CPSC 320 2020S2: Tutorial 4

1 Knapsack with Structured Weights: Part 1

The *Knapsack Problem* is a classic optimization problem in computer science. The most common version is as follows: consider a set of n items, numbered $1, ..., n$. Each item i has a positive weight $w_i > 0$ and a positive value $v_i > 0$. Furthermore, we're given an overall weight capacity C. The problem is to select a subset $S \subseteq \{1, ..., n\}$ that maximizes the total value of S, but keeps the total weight below C. Formally, we want to maximize $\sum_{i \in S} v_i$ subject to $\sum_{i \in S} w_i \leq C$.

The knapsack problem is known to be hard: there are no algorithms known that find the optimal solution to all instances in better than worst-case exponential time (we'll study this concept more at the end of the course, when we consider NP-completeness).

However, for this tutorial question, we'll consider a variation of the problem. In particular, we'll set the capacity as $C = 1$, and restrict that all weights are either 1/2, 1/4, 1/8, or 1/16. An intuitive greedy approach you might think of to try first is by sorting the items in decreasing order of v_i/w_i, which is the "value density" of each item. Then, go down the list in decreasing order, selecting as many items as fit.

1. Give a small instance of the problem where this greedy algorithm achieves an optimal solution.
2. Give a small instance of the problem where this "value density" greedy algorithm does **not** give an optimal solution. What goes wrong with this greedy approach?

Again, the original Knapsack Problem is a "hard" problem and an efficient algorithm may not exist. However, thanks to the weight restrictions, there *is* an efficient greedy algorithm that gives an optimal solution to this version of the problem! We'll derive this algorithm and prove that it works.

To avoid special cases in our proofs, assume that we always have an unlimited supply of extra items whose value is 0 in every weight class

3. Prove that there is always an optimal solution that uses an **even** number of 1/16-weight items.
4. Prove that if an optimal solution uses c items of weight 1/16, it uses c of the highest value 1/16-weight objects.

2 Knapsack with Structured Weights: Part 2

To be continued next week...

Ruolin Li
31764160

Tutorial 4

2020/07/17

1. maximize the total value, but keep the total weight below C.

1) items for each item, it has a value and a weight and the weight capacity = 1.

greedy algorithm: look for the largest value density.

Weights (w_i) $w_i = \{1/2, 1/4, 1/8, 1/16\}$.

items	1	2	3
value	2	1	1/2
weight	1/2	1/2	1/2
value density	4	2	1

→ $S = \{1, 2\}$ since the weight capacity = 1, we only can choose 2 items from $\{1, 2, 3\}$.

→ We choose the items with large value density.

2)

items	1	2	3	4
value	3	4	2	2
weight	1/2	1/2	1/4	1/4
τ	6	8	8	8

The optimal solution $S = \{2, 3, 4\}$.

items	1	2	3
value	3	3	2
weight	1/2	1/2	1/4
τ	6	6	8

greedy algorithm: → it has a larger value density.

① We choose 3 first, then we can choose 2. $S = \{2, 3\}$ $\sum_{i \in S} v_i = 5$

② $S = \{1, 2\}$ $\sum_{i \in S} v_i = 6$. ← This is the optimal solution.

Because the value and the weight is small compare to others.

3) Suppose that in that optimal solution we have x_1 items of weight 1/2, x_2 items of weight 1/4, x_3 items of weight 1/8, and x_4 items of weight 1/16.

Because:

Total selected weight $= \frac{1}{16}(8x_1 + 4x_2 + 2x_3 + x_4) \leq 16/16$ and x_1, x_2, x_3 are integers and all multiplied by an even number, so $[8x_1 + 4x_2 + 2x_3 + x_4]$ is always even.

We want to show that x_4 is always even in optimal solution.

→ If x_4 is odd, then $[8x_1 + 4x_2 + 2x_3] + x_4$ is odd as well, and which is ≤ 15.

If we have odd number x_4, we still have at least 1/16 space to spare. Assuming we have unlimited supply, then we can add another 1/16 (weight), so the original solution is not optimal.

4) Assume not, such that we have chosen c items of weight 1/16 but not the highest value c items of that weight in an optimal solution.

→ We can always swap the lowest-valued 1/16 th item in our solution with another higher-valued 1/16 th weight item.

We can get another solution with a higher total value, so the original solution is not optimal, which is a contradiction.