Recall \( P \neq NP \)

NP-complete - hardest problems in NP - open - to find poly time alg

( Ladner proved: if \( P \neq NP \) then there are inf. many problems between \( P \) and \( NP \) compete.

Major theoretical and practical question: is \( P = NP \)?

Previous section of course - does randomness help? open (though it's a good tool to have in your arsenal, and gives faster/simpler algorithms)

What to do with NP-complete problems?
- heuristics
- exact exponential time (as efficient as possible)
- other models of computing (quantum, natural)
- \( \star \) approximation algns
- parameterized complexity (next section of course)

give up on exact
but ask for guarantee on quality of solution
An approximation alg. for optimization problem finds in poly. time a solution "close to" the opt. "close to" – difference is small or ratio of solution to opt. is good.

An extreme example

Edge-colouring in graph: Given a graph colour the edges s.t. if two edges are incident, they have different colours.

![Triangle with edges coloured](image)

This problem is NP-complete: Given G, k ∈ N, can you edge colour G with k colours?

Vizing's Thm

\[ \Delta \leq \min \text{ # colours } \leq \Delta + 1 \]

\[ \Delta = \max \text{ degree} \]

\[ \{ \text{t edges} \} \text{ need at least } t \text{ colours.} \]

Furthermore, there is poly. time alg. to colour any graph with \( \Delta + 1 \) colours.

So we can approximate within \( +1 \) of opt.

This is rare. Usually we will just get good ratio of approx to opt.
**Vertex Cover**: Given a graph $G = (V, E)$ find a min. size vertex cover — a set $U \subseteq V$ s.t. every edge has at least one endpoint in $U$.

![Graph with vertex cover highlighted]

This is min. vertex cover.

**Application**: monitor all links in network.

(decision version of) Vertex Cover is NP-complete.

$3SAT \leq_p \text{ Independent Set} \leq_p \text{ Vertex Cover}$.

Argument:

- find max set of vertices, no two joined by edge.
- $U$ is a vertex cover iff $V \setminus U$ is an ind. set.

$G$

vertex cover

ind. set

$U$ is a vertex cover iff $V \setminus U$ is an ind. set
EX: Prove that if there is an approx alg. for vertex cover that’s good within additive constant (as for edge colouring) then \( P = NP \).

**Greedy Algorithm**

\[
C \leftarrow \emptyset \\
\text{repeat} \\
\quad C \leftarrow C \cup \{ \text{vertex of max degree} \} \\
\quad \text{remove covered edges} \\
\text{until no edges remain} \\
\text{we will show} \quad |C| \leq O(\log n) |\text{OPT}| \\

EX: Show the greedy alg. can give \( \frac{|C|}{|\text{OPT}|} \in \Omega(\log n) \)

**more general problem:**

**Set Cover Problem:** Given a collection of sets

\[
S = \{ S_1, S_2, \ldots, S_k \} \quad S_i \subseteq \{1, \ldots, n\} \\
\text{find a min. size subcollection s.t. every element } 1 \ldots n \text{ is covered} \\
i.e. \quad C \subseteq \{1, \ldots, k\} \\
\forall i \in \{1, \ldots, n\} \quad i \in S_j \text{ for some } j \in C.
\]

e.g.

\[
S_1 = \{1, 2\}, \quad S_2 = \{1, 3\}, \quad S_3 = \{2, 3, 4\}, \quad S_4 = \{1, 4\} \\
S_1, S_3 \text{ cover} \\
\]

**Application:** Set = type of pizza elements = people

\[
element \in \text{set} = \text{person eats that type of pizza} \\
\text{find min. # of pizza types to feed all.}
\]
Vertex Cover is the special case of Set Covering where element = edge set = edges incident to a vertex

\[ S(\{1\}) = \{1,3,4,5\} \]
\[ S(\{2\}) = \{1,2,5\} \]

Is every Set Cover Problem a Vertex Cover Problem? No. The special property of Vertex Cover is that each element is in only 2 sets.

Greedy Alg. for Set Cover:

\[ C \leftarrow \emptyset \]

while there are uncovered elements

\[ S_i \leftarrow \text{a set that covers max # uncovered elements} \]

\[ C \leftarrow C \cup S_i \]

end

Ex.

pick set 1
pick set 2
pick set 3

greedy is not opt. here.
Theorem. The greedy alg. is poly. time with approx. factor $O(\log n)$.

i.e. $|c| = O(\log n) |\text{OPT}|$ where $\text{OPT} = \min\text{ set cover}$.

Proof. from Vazirani's book - better than pf. in CLRS.

Distribute cost of choosing a set $S_i$ (cost is 1) uniformly over newly covered elements. Let $c(e) = \text{cost of element } e$.

size of greedy soln. = # sets chosen = $\sum_e c(e)$

eg. if $e$ is covered by first chosen set $S$, $c(e) = \frac{1}{|S|}$

Since $S$ is max size set $\geq$ max size set in $\text{OPT}$

$\geq \frac{\text{avg. size set in } \text{OPT}}{|\text{OPT}|} \geq \frac{n}{|\text{OPT}|}$ clever!

$\therefore c(e) = \frac{1}{|S|} \leq \frac{|\text{OPT}|}{n}$

more generally, let $e_1, e_2, \cdots, e_n$

be ordering of elements as they are covered. (lots of ties)

Consider $e_i$. Suppose first greedy set to cover it is $S'$

Suppose $S'$ covers $t$ new elements. Then $c(e_i) = \frac{1}{t} \bigstar$

Previous sets cover $\leq i-1$ elements

$e_1, \cdots, \underbrace{e_i}_{\text{newly covered by } S'}\cdots, e_n$

$\therefore \#\text{uncovered elements when we chose } S' \text{ is } \geq n-i+1$

$S'$ covers $t$ of those, $t = \max_{S \in \text{OPT}} \#\text{new in } S \geq \max_{S \in \text{OPT}} \#\text{new in } S$

$\geq \frac{\text{avg. } \#\text{ new in } S}{\text{OPT}} \geq \frac{n-i+1}{|\text{OPT}|}$ because the sets in OPT cover all $n-i+1$ elements.

thus $t \geq \frac{n-i+1}{|\text{OPT}|}$. By $\bigstar$ $c(e_i) \leq \frac{|\text{OPT}|}{n-i+1}$.
Suppose that the optimum cover has $k$ sets. Then at a stage where $r$ items remain to be covered, some set covers $r/k$ of them. Therefore, choosing the set that covers the most new points reduces the number of uncovered points to $r - r/k = r(1 - 1/k)$. If we start with $n$ points and repeat this process $n$ times, $r \leq n(1 - 1/k)^j$. Since $r$ is an integer, we are done when $n(1 - 1/k)^j < 1$. This happens when $j = O(k \log n)$. Hence, $\alpha = j/k = O(\log n)$.

**Next Day:** An approx. alg. for vertex cover with approx. factor 2.