Lecture #8

CS 170

Spring 2021

Greedy Algorithms   Chap 5

General algorithmic approach – at every step, choose what to do based on local information, without considering next steps

Sometimes works well: optimal

minimum spanning trees, Huffman encoding

Horn formulas (used in Prolog)

Sometimes just approximation

set cover (no poly time exact alg)

Sometimes not so good: chess...

Example: Student's Problem

You have n HW assignments $a_1, a_2, \ldots, a_n$
with deadlines (all integers) $d_1, d_2, \ldots, d_n$

Each assignment takes 1 hour

In what order should you do assignments to maximize the number turned in on time?

Example:

	a_2	a_4	a_5	a_6		
	a_1	a_2	a_3	a_4	a_5	a_6
	1	1	2	2	4	5
	a_1	a_3	a_5	a_6		

greedy: pick assignment with next deadline

optimal: in first 2 hrs, can do 2 assignments
then do all rest $\Rightarrow$ 4 max

Strategy: repeat: pick unexpired a_i with closest deadline

Claim: optimal strategy: proof by contradiction: assume not

S = optimal strategy $\neq G$ = greedy strategy

$= (a_{s_1}, a_{s_2}, \ldots a_{s_t})$ $\quad = (a_{g_1}, a_{g_2}, \ldots, a_{g_{t'}})$

Modify S to become G $\quad$ if $t' = t-1$, then G could do a_{s_t}

Let k be task where S and G differ

$s_1 = g_1, s_2 = g_2, \ldots s_{k-1} = g_{k-1}, \; s_k \neq g_k$

Change S to S' with $s_k = g_k$, S' still optimal

Case 1: a_{g_k} doesn't appear in $S \Rightarrow$ replace a_{s_k} by a_{g_k}, ok since a_{g_k} hasn't expired yet, since chosen by G

Case 2: a_{g_k} does appear in S, say $a_{s_\ell} = a_{g_k}$ $\quad \ell > k$

$\Rightarrow$ swap a_{s_ℓ} and a_{s_k}; OK to do a_{s_ℓ} earlier, still not expired since G (greedily) chose a_{g_k} as earliest to expire

$d_{g_k} = d_{s_\ell} \leq d_{s_k} \Rightarrow$ can do a_{s_k} at same time as a_{s_ℓ}

Finally G can't be shorter than S, because G could do same assignment as S $\quad$ 3

Set Cover

Given: $V = \{1, 2, \dots, n\} = [n]$

Collection of subsets $S_1, S_2, \dots S_m \subseteq V$

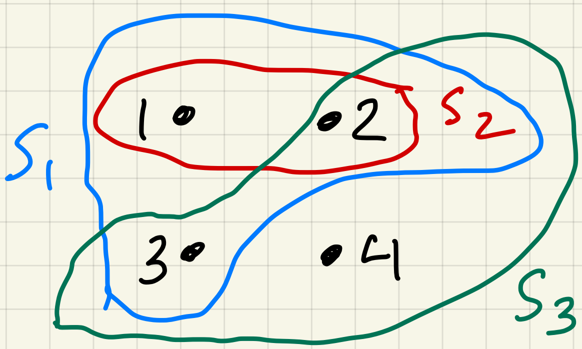
such that $\bigcup_{i=1}^{m} S_i = V$

Find: Fewest S_i that cover V:

$J \subseteq [m]$, $\bigcup_{i \in J} S_i = V$,

$|J|$ as small as possible

$V = [4]$



$V = S_1 \cup S_3 = S_3 \cup S_2$

$\underset{\text{optimal}}{|J| = 2}$, $|J| = 1$ not possible

Ex

$V = \{\text{town}_1, \dots, \text{town}_n\}$

$S_i = \{$all towns in distance ≤ 30 miles from town $i\}$

Goal: build as few schools as needed so everyone ≤ 30 miles from nearest school

Greedy Strategy:

At each step, pick S_i that covers the most uncovered points

Greedy Set Cover ... $V=[n]$

$J=\emptyset$ ... collection of S_i, $i\in[m]$

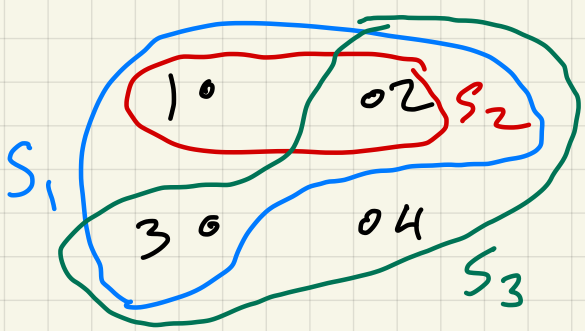
while $S_J \neq V$ ... $S_J = \bigcup_{i\in J} S_i$

  pick $i\notin J$ with largest $|S_i \setminus S_J|$

  ... covers most new points

  $J = J\cup\{i\}$

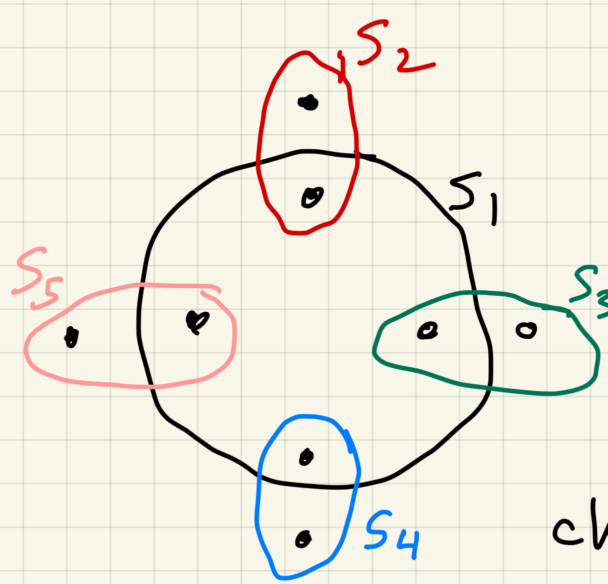
Correct?



1) pick S_1 or S_3 $|J|=2$

2) pick S_3 or S_1 S_2 best

done done



1) pick S_1

2) pick all other S_i: $|J|=5$

not best: choose $S_2,\dots,S_4$ $|J|=4$

How well can we solve Set Cover?

Exact answer in time $O(\text{polynomial}(n,m))$?
$n = \#$ points, $m = \#$sets

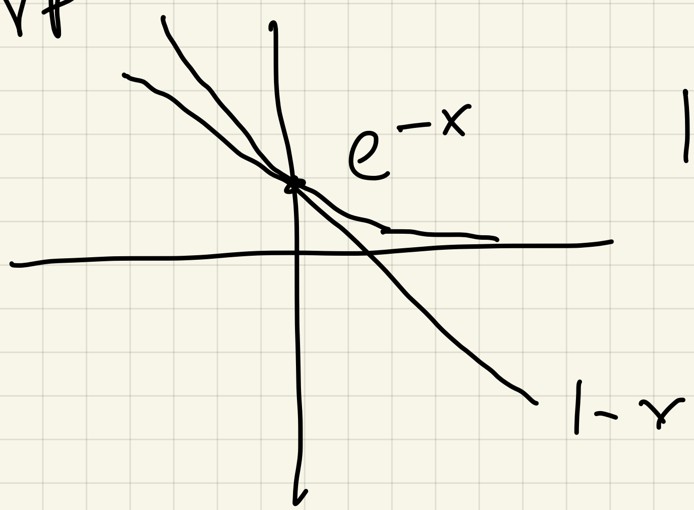
win \$1M (Millenium Prize) $P=NP$

Thm: If $k =$ exact answer (fewest #sets)

Greedy finds $\le k \cdot \ln n$ sets

Fact: anything better, eg $\le k \cdot \sqrt{\ln n}$
implies $P=NP$

Fact from calculus



$1-x \le e^{-x}$

Thm: $n = \#$points, $m = \#$sets, $k =$ fewest $\#$sets in a set cover

Greedy Alg. find $\leq k \cdot \ln n$ sets

proof: $n_t = \#$ uncovered points after t steps, $n_0 = n$

show n_t decreases "quickly": $n_t \leq c\, n_{t-1}$ where $c < 1$

$\Rightarrow n_t \leq c^t n_0 = c^t \cdot n$:

choose $t = \#$sets chosen by Greedy alg big enough

$n_t \leq c^t \cdot n < 1 \Rightarrow n_t = 0 \Rightarrow$ done

$\ln(c^t \cdot n) < \ln(1) = 0 \Rightarrow t \cdot \ln c + \ln n < 0 \Rightarrow t > \dfrac{\ln n}{\ln(1/c)}$ since $c < 1$

claim: $n_t \leq n_{t-1} - \dfrac{n_{t-1}}{k} = (1 - \frac{1}{k}) n_{t-1} = c \cdot n_{t-1}$

proof: after step $t-1$, n_{t-1} uncovered pts

$\Rightarrow n_{t-1}$ pts covered by k sets

$\Rightarrow$ some unchosen set covers at least $\dfrac{n_{t-1}}{k}$ pts

$\Rightarrow$ greedy choice covers $\geq \dfrac{n_{t-1}}{k}$ pts

$\Rightarrow n_t \leq n_{t-1} - \dfrac{n_{t-1}}{k}$

choose $t \geq k \cdot \ln n \geq \dfrac{\ln n}{\ln(\frac{1}{c})} \Rightarrow$ done

$c = 1 - \frac{1}{k} \leq e^{-1/k}$

$\frac{1}{c} \geq e^{1/k}$

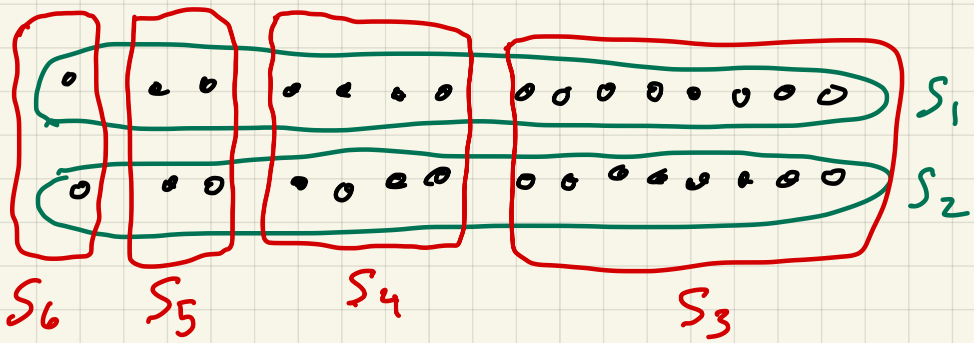
$\ln(\frac{1}{c}) \geq \frac{1}{k}$

$\Rightarrow \dfrac{1}{\ln(\frac{1}{c})} \leq k$

Fact: Greedy Alg can attain $\Omega(k \cdot \ln n)$

$$n = 2(2^e - 1) \sim 2^{e+1}$$

Ex: $n = 30$:
$e = 4$



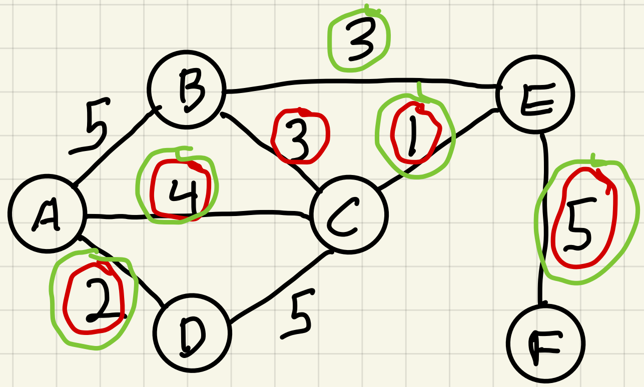
Optimal:
$S_1 \cup S_2$
$k = 2$
$|S_1| = |S_2| = 15$

Greedy picks S_3, $|S_3| = 16$
S_4, S_5, S_6

Optimal: $k = 2$

Greedy: pick e subset $= \log_2 n - 1$
$= \Theta(k \cdot \ln n)$

Introduction to Minimum Spanning Trees (MSTs)



Given an undirected graph $G=(V,E)$ with edge weights $w(e)>0$

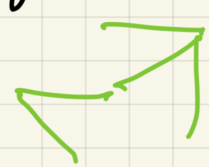
Find a subset $T \subseteq E$ such that

1. (V,T) connected
2. sum of weights of $T = \sum_{e \in T} w(e)$ minimized

Fact: T has no cycles, i.e a tree

What would be a greedy algorithm?

add cheapest edge to T as long as no cycle



Next time: Algorithms for MSTs

Kruskal's Alg. and Prim's Alg.

Cost: $O(|E| \cdot \log |V|)$

or $O(|E| \log^* |V|)$

$\log^* |V| \leq 5$ as long

$|V| \leq (\underbrace{\#\text{ particles in universe}}_{10^{80}})^{246}$

(very likely)

$|V| \leq 2^{65536} = \left.2^{2^{2^{2^{2}}}}\right\} 5$