A.B.B. Dynamic vs. A.B.B. Greedy

Edisson López Díaz

August 5, 2018

1 "Example" mode

Initial problem

Six keys random were generated, with these we proceed to find the optimal solution with the Greedy algorithm and the dynamic programming algorithm.
Keys generated:

Key	Weight	Probability
KeyA	743	0.308043
KeyB	124	0.051410
KeyC	837	0.347015
KeyD	149	0.061774
KeyE	512	0.212272
KeyF	47	0.019486

Forced sum of the probability of the keys: 1.0

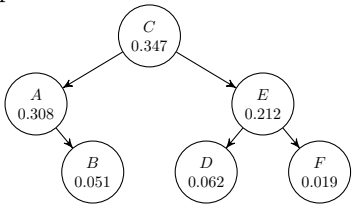
1.1 Greedy Aagorithm

From the previous keys we create an equivalent to the R table used in the dynamic programming algorithm. Choosing each time the key of maximum probability to be the root of the tree, separating the rest of the keys into two groups: the minor ones that the selected root and the majors that this root. This recursively.

Tabla R

•	**0**	**1**	**2**	**3**	**4**	**5**	**6**
1	0	1	**1**				**3**
2		0	2				
3			0	3			
4				0	4		**5**
5					0	5	
6						0	6
7							0

Optimal tree created from table R



Computing time: 39 microseconds

1.2 Dynamic algorithm

From the previous keys the following table is created:

Tabla A

•	**0**	**1**	**2**	**3**	**4**	**5**	**6**
1	0.000	0.308	0.000	0.000	0.000	0.000	0.000
2		0.000	0.051	0.000	0.000	0.000	0.000
3			0.000	0.347	0.000	0.000	0.000
4				0.000	0.062	0.000	0.000
5					0.000	0.212	0.000
6						0.000	0.019
7							0.000

Now proceed to complete Tabla A and R, calculating and choosing the value of each cell of said tables.

Calculation for the cell **[5][6]**:
A[5][4] + A[6][6] = 0.000 + 0.019 = 0.019 *min*
A[5][5] + A[7][6] = 0.212 + 0.000 = 0.212
K winner: 5
Selected: [5][4] + [6][6] + Ks = **0.251244**

Calculation for the cell **[4][5]**:
A[4][3] + A[5][5] = 0.000 + 0.212 = 0.212 *min*
A[4][4] + A[6][5] = 0.062 + 0.000 = 0.062 *min*
K winner: 5
Selected: [4][4] + [6][5] + Ks = **0.335821**

Calculation for the cell **[3][4]**:
A[3][2] + A[4][4] = 0.000 + 0.062 = 0.062 *min*
A[3][3] + A[5][4] = 0.347 + 0.000 = 0.347
K winner: 3
Selected: [3][2] + [4][4] + Ks = **0.470564**

Calculation for the cell **[2][3]**:
A[2][1] + A[3][3] = 0.000 + 0.347 = 0.347 *min*
A[2][2] + A[4][3] = 0.051 + 0.000 = 0.051 *min*
K winner: 3
Selected: [2][2] + [4][3] + Ks = **0.449834**

Calculation for the cell **[1][2]**:
A[1][0] + A[2][2] = 0.000 + 0.051 = 0.051 *min*
A[1][1] + A[3][2] = 0.308 + 0.000 = 0.308
K winner: 1
Selected: [1][0] + [2][2] + Ks = **0.410862**

Calculation for the cell **[4][6]**:
A[4][3] + A[5][6] = 0.000 + 0.251 = 0.251 *min*
A[4][4] + A[6][6] = 0.062 + 0.019 = 0.081 *min*
A[4][5] + A[7][6] = 0.336 + 0.000 = 0.336
K winner: 5
Selected: [4][4] + [6][6] + Ks = **0.374793**

Calculation for the cell **[3][5]**:
A[3][2] + A[4][5] = 0.000 + 0.336 = 0.336 *min*

A[3][3] + A[5][5] = 0.347 + 0.212 = 0.559
A[3][4] + A[6][5] = 0.471 + 0.000 = 0.471
K winner: 3
Selected: [3][2] + [4][5] + Ks = **0.956882**

Calculation for the cell **[2][4]**:
A[2][1] + A[3][4] = 0.000 + 0.471 = 0.471 *min*
A[2][2] + A[4][4] = 0.051 + 0.062 = 0.113 *min*
A[2][3] + A[5][4] = 0.450 + 0.000 = 0.450
K winner: 3
Selected: [2][2] + [4][4] + Ks = **0.573383**

Calculation for the cell **[1][3]**:
A[1][0] + A[2][3] = 0.000 + 0.450 = 0.450 *min*
A[1][1] + A[3][3] = 0.308 + 0.347 = 0.655
A[1][2] + A[4][3] = 0.411 + 0.000 = 0.411 *min*
K winner: 3
Selected: [1][2] + [4][3] + Ks = **1.117330**

Calculation for the cell **[3][6]**:
A[3][2] + A[4][6] = 0.000 + 0.375 = 0.375 *min*
A[3][3] + A[5][6] = 0.347 + 0.251 = 0.598
A[3][4] + A[6][6] = 0.471 + 0.019 = 0.490
A[3][5] + A[7][6] = 0.957 + 0.000 = 0.957
K winner: 3
Selected: [3][2] + [4][6] + Ks = **1.015340**

Calculation for the cell **[2][5]**:
A[2][1] + A[3][5] = 0.000 + 0.957 = 0.957 *min*
A[2][2] + A[4][5] = 0.051 + 0.336 = 0.387 *min*
A[2][3] + A[5][5] = 0.450 + 0.212 = 0.662
A[2][4] + A[6][5] = 0.573 + 0.000 = 0.573
K winner: 3
Selected: [2][2] + [4][5] + Ks = **1.059702**

Calculation for the cell **[1][4]**:
A[1][0] + A[2][4] = 0.000 + 0.573 = 0.573 *min*
A[1][1] + A[3][4] = 0.308 + 0.471 = 0.779
A[1][2] + A[4][4] = 0.411 + 0.062 = 0.473 *min*
A[1][3] + A[5][4] = 1.117 + 0.000 = 1.117
K winner: 3
Selected: [1][2] + [4][4] + Ks = **1.240879**

Calculation for the cell **[2][6]**:
A[2][1] + A[3][6] = 0.000 + 1.015 = 1.015 *min*
A[2][2] + A[4][6] = 0.051 + 0.375 = 0.426 *min*
A[2][3] + A[5][6] = 0.450 + 0.251 = 0.701
A[2][4] + A[6][6] = 0.573 + 0.019 = 0.593
A[2][5] + A[7][6] = 1.060 + 0.000 = 1.060
K winner: 3
Selected: [2][2] + [4][6] + Ks = **1.118159**

Calculation for the cell **[1][5]**:
A[1][0] + A[2][5] = 0.000 + 1.060 = 1.060 *min*
A[1][1] + A[3][5] = 0.308 + 0.957 = 1.265
A[1][2] + A[4][5] = 0.411 + 0.336 = 0.747 *min*
A[1][3] + A[5][5] = 1.117 + 0.212 = 1.330
A[1][4] + A[6][5] = 1.241 + 0.019 = 1.260
K winner: 3
Selected: [1][2] + [4][5] + Ks = **1.727197**

Calculation for the cell **[1][6]**:
A[1][0] + A[2][6] = 0.000 + 1.118 = 1.118 *min*
A[1][1] + A[3][6] = 0.308 + 1.015 = 1.323
A[1][2] + A[4][6] = 0.411 + 0.375 = 0.786 *min*
A[1][3] + A[5][6] = 1.117 + 0.251 = 1.369
A[1][4] + A[6][6] = 1.241 + 0.019 = 1.260
A[1][5] + A[7][6] = 1.727 + 0.000 = 1.727
K winner: 3
Selected: [1][2] + [4][6] + Ks = **1.785655**

After the calculation process, the following tables are obtained:

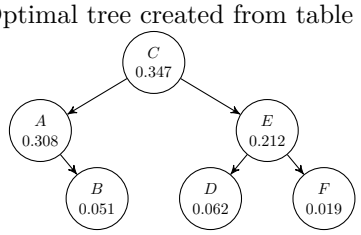
Tabla A

•	**0**	**1**	**2**	**3**	**4**	**5**	**6**
1	0.000	0.308	0.411	1.117	1.241	1.727	1.786
2		0.000	0.051	0.450	0.573	1.060	1.118
3			0.000	0.347	0.471	0.957	1.015
4				0.000	0.062	0.336	0.375
5					0.000	0.212	0.251
6						0.000	0.019
7							0.000

Tabla R

•	**0**	**1**	**2**	**3**	**4**	**5**	**6**
1	0	1	1	3	3	3	3
2		0	2	3	3	3	3
3			0	3	3	3	3
4				0	4	5	5
5					0	5	5
6						0	6
7							0

Optimal tree created from table R



Computing time: 504 microseconds

Conclusions

- The two generated trees (greedy and dynamic programming) share the same topology, so both found the optimal solution.
- The computing time of the dynamic algorithm was 12 was less fast than the Greedy algorithm.
- With six keys the Greedy algorithm approximately the 32% of the time finds the optimal solution.