National Taiwan University of Science and Technology
Department of Electrical Engineering

Algorithm Design and Application
Sample Solutions to Homework #2

1. (c) **X**, 912 should not be searched after 911.

   (e) **X**, 299 should not be searched after 347.

2. (a) See Figure 1(a).

   (b) See Figure 1(b).

   (c) See Figure 1(c).

   (d) See Figure 1(d).

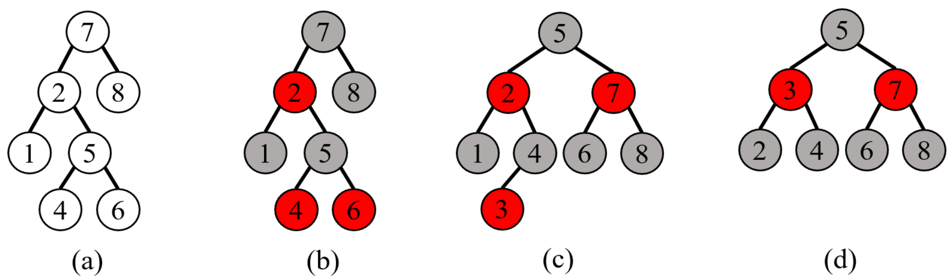

Figure 1: Sample trees for Problem 2.

3. The m and s tables computed by MATRIX-CHAIN-ORDER for $n = 6$ and the sequence of dimensions $< 5, 10, 3, 12, 5, 50, 6 >$ are shown in Figure 2. The minimum number of scalar multiplications to multiply the six matrices is $m[1, 6] = 2010$ and its corresponding parenthesization is $((A_1A_2)((A_3A_4)(A_5A_6)))$.

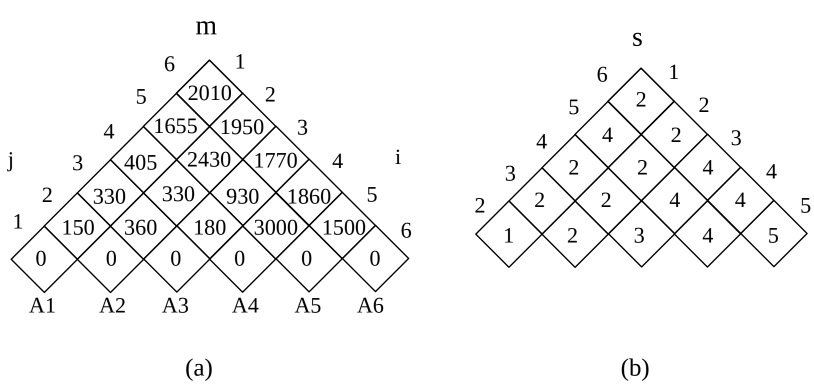

Figure 2: The m and s tables computed by MATRIX-CHAIN-ORDER for $n = 6$ and the sequence of dimensions $< 5, 10, 3, 12, 5, 50, 6 >$.

4. The c and b table computed by LCS-LENGTH on the sequence $X =< 1, 0, 0, 1, 0, 1, 0, 1 >$ and $Y =< 0, 1, 0, 1, 1, 0, 1, 1, 0 >$ is shown in Figure 3. The longest common subsequence of X and Y is $< 1, 0, 0, 1, 1, 0 >$.

i	j	0	1	2	3	4	5	6	7	8	9
	y_j		0	1	0	1	1	0	1	1	0
0	x_i	0	0	0	0	0	0	0	0	0	0
1	1	0	↑ 0	↖ 1	← 1	↖ 1	↖ 1	← 1	↖ 1	↖ 1	← 1
2	0	0	↖ 1	↑ 1	↖ 2	← 2	← 2	↖ 2	← 2	← 2	↖ 2
3	0	0	↖ 1	↑ 1	↖ 2	↑ 2	↑ 2	↖ 3	← 3	← 3	↖ 3
4	1	0	↑ 1	↖ 2	↑ 2	↖ 3	↖ 3	↑ 3	↖ 4	↖ 4	← 4
5	0	0	↖ 1	↑ 2	↖ 3	↑ 3	↑ 3	↖ 4	↑ 4	↑ 4	↖ 5
6	1	0	↑ 1	↖ 2	↑ 3	↖ 4	↖ 4	↑ 4	↖ 5	↖ 5	↑ 5
7	0	0	↖ 1	↑ 2	↖ 3	↑ 4	↑ 4	↖ 5	↑ 5	↑ 5	↖ 6
8	1	0	↑ 1	↖ 2	↑ 3	↖ 4	↖ 5	↑ 5	↖ 6	↖ 6	↑ 6

Figure 3: The c and b table computed by LCS-LENGTH on the sequence $X =< 1,0,0,1,0,1,0,1 >$ and $Y =< 0,1,0,1,1,0,1,1,0 >$.

5. The e, w, and *root* tables computed by OPTIMAL-BST for the given probabilities are shown in Figure 4(a), (b) and (c). The lowest expected search cost of any binary search tree for the given probabilities is 3.12 and the corresponding structure is shown in Figure 4(d).

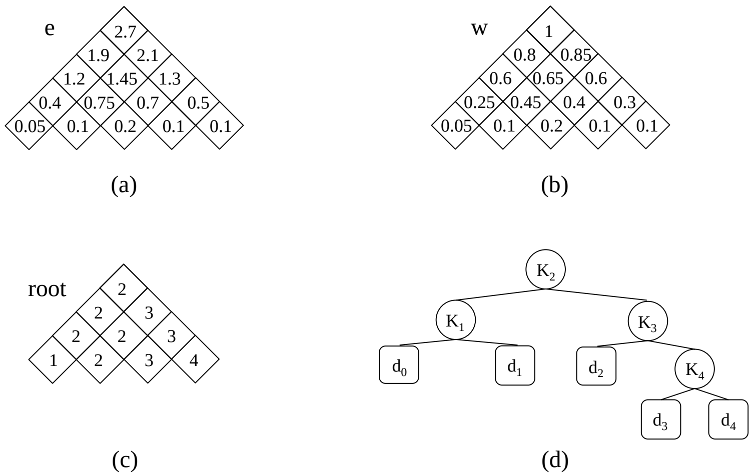

Figure 4: The e, w, and $root$ tables computed by OPTIMAL-BST for the given probabilities, and the structure of a binary search tree with the lowest expected search cost.

6. (a)

$$P[i,j] = \begin{cases} 0 & ,i = j = 0 \\ 1 & ,i = 0, j > 0 \\ 0 & ,j = 0, i > 0 \\ p \times P[i-1,j] + q \times P[i,j-1] & ,i,j > 0 \end{cases}$$

(b) $P[2,2] = 0.4 \times P[1,2] + 0.6 \times P[2,1]$
$= 0.4 \times (0.4 \times P[0,2] + 0.6 \times P[1,1]) + 0.6 \times (0.4 \times P[1,1] + 0.6 \times P[2,0])$
$= 0.4 \times (0.4 \times P[0,2] + 0.6 \times (0.4 \times P[0,1] + 0.6 \times P[1,0])) + 0.6 \times (0.4 \times (0.4 \times P[0,1] + 0.6 \times P[1,0]) + 0.6 \times P[2,0])$
$= 0.4 \times (0.4 \times 1 + 0.6 \times (0.4 \times 1 + 0.6 \times 0)) + 0.6 \times (0.4 \times (0.4 \times 1 + 0.6 \times 0)) + 0.6 \times 0)$
$= 0.4 \times 0.64 + 0.6 \times 0.16$
$= 0.352$
(See Figure 5.)

		j		
		0	1	2
i	0		1	1
	1	0	0.4	0.64
	2	0	0.16	0.352

Figure 5: The probability of the game is show in table which is computed by the recurrence found in (a)

(c) pseudo code

```
Winning(P, i, j):
    let P[0...n, 0...n] be a new table
    let p be the probability
    let q = 1 − p
    for i = 0 to n
        for j = 0 to n
            if j = 0
                P[i, j] = 0
            else if i = 0
                P[i, j] = 1
            else
                P[i, j] = p × P[i − 1, j] + q × P[i, j − 1]
    return P[n, n]
```

The time complexity is $O(n^2)$
The space complexity is $O(n^2)$

7. (a) We denote $S(i, j)$ as whether $< z_1 z_2 ... z_i + j >$ is a shuffle of $< x_1 x_2 ... x_i >$ and $< y_1 y_2 ... y_i >$ $\forall i, j, 1 \leq i \leq m, 1 \leq j \leq n$. The optimal substructure of the problem gives the following recursive formula:

$$S(i,j) = \begin{cases} 1 & , (i = 0 \ and \ j = 1 \ and \ y_1 = z_1) \ or \ (j = 0 \ and \ i = 1 \ and \ x_1 = z_1), \\ 0 & , z_{i+j} \neq x_i \ and z_{i+j} \neq y_j, \\ S(i-1, j) & , z_{i+j} = x_i \ and z_{i+j} \neq y_j, \\ S(i, j-1) & , z_{i+j} = y_j \ and z_{i+j} \neq x_i, \\ S(i-1, j) \vee S(i, j-1) & , z_{i+j} = x_i = y_j. \end{cases}$$

(b) pseudo code

```
Is_Shuffle(X, Y, Z, m, n):
    let S[0...m, 0...n] be a new table
    for i = 0 to m
        for j = 0 to n
            if (i = 0 and j = 1 and y_1 = z_1)or(j = 0 and i = 1 and x_1 = z_1)
                S[i, j] = true
            else if z_{i+j} ≠ x_i and z_{i+j} ≠ y_j
                S[i, j] = false
            else if z_{i+j} = x_i and z_{i+j} ≠ y_j
                S[i, j] = S[i − 1, j]
            else if z_{i+j} = y_j and z_{i+j} ≠ x_i
                S[i, j] = S[i, j − 1]
            else if z_{i+j} = x_i = y_j
                S[i, j] = S[i − 1, j] ∨ S[i, j − 1]
    return S[m, n]
```

(c) The time complexity is $O(mn)$