UWI

Design and Analysis of Algorithms

Tutorial 8 - Sample Solutions

Greedy Approach

It is important that we learn the theory behind the greedy approach.

1. Explain the theory behind the greedy approach

   (a) What is meant by the greedy approach?

   The greedy approach is based on the principle that a sequence of locally optimal choices will yield a globally optimal solution to the entire problem.

   (b) Does the greedy approach always lead to a globally optimal solution.

   No, a sequence of locally optimal choices does not always yield a globally optimal solution.

2. Solve the 0/1 knapsack problem using the greedy approach for the instance shown in table 1 with a knapsack capacity of **12**.

Item	Weight	Value
1	5	\$20
2	3	\$15
3	9	\$30
4	4	\$25

Table 1: A small instance of the 0/1 knapsack problem

To solve the 0/1 knapsack problem using the greedy approach, we compute the value-to-weight ratios and select the items in decreasing order of these ratios. These calculations are shown in table 2.

Item	Weight	Value	$\frac{v_i}{w_i}$
4	4	\$25	$6\frac{1}{4}$
2	3	\$15	5
1	5	\$20	4
3	9	\$30	$3\frac{1}{3}$

Table 2: Value to weight ratios for the above instance of the knapsack problem

From the table we can greedily select items that can hold in the knapsack by decreasing order of value. We select item 4 which gives us a total weight of 4 and a value of $25. We then select item 2, which gives us a total weight of 7 and an overall value of $40. We also select item 1, which maxes out the capacity of 12 with an overall value of $60. Since the knapsack is full, we do not select item 3.

3. Solve the assignment problem using the greedy row-by-row and column-by-column approaches for the instance shown in table 3.

	Job 1	Job 2	Job 3
Person 1	5	7	9
Person 2	3	8	5
Person 3	7	4	9

Table 3: A small instance of the assignment problem

Using the row-by-row approach, we greedily select the most cost effectively job for each person starting with person 1. Thus, we select job 1 for person 1. This means that no one else can get job 1. Given than no one else can get job 1, we select job 3 for person 2. This leaves job 2 for person 3. Using this selection, we end up with a total cost of 14, which is the globally optimal solution.

	Job 1	Job 2	Job 3
Person 1	**5**	7	9
Person 2	3	8	**5**
Person 3	7	**4**	9

Table 4: A solution to the above assignment problem using the greedy row-by-row approach

Using the column-by-column approach, we greedily select the most cost effectively person for each job starting with job 1. Thus, we select person 2 for job 1. Given than no other job can get person 2, we select person 3 for job 2. This leaves person 1 for job 3. Using this selection, we end up with a total cost of 16, which is **not** the globally optimal solution.

	Job 1	Job 2	Job 3
Person 1	5	7	**9**
Person 2	**3**	8	5
Person 3	7	**4**	9

Table 5: A solution to the above assignment problem using the greedy column-by-column approach

4. You are given the following data for this question.

Symbol	A	B	C	D	-
Frequency	0.4	0.1	0.2	0.15	0.15

Table 6: Data for a the question

(a) Construct a huffman code from the above data.

We start by arranging the symbols by increasing order of frequency. Then we group them the two smallest symbols together

Symbol	B	D	-	C	A
Frequency	0.1	0.15	0.15	0.2	0.4

Table 7: Ordering symbols by frequency

Therefore, we group symbols B and D, to form a new group with a frequency of 0.25.

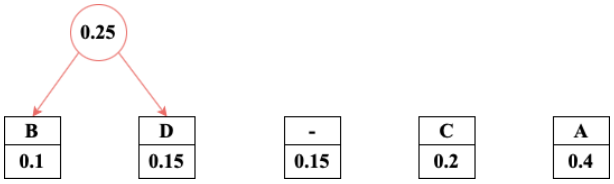


Figure 1: Grouping symbols B and D

Next we observe that symbols - and C have the least frequencies of the new groupings. Therefore, we group those two together to form a frequency of 0.35.

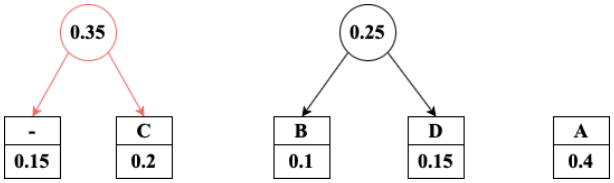


Figure 2: Grouping symbols - and C

Next we group the groups containing B and D, and - and C. This is because they are the two groups with the least frequencies so far.

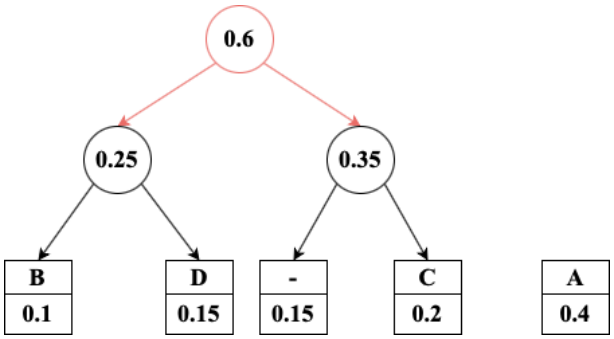


Figure 3: Grouping symbols B and D and - and C

The last step is to group symbol A and the rest. Note that A has a lower frequency than the rest grouped, so it would be on the left. We can also fill in the 1s and 0s for our tree branches.

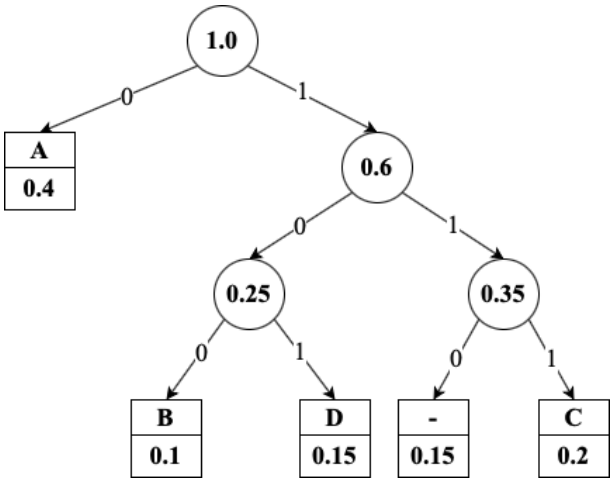


Figure 4: Final grouping

From the above diagram, we can deduce the following values:

Symbol	A	B	C	D	-
Code	0	100	111	101	110

Table 8: Final Codes

(b) Encode ABACABAD using the above generated code.

0100011101000101

(c) Decode 100010111001010 using the above generated code.

BAD-ADA

Exercise

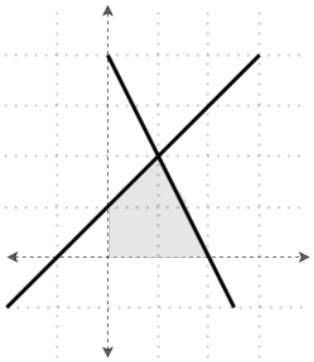
1. Solve the following linear programming problem geometrically. Maximize $3x + y$, subject to $-x + y \leq 1$ and $2x + y \leq 4$, where $x \geq 0$ and $y \geq 0$.

   We must first sketch the problem's feasible region given the problem's constraints.

   For $-x + y \leq 1$, when $x = 0 \implies y = 1$ and when $y = 0 \implies x = -1$.

   For $2x + y \leq 4$, when $x = 0 \implies y = 4$ and when $y = 0 \implies x = 2$.

   We then draw the two lines on a graph and show its feasible region.



   We notice that the edges of the feasible region correspond to the following coordinates: (0,0), (0,1), (2,0) and (1,2). We then calculate the value of $3x + y$ at each of these feasible regions.

Point	Value	
(0,0)	3(0)+(0)	0
(0,1)	3(0)+(1)	1
(2,0)	3(2)+(0)	6
(1,2)	3(1)+(2)	5

Table 9: Values of feasible region coordinates

   We notice from table 9 that point with the highest value is (2,0). This means that this is the optimal solution.

2. Solve the all-pairs shortest-path problem by Floyd's algorithm for the following diagram:

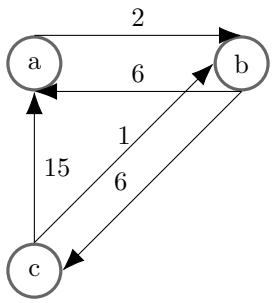


Figure 5: A small instance of the travelling salesman problem

We must first convert the above diagram to a matrix W or $D^{(0)}$.

$$D^{(0)} = \begin{bmatrix} 0 & 2 & \infty \\ 6 & 0 & 6 \\ 15 & 1 & 0 \end{bmatrix}$$

Next we consider intermediate vertices numbered not higher than 1, or just a. We notice that c can go through a to b, but not at a lower cost than 1, so nothing happens.

$$D^{(1)} = \begin{bmatrix} 0 & 2 & \infty \\ 6 & 0 & 6 \\ 15 & 1 & 0 \end{bmatrix}$$

Next we consider intermediate vertices numbered not higher than 2, or a and b. We notice that a can go thorough b to c at a cost of 8. We also notice that c and go through b to a at a lowered cost of 7. We end up with the following matrix.

$$D^{(2)} = \begin{bmatrix} 0 & 2 & 8 \\ 6 & 0 & 6 \\ 7 & 1 & 0 \end{bmatrix}$$