intervals) for all sitters chosen, i.e.

$$\sum_{b_i \text{ chosen}} f_i - s_i.$$

2. (15 points) Professor Audubon has recorded the song of a bird in his backyard. The record consists of a time series  $Q$  of  $n$  successive pitch measurements  $q_1 \dots q_n$ , taken once every 100 milliseconds. The professor wants to divide the song into its individual notes (intervals of constant pitch). Unfortunately, experimental noise randomly perturbs the pitches measured over the course of a single note, so that they do not remain constant.

To approximately divide the song into notes, the professor proposes the following approach. Let the *variance* of an interval  $q_i \dots q_j$  of  $Q$ , denoted  $\sigma^2(i, j)$ , be defined in the usual statistical sense as

$$\sigma^2(i, j) = \sum_{k=i}^j (q_k - E(i, j))^2,$$

where

$$E(i, j) = \frac{1}{j - i + 1} \sum_{k=i}^j q_k$$

is the mean pitch of measurements  $i..j$ . If the measurement error in each pitch is small, then the variance of the interval corresponding to a single note should be low, while the variance over intervals combining distinct notes should be high. Hence, we want to divide  $Q$  into notes such that the sum of variances for all notes is minimized.

There is one important problem with the above approach. The number of notes is unknown *a priori*. One could simply create a new note for each measurement, making the variance of each note (and hence the sum of variances for all notes) zero! To avoid this trivial solution, we penalize a proposed division into  $m$  notes by adding  $m \cdot p$  to its cost, where  $p > 0$  is a user-defined penalty.

To summarize, the goal is to divide the song  $Q$  into notes by selecting  $m \leq n$  break points  $b_1 \dots b_m$  (the ends of each note) so as to minimize

$$m \cdot p + \sum_{i=1}^{m+1} \sigma^2(b_{i-1} + 1, b_i)$$

where  $b_0 = 0$  and  $b_{m+1} = n$ .

Give an efficient algorithm for this problem. Your solution should run in time  $O(n^2)$ . *Hint:* show how to compute  $\sigma^2(i, j)$  in constant time for any  $(i, j)$ .

3. (10 pts) Consider the mass spectrometry problem discussed in Homework 1. Recall that the goal is to match each measured mass  $x_i$  to a known mass  $w_i$  so as to minimize the sum of absolute differences  $\sum_i |w_i - x_i|$ .

Suppose that the numbers of measured and known masses are  $n$  and  $m > n$  respectively. As before, we may not match two measured masses to the same known mass; however, in this version of the problem, not all known masses must be used. Given an efficient algorithm to match the measured masses to known masses so as to minimize their sum of absolute differences. Be sure to analyze your time complexity as a function of both  $m$  and  $n$ .

(**Note:** you may want to look at the solution to the  $m = n$  case in Homework 1 before attempting this problem! You need not re-prove anything about that algorithm on this homework.)

4. (15 points) Consider the problem of word-wrapping a paragraph. A paragraph is an ordered list of  $n$  words, where word  $w_i$  is  $\ell_i$  letters long. You want to divide the paragraph into a sequence of lines, each containing at most  $L$  letters. (No word is more than  $L$  letters long.) Suppose a line contains words  $w_i \dots w_j$ . The total length  $W(i, j)$  of this line is defined by

$$W(i, j) = j - i + \sum_{k=i}^j \ell_k.$$

This length accounts for a single space between successive pairs of words on the line. The *slop*  $S(i, j)$  of this line is defined to be  $L - W(i, j)$ , the total number of unused spaces at the

end of the line. Note that in any feasible solution, the slop of each line must be non-negative. (The squared slop criterion is a simplified version of what is actually used in, e.g., TeX for paragraph wrapping.)

Just to make things concrete, consider the example paragraph “Now is the time for all good men.”, and suppose  $L = 10$ . One feasible solution is

```
Now is the
time for
all good
men.
```

This solution has four lines of lengths 10, 8, 8, and 4; the corresponding slops are 0, 2, 2, and 6.

Your goal is to find a division of the input paragraph into lines that minimizes the sum, over all lines *except the last*, of the *squared* slop of each line. (We omit the last line because it can in general be much shorter than the others.) For example, the total cost of the above solution is  $0^2 + 2^2 + 2^2 = 8$ .

Give an efficient algorithm for this problem.

## Advanced Problem

**This problem is *required* only for CSE 541 students.** 441 students may receive extra credit for a correct solution.

5. (15 points) You are compiling Linkin Park’s greatest hits collection. Your goal is to construct a  $k$ -volume CD box set packed with a retrospective of the group’s music. As input, you are given a list of  $n$  songs by the group, in fixed chronological order. Song  $i$  in the list is  $\ell_i$  minutes long. Each CD is  $m$  minutes long.

Your goal is to choose a subset of these songs for the  $k$  CDs in the box set. The assignment of songs to CDs must respect the input order; that is, if songs  $i$  and  $j$  are both in the chosen subset, and  $j > i$ , then either  $j$  follows  $i$  on the same CD, or  $j$  occurs on a later CD than  $i$ . No song may be split across CDs.

Give an efficient algorithm (in  $n$ ) to find the **largest number of songs** that can be placed on  $k$  CDs while respecting the input order. Analyze the complexity of your algorithm as a function of  $n$  and  $k$ . (*Hint*: a good way to make progress is to compute the smallest number of CDs needed to store each set of  $s$  songs.)