Greedy Algorithms

A greedy algorithm selects the best available options at each step, without reconsidering previous choices, aiming local optimization.

Top-down approach :-

→ Works on top down approach, makes decision based on current best options.

→ The algorithm doesn't reverse decisions even if they turn out to be sub optimal.

Elements of Greedy Approach :-

1. Greedy Choice Property : Optimal decisions made at each step
2. Optimal Substructure : subproblems contribute to global optimum.
3. Greedy Strategy : Rule for selecting the best option at each step.
4. Local Optimization : Focus on immediate best solution
5. No Backtracking : Decisions are final can get reversed.

Advantage :-

→ Easy and Fast

→ Sometimes performs better than other algorithms sometimes.

Caching using Greedy Algorithm :-

The caching problems arise from limitation of finite space.

Assume cache 'C' has k pages

C = | A | B | C | D | E |

Now we want to process a sequence of m items which must be placed in cache before they are processed.

If $m <= k$

m = | A | F |

C = | F | B | C | D | E |

* A can be put into cache without any problem.
* F for F since it's not in cache remove A from cache and place F

If $m > k$

→ cache hit when item is only present in both.
→ cache miss if not present in cache.
In cache miss ; bring the requested item to cache, ~~then~~ and evict another.

Greedy Strategies

FIFO :- Oldest page gets evicted
LIFO :- The newest page gets evicted.

Fractional Knapsack Problem :-

The FKP is a classic optimization problem where the goal is to maximize the total value (profit) of items selected, giving constraint on maximum weight the knapsack can hold.

Three Greedy Approaches :-

1. Selecting the items based on maximum profit :-
   (i) Sort the items in highest to low profit.
   (ii) Start selecting one by one until knapsack is full.
   (iii) If the knapsack is not completely filled, consider the
   (iv) fractional parts of next items.

2. Selecting the items based on minimum weight :-
   (i) Sort the items in asending order of their weight.
   (ii) Start selecting items one by one until knapsack is full.
   (iii) If the knapsack is not completely filled, consider fractional parts of the next item.

3. Selecting items based on profit to weight ratio :-
   (i) Calculate the profit-to-weight ratio for each item.
   (ii) Sort the items in descending orders of this ratio.
   (iii) Start selecting items one by one until the knapsack is full
   (iv) If the knapsack is not completely filled, consider fractional parts of the next item.

* Greedy approach, particularly using the profit-to-weight ratio, is preffered because of near optimal solution.

Applications :-

1. Resource allocation.
2. Scheduling
3. optimization

Example :-

Items	Profit (P)	Weight (w)	P/w
1	5	1	5/1 = 5
2	10	3	10/3 = 3.3
3	15	5	15/5 = 3
4	7	4	7/4 = 1.7
5	8	1	8/1 = 8
6	9	3	9/3 = 3
7	4	2	4/2 = 2

Weight of Knapsack (w) = 15
no. of items (n) = 7

Arranging table with P/w descending items. →

Item	Profit	Weight	Remaining weight
5	8	1	15 - 1 = 14
1	5	1	14 - 1 = 13
2	10	3	13 - 3 = 10
3	15	5	10 - 5 = 5
6	9	3	5 - 3 = 2
7	4	2	2 - 2 = 0

* Item 4 is not added as the knapsack is filled adding item '4' will result to overflow.

Total Profit :- $8+5+10+15+9+4=51$

Time Complexity :-

$O(n \log n)$

Selection step → sorting

P.T.O