CSCE 750, Fall 2012, Assignment 7

October 21, 2014

Page 314 Exercise 13.2-3, 13.2-4

Page 322 Exercise 13.3-1, 13.3-2, 13.3-3, 13.3-5

Pages 333–336 Problem 13-4(a,b,c,d) (Hint for part (b): this is similar to the analysis of randomized Quicksort.)

Page 370 Exercises 15.1-2, 15.1-3

Page 378 Exercises 15.2-1, 15.2-5

Pages 389–390 Exercises 15.3-3, 15.3-4

Pages 396–397 Exercises 15.4-1, 15.4-2, 15.4-4, 15.4-5, 15.4-6 (optional)

Pages 404–405 Problems 15-1, 15-10, 15-12

Not in the text

Instance of rod-cutting You are given the following table of prices you can get for rod lengths $i = 1, \ldots, 12$ (in inches):

<table>
<thead>
<tr>
<th>$i$</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_i$</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>9</td>
<td>4</td>
<td>6</td>
<td>10</td>
<td>12</td>
<td>8</td>
<td>7</td>
<td>11</td>
</tr>
</tbody>
</table>

Using the dynamic programming algorithm given in the book or in class, find the maximum possible revenue for a 12-inch rod, and a series of cuts that achieves this revenue.

Knapsack Problem You are going on a camping trip, and you have a cooler with a capacity of $n$ ounces. The local Walmart sells soda bottles of $k$ different sizes, say $a_1, \ldots, a_k$, each a whole number of ounces. Can you stock your cooler completely (to capacity) by buying sodas in some combination of the available sizes? Assume you can buy any number of bottles of any size, and that you can pack your cooler efficiently (no empty gaps).

In other words, you are given positive integers $n$ and $a_1, \ldots, a_k$, and you want to know if there exist integers $b_1, \ldots, b_k \geq 0$ such that

$$n = \sum_{i=1}^{k} b_i a_i.$$

That is, can you buy $b_i$ bottles of size $a_i$ for $i = 1, \ldots, k$ to fill up the cooler completely?
1. Give a simple recursive solution to this problem (don’t worry about efficiency).

2. Give an algorithm that solves this problem in $O(nk)$ time using $O(n+k)$ space (assume the $a_i$ values are given in an array $A[1\ldots k]$). [Hint: Build a table $m[0\ldots n]$ of Boolean values, where each entry $m[i]$ records the answer for a cooler of capacity $i$.]

3. Adapt your algorithm to output values $b_1,\ldots,b_k$ (nonnegative integers) that maximize the sum above without exceeding $n$.

**Subset Sum Problem** This is just like the knapsack problem above, except you are constrained to buy no more than one bottle of each size. In other words, each $b_i$ must be in $\{0,1\}$. Give an algorithm that solves this problem in time $O(nk)$ using $O(nk)$ space. [Hint: Use a table $c[0\ldots n,0\ldots k]$ of Boolean values, where each entry $c[i,j]$ records the answer for capacity $i$ and available sizes $a_1,\ldots,a_j$.]