COL 351 Lecture 16 2023/02/15

Topic : Fractional Knapsack
Assignment Scheduling

Fractional Knapsack

Input: n divisible objects ; W : capacity of knapsack
p_i : price of ith object w_i : weight of ith object.

Output: Pick fraction x_i of object i such that total weight $\leq W$, total profit is max.

i.e. $x_1 \cdots x_n$ so that $\sum_i w_i x_i \leq W$

$x_i \in [0,1]\ \forall i$, $\max \sum_i p_i x_i$

Suppose $x_i^* \cdots x_n^*$ is optimum solution. $\therefore \sum_i w_i x_i^* = W$

Suppose $x_i^* > 0$ and $x_j^* < 1$

Remove δ_i amount of i, add δ_j amount of j

$\delta_i \leq x_i^* \quad \delta_j \leq 1 - x_j^*, \quad w_j \delta_j = w_i \delta_i \longrightarrow ①$

Δ profit $= -p_i \delta_i + p_j \delta_j$ $\therefore$ Exchange profitable iff $p_j \delta_j > p_i \delta_i \to ②$

$② / ① \Rightarrow \dfrac{p_j}{w_j} > \dfrac{p_i}{w_i}$

For optimality, we need:

$\dfrac{p_j}{w_j} > \dfrac{p_i}{w_i} \Rightarrow x_i^* = 0$ or $x_j^* = 1$.

Greedy Algorithm for Fractional Knapsack:

1. Sort items in non-inc. order of p/w.
2. For each i in the above sorted order:

    If i can fit into the knapsack, put it in.

    Else Pack the max possible amount of i in the knapsack and break.

Proof template for greedy algorithms:

Claim: $\forall i$, $\exists$ an opt solution, say OPT_i, which agrees with the greedy alg's first i decisions.

Prove by induction on i; base case $i=0$ obvious.

For $i = n$ (# decisions), claim implies $ALG = OPT_n$ is optimal.

Claim: Suppose

$$\frac{p_1}{w_1} \geq \frac{p_2}{w_2} \geq \cdots \geq \frac{p_n}{w_n}$$

Let ALG's solution be $(y_1, y_2, \ldots, y_n)$.

$\forall i \; \exists \, x_{i+1} \ldots x_n$ such that

$OPT_i = (y_1, y_2, \ldots y_i, x_{i+1}, \ldots x_n)$ is an opt solution.

Proof: Induction on i; Base case $i=0$ is obvious.

IH: $\exists$ an opt solution $OPT_{i-1} = (y_1, \ldots, y_{i-1}, x_i, x_{i+1} \ldots x_n)$

$x_i \leq y_i$ by def of alg.

If $x_i = y_i$, take $OPT_i = OPT_{i-1}$; claim proved.

Suppose $x_i < y_i \leq 1$

Since both ALG and OPT_{i-1} fill up the knapsack completely, $\exists \, j > i$ such that $x_j > y_j \geq 0$

Remove δ amount of j, from OPT_{i-1} add $\frac{\delta w_j}{w_i}$ amount of i, for

$$\delta = \min\left(x_j, \frac{w_i}{w_j}(1 - x_i)\right) > 0$$

$$\Delta \text{ value} = -p_j \delta + \frac{\delta w_j}{w_i} p_i = \delta w_j \left(\frac{-p_j}{w_j} + \frac{p_i}{w_i}\right) \geq 0$$

Can use this process to increase x_i' to y_i, while maintaining optimality. Call the resulting solution OPT_i

Assignment Scheduling

Input: n assignments.

Assignment i has deadline d_i, value p_i

Each assignment takes 1 day.

Output: Subset of assignment to be done + schedule, which maximizes total value.

(You get p_i for each assignment i that you finish before deadine d_i)

Ideas? Observations? Exchange argument?