**Optimal Binary Search Algorithm**

**Dynamic Programming Approach**

**Step 1:** Read n symbols with probability pi.

**Step 2:** Create the table C[i,j] , 1<= i <= j+1 <= n

**Step 3:** Set C[i,j] = pi and C[i-1,j] = 0 for i **∈** [n]

**Step 4:** Recursively compute the following relation

C[i,j] = C [i,k-1] + C [k+1,j] + ∑ Pm for all i and j

**Step 5:** Return C [1….n] as the maximum cost of constructing a BST

**Step 6:** End

**Example:**

Using the dynamic programming approach, construct an optimal BST for the following keys:

Key 1: Danny P1 = 2/7

Key2: Ian P2 = 1/7

Key 3: Radha P3 = 3/7

Key 4: Zee P4 = 1/7

**Solution**

**Step 0:**

Create a table C , Root table R

Dimensions (n+1) x n

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 |
| 1 |  |  |  |  |  |
| 2 |  |  |  |  |  |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
| 5 |  |  |  |  |  |

**Initial Table C R - Root Table**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 |  |  | 0 | 1 | 2 | 3 | 4 |
| 1 | 0 |  |  |  |  |  | 1 | 0 |  |  |  |  |
| 2 |  | 0 |  |  |  |  | 2 |  | 0 |  |  |  |
| 3 |  |  | 0 |  |  |  | 3 |  |  | 0 |  |  |
| 4 |  |  |  | 0 |  |  | 4 |  |  |  | 0 |  |
| 5 |  |  |  |  | 0 |  | 5 |  |  |  |  | 0 |

**Step 1:**

C1,1 = P1 = 2/7

C2,2 = P2 = 1/7

C3,3 = P3 = 3/7

C4,4 = P4 = 1/7

**C - Table R - Table**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 0 | 1 | 2 | 3 | 4 |  |  | 0 | 1 | 2 | 3 | 4 |
| 1 | 0 | 2/7 |  |  |  |  | 1 | 0 | 1 |  |  |  |
| 2 |  | 0 | 1/7 |  |  |  | 2 |  | 0 | 2 |  |  |
| 3 |  |  | 0 | 3/7 |  |  | 3 |  |  | 0 | 3 |  |
| 4 |  |  |  | 0 | 1/7 |  | 4 |  |  |  | 0 | 4 |
| 5 |  |  |  |  | 0 |  | 5 |  |  |  |  | 0 |

**Step 2:**

Compute the next super diagonal [1,2] [2,3] [3,4]

C[1,2] i=1, j=2, Therefore k=1,2

C [i,k-1] + C [k+1,j] + ∑ Pm

C[1,2] = Min C[1,0] + C[2,2] + P1 + P2 When k=1

C[1,1] + C[3,2] + P1 + P2 When k=2

= Min {0 + 1/7+3/7 , 2/7 +0+3/7}

= Min {4/7 , 5/7}

= 4/7

The minimum is when k=1

C[2,3] i=2, j=3, Therefore k=2,3

C [i,k-1] + C [k+1,j] + ∑ Pm

C[2,3] = Min C[2,1] + C[3,3] + P2 + P3 When k=2

C[2,2] + C[4,3] + P2 + P3 When k=3

= Min {0 + 3/7+(1/7 +3/7) , 1/7 +0+(1/7+3/7)}

= Min {7/7 , 5/7}

= 5/7

The minimum is when k=3

C[3,4] i=3, j=4, Therefore k=3,4

C [i,k-1] + C [k+1,j] + ∑ Pm

C[2,3] = Min C[3,2] + C[4,4] + P3 + P4 When k=3

C[3,3] + C[5,4] + P3 + P4 When k=4

= Min {0 + 1/7+(3/7 +1/7) , 3/7 +0+(3/7 +1/7)}

= Min {5/7 , 7/7}

= 5/7

The minimum is when k=3

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **C - Table** | |  |  |  |  |  |  |  | **R - Table** | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 0 | 1 | 2 | 3 | 4 |  |  | 0 | 1 | 2 | 3 | 4 |
| 1 | 0 | 2/7 | 4/7 |  |  |  | 1 | 0 | 1 | 1 |  |  |
| 2 |  | 0 | 1/7 | 5/7 |  |  | 2 |  | 0 | 2 | 3 |  |
| 3 |  |  | 0 | 3/7 | 5/7 |  | 3 |  |  | 0 | 3 | 3 |
| 4 |  |  |  | 0 | 1/7 |  | 4 |  |  |  | 0 | 4 |
| 5 |  |  |  |  | 0 |  | 5 |  |  |  |  | 0 |

**Step 3:**

Compute the next super diagonal [1,3] [2,4]

C[1,3] i=1, j=3, Therefore k=1,2,3

C [i,k-1] + C [k+1,j] + ∑ Pm

C[1,3] = C[1,0] + C[2,3] + P1 + P2 + P3 When k=1

Min C[1,1] + C[3,3] + P1 + P2 + P3 When k=2

C[1,2] + C[4,3] + P1 + P2 + P3 When k=3

= Min {0 + 5/7+(2/7 +1/7+3/7) , 2/7 +3/7 +(2/7 +1/7+3/7) , 4/7+0+(2/7 +1/7+3/7) }

= Min {11/7 , 11/7 , 10/7}

= 10/7

The minimum is when k=3

C[2,4] i=2, j=4, Therefore k=2,3,4

C [i,k-1] + C [k+1,j] + ∑ Pm

C[2,4] = C[2,3] + C[5,4] + P2 + P3 + P4 When k=2

Min C[2,2] + C[4,4] + P2 + P3 + P4 When k=3

C[2,3] + C[5,4] + P2 + P3 + P4 When k=4

= Min {5/7+ 0 + (1/7+3/7 +1/7) , 1/7 +1/7 + (1/7+3/7 +1/7) , 6/7+0+ (1/7+3/7 +1/7)}

= Min {10/7 , 7/7 , 11/7}

= 7/7

The minimum is when k=3

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **C - Table** | |  |  |  |  |  |  |  | **R - Table** | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 0 | 1 | 2 | 3 | 4 |  |  | 0 | 1 | 2 | 3 | 4 |
| 1 | 0 | 2/7 | 4/7 | 10/7 |  |  | 1 | 0 | 1 | 1 | 3 |  |
| 2 |  | 0 | 1/7 | 5/7 | 7/7 |  | 2 |  | 0 | 2 | 3 | 3 |
| 3 |  |  | 0 | 3/7 | 5/7 |  | 3 |  |  | 0 | 3 | 3 |
| 4 |  |  |  | 0 | 1/7 |  | 4 |  |  |  | 0 | 4 |
| 5 |  |  |  |  | 0 |  | 5 |  |  |  |  | 0 |

**Step 4:**

Compute the next super diagonal [1,4]

C[1,4] i=1, j=4, Therefore k=1,2,3,4

C [i,k-1] + C [k+1,j] + ∑ Pm

C[1,3] = C[1,0] + C[2,4] + P1 + P2 + P3 + P4 When k=1

Min C[1,1] + C[3,4] + P1 + P2 + P3 + P4 When k=2

C[1,2] + C[4,4] + P1 + P2 + P3+ P4 When k=3

C[1,3] + C[5,4] + P1 + P2 + P3+ P4 When k=4

= Min {0 + 7/7+(7/7) , 2/7 +5/7 +(7/7) , 4/7+1/7+(7/7) , 10/7+1/7+(7/7) }

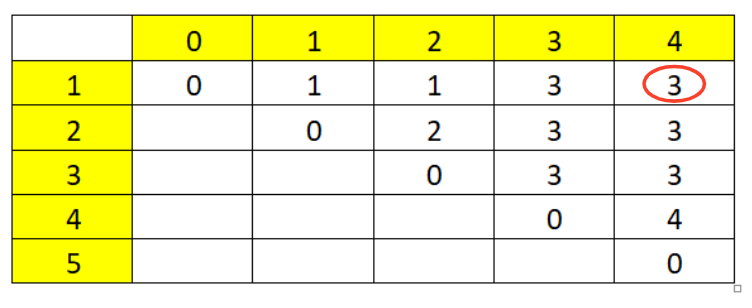
= Min {14/7 , 14/7 , 12/7 , 18/7}

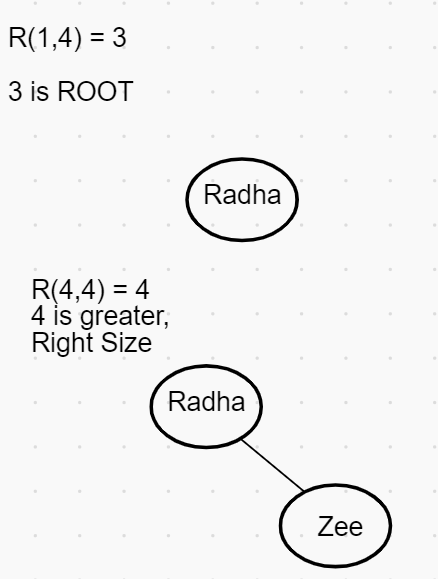
= 12/7

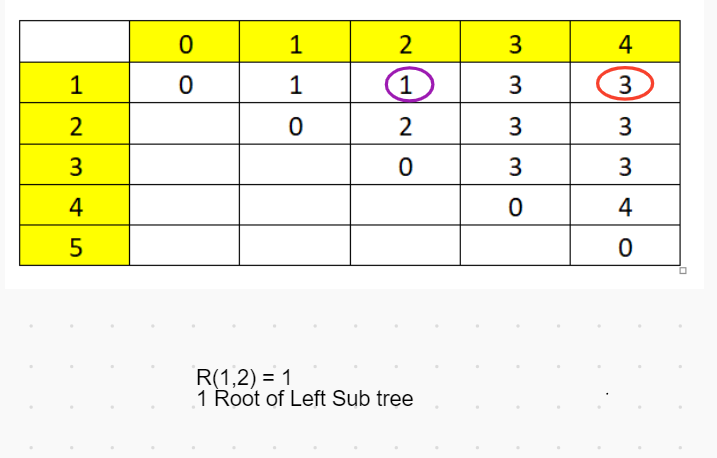
The minimum is when k=3

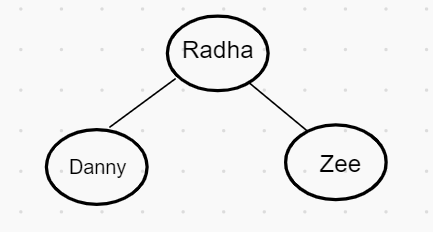
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **C - Table** | |  |  |  |  |  |  |  | **R - Table** | |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | 0 | 1 | 2 | 3 | 4 |  |  | 0 | 1 | 2 | 3 | 4 |
| 1 | 0 | 2/7 | 4/7 | 10/7 | 12/7 |  | 1 | 0 | 1 | 1 | 3 | 3 |
| 2 |  | 0 | 1/7 | 5/7 | 7/7 |  | 2 |  | 0 | 2 | 3 | 3 |
| 3 |  |  | 0 | 3/7 | 5/7 |  | 3 |  |  | 0 | 3 | 3 |
| 4 |  |  |  | 0 | 1/7 |  | 4 |  |  |  | 0 | 4 |
| 5 |  |  |  |  | 0 |  | 5 |  |  |  |  | 0 |

Final BST Tree

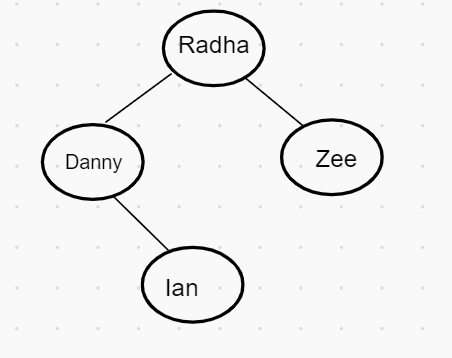








Remaining Only one – Node 2



Key 1: Danny P1 = 2/7 Key2: Ian P2 = 1/7

Key 3: Radha P3 = 3/7 Key 4: Zee P4 = 1/7

Verification:

1 x P3 + 2 x (P1 + P4) + 3 x P2

= 3/7 + 6/7 + 3/7

= 12/7