EECS3101 notes:: Greedy algorithms

Jerry Wu

2022-03-09

The Knapsack problem (discrete) continued

Problem description

> *"I try to be vague about the class average so students don't obsess over their ranking in the class"-Larry YL Zhang 2023*

Given n items, each having an integer weight w_i and value v_i where $i \in [1, n]$ and given a knapsack with weight capacity W. Select items to fill the knapsack such that the total value is maximized.

There are two versions of this problem::

- Continuous:: can take any real valued fraction of any item. A greedy algorithm would work.
- Discrete:: Each item is either taken or not taken. Would require dynamic programming.

Example

Given a list of boxes with a capacity ($w \in \mathbb{N}$) and a whole dollar value ($v \in \mathbb{N}$), find the highest value combination of boxes to fill the knapsack. Capacity of the bag is 50, i.e. $W = 50$.

Box 1:: $w = 10, v = 95$

Box 2:: $w = 10, v = 95$

Box 3:: $w = 10, v = 95$

Box 4:: $w = 10, v = 95$

Box 5:: $w = 10, v = 95$

Box 6:: $w = 20, v = 180$

Box 7:: $w = 45, v = 450$

Optimal substructure

Suppose we know that item at index $k \in \mathbb{N}$ is selected. Then the solution for $W - w_k$ must be optimal.

Subproblems

We can express the answer to the subproblems $ans[w][i]$ as the maximum value of all the items contained within the bag of capacity w using items 1 through i.

$$Ans(i, w) = \begin{cases} 0 & i = 0 \vee w = 0 \\ Ans(i-1, w) & w_i > w \\ \max\{Ans(i-1, w), v_i + Ans(i-1, w - w_i)\} & w \leq w_i \end{cases}$$

"Please talk to me if you didn't do as well as you wanted to on the midterm. My office hours are like a therapy session."-Larry YL Zhang 2023

So from here, we ask ourselves the following::

- Which cell has the final answer?
- What is the order to fill the array?
- What is the space/time complexity of the solution? We have that the complexity is $\Theta(n \times w)$

Greedy algorithms

The general idea

"Every 2 year old knows the greedy algorithm"-S. Datta, LE EECS

- In order to get an optimal solution, we can just keep grabbing what looks best.
- No backtracking, i.e. once a choice is made, never revert it.
- The trickier part is to **prove** that the solution produced by the greedy algorithm is indeed **optimal**.

Difference between DP and greedy

- In DP, we do not know what an optimal solutions like *a priori*, so we need to **search** a range of possibilities (the recursion tree).
- In greedy, we believe we know a best choice already, and we focus on finding that best choice.

We can represent the general idea through pseudocode.

```
while(!requirement is met):
    choose next best object
    if it conflicts with a previous chosen object:
        reject the object
    else:
        commit to the object
```

Invariant

- After each iteration, $\exists$ at least one optimal solution that is consistent with the choices made so far.
- This invariant is the key to proving the optimality of the greedy solution.
- We often prove this by contradiction.
- The invariant is known as the **greedy choice property**.

Example 1: A trivial proof

- Given a set of n integers ($n > 2$), find the maximum sum of any $\frac{n}{2}$ elements selected from the set.
- By way of common sense, we can just sort the integers in decreasing order and pick the $\frac{n}{2}$ largest elements. However, we need to prove this!

Proof

Define array SG as the choices made in the greedy solution, and SO to be the choices made by an optimal solution. Assume $SO \neq SG \wedge Opt(SO) > Opt(SG)$. Sort both arrays in decreasing order and let k be the first index where $SG[k] \neq SO[k]$. We construct an SO' such that $Opt(SO') > Opt(SO)$.

Going back to the definition of greedy, if we arrive at $SG[k]$, then $SG[k] > SO[k]$ is indeed true since greedy entails that the element at the current index is the **best/largest** possible value, thus we can place it at $SO'[k]$. Because SO' will be sorted in descending order, it is completely fine to make this placement, whereby we arrive at our contradiction. ■

Coin change

Given a set of denominations (quarters, dimes, nickels, and pennies), find the minimum number of coins that total to a desird amount of money. For example::

> *"We can assume we have infinite money here."-Larry YL Zhang 2023*

- Denominations: 1,3,4 cents
- What would be the greedy solution?

- What is the optimal solution for making 6 cents?
- Does the greedy solution give the optimal solution?
- What if we used real life denominations? (1,5,10,25) We can actually always use the greedy approach for the denominations we have. Let us prove it.

Proof that capitalism is built on greed

Assume denominations are 1,5,10,25 cents.

Consider the greedy solution SG and assume a different, more optimal solution SO for the sake of contradiction. Sort both in **descending order**. Knowing that $SG[k]$ is always the largest possible value up to k, we know $SG[k] > SO[k]$ due to the affomentioned property. Take the following cases to construct SO':

- $SG[k] = 1$, no $SO[k]$. Every coin following the first one would be 1 as well.
- $SG[k] = 5, SO[k] = 1$. Everything following the 5 would be a 1, so we can merge exactly 5 of the 1s into a 5.
- $SG[k] = 10, SO[k] = 5$. Everything following the 10 would be 5s followed by 1s, so we can merge the 1s into 5s and 5s into 10s.
- $SG[k] = 25, SO[k] = 10$. Same principles, but merging 10s and 5s into 25s.

This solution is indeed more optimal than SO. So it follows that we have a contradiction, thus SG is optimal. ■

The continuous knapsack problem

We will now revisit the knapsack problem.

Given n items, each having an integer weight w_i and value v_i where $i \in [1, n]$ and given a knapsack with weight capacity W. Select items to fill the knapsack such that the total value is maximized. We can take any real valued fraction of the item this time.

To do this, we can sort each item by a ratio of value to weight i.e. the density of value (descending order of $\frac{v_i}{w_i}$). Let us prove that we can solve it with a greedy approach.

Construct SG such that each element contains a ratio of value and density, and a hypothetical SO that is more optimal for the sake of contradiction.

$$SG[i] = \frac{v_{g,i}}{w_{g,i}} \forall i, SO[i] = \frac{v_{o,i}}{g_{o,i}} \forall i \implies Opt(SO) > Opt(SG)$$

We can conclude that if the weight of a bigger item has a higher density of value than one of a smaller size, we cannot replace the smaller item. However, since we are in a continuous context, we can take a part of the item, i.e. cut it off and replace it with the smaller sized item. **A good example would be replace 1 gram of silver with 1 gram of gold.** So we now have a contradiction, since::

$$v_o \to \frac{v_g}{w_g} \times w_o > v_o \blacksquare$$

We apply the same principles for the case when $w_o \leq v_o$.