OUTLINES OF DYNAMIC PROGRAMMING SOLUTIONS

DR. GODFREY C. MUGANDA

1. Problem 6.1

A *contiguous subsequence* of a list S is a subsequence made up consecutive elements of S. For instance, if S is

5, 15, -30, 10, -5, 40, 10

then 15, -30, 10 is a contiguous subsequence but 5, 15, 40 is not. Give a linear-time algorithm for the following task:

Input:	A list of numbers $a_1, a_2, \dots, a_n$.
Output:	The contiguous subsequence of maximum sum (a subsequence of length zero has sum zero).

For the preceding example, the answer would be 10, -5, 40, 10, with a sum of 55.

Hint: For each $j \in \{1, 2, \dots, n\}$, consider the contiguous subsequences ending exactly at position j.

Let $S[j] =$ be the maximum sum of a contiguous subsequence that ends at j.

Let $I[j] =$ be the index of the beginning of the maximum sum subsequence.

There are two choices at j:

(1) Extend the contiguous subsequence that ends at $j-1$, getting a subsequence with sum $S[j-1] + a[j]$.
(2) Do not extend: which means you cut off the subsequence that ends at $j-1$ and start a new subsequence that begins and ends at j, whose sum will be $a[j]$.

The recurrence relation is

$$S[j] = \max\{S[j-1] + a[j], a[j]\}$$

$$I[j] = \begin{cases} I[j-1], & \text{if } S[j-1] + a[j] \geq a[j] \\ j, & \text{otherwise.} \end{cases}$$

which is equivalent to

$$I[j] = \begin{cases} I[j-1], & \text{if } S[j-1] \geq 0 \\ j, & \text{otherwise.} \end{cases}$$

Boundary conditions are $S[1] = a[1]$ and $I[1] = 1$.

2. Problem 6.2

You are going on long trip. You start on the road at mile post 0. Along the way there are n hotels, at mile posts $a_1 < a_2 < \cdots < a_n$, where each a_i is measured from the starting point. The only places you are allowed to stop are at these hotels, but you can choose which of these hotels to stop at. You must stop at the final hotel (at distance a_n), which is your destination.

You would ideally like to travel 200 miles a day, but this may not be possible (depending on the spacing of the hotels). If you travel x miles during a day, the penalty for that day is $(200 - x)^2$. You want to plan your trip so as to minimize the total penalty – that is, the sum, over all travel days, of the daily penalties.

Give an efficient algorithm that determines the optimal sequence of hotels at which to stop.

Let $P[j]$ = minimum total penalty for stopping at hotel j.

Let $\pi[j]$ = the previous hotel stopped at before stopping at hotel j.

If you are at hotel j and the previous stop was at hotel k, where $0 \leq k < j$ (we assume that hotel 0 is the starting point and is at mile marker $a_0 = 0$), then you traveled $a_j - a_k$ miles in one day and incurred a penalty of

$$(200 - (a_j - a_k))^2 .$$

To get the minimum penalty, you must consider all previous stops k and take the minimum. We get the recurrence relation

$$P[j] = \min_{0 \leq k < j} \{P[k] + (200 - (a_j - a_k))^2\}$$

We set $\pi[j]$ to be the k that achieves the minimum in the above recurrence relation.

Boundary condition is $P[0] = 0$ and $\pi[0] = -1$.

3. Problem 6.3

Yuckdonald's is considering opening a series of restaurants along Quaint Valley Highway(QVH). The n possible locations are along a straight line, and the distances of these locations from the start of QVH are, in miles and in increasing order, $m_1, m_2, \ldots, m_n$. The constraints are as follows:

- At each location, Yuckdonald's may open at most one restaurant. The expected profit from opening a restaurant at location i is p_i, where $p_i > 0$ and $i = 1, 2, \ldots, n$.
- Any two restaurants should be at least k miles apart, where k is a positive integer.

Give an efficient algorithm to compute the maximum expected total profit subject to the given constraints.

Let $\pi[j]$ = maximum expected profit from building restaurants at mile markers selected from $m_1, m_2, \ldots, m_j$.

The decision at j is whether to put a restaurant there or not. We can record this decision with a boolean variable $d[j]$:

$$d[j] = \begin{cases} T & \text{if we put a restaurant at } j, \\ F & \text{otherwise} \end{cases}$$

If we do not put a restaurant at j, then

$$\pi[j] = \pi[j-1],$$

which is the maximum expected profit of building restaurants at mile markers selected from $m_1, m_2, \ldots, m_{j-1}$.

If we do put a restaurant at j, then we get an expected profit of p_j in addition to the maximum profit we can get from putting restaurants at mile markers selected from $m_1, m_2, \ldots, m_i$, where $i < j$ and is the largest i such that $m_j - m_i \geq k$. In that case,

$$\pi[j] = p_j + \pi[i]$$

where i is the largest index such that $i < j$ and $m_j - m_i \geq k$.

The recurrence relation is

$$\pi[j] = \max\{\pi[j-1], p_j + \pi[i]\}$$

where i is the largest index such that $i < j$ and $m_j - m_i \geq k$.

We set $d[j]$ to the same value as the boolean expression $p_j + \pi[i] > \pi[j-1]$.

$$d[j] = \begin{cases} T & \text{if } p_j + \pi[i] > \pi[j-1], \\ F & \text{otherwise} \end{cases}$$

Boundary conditions are $\pi[1] = p_1$ and $d[i] = T$.

4. Problem 6.6

Let us define a multiplication operation on three symbols a, b, c according to the following table; thus $ab = b$, $ba = c$, and so on. Notice that the multiplication operation defined by the table is neither associative nor commutative.

	a	b	c
a	b	b	a
b	c	b	a
c	a	c	c

Find an efficient algorithm that examines a string of these symbols, say *bbbbac*, and decides whether or not it is possible to parenthesize the string in such a way that the value of the resulting expression is a. For example, in input *bbbbac* your algorithm should return *yes* because $(b(bb))(ba)c) = a$

This is not a maximization or minimization problem: We are being asked to determine whether a certain boolean expression evaluates to true, the boolean expression begin

This string of symbols evaluates to a.

Consider the boolean statement $P[i, j, x]$:

$P[i, j, x]$: the string $s[i, \ldots, j]$ can be parenthesized to evaluate to x.

We are interested in $P[1, n, a]$, which states that the entire string can be parenthesized to evaluate to a.

Looking at the multiplication table, we see that there are three ways for a to be a product:

$$ac = a, \qquad bc = a, \qquad ca = a$$

We then see that

$$\begin{aligned} P[i, j, a] = & \bigvee_{k=i}^{j-1} (P[i, k, a] \wedge P[k+1, j, c]) \\ & \vee \bigvee_{k=i}^{j-1} (P[i, k, b] \wedge P[k+1, j, c]) \\ & \vee \bigvee_{k=i}^{j-1} (P[i, k, c] \wedge P[k+1, j, a]) \end{aligned}$$

Likewise, we see from the multiplication table that there are three ways to form b:

$$aa = b, \qquad ab = b, \qquad bb = b$$

From these we get the recurrence relations

$$\begin{aligned} P[i,j,b] = &\bigvee_{k=i}^{j-1} (P[i,k,a] \wedge P[k+1,j,a]) \\ \vee &\bigvee_{k=i}^{j-1} (P[i,k,a] \wedge P[k+1,j,b]) \\ \vee &\bigvee_{k=i}^{j-1} (P[i,k,b] \wedge P[k+1,j,b]) \end{aligned}$$

And finally, there are three ways to form c

$$ba = c, \qquad cb = c, \qquad cc = c$$

From these we get the recurrence relations

$$\begin{aligned} P[i,j,c] = &\bigvee_{k=i}^{j-1} (P[i,k,b] \wedge P[k+1,j,a]) \\ \vee &\bigvee_{k=i}^{j-1} (P[i,k,c] \wedge P[k+1,j,b]) \\ \vee &\bigvee_{k=i}^{j-1} (P[i,k,c] \wedge P[k+1,j,c]) \end{aligned}$$

5. Problem 6.25

Consider the following 3-Partition problem. Given integers $a_1, a_2, \ldots, a_n$ we want to determine whether it is possible to partition $\{1, \ldots, n\}$ into three disjoint subsets I, J, K such that

$$\sum_{i \in I} a_i = \sum_{i \in J} a_i = \sum_{i \in K} a_i = \frac{1}{3} \sum_{i=1}^{n} a_i$$

For example, for input (1, 2, 3, 4, 4, 5, 8) the answer is *yes*, because there is a partition (1, 8), (4, 5), (2, 3, 4). On the other hand, for input (2, 2, 3, 5) the answer is *no*.

Devise and analyze a dynamic programming algorithm for the 3partition problem that runs in time polynomial in n and in $\sum_i a_i$.

This is another problem where we are trying to determine whether a boolean statement is true: Can a set of integers $a[1, \ldots, n]$ be partitioned into three parts with equal sum?

For integers r, s, and t and index j $(1 \leq j \leq n)$ define $P[r, s, t, j]$ to be the statement

$$\begin{aligned} P[r, s, t, j] = {} & \text{The sums of elements in } I, J, K \text{ are respectively } r, s, t \text{ after} \\ & \text{the elements in } a[1, \ldots, j] \text{ have been distributed.} \end{aligned}$$

Let

$$T = \frac{1}{3} \sum_{i=1}^{n} a_i.$$

Then the question we are trying to decide is the statement $P[T, T, T, n]$.

The recurrence relation is

$$P[r, s, t, j] = P[r - a_j, s, t, j - 1] \vee P[r, s - a_j, t, j - 1] \vee P[r, s, t - a_j, j - 1]$$

with the boundary condition being $P[0, 0, 0, 0]$, or if you like, you can take the boundary conditions to be

$$P[a_1, 0, 0, 1] \qquad P[0, a_1, 0, 1] \qquad P[0, 0, a_1, 1].$$