CSE-281: Data Structures and Algorithms

Greedy Methods

Ref: Schaum's Outline Series, Theory and problems of Data Structures
By Seymour Lipschutz

And Online Resource

Eftekhar Hossain
Lecturer
Dept. of ETE, CUET

Topics to be Covered

- Introduction to Greedy Algorithms
- Greedy Choice Property
- Knapsack Problem

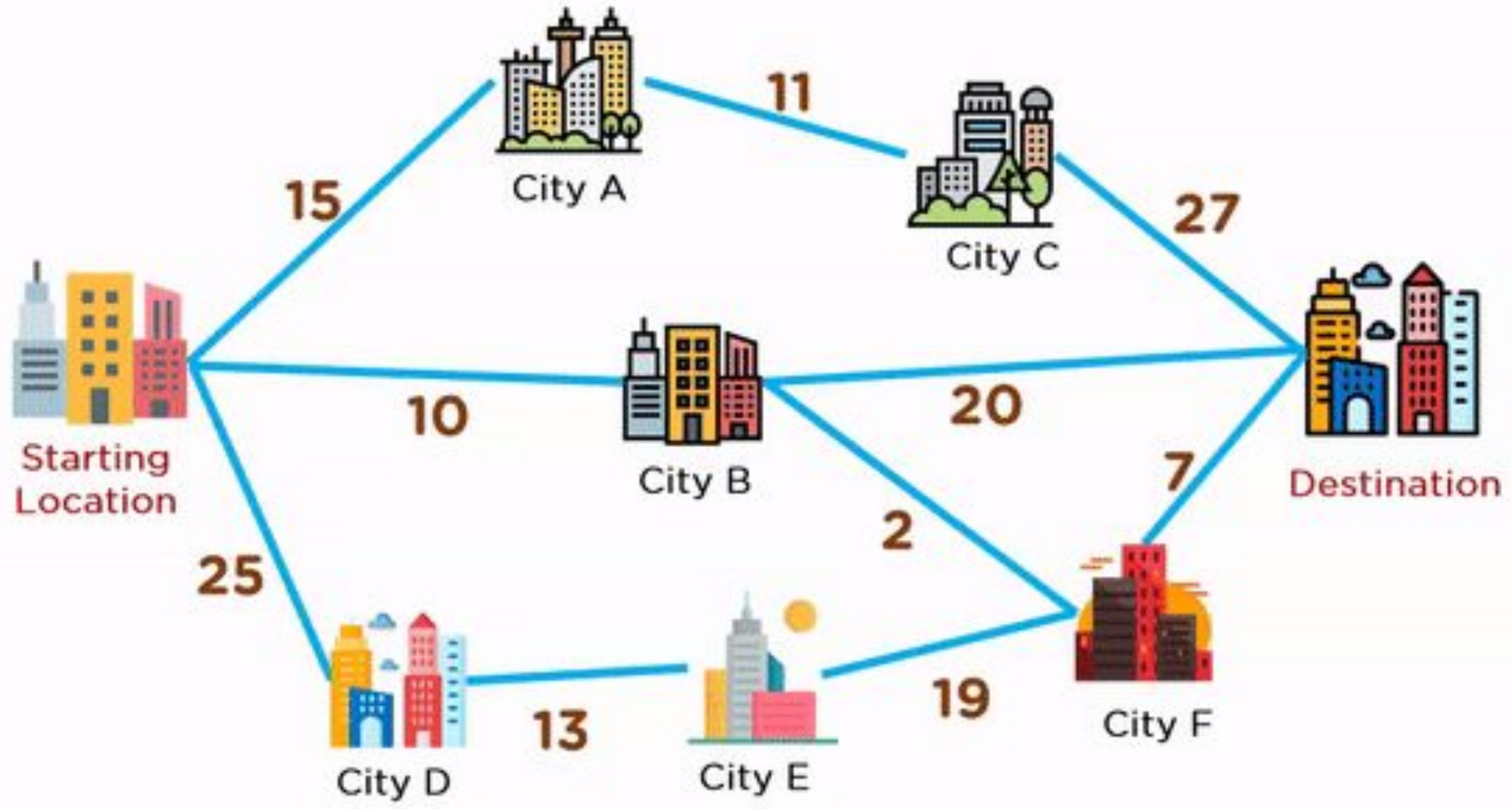
Greedy Method

- A greedy algorithm is an approach for solving a problem by selecting the **best option** available at the moment. It doesn't worry whether the current best result will bring the **overall optimal** result.

- The algorithm *never reverses* the earlier decision even if the choice is wrong. It works in a top-down approach.

- This algorithm may not produce the best result for all the problems. It's because it always goes for the local best choice to produce the global best result.

Greedy Method



Greedy Properties

- However, we can determine if the algorithm can be used with any problem if the problem has the following properties:

1. Greedy Choice Property

- If an optimal solution to the problem can be found by choosing the best choice at each step without reconsidering the previous steps once chosen, the problem can be solved using a greedy approach. This property is called greedy choice property.

Greedy Properties

2. Optimal Substructure

- If the optimal overall solution to the problem corresponds to the optimal solution to its subproblems, then the problem can be solved using a greedy approach. This property is called optimal substructure.

Greedy Method

- Applications:
  - Knapsack
  - Job Sequencing
  - Minimum Spanning Tree
  - Huffman Coding
  - Dijkstra Algorithm
  - Optimal Merge Pattern

Knapsack Problem

The classic Knapsack problem is:

A thief breaks into a store and wants to fill his knapsack of capacity K with goods of as much value as possible.

Decision version: Does there exist a collection of items that fits into his knapsack and whose total value is ***maximized*** ?

- Input
  - Capacity **K**
  - n items with weights w_i and values v_i
- Output: a set of items S such that
  - the sum of weights of items in **S** is at most **K**
  - and the sum of values of items in S is maximized

Fractional Knapsack Problem

Fractional Knapsack Problem: Can items be picked up partially?

The thief's knapsack can hold 100 gms and has to choose from:

30 gms of gold dust at Rs 1000 /gm
60 gms of silver dust at Rs 500/gm
30 gms of platinum dust at Rs 1500/gm

Note: Optimal fills the Knapsack upto full capacity.
Proof: Else the remaining capacity can be filled with some item, picking it partially if the need be.

Fractional Knapsack Procedure

1. Sort the items in the ascending order of value/weight ratio (cost effectiveness).
2. If the next item cannot fit into the knapsack, break it and pick it partially just to fill the knapsack.

- n objects, each with a weight $\mathbf{w_i > 0}$
  a profit $\mathbf{p_i > 0}$
  capacity of knapsack: $\mathbf{M}$

Maximize $\sum_{1 \le i \le n} p_i x_i$

Subject to $\sum_{1 \le i \le n} w_i x_i \le M$

$$\mathbf{0 \le x_i \le 1, \quad 1 \le i \le n}$$

Knapsack Example

Object (X): 1 2 3

Profit (P): 25 24 15

Weight (W): 18 15 10

P/W : 1.32 1.6 1.5

$n = 3$,

Knapsack Capacity $M = 20$

Object (X)	Weight (W)	Profit	Remaining Weight
2	15	24	20- 15 = 5
3	5	15* (5/10) = 7.5	0
	Total Profit =	31.5	

Optimal solution: $x_1 = 0, x_2 = 1, x_3 = 1/2$

Knapsack Example

Solve the Problem

$n = 4,$
$M = 23$

$(V1, V2, V3, V4) = (10, 5, 5, 15)$
$(W1, W2, W3, W4) = (20, 5, 15, 5)$

Thank You