Greedy Method

This is another approach that is often used to design algorithms for solving.

Greedy algorithms **do not always** yield a genuinely optimal solution. In such cases the greedy method is frequently the basis of a *heuristic approach*.
• Even for problems which can be solved exactly by a greedy algorithm, establishing the **correctness** of the method may be a non-trivial process.
The greedy method has,
• Most straightforward design technique
– Most problems have n inputs
– Solution contains a subset of inputs that satisfies a given constraint
– Feasible solution: Any subset that satisfies the constraint
– Need to find a feasible solution that maximizes or minimizes a given objective
Function – optimal solution
• Used to determine a feasible solution that may or may not be optimal
– At every point, make a decision that is locally optimal; and hope that it leads to a globally optimal solution
– Leads to a powerful method for getting a solution that works well for a wide range of applications
– May not guarantee the best solution
• Ultimate goal is to find a feasible solution that minimizes [or maximizes] an objective function; this solution is known as an optimal solution.

Knapsack problem

Knapsack problem is one of the classical optimization problems.
Description: given a set of items each with a cost and a value, determine the number of each item to include in a collection so that the total cost does not exceed some given cost and the total value is as large as possible.

Different variations:

1. Unbounded knapsack problem: no limits on the number of each item
2. 0-1 knapsack problem: number of each item is either 0 or 1.
3. Fractional knapsack problem: number of each item can be a fractional.
4. Subset problem:
    - D Decision problem
    - D 0-1 problem
    - D For each item cost is equal to value

The goal of Knapsack problem is,

- Choose only those objects that give maximum profit.
- The total weight of selected objects should be <=W

Formulation of the problem

Maximize $\sum_{i=1}^{N} P_i X_i$ ……..(1)
Subjected to $\sum_{i=1}^{N} W_i X_i \leq$ M……..(2)
Where ($0\leq X_i \leq 1$ & 1≤Ii≤N)……..(3)

```
Algorithm Knapsack_greedy(W,n)
For i:=1 to n do
If(w[i]<W) then
  X[i]=1
  W=W-w[i]
If(i<=n) then
  X[i]=W/w[i];
```

Example:

Consider that there are three items. Weight and profit value of each item is as given below. Find optimal solution for the Fractional knapsack problem Where N=3:

i	Wi	Pi
1	18	25
2	15	24
3	10	15

And weight W=20.

Solution:

X1,X2,X3	$\sum_{i=1}^{3} W_i X_i \le 20$	$\sum_{i=1}^{3} P_i X_i$	The Criterion
1,2/15,0	20	28.2	Selection according to higher profit
0,2/3,1	20	31	Selection according to less weight
0,1,1/2	20	31.5	Selection according to higher profit per unit weight P_i/W_i

- Selection according to higher profit **:** In this case X1 has Maximum profit therefore
  - ✓ X1=1, — 2 unit place left 20-18=2
  - ✓ X2=2/15 — X[i]=W/w[i]
  - ✓ X3=0, — Where cannot place item 3 in bag

- Selection according to less weight:
  - ✓ X3=1 — less weight (20-10=10)
  - ✓ X2= 10/15=2/3
  - ✓ X1=0, — Where cannot place item 1 in bag

- Selection according to higher profit per unit weight P_i/W_i

  P1/W1=25/18=1.4

  P2/W2=24/15=1.6

  P3/W3=15/10=1.5

  X2=1 (because P2/W2 is maximum) 20-15= 5, Next X3 is maximum

  X3=5/10=1/2

  X1=0, Where cannot place item 1 in bag

Let us compute $\sum wixi$

1. 2. 1*18+2/15*15+0*8=20
2. 0*18+2/3*15+10=20
3. 0*18+1*15+1/2*10=20

Let us compute $\sum Pixi$

1. 2. 1*30+2/15*21+0*18=32.8
2. 3. 0*30+2/3*21+18=32
3. 4. 0*30+1*21+1/2*18=30