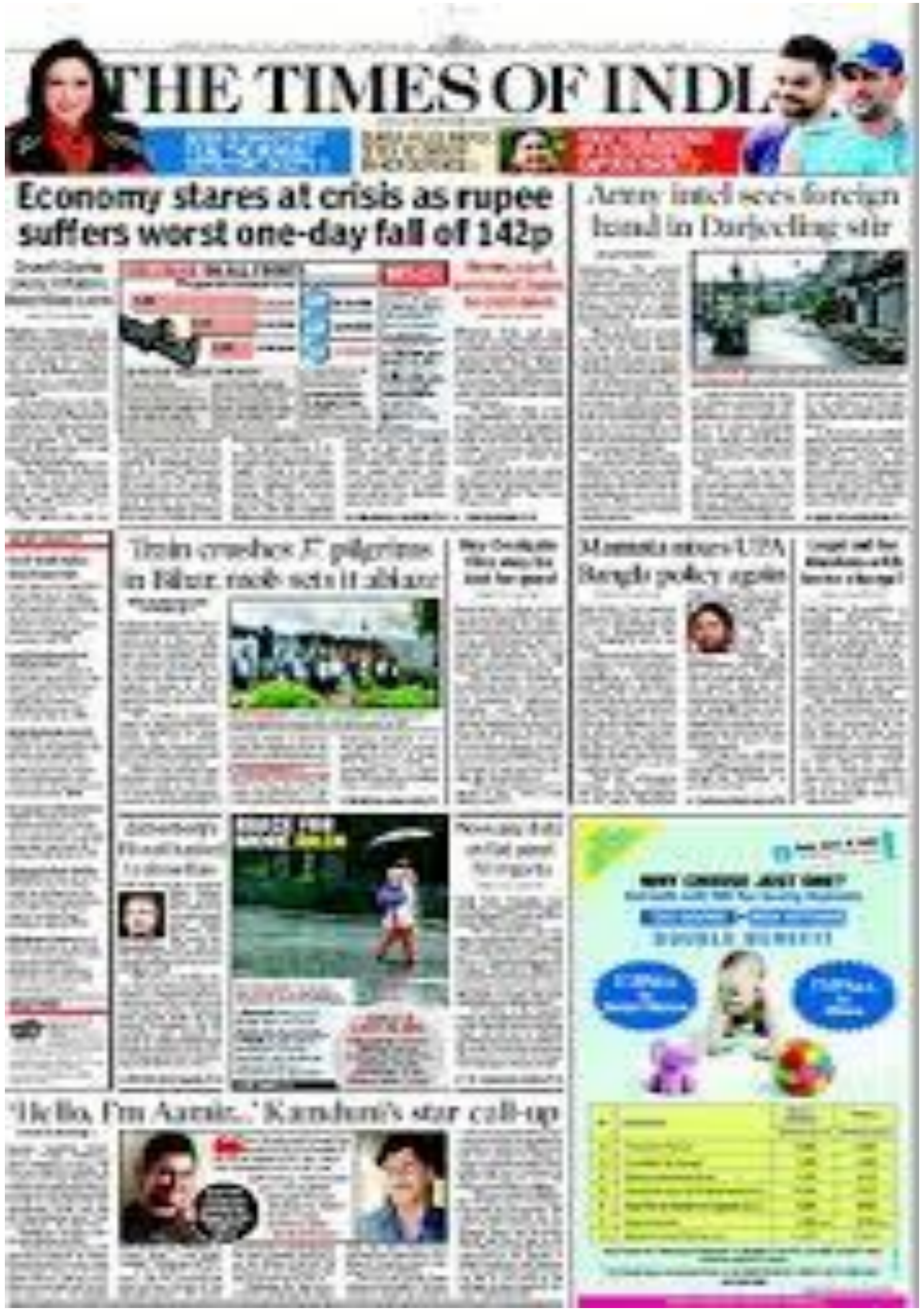
Knapsack Problem


THE TIMES OF IND

Economy stares at crisis as rupee suffers worst one-day fall of 142p

Army intel sees foreign hand in Darjeeling stir

Train crushes 37 pilgrims in Bihar, mob sets it ablaze

Mamata nixes UPA Bangla policy again

'Hello, I'm Aamir..' Kamduni's star call-up

WHY CHOOSE JUST ONE?

DOUBLE BENEFIT

- An editor has 5 news items. The table shows the importance of each item (between 1 and 10) and number of lines for each item.

news	lines	importance
1	120	5
2	150	5
3	200	4
4	150	8
5	140	3

- The problem is to edit the report, by deleting lines or rewriting, so that overall importance is maximized, limiting the total lines to 600 (Capacity is 600).
- Let x_i be the amount of news used (number of lines used/number of lines), and let p_i be the importance of that news, and let the capacity be C.
- The importance of the resulting draft would be $\sum x_i p_i$.
- The problem is to maximize this sum subject to the overall news capacity C.

- This is an example of class of problems known as Knapsack Problem.

- Given n objets and a knapsack of capacity C,
- where each object has weight w_i ,and earns profit p_i ,
- find values of x_i that maximizes the total profit $\sum x_i p_i$,
- subject to the constraint that $\sum x_i w_i < C$.

- The constraint guarantees that the capacity of the knapsack is not exceeded.

Knapsack Problem.

- Given n objets and a knapsack of capacity C, where each object has weight w_i . and earns profit p_i , find values of x_i that maximizes the total profit $\sum x_i p_i$, subject to the constraint that $\sum x_i w_i < C$.
- A greedy algorithm would examine the objects with maximum possible profit to decide how much of each object be selected.

- Since we want to maximize the profit and minimize the weight, we should maximize the ratio of profit to weight.
- Greedy algorithm selects the objects in terms of decreasing order of this ratio.
- If Capacity of Knapsack is not exceeded, It selects all objects.
- If it is going to exceed, then take whatever portion of last object can be taken to fill the knapsack.

- We now calculate profit to weight ratio for the news items.

news	lines	importance	ratio p/w
1	120	5	5/120 = .0417
2	150	5	.0333
3	200	4	.0200
4	150	8	.0533
5	140	3	.0214

We select the items in order of 4,1,2, 5, 3

Knapsack weight = 150+120+150+140 = 560.

Now we cannot take all of item 3,

so we take only 40/200 (that is 0.2) of item 3.

- We now calculate profit to weight ratio for the news items.

news	lines	importance	ratio p/w	profit
1	120	5	5/120 = .0417	5
2	150	5	.0333	5
3	200	4	.0200	0.2x4= 0.8
4	150	8	.0533	8
5	140	3	.0214	3

We select the items in decreasing order of 4,1,2, 5, 3

Knapsack weight of items 4,1,2, 5 = 150+120+150+140 = 560.

Now we cannot take all of item 3,

so we take only 40/200 (that is 0.2) of item 3.

Total Profit = 21.8

- Prob 2
- 3 Sugar bottles wiith weight and profit given below. Fill up sugar in bottle of size 20.

items	Weight	Profit	ratio p/w
1	14	24	1.71
2	18	24	1.11
3	10	16	1.6

We select the items in order of 1, 3, 2

Knapsack Capacity = 20.

Knapsack Capacity after item 1 = 20 – 14 = 6.

we can take only 6/10 of item 3.

Knapsack Capacity after item 3 = 0, (No space for item 2)

NET PROFIT: 24 + 16 (0.6) = 24 + 9.6 = 33.6

Complexity of Knapsack

- All profits need to be sorted.
- Sorting step takes O(n log n)
- Looping through all the items takes O(n) time.
- So overall complexity is O(nlog n).

- A brute force approach would involve trying out all possible combinations of weights.
- For n items, there can be 2^n subsets of weights, $O(2^n)$ is practically impossible to solve if n is large.
- If n is 20, it is close to 1 million steps.

- These type of problems are known as Continuous Knapsack problems.
- The greedy approach generates the optimal (best) solution .
- However, there is another type of Knapsack problems for which solution may not be optimal.

- https://www.javatpoint.com/fractional-knapsack-problem
- https://www.tutorialspoint.com/design_and_analysis_of_algorithms/design_and_analysis_of_algorithms_fractional_knapsack.htm

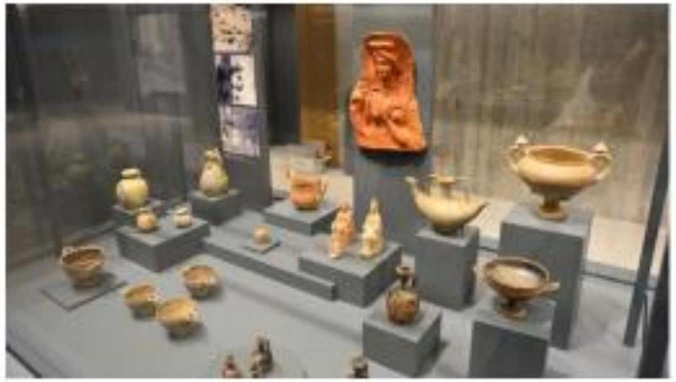
The 0/1 Knapsack Problem

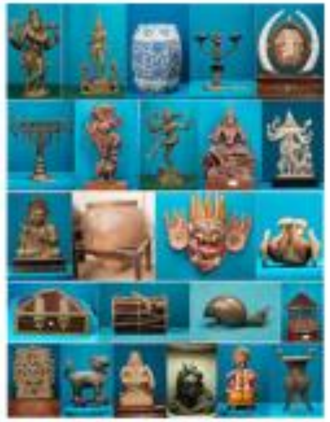
- The 0/1 Knapsack problem is slightly different.
- Here you cannot take portion of an item.
- You either take the whole item, or leave it.

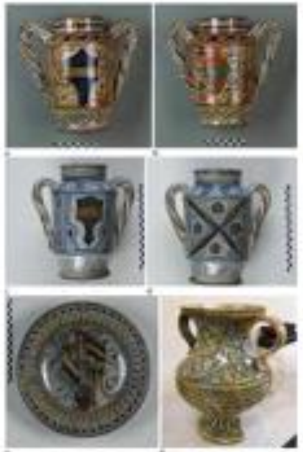
Thief in a museum

What should he take?

- The burglar wishes to carry away the most valuable items subject to the weight constraint.
- No point taking fraction of an object, so he must make a decision to take the object entirely or leave it







- Prob 3 (0/1 Knapsack Problem)
- A thief wants to maximize his profit by stealing items from a museum. He has got a bag which can hold 60 Kg of goods.

items	Price	weight	ratio p/w
1	1000	10	100
2	2800	40	70
3	1200	20	60

The Greedy approach selects the items in order of 1, 2, 3

Knapsack Capacity = 60.

Knapsack Capacity after item 1 = 60 – 10= 50.

Knapsack Capacity after item 2 = 50 – 40= 10.

Profit after selling the items : 1000+2800 = 3800.

However, this is not the Optimal solution (best solution).

The best solution is to take items 2 and 3, which gives a profit of 4000.

- Thus the Greedy approach may sometimes fail for 0/1 Knapsack problems.

Job Scheduling Problem

- The goal is to schedule the jobs such that task completion profit is maximized.
- The job needs to be completed before deadline for the job.

- A sequence of jobs is feasible if all the jobs end by their deadline.
- A set of jobs is called a feasible set, if at least one sequence is possible.
- The sequence associated with maximum profit is called the optimal sequence.

- You are given a set of n jobs.
- Each job takes **one** unit of time to complete, and *only one job can be scheduled at a time.*
- Suppose the unit is one day.
- Each job has a deadline. If deadline is 2, it means job can be delivered after 2 days.
- We earn the profit associated with a job, if and only if the job is completed by its deadline.
- Find the job scheduling of the given jobs that ensure maximum profit.

- Consider a problem of scheduling 4 jobs on 5th Feb.

Job	Deadline	Profit
1	2	60
2	1	30
3	2	40
4	1	80

- Consider job Sequence {1,2}. Job 1 is to be delivered on 7th Feb and Job 2 is to be delivered on 6th Feb.
- Job 1 can be scheduled on 5th Feb. This would be delivered on 6th Feb.
- Job 2 can be scheduled only after job 1 is delivered. But job 2 cannot be scheduled as it is already 6th Feb.
- So sequence {1, 2} is not feasible.
- For same reason sequences {1,4}, {2,4}, {1,2,3} ,. . . . . . . .are not feasible

Job	Deadline	Profit
1	2	60
2	1	30
3	2	40
4	1	80

- Consider now the job Sequence {2,1}. Job 2 is to be delivered on 6th Feb and Job 1 is to be delivered on 7th Feb.
- Job 2 can be scheduled on 5th Feb. This would be delivered on 6th Feb.
- Job 1 can be scheduled after that (on 6th Feb), and this can be delivered on 7th Feb in time.
- So both jobs will be delivered in time. We say that sequence {2, 1} is feasible.

- Here are the feasible sequences and corresponding profits.
- Job sequence **Total Profit**

Job sequence	Total Profit	
2,1	90	(60 + 30)
1,3	100	
2,3	70	
4,1	140	
4,3	120	

- Max profit is associated with Sequence {4,1}.
- Optimal order is Sequence {4,1}.
- Note that Sequence {1,4} is not feasible, as deadlines are violated.

Greedy approach to Scheduling Algorithm

- First, sort the jobs in the decreasing order of their profits.
- Then find the highest deadline among all deadlines.
- Next, we need to assign time slots to individual job ids.
- First, check if the maximum possible time slot for the job, i.e., its deadline, is assigned to another job or not.
  - If it is not filled yet, assign the slot to the current job id.
  - Otherwise, search for any empty time slot less than the deadline of the current job.
  - If such a slot is found, assign it to the current job id, and move to the next job id.

- Consider a problem of scheduling 5 jobs.
-

Job	Deadline	Profit
1	2	100
2	1	19
3	2	30
4	1	25
5	3	15

- Sort the jobs in decreasing order of profit.
-

Job	Deadline	Profit
1	2	100
3	2	30
4	1	25
2	1	19
5	3	15

- Consider first the job with highest profit (job 1)
- Next highest profitable job is job 3. In 2 days, job 1 and job 3 can be delivered.
- However, job 4 and job 2 cannot be scheduled as they don't have enough deadline.
- But job 5 can be scheduled, as its deadline permits job to be scheduled on 3^{rd} day.
- So { 1, 3, 5} is an optimal solution. Note {3, 1, 5} is also an optimal solution.

complexity analysis

Main computation of scheduling problem is the sorting step

- This takes O(n log n) time.
- All other steps take constant time.
- Therefore complexity of scheduling problem is O(n log n).

- read up more about this topic:
- https://www.tutorialspoint.com/design_and_analysis_of_algorithms/design_and_analysis_of_algorithms_job_sequencing_with_deadline.htm