Lecture 26 — Some more NP-complete problems

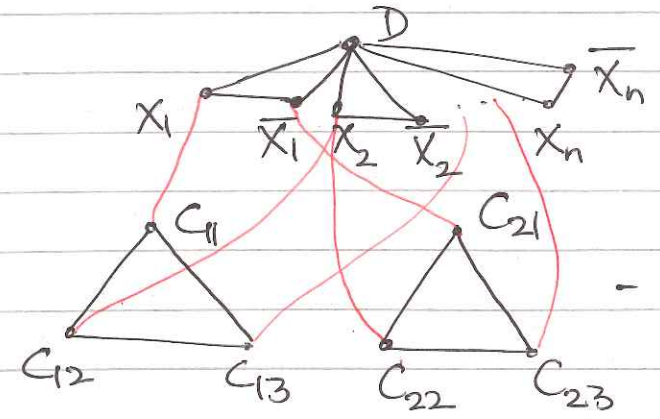
4) 3COL is NP-complete.

We will use a variant of SAT called NAE-SAT (not all equal SAT) to show the NP-hardness of 3COL. In NAE-SAT we want to know if there is a true/false assignment to the input formula ϕ so that

* in every clause there is at least 1 true literal and 1 false literal.

NAE-SAT is also NP-complete.

We will now show that NAE-SAT $\leq_p$ 3COL. Given a 3CNF formula ϕ that is the input to NAE-SAT, we will build a graph G_ϕ as described below so that $\phi \in$ NAE-SAT $\iff G_\phi \in$ 3COL.



The graph G_ϕ

Here the top vertex D is a "dummy". $x_1, \ldots, x_n$ are the n variables in ϕ. Let first clause is $(x_1 \vee x_2 \vee \overline{x}_3)$, the second clause is $(\overline{x}_1 \vee x_2 \vee x_4)$, ...

Claim 1. If $\phi \in$ NAE-SAT then $G_\phi \in$ 3COL.

Colour the vertex D green. Assign red colour to all true literals and blue to all false literals in the top layer.

The vertices in the triangles in the lower layer can be coloured red/blue/green appropriately so that the resulting colouring is valid.

(Please check this.)

Claim 2. If $G_\phi \in$ 3COL then $\phi \in$ NAE-SAT.

In this 3-colouring of G_ϕ, assume (without loss of generality) that the vertex D is coloured green. So among $x_i, \bar{x}_i$ (for $1 \le i \le n$), one is coloured red and the other is coloured blue.

- take all red vertices in the top layer as true literals and all blue vertices in the top layer as false literals.

Exercise. Check that the above true/false assignment to the variables in ϕ in a not-at-equal satisfying assignment.

5) Max-Cut $= \{(G, k) : G$ has a cut of size $\ge k\}$.

We will show that NAE-SAT $\le_p$ Max-Cut.

Let ϕ be the input 3CNF formula to NAE-SAT.

We will build a graph G_ϕ such that

$\phi \in$ NAE-SAT $\iff (G_\phi, 8m) \in$ Max-Cut.

Here m is the number of clauses in ϕ.

Let n_i be the number of occurrences of $x_i / \bar{x}_i$ in ϕ. So $\sum_i n_i = 3m$.

The graph G_ϕ has $2n$ vertices: $x_1, \ldots, x_n$, $\bar{x}_1, \ldots, \bar{x}_n$. The edges are as follows:

* $2n_i$ parallel edges between x_i & $\bar{x}_i$ (for $1 \le i \le n$)
* a triangle for each clause in ϕ.

Claim 3. If $\phi \in$ NAE-SAT then $(G_\phi, 8m) \in$ Max-Cut.

Consider the cut $(S, V-S)$ where $S = \{$vertices that corr. to true literals$\}$ and $V-S = \{$vertices that corr. to false literals$\}$

So for each i, we have x_i and $\bar{x}_i$ on opposite sides.

For each triangle: 2 vertices are on one side and 1 vertex on the opposite side.

So the number of edges crossing this cut $= \sum_i 2n_i + 2m = 8m$.

Claim 4. If $(G_\phi, 8m) \in$ Max-Cut then $\phi \in$ NAE-SAT.

We are given that G_ϕ has a cut $(S, V-S)$ with $8m$ edges crossing this cut. The only way we can have $8m$ edges crossing a cut is to have x_i & $\overline{x_i}$ on opposite sides! This will add up to $\sum_i 2n_i = 6m$ edges.

What about the remaining $2m$ edges? Observe that each triangle can contribute ≤ 2 edges to any cut. So to add up to $2m$ edges, each triangle has to contribute 2 edges to the cut — since there are m triangles, this will add up to $2m$.

View vertices on one side of the cut as true literals & vertices on the other side as false literals. This corresponds to a not-all-equal satisfying assignment. (Please check this.)

6) Set Cover: We have a universe $\mathcal{U} = \{1, \ldots, m\}$ and a collection of subsets $S_1, \ldots, S_n$ of $\mathcal{U}$ such that $S_1 \cup \cdots \cup S_n = \mathcal{U}$.

The optimization version of this problem: what is the least number of sets to be taken so that their union is $\mathcal{U}$?

The decision version: Is there a sub-collection of k sets $S_{i_1}, \ldots, S_{i_k}$ such that $S_{i_1} \cup \cdots \cup S_{i_k} = \mathcal{U}$?

It is easy to see that Vertex Cover $\leq_p$ Set Cover. Take $\mathcal{U} = \{e_1, \ldots, e_m\}$ (all the edges in the i/p graph) and $S_i = \{$edges incident to vertex $i\}$ for each vertex i in G

Observe that this instance has a set cover of size k $\iff$ G has a vertex cover of size k.

7) Integer Programming

We are given an $m \times n$ matrix A with integer values and a column vector b with m entries.

Is there an integer-valued vector x such that

$$Ax \leq b ?$$

It is easy to see that $3SAT \leq_P$ Int-Pgmming. Given a 3CNF formula ϕ, we can write it down as an integer program s.t. ϕ is satisfiable $\Leftrightarrow$ our IP has an integral solution.

Exercise. Please show the above reduction.

Introduction to Approximation Algorithms

Vertex Cover: This problem has an easy 2-approximation algorithm.

1. Find a maximal matching M in the i/p graph G.
2. For each edge e in M: include both endpoints of e in our set C.

Claim. C is a vertex cover. (Please show this.)

We now claim $|C| \leq 2k$ where k = size of minimum vertex cover of G. This is because $k \geq |M|$ and $|C| \leq 2|M|$.

Suppose every vertex has a weight associated with it and we want to find a min-weight vertex cover. Let us write the LP for vertex cover.

$$\min \sum_{u \in V} w_u \cdot x_u$$

subject to

$$x_u + x_v \geq 1 \quad \text{for each edge } (u,v)$$

$$x_u \geq 0 \quad \forall\, u \in V.$$

The approx. algo: (1) Solve the above LP. Let x^* be the optimal solution.

(2) for each vertex u do: if $x_u^* \geq 1/2$ then add u to our vertex cover C else do not add u to C.

Claim. The set returned by this algorithm is a vertex cover of G.

- for every edge (u, v) in G, since $x^*_u + x^*_v \geq 1$ at least one of x^*_u, x^*_v is $\geq 1/2$. Thus this algo. computes a vertex cover.

Claim. w(vertex cover comp. by this algo) $\leq 2\, w(OPT)$

Observe that $w(OPT) \geq w(x^*)$ and w(our vertex cover) $\leq 2\, w(x^*)$.

Set Cover - A greedy algorithm

Initially all elements in the universe $\mathcal{U}$ are uncovered. $\{1, \ldots, n\}$

1. $C = \emptyset$.
2. while there exist one or more uncovered elements do:
   - find the set S that minimizes the ratio $\dfrac{\text{cost}(S)}{\text{no. of uncovered elements in } S}$
   - add S to C.
3. return C.

* We will now show this is an $\ln n$-approx. algorithm. Renumber the elements of $\mathcal{U}$ in the order that they got covered in our algorithm, where ties are resolved arbitrarily. Let $e_1, \ldots, e_n$ be this numbering. Define $\text{price}(e_i) = \dfrac{\text{cost}(S)}{\text{no. of uncovered elmts in } S \text{ at the time } S \text{ got picked}}$ where S is the set that covers e_i for the first time in our algorithm.

$$\text{cost}(C) = \sum_{S \in C} \text{cost}(S) = \sum_{i=1}^{n} \text{price}(e_i)$$

Claim. For every $i \in \{1, \ldots, m\}$, we have $\text{price}(e_i) \leq \frac{OPT}{m-i+1}$

Proof. Let $\{S_1, \ldots, S_k\}$ form the optimal set cover. So $c(S_1) + \cdots + c(S_k) = OPT$.
This means $|S_1| \cdot \frac{c(S_1)}{|S_1|} + \cdots + |S_k| \cdot \frac{c(S_k)}{|S_k|} = OPT$

Since $|S_1| + \cdots + |S_k| \geq m$, there has to be some j such that $|S_j| \cdot \frac{c(S_j)}{|S_j|} \leq \frac{OPT}{m}$.

Otherwise the lhs above will be
$$> (|S_1| + \cdots + |S_k|) \frac{OPT}{m} \geq OPT.$$

So in the first step of the algorithm, when we choose a set S that minimizes $\frac{c(S)}{|S|}$, we have $\text{price}(e_1) \leq \frac{OPT}{m}$.

Consider any later iteration in the algorithm when elements $e_1, \ldots, e_{k-1}$ have been covered and we are now picking a set S' that covers $e_k, e_{k+1}, \ldots$. We will show $\text{price}(e_k) \leq \frac{OPT}{m-k+1}$.

It is the same idea as in the first iteration. Delete $e_1, \ldots, e_{k-1}$ from the universe. Let $S_1', \ldots, S_r'$ be the non-empty sets in OPT now. Using our old reasoning, for at least one of these sets S_j' we have $\frac{c(S_j')}{|S_j'|} \leq \frac{OPT}{m-k+1}$, where $|S_j'|$ = no. of uncovered elmts in S_j'. ∎

Thus the cost of our set cover $C = \sum_{i=1}^{m} \text{price}(e_i)$
$$\leq OPT \left(\frac{1}{m} + \frac{1}{m-1} + \cdots + 1 \right)$$
$$= OPT \cdot H_m \approx (\ln m) \cdot OPT$$

This proves the approximation guarantee of the greedy algorithm for set cover.