Lindsey Bieda and Joe Frambach
Dynamic Programming Problems
10.05.2011

18. The input to this problem is a set of n gems. Each gem has a value in dollars and is either a ruby or an emerald. Let the sum of the values of the gems be L. The problem is to determine if it is possible to partition of the gems into two parts P and Q, such that each part has the same value, the number of rubies in P is equal to the number of rubies in Q, and the number of emeralds in P is equal to the number of emeralds in Q. Note that a partition means that every gem must be in exactly one of P or Q. You algorithm should run in time polynomial in $n + L$.

First, we must assert that a solution can exist. We split the input I into two arrays, Rubies and Emerils.
Assert that $||R|| \ mod\ 2 \ == \ 0$ and $||E|| \ mod\ 2 \ == \ 0$.
Assert that $L \ mod\ 2 \ == \ 0$.

The tree is constructed with nodes $[level,\ \#rubies,\ \#emeralds,\ value] \ = \ boolean$, where the indexes show the current state of partition P or Q after deciding on gem $level$. Which partition specifically doesn't matter, since at the end they will be identical. $boolean$ would be used for a traceback algorithm later, but here it only matters that the cell is defined or not defined.

Pruning Rules:
Prune nodes with more than $||R||/2$ rubies or more than $||E||/2$ Emerils, or has a value greater than $L/2$.

Initialization:
$P[0, 0, 0, 0] = 0$.

Building the tree:
For g = 0 to $||I||$:
    If I_g is a ruby:
        For r = 0 to $||R||/2$:
            For e = 0 to $||E||/2$:
                For $\ell = 0$ to $L/2$:
                    $A[g + 1, r, e, \ell] = 0$
                    $A[g + 1, r + 1, e, \ell + I_g] = 1$
    Else (I_g is an Emeril):
        For r = 0 to $||R||/2$:
            For e = 0 to $||E||/2$:
                For $\ell = 0$ to $L/2$:
                    $A[g + 1, r, e, \ell] = 0$
                    $A[g + 1, r, e + 1, \ell + I_g] = 1$

If $A[||I||, \frac{||R||}{2}, \frac{||E||}{2}, \frac{L}{2}]$ is defined, then a solution exists. Bam.

19. The input to this problem consists of an ordered list of n words. The length of the ith word is w_i, that is the ith word takes up w_i spaces. (For simplicity assume that there are no spaces between words.) The goal is to break this ordered list of words into lines, this is called a layout. Note that you can not reorder the words. The length of a line is the sum of the lengths of the words on that line. The ideal line length is L. No line may be longer than L, although it may be shorter. The penalty for having a line of length K is $L - K$. *The total penalty is the* **maximum** *of the line penalties.* The problem is to find a layout that minimizes the total penalty. Give a polynomial time algorithm for this problem.

Pruning rules: 1) Prune all nodes at the same level that have the same words on the last line except for the the one with the minimum total penalty.

Initalization:
A[*,*] = INF
A[1, w_1] = 0

For $i = 0$ to n:
  For $\ell = 0$ to L:
    $A[i+1, w_{i+1}] = min(A[i+1, w_{i+1}], L - \ell)$
    $A[i+1, \ell + w_{i+1}] = min(A[i+1, \ell + w_{i+1}], A[i, \ell])$

The solution is at

$$\min_{0 \leq \ell \leq L} (A[n, \ell] + L - \ell)$$