Name: Inchara Raveendra

SCU ID: 00001653600

Assignment - 5

15.1-1
Show that equation (15.4) follows from equation (15.3) and the initial condition $T(0) = 1$.

We can prove that $T(n) = 2^n$ is a solution using Substitution method.

For $n=0$, $T(n) = 2^0 = 1$

For $n > 0$,

$$T(n) = 1 + \sum_{j=0}^{n-1} 2^j$$

$$= 1 + \frac{2^n - 1}{2-1}$$ [From summation of terms in a geometric progression]

$$= 2^n$$

15.1-2
Show, by means of a counterexample, that the following "greedy" strategy does not always determine an optimal way to cut rods. Define the ***density*** of a rod of length i to be p_i/i, that is, its value per inch. The greedy strategy for a rod of length n cuts off a first piece of length i, where $1 \leq i \leq n$, having maximum density. It then continues by applying the greedy strategy to the remaining piece of length $n - i$.

Consider the below table for length, density and price

length	i	1	2	3	4
price	p_i	1	20	33	36
density	p_i/i	1	10	11	9

For $n=4$ which is the problem size, we can cut a rod of length 3 at 33\$ and of length 1 at 1\$ $\Rightarrow$ $33+1 = 34\$$

But, the optimal way to cut would be two rods of length 2 each $\Rightarrow$ $20+20 = 40\$$

$\therefore$ Greedy strategy with maximum density did not fetch optimal solution and does not always determine optimal way to cut rods.

15.1-4

Modify MEMOIZED-CUT-ROD to return not only the value but the actual solution, too.

```
MEMOIZED-CUT-ROD(p, n)
    let r[0....n] and s[0....n] be new arrays
    for i=0 to n
        r[i] = -∞
    (val, s) = MEMOIZED-CUT-ROD-AUX (p, n, r, s)
    print "Optimal value is" + val + " and cuts are
                        at" + s
```

```
j = n
while j > 0
    print s[j]
    j = j - s[j]


MEMOIZED-CUT-ROD-AUX(p, n, r, s)
  if r[n] ≥ 0
      return r[n]
  if n == 0
      q = 0
  else q = -∞
      for i = 1 to n
          (val, s) = MEMOIZED-CUT-ROD-AUX(p, n-i, r, s)
          if q < p[i] + val
              q = p[i] + val
              s[n] = i
  r[n] = q
  return (q, s)
```

15.1-5

The Fibonacci numbers are defined by recurrence (3.22). Give an $O(n)$-time dynamic-programming algorithm to compute the nth Fibonacci number. Draw the subproblem graph. How many vertices and edges are in the graph?

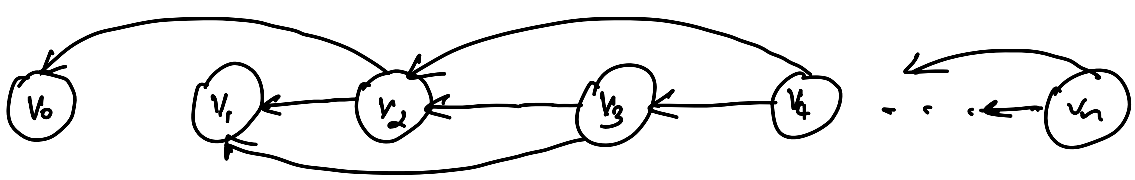
```
FIBONACCI (n)
    let fib[0.....n] be a new array
    fib[0] = 1
    fib[1] = 1
    for i = 2 to n
        fib[i] = fib[i-1] + fib[i-2]
    return fib[n]
```

Number of vertices = $n-1$

Vertices v_0, v_1 have 0 leaving edge

Vertices $v_2, v_3, \ldots v_n$ have 2 leaving edges.

$\therefore$ Total edges $= 2n-2$



15.3-5

Suppose that in the rod-cutting problem of Section 15.1, we also had limit l_i on the number of pieces of length i that we are allowed to produce, for $i = 1, 2, \ldots, n$. Show that the optimal-substructure property described in Section 15.1 no longer

When there is a limit l_i on number of pieces of size i, sub-problems can no longer be solved independently.

Ex:

Length i	1	2	3	4
Price p_i	15	20	33	36
Limit l_i	2	1	1	1

There are 3 possible solutions :

a) $i = 4$ Price = 36\$

b) $i = 1$ and $i = 3$ Price = 15+33 = 48\$

c) $i = 1, 1$ and 2 Price = 15+15+20 = 50\$

When we look at subproblem for length 2, it has two solutions

a) $i = 2$ Price = 20\$

b) $i = 1, 1$ Price = 15+15 = 30\$

$\therefore$ Optimal solution for length 2 = {1, 1}

But we cannot use this since there is a limit on the number of pieces that are permitted

15.5-1

Write pseudocode for the procedure CONSTRUCT-OPTIMAL-BST(*root*) which, given the table *root*, outputs the structure of an optimal binary search tree. For the example in Figure 15.10, your procedure should print out the structure

k_2 is the root
k_1 is the left child of k_2
d_0 is the left child of k_1
d_1 is the right child of k_1
k_5 is the right child of k_2
k_4 is the left child of k_5
k_3 is the left child of k_4
d_2 is the left child of k_3
d_3 is the right child of k_3
d_4 is the right child of k_4
d_5 is the right child of k_5

corresponding to the optimal binary search tree shown in Figure 15.9(b).

```
CONSTRUCT-OPTIMAL-BST (root, i, j, last)
    if i == j
        return
    if last == 0
        print root[i, j] + " is the root"
    else if j < last
        print root[i, j] + " is the left child of " + last
    else
        print root[i, j] + " is the right child of " + last
    CONSTRUCT-OPTIMAL-BST (root, i, root[i, j] - 1, root[i, j])
    CONSTRUCT-OPTIMAL-BST (root, root[i, j] + 1, j, root[i, j])
```

15.5-2

Determine the cost and structure of an optimal binary search tree for a set of $n = 7$ keys with the following probabilities:

i	0	1	2	3	4	5	6	7
p_i		0.04	0.06	0.08	0.02	0.10	0.12	0.14
q_i	0.06	0.06	0.06	0.06	0.05	0.05	0.05	0.05

i \ j	0	1	2	3	4	5	6	7
1	0	0.04	0.14	0.3	0.36	0.58	0.9	1.28
2	0	0	0.06	0.2	0.24	0.46	0.74	1.12
3	0	0	0	0.08	0.12	0.32	0.56	0.92
4	0	0	0	0	0.02	0.14	0.38	0.66
5	0	0	0	0	0	0.1	0.32	0.6
6	0	0	0	0	0	0	0.12	0.38
7	0	0	0	0	0	0	0	0.14

$$e(i,j) = \min_{i \le r \le j} \{ e(i, r-1) + e(r+1, j) \} + w(i,j) \quad i \le j$$

$$w(i,j) = \sum_{i=1}^{j} p_i$$

$$e(1,1) = \min_{1 \le r \le 1} \{ e(1,\cancel{0}) + e(\cancel{2},1) \} + w(1,1) \quad r=1$$

$$= P_1 = 0.04 \qquad \boxed{r=1}$$

|||ly $e(2,2) = 0.06$ ($r=2$) $\quad e(3,3) = 0.08$ ($r=3$)

$e(4,4) = 0.02$ ($r=4$) $\quad e(5,5) = 0.1$ ($r=5$)

$e(6,6) = 0.12$ ($r=6$) $\quad e(7,7) = 0.14$ ($r=7$)

$$e(1,2) = \min_{1 \le r \le 2} \{ \overset{r=1}{e(1,\cancel{0}) + e(2,2)},\ \overset{r=2}{e(1,1) + e(3,\cancel{2})} \} + w(1,2)$$

$$= \min \{ 0.06,\ 0.04 \} + P_1 + P_2$$

$$= 0.04 + 0.1 = 0.14 \qquad \boxed{r=2}$$

$$\begin{array}{c} \\ \end{array}\quad r = 2 \qquad\qquad\qquad r = 3$$

$$e(2,3) = \min_{2 \le r \le 3} \{ \cancelto{0}{e(2,1)} + e(3,3),\ e(2,2) + \cancelto{0}{e(4,3)} \} + w(2,3)$$

$$= \min \{ 0.08,\ 0.06 \} + P_2 + P_3$$

$$= 0.06 + 0.14 = 0.2 \qquad \boxed{r = 3}$$

$$r = 1 \qquad\qquad\qquad r = 2$$

$$e(1,3) = \min_{1 \le r \le 3} \{ \cancelto{0}{e(1,0)} + e(2,3),\ e(1,1) + e(3,3),\ e(1,2) + \cancelto{0}{e(4,3)} \} + P_1 + P_2 + P_3$$

$$r = 3$$

$$= \min \{ 0.2,\ (0.04 + 0.08),\ 0.14 \} + 0.18$$

$$= 0.12 + 0.18 = 0.3 \qquad \boxed{r = 2}$$

$$r = 3 \qquad\qquad\qquad r = 4$$

$$e(3,4) = \min_{3 \le r \le 4} \{ \cancelto{0}{e(3,2)} + e(4,4),\ e(3,3) + \cancelto{0}{e(5,4)} \} + w(3,4)$$

$$= \min \{ 0.02,\ 0.08 \} + P_3 + P_4$$

$$= 0.02 + 0.08 + 0.02$$

$$= 0.12 \qquad \boxed{r = 3}$$

$$r = 2 \qquad\qquad\qquad r = 3$$

$$e(2,4) = \min_{2 \le r \le 4} \{ \cancelto{0}{e(2,1)} + e(3,4),\ e(2,2) + e(4,4),\ e(2,3) + \cancelto{0}{e(5,4)} \} + P_2 + P_3 + P_4$$

$$r = 4$$

$$= \min\{0.12,\ 0.06+0.02,\ 0.2\} + 0.16$$

$$= 0.08 + 0.16$$

$$= 0.24 \qquad \boxed{r=3}$$

$$e(1,4) = \min_{1\le r\le 4}\Big\{ \overset{r=1}{\overset{0}{\cancel{e(1,0)}}+e(2,4)},\ \overset{r=2}{e(1,1)+e(3,4)},$$

$$\underset{r=3}{e(1,2)+e(4,4)},\ \underset{r=4}{e(1,3)+\overset{0}{\cancel{e(5,4)}}}\Big\}$$

$$+ p_1+p_2+p_3+p_4$$

$$= \min\{0.24,\ 0.04+0.12,\ 0.14+0.02,$$

$$0.3\} + 0.2$$

$$= 0.16+0.2$$

$$= 0.36 \qquad \boxed{r=2}$$

$$e(4,5) = \min_{4\le r\le 5}\Big\{ \overset{r=4}{\overset{0}{\cancel{e(4,3)}}+e(5,5)},\ \overset{r=5}{e(4,4)+}$$

$$\overset{0}{\cancel{e(6,5)}}\Big\} + p_4+p_5$$

$$= \min\{0.1,\ 0.02\} + 0.12$$

$$= 0.14 \qquad \boxed{r=5}$$

$$e(3,5) = \min_{3\le r\le 5}\Big\{ \overset{r=3}{\overset{0}{\cancel{e(3,2)}}+e(4,5)},\ \overset{r=4}{e(3,3)+}$$

$$e(5,5),\ e(3,4)+\overset{0}{\cancel{e(6,5)}}\Big\}$$

$$r=5 \quad + p_3+p_4+p_5$$

$$= \min\{0.14,\ 0.08+0.1,\ 0.12\} + 0.2$$

$$= 0.12+0.2$$

$$= 0.32 \qquad \boxed{r=5}$$

$$e(2,5) = \min_{2\le r\le 5}\left\{ \overset{r=2}{\overset{0}{\cancel{e(2,1)}}+e(3,5)},\ \overset{r=3}{e(2,2)+e(4,5)},\ \underset{r=4}{e(2,3)+e(5,5)},\ \underset{r=5}{e(2,4)+\overset{0}{\cancel{e(6,5)}}} \right\} + p_2+p_3+p_4+p_5$$

$$= \min\{0.32,\ 0.06+0.14,\ 0.2+0.1,\ 0.24\} + 0.26$$

$$= 0.2+0.26$$

$$= 0.46 \qquad \boxed{r=3}$$

$$e(1,5) = \min_{1\le r\le 5}\left\{ \overset{r=1}{\overset{0}{\cancel{e(1,0)}}+e(2,5)},\ \overset{r=2}{e(1,1)+e(3,5)},\ \underset{r=3}{e(1,2)+e(4,5)},\ \overset{r=4}{e(1,3)+e(5,5)},\ \underset{r=5}{e(1,4)+\overset{0}{\cancel{e(6,5)}}} \right\} + p_1+p_2+p_3+p_4+p_5$$

$$= \min\{0.46,\ 0.04+0.32,\ 0.14+0.14,\ 0.3+0.1\} + 0.3$$

$$= 0.28+0.3$$

$$= 0.58 \qquad \boxed{r=3}$$

$$e(5,6) = \min_{5\le r\le 6}\Big\{ \overset{r=5}{\cancel{e(5,4)}} + e(6,6),\ \overset{r=6}{e(5,5) + \cancel{e(7,6)}} \Big\} + p_5 + p_6$$

$$= \min\{0.12,\ 0.1\} + 0.22$$

$$= 0.1 + 0.22$$

$$= 0.32 \qquad \boxed{r = 5}$$

$$e(4,6) = \min_{4\le r\le 6}\Big\{ \overset{r=4}{\cancel{e(4,3)}} + e(5,6),\ \overset{r=5}{e(4,4) + e(6,6)},\ \overset{r=6}{e(4,5) + \cancel{e(7,6)}} \Big\} + p_4 + p_5 + p_6$$

$$= \min\{0.32,\ 0.14,\ 0.14\} + 0.24$$

$$= 0.14 + 0.24$$

$$= 0.38 \qquad \boxed{r = 5}$$

$$e(3,6) = \min_{3\le r\le 6}\Big\{ \overset{r=3}{\cancel{e(3,2)}} + e(4,6),\ \overset{r=4}{e(3,3) + e(5,6)},\ \overset{r=5}{e(3,4) + e(6,6)},\ \overset{r=6}{e(3,5) + \cancel{e(7,6)}} \Big\} + p_3 + p_4 + p_5 + p_6$$

$$= \min\{0.38,\ 0.08 + 0.32,\ 0.12 + 0.12,\ 0.32\} + 0.32 \qquad \boxed{r = 5}$$

$$= 0.24 + 0.32 = 0.56$$

$$e(2,6) = \min_{2 \le r \le 6} \Big\{ \overset{r=2}{e(2,\cancelto{0}{1}) + e(3,6)},\ \overset{r=3}{e(2,2)+e(4,6)},\ \underset{r=4}{e(2,3)+e(5,6)},\ \underset{r=5}{e(2,4)+e(6,6)},$$

$$\underset{r=6}{e(2,5)+e(7,\cancelto{0}{6})} \Big\} + p_2 + p_3 + p_4 + p_5 + p_6$$

$$= \min\{0.56,\ 0.06+0.38,\ 0.2+0.32,\ 0.24+0.12,\ 0.46\} + 0.38$$

$$= 0.36 + 0.38 \qquad \boxed{r=5}$$

$$= 0.74$$

$$e(1,6) = \min_{1 \le r \le 6} \Big\{ \overset{r=1}{e(1,\cancelto{0}{0}) + e(2,6)},\ \overset{r=2}{e(1,1)+e(3,6)},\ \underset{r=3}{e(1,2)+e(4,6)},\ \underset{r=4}{e(1,3)+e(5,6)},$$

$$\underset{r=5}{e(1,4)+e(6,6)},\ \underset{r=6}{e(1,5)+e(7,\cancelto{0}{6})} \Big\}$$

$$+ p_1 + p_2 + p_3 + p_4 + p_5 + p_6$$

$$= \min\{0.72,\ 0.04+0.56,\ 0.14+0.38,\ 0.3+0.32,\ 0.36+0.12,\} + 0.42$$

$$0.58$$

$$= 0.48 + 0.42 = 0.9 \qquad \boxed{r=5}$$

$$e(6,7) = \min_{6 \le r \le 7} \{ \underbrace{\overset{0}{\cancel{e(6,5)}} + e(7,7)}_{r=6},\ \underbrace{e(6,6) + \overset{0}{\cancel{e(8,7)}}}_{r=7} \} + p_6 + p_7$$

$$= \min \{ 0.14,\ 0.12 \} + 0.26$$

$$= 0.38 \qquad \boxed{r = 7}$$

$$e(5,7) = \min_{5 \le r \le 7} \{ \underbrace{\overset{0}{\cancel{e(5,4)}} + e(6,7)}_{r=5},\ \underbrace{e(5,5) + e(7,7)}_{r=6},\ \underbrace{e(5,6) + \overset{0}{\cancel{e(8,7)}}}_{r=7} \} + p_5 + p_6 + p_7$$

$$= \min \{ 0.38,\ 0.24,\ 0.32 \} + 0.36$$

$$= 0.24 + 0.36$$

$$= 0.6 \qquad \boxed{r = 6}$$

$$e(4,7) = \min_{4 \le r \le 7} \{ \underbrace{\overset{0}{\cancel{e(4,3)}} + e(5,7)}_{r=4},\ \underbrace{e(4,4) + e(6,7)}_{r=5},\ \underbrace{e(4,5) + e(7,7)}_{r=6},\ \underbrace{e(4,6) + \overset{0}{\cancel{e(8,7)}}}_{r=7} \} + p_4 + p_5 + p_6 + p_7$$

$$= \min \{ 0.6,\ 0.02 + 0.38,\ 0.14 + 0.14,\ 0.38 \} + 0.38$$

$$= 0.28 + 0.38 = 0.66 \qquad \boxed{r = 6}$$

$$r=3 \qquad\qquad r=4$$

$$e(3,7) = \min_{3\le r\le 7} \{ e(3,\cancel{2}^{\,0}) + e(4,7),\ e(3,3)+e(5,7),$$

$$e(3,4)+e(6,7),\ e(3,5)+e(7,7),$$

$$r=5 \qquad\qquad r=6$$

$$e(3,6)+e(\cancel{8,7}^{\,0}) \} + p_3+p_4+p_5+p_6+p_7$$

$$r=7$$

$$= \min\{ 0.66,\ 0.08+0.6,\ 0.12+0.38,$$

$$0.32+0.14,\ 0.56 \} + 0.46$$

$$= 0.46+0.46$$

$$= 0.92 \qquad \boxed{r=6}$$

$$r=2 \qquad\qquad r=3$$

$$e(2,7) = \min_{2\le r\le 7} \{ e(2,\cancel{1}^{\,0}) + (3,7),\ e(2,2)+$$

$$e(4,7),\ e(2,3)+e(5,7),\ e(2,4)+e(6,7),$$

$$r=4 \qquad\qquad r=5$$

$$e(2,5)+e(7,7),\ e(2,6)+e(\cancel{8,7}^{\,0}) \} +$$

$$r=6 \qquad\qquad r=7$$

$$p_2+p_3+p_4+p_5+p_6+p_7$$

$$= \min\{ 0.92,\ 0.06+0.66,\ 0.2+0.6,$$

$$0.24+0.38,\ 0.46+0.14,\ 0.72 \}$$

$$+\ 0.52$$

$= 0.6 + 0.52 = 1.12$ $\boxed{r = 6}$

$$e(1,7) = \min_{1 \le r \le 7} \{ \overbrace{e(1,0) + (2,7)}^{r=1}, \overbrace{e(1,1) + e(3,7)}^{r=2},$$

$$\underbrace{e(1,2) + e(4,7)}_{r=3}, \underbrace{e(1,3) + e(5,7)}_{r=4},$$

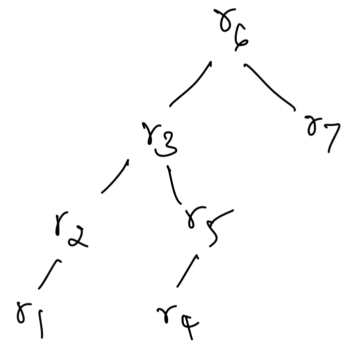
$$\underbrace{e(1,4) + e(6,7)}_{r=5}, \underbrace{e(1,5) + e(7,7)}_{r=6},$$

$$\underbrace{e(1,6) + e(8,7)}_{r=7} \} + P_1 + P_2 + P_3 + P_4 + P_5 + P_6 + P_7$$

$= \min\{ 1.12,\ 0.04 + 0.92,\ 0.14 + 0.66,$
$0.3 + 0.6,\ 0.36 + 0.38,$ $+ 0.56$

$0.58 + 0.14,\ 0.9 \}$

$= 0.72 + 0.56$ $\boxed{r = 6}$
$= 1.28$



Optimal BST

15-1 Longest simple path in a directed acyclic graph

Suppose that we are given a directed acyclic graph $G = (V, E)$ with real-valued edge weights and two distinguished vertices s and t. Describe a dynamic-programming approach for finding a longest weighted simple path from s to t. What does the subproblem graph look like? What is the efficiency of your algorithm?

To find the longest weighted simple path from s to t in a directed acyclic graph $G=(V,E)$ with real-valued weights, we can use dynamnic programming.

$L(v) \rightarrow$ length of the longest path
$w(u,v) \rightarrow$ weight of the edge from u to v

$L(v)$ can be calculated as follows:
1) Set $L(s)$ to 0 and $L(v)$ to $-\infty$ for all other vertices v in V
2) For each vertex v,
For each incoming edge (u,v) in E, update $L(v)$ to $\max(L(v), L(u) + w(u,v))$

3) Return $L(t)$ as length of longest path from s to t

Subproblem:

A subgraph of G that includes all vertices from s. It has the same structure as original graph, but edges and weights are modified based on the above recurrence.

Time complexity $= O(|V| + |E|)$

$|V| \rightarrow$ Vertices in the graph

$|E| \rightarrow$ Edges in the graph

Using depth-first search, a topological sort of vertices can be done in $O(|V| + |E|)$

Then, one pass through each edge takes $O|E|$ time.

$\therefore$ Overall complexity is $O(|V| + |E|)$