APPROX. ALGOS (chapter 11.)

Sec. 11.1 Greedy lower bounds
11.3 Weighted set cover

A LOAD BALANCING PROBLEM: (NP-HARD!)

M machines
N Jobs

each job has a weight $w_i > 0$
i.e. $w_1, \ldots, w_N > 0$.

GOAL: Minimize the maximum load of any one machine.

THIS PROBLEM IS NP-HARD

[Note: This is not a decision problem. BUT you can embed a decision problem into this].

A problem is NP hard if a polynomial time solution for this problem implies $P = NP$.

An aside:

"SUBSET-SUM" is NP-COMPLETE

INPUT: $w_1, \ldots, w_n \geq 0$ TARGET W

Q: Is there a subset of $\{w_1, \ldots, w_n\}$ s.t. the sum of the elements of this subset is equal to W?

EXERCIZE: SUBSET SUM $\leq_p$ PARTITION.

"PARTITION" - problem: INPUT: $w_1, \ldots, w_n \geq 0$

Does there exist some subset of w_i's such that they add up to $\sum w_i / 2$?

We can reduce the PARTITION PROBLEM to the LOAD BALANCING PROBLEM. this shows that the L.B.P. is NP-Hard.
$\Rightarrow$ It is unlikely there is a polytime algorithm.

We will give an approximation algorithm that gives us the right answer upto a factor of α.

i.e. $$[\text{OUR ANSWER}] \leq \alpha \times [\text{CORRECT ANSWER}]$$

called an "α-approximation algorithm".
(note: $\alpha \geq 1$).

This gives us a 2-approximation

Algorithm 1

1. Process the jobs one at a time in arbitrary order
2. To process the job, put it on the least loaded machine.

Analysis : Note: Actually a proof by induction

Let L^* be the optimal max load.

Suppose we put a job of weight w on a machine with weight L at some point.

claim $w \leq L^*$ and $L \leq L^*$

Proof idea: The first inequality is easy.

Second inequality follows from an averaging argument. (Note that the total load is $\geq mL$, and so $L^* \geq mL/m$)

claim : $2L^* \geq w + L$

Pf : from previous claim.

Since this holds at every step, this holds at the final step. Which means this holds for our algorithm and our answer $\leq 2L^*$.

Gives us a 1.5 – approx algo.

Algorithm 2

1. Sort weights in non-increasing order
2. Process the job by putting it on the least loaded machine.

Analysis :

Similar to previous case. Analyze the algorithm at an arbitrary step. Suppose you put a weight ω on a machine of load L

Case 1 $\quad L = 0$

$$L + \omega = \omega \leq L^*$$

Case 2 $\quad L > 0$

At least $m+1$ jobs have the weight $\geq \omega$

$\Rightarrow \quad L^* \geq 2\omega$

$\therefore \quad \omega \leq L^*/2$

still have $\quad L \leq L^*$

then $\quad L + \omega \leq L^* + \dfrac{L^*}{2} = 1.5\ L^*$

WEIGHTED SET COVER

This is NP-HARD. How you show this is:

WEIGTED SET COVER $\geq_p$ SET COVER $\geq_p$ VERTEX COVER

PROBLEM:

U : universe of size m

$S_1, \ldots, S_k$ are subsets of U. such that $\bigcup_{i=1}^{k} S_i = U$.

each S_i has a weight $w_i > 0$.

Def. A subset of $\{S_1, \ldots, S_K\}$ (for example $\{S_1, S_7, S_5\}$) is a "set cover" if $S_1 \cup S_5 \cup S_7 = U$.

Def. The weight of a set cover is the sum of the weights of the sets involved in the set cover. (For instance, in the above example, weight of $\{S_1, S_5, S_7\}$ is $w_1 + w_5 + w_7$.)

GOAL Find a set cover $S_{\alpha_1}, \ldots, S_{\alpha_t}$ s.t. $\sum_{i=1}^{t} w_{\alpha_i}$ is minimized.

Algorithm to solve weighted set cover:

Idea: Repeatedly apply a $\boxed{\text{greedy rule}}$ to select the next S_i until you get a set cover.

Q: What greedy rule?

A: " choose S_i that minimizes $\dfrac{w_i}{|S_i|}$ "

where $|S_i|$ = number of elements in S_i.

Algorithm:

* At each stage pick the S_i that minimizes $\dfrac{w_i}{|S_i \cap T|}$

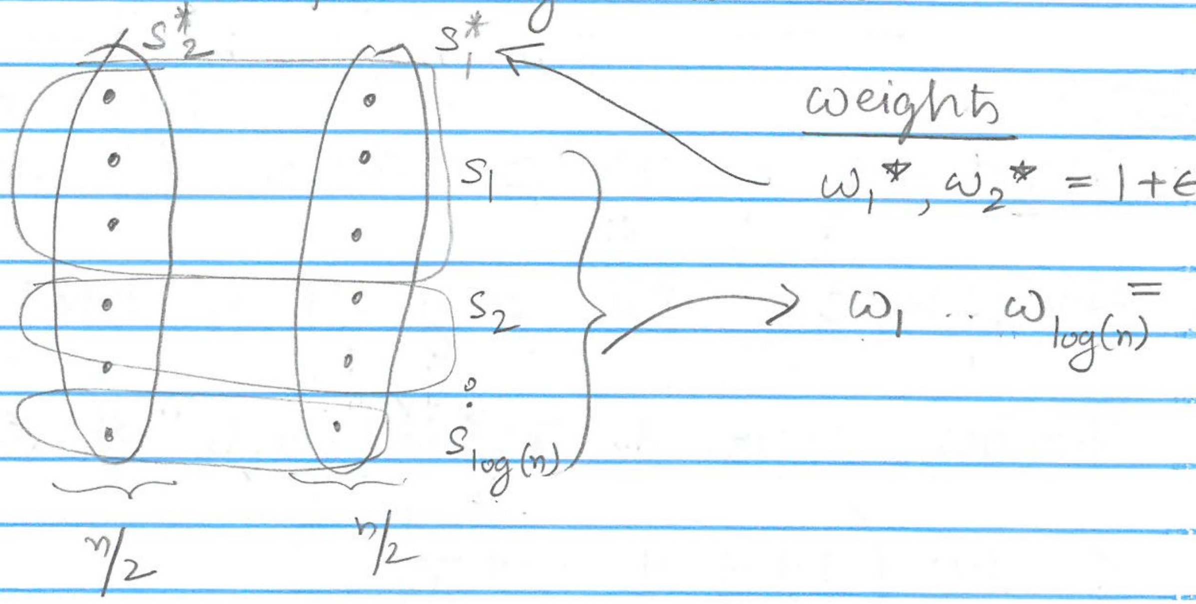
where T = set of uncovered elements.

i.e. we only consider the number of newly covered elements by S_i.

NOTE : $\Omega(\log n)$ lower bound :

(i.e. the greedy algorithm gives an α-approx algo where $\alpha > \log(n)$)

Consider the following instance.



Greedy will never pick S_1^* or S_2^* but will keep picking S_i's

$\therefore$ $OPT = w_1^* + w_2^* = 2(1+\epsilon)$

Greedy answer $= \log(n)$.

tldr; $\longrightarrow$ We cannot hope to prove that our greedy algorithm gives us something better than a $\log(n)$ - approx.

Analysis :

We get a $O(\log(n))$ approximation ratio.

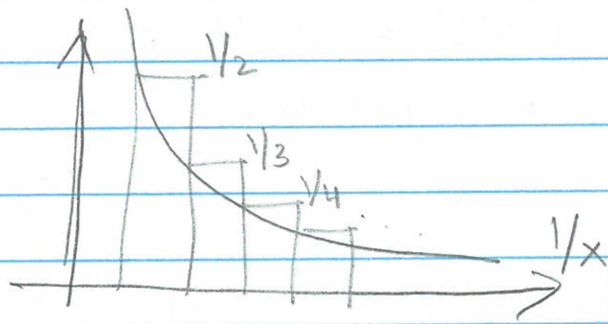
→ i.e. YOUR ANSWER $\leq O(\log(n))$ CORRECT ANS.

idea : "CHARGING SCHEME" → "How much do you charge for elements covered?"

MAIN LEMMA : For any set S_i total of charges assigned to elements of S_i

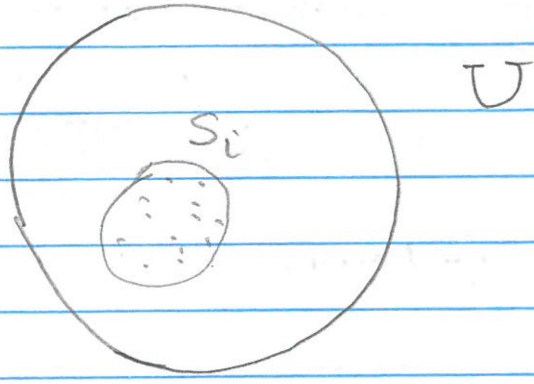
$$\leq w_i \left(1 + \frac{1}{2} + \cdots + \frac{1}{|S_i|}\right)$$

BUT $\displaystyle\sum_{n=1}^{k} \frac{1}{n} < \log(k)$



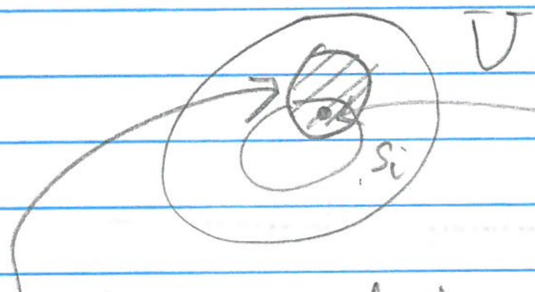
$\int \frac{1}{x} = \log(x)$

Idea of proof of main Lemma:



for any subset S_i, we eventually cover all elements.

Because greedy: charge associated with first element of S_i covered $\leq \frac{\omega}{|S_i|}$



first covered

maybe covered by S_{10}.

Because greedy: charge associated with second element of S_i that is covered.

$\leq \frac{\omega}{|S_i|-1}$

FIRST COVERED

SECOND COVERED.

Because one elt. has already been covered going on, we see that the charge associated with the j^{th} of S_i element covered is $\leq w_i / |S_i| - j + 1$