Homework NUMBER 5
Colorado CSCI 5454

Ashwin Viswamithiran asvi7266

September 28, 2022

People I studied with for this homework: Rithik Kumar, Sruthi Sampath Kumar

Other external resources used:

- Office Hours

,

1 Problem 1 (12 points)

Recall the knapsack problem (no duplicates):

- **Input**: values v1,...,vn; weights w1,...,wn; weight limit W .
- **Output**: subset of items with total weight W .
- **Objective**: maximize total value of the subset.

Assume that $w_i \leq$ W for all i, i.e. each item can fit on its own. To start, consider the following algorithm, Greedy, that tries to add items in order of the most "bang per buck" (value per weight):

- For each item i, let $ai = vi/wi$.
- Sort the items from largest ai to smallest.
- Add items in this order until the next one does not fit; then stop.

1.1 Part a (2 points)

Consider the following input: $v_1 = 1$, $w_1 = 1$; $v_2 = 5$, $w_2 = 10$; $v_3 = 9$, $w_3 = 10$; $W = 10.1$.

- What is the value of the optimal solution and which items are in it?
- What is the value of Greedy's solution and which items does it choose?
- What is the ratio of Greedy to optimal in this example?

Solution:

- To get the optimal solution we cannot pick up w_1 because if we pick it then the two remaining weights both are 10 and it will exceed 10.1 so, we will choose the weight w_3 as 10 with value v_3 as 9, hence the optimal value is 9.
- Let a_i be the ratio of v_i and w_i, following it in decreasing order of a_i for all the weights, the order is a_1, a_3 and a_2 (or) v_1/w_1, v_3/w_3 and v_2/w_2. That is ...
  - 1/1, 9/10 and 5/10

  Now, following the greedy approach we will have to pick up v_1 and w_1 because a_1 is the highest, now since we have already picked up w_1 we cannot pick up any of the remaining weights because then the total weight will exceed 10.1, hence the maximum value following greedy approach is 1.
- The ratio of Greedy to Optimal in this example would be: 1/9.

1.2 Part b (2 points)

Now consider a hypothetical algorithm CheatingGreedy. This is the same as Greedy, but it gets to include the last item in the loop, the one that does not fit, in its solution. So CheatingGreedy actually will violate the weight constraint W , but analyzing it will help us find a good non-cheating algorithm. First, what is the performance of CheatingGreedy on the example in part (a)?

Solution:

Let a_i be the ratio of v_i and w_i, following it in decreasing order of a_i for all the weights, the order is a_1, a_2 and a_3 (or) v_1/w_1, v_2/w_2 and v_3/w_3. That is ...

- 1/1, 9/10 and 5/10

Here the CheatingGreedy algorithm will take the weights in the above given order, so first it will take up v_1 and w_1 then the total weight added would be 1 and and the next it will take w_2 and the total weight added will become 11 which is greater than 10.1 and the algorithms ends here.
Hence, the answer obtained using CheatingGreedy algorithm would be 10.

1.3 Part c (4 points)

Next, argue that CheatingGreedy's performance is at least as large as Opt. Hint: if we let W' be the total weight used by CheatingGreedy, first argue that CheatingGreedy achieves the optimal performance for constraint W'.

Solution:

Let's try to solve this:

$$val = \sum_{i \in S} v_i \leq V_1$$

Taking the following conventions:
Items in CheatingGreedy be j. CheatingGreedy = {1,...,j} (any number of these j items)

We will try to prove that CheatingGreedy Algorithm is at least as good as S.
Let us take X to be the set of items in CheatingGreedy.

$X^{'}$ = set of bad items in S

$$\begin{aligned} X &= \text{CheatingGreedy} \cap S \\ &\Rightarrow \{1, ..., j\} \cap S \\ X^{'} &= S \backslash X \qquad\qquad (\forall k \in X^{'}, a_k \leq a_j) \end{aligned}$$

Items in $A = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}$, CheatingGreedy $= \{1, 2, 3, 4\}$ and $S = \{1, 3, 7, 8, 9\}$
Let's take the above assumption and,

$$X = \{1, 3\}$$

.

$$X^{'} = \{7, 8, 9\}$$

. Let Y be the set of rest of the items.

$$\begin{aligned} Y &= \text{CheatingGreedy} \backslash X \\ &= \{2, 4\} \end{aligned}$$

We know, $\forall\ i \in Y \text{and} i \leq j, a_i \geq a_j$

$$\therefore a_i \geq a_j \geq a_k$$

Now, we need to prove CheatingGreedy $\geq S(W^{'})$ To prove,

$$\sum_{i \in X} v_i + \sum_{i \in Y} v_i \geq \sum_{k \in X'} v_k + \sum_{i \in X} v_i$$

$$\begin{aligned}
\sum_{i\in X} v_i &= \sum_{i\in Y} a_i w_i && (a_i = \frac{v_i}{w_i}) \\
&\geq \sum_{i\in Y} a_j w_i && (a_i \geq a_j) \\
&\geq a_j(rest\ of\ \ the \text{ CheatingGreedy}) \\
\text{Hence} \sum_{i\in X} v_i &\geq a_j(rest\ of\ \ the \text{ CheatingGreedy})
\end{aligned}$$

$$\begin{aligned}
\sum_{k\in X'} v_k &= \sum_{k\in X'} a_k w_k && (a_k = \frac{v_k}{w_k}) \\
&\leq \sum_{k\in X'} a_j w_k && (a_j \geq a_k) \\
&\leq a_j(rest\ of\ \ the\ S) \\
\text{Hence} \sum_{k\in X'} v_k &\leq a_j(rest\ of\ \ the\ S)
\end{aligned}$$

Since we have this above two conclusions, we can say that

$$\sum_{i\in Y} v_i \geq \sum_{k\in X'} v_k$$

CheatingGreedy uses the capacity $W^{'} \geq W$.
Thus, we can conclude CheatingGreedy $\geq OPT(W^{'})$
$\therefore OPT(W^{'}) \geq OPT(W)$
If capacity $= W^{'}$, we can use the solution of $OPT(W)$.
therfore we can conclude that CheatingGreedy $\geq\ OPT$

1.4 Part d (2 points)

Our final algorithm is called CarefulGreedy. First, it runs Greedy. Let i be the first item that Greedy cannot fit. Let V be the total value of the items taken by Greedy. Then CarefulGreedy outputs:

- just item i, if $v_i \geq V$;
- the Greedy solution, otherwise.

Solution:

Let a_i be the ratio of v_i and w_i, following it in decreasing order of a_i for all the weights, the order is a_1, a_2 and a_3 (or) v_1/w_1, v_2/w_2 and v_3/w_3. That is ...

- 1/1, 9/10 and 5/10

Following the greedy approach, the algorithm will take first the v_1, w_1 and ends. But the CarefulGreedy will compare the values of v_1 with the next weight, now since $w_1 = 1$ is lesser than the $v_3 = 9$, the algorithm will take that item for the previous hence the answer obtained using CarefulGreedy alogorithm will be 9.

1.5 Part e (2 points)

Prove that CarefulGreedy has an approximation ratio of 0.5.

- Hint 1: first show that CarefulGreedy $\geq$ 0.5 CheatingGreedy.
- Hint 2: what is the performance of CheatingGreedy in terms of V and v_i ?

Solution:

Let us take the following conventions:
V_1 be the total value of Greedy algorithm for taking 1,2.....A_n items for a weight capacity of W.
V_2 b the total value of CheatingGreedy algorithm for taking 1,2.....A_n,A_{n+1} items for a weight capacity of W'.

Let V_c be total value for CarefulGreedy algorithm.
Now, we will go through case by case for CarefulGreedy:
Case: 1 $V_1 = V_2$.

$$Careful/cheating = V_c/V_2$$
We know that $V_1 = V_2$ and $V_c = V_1 + V_2$
Which implies,
$$Careful/cheating = V_2/2V_2$$
$$CheatingGreedy = (2) * CarefulGreedy$$

Case: 2 $V_1 > V_2$

$$Careful/cheating = V_c/V_2$$

2 Problem 2 (8 points)

The **Online Vertex Cover** problem is as follows.
The instance consists of an unweighted, undirected graph $G = (V, E)$, initially hidden. Each round, one edge of the graph is revealed. The algorithm must maintain a vertex cover of the revealed graph at all times.
After each edge arrives, the algorithm may add any vertex seen so far to its set. However, it may never remove any vertex once added.

2.1 Part a (4 points)

Consider the following algorithm:

- Initially, the vertex cover W is empty.
- When an edge arrives, if neither endpoint is in W , then pick one endpoint and add it to W .

Prove that this algorithm does not guarantee a competitive ratio of C for any C.
Solution:

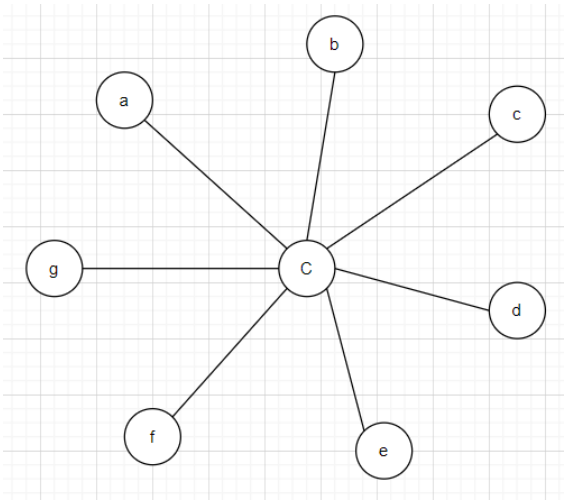

Figure 1: Star graph with a central node as C

Let's take this graph with a central node for proof.
At the beginning the vertex cover W is empty.
Whenever there is an incoming of a new edge, if both of the endpoint(vertices) are in W, we add either of the vertex to our online vertex.

In this graph:
when edge {d,f} is incoming and in the worst case edge f is added to the online vertex set.

similarly, we have C edges and in every case, the selection of algorithm is such that it prefers the radial node over the central node.
At the completion of it the graph should contain C edges connecting the central node with each radial node and also the number of vertices in W would be C.

For this graph the optimal number of elements in W should be 1, but the algorithm returns us C,
hence the competitive ratio of the given Algorithm to OPT is C.

Now lets take the case where we try to increase the vertices by one by adding another edge between the central node and the new incoming radial node.
So now the updated number of vertices = C+1.
Now lets assume the algorithm takes the worst case scenario of adding only the radial nodes to W which implies the number of elements in W is increased by 1 which is C+1 but the OPT will still hold 1.
hence the competitive ratio of the given Algorithm to OPT is C+1.

This example proves that there is no fixed competitive ratio of C for any C.

Part b (4 points)

Give a different algorithm for online vertex cover and prove that it is has a competitive ratio of 2.

Solution:
We make these claims to come up with an algorithm for online vertex cover and prove that the algorithm gives a competitive ratio of 2.

Claim 1: For any matching M in a graph,

$$|M| \leq |OPT| \text{ (Here our optimal solution is the minimum vertex cover)}$$

Proof: Lets take the below graph for example,

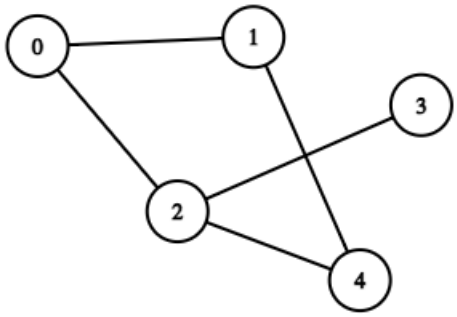

Figure 2: Example graph

Let's imagine that when we use the greedy algorithm in this situation to discover the maximum matching, the output is choosing any edge between 0 and 2 from the two sets. One of the vertices between 2 and 3 must be added in order to attain vertex cover, however it cannot be added because it violates the matching definition.
In situations where a particular vertex does not belong in the matching, we should add one of the vertex of that particular edge to achieve vertex cover.
For a graph, the vertex cover will at least contain all the vertices that are included in the maximum matching.

Claim 2: In order to obtain the vertex cover, we must include both endpoints of all edges that are present in M.

Proof: We can generate a vertex cover by using the ends of both matching vertices after determining the maximum matching M for a graph.

As we include both vertices in a matching, the size of our vertex cover is 2*$|M|$ and we can conclude that Greedy also gives 2*$|M|$.

From claim 1, we know that OPT $\geq$ M, with which we can prove that OPT $\geq \frac{1}{2}$*Greedy.

Rearranging the above statement gives us 2*OPT $\geq$ Greedy. We know an algorithm is

C-competitive from notes if there is a guarantee of approximation by factor C such that ALG $\leq$ C* (OPT). As a result, we have provided proof to support our statements about the online vertex cover, which has a ratio of 2.