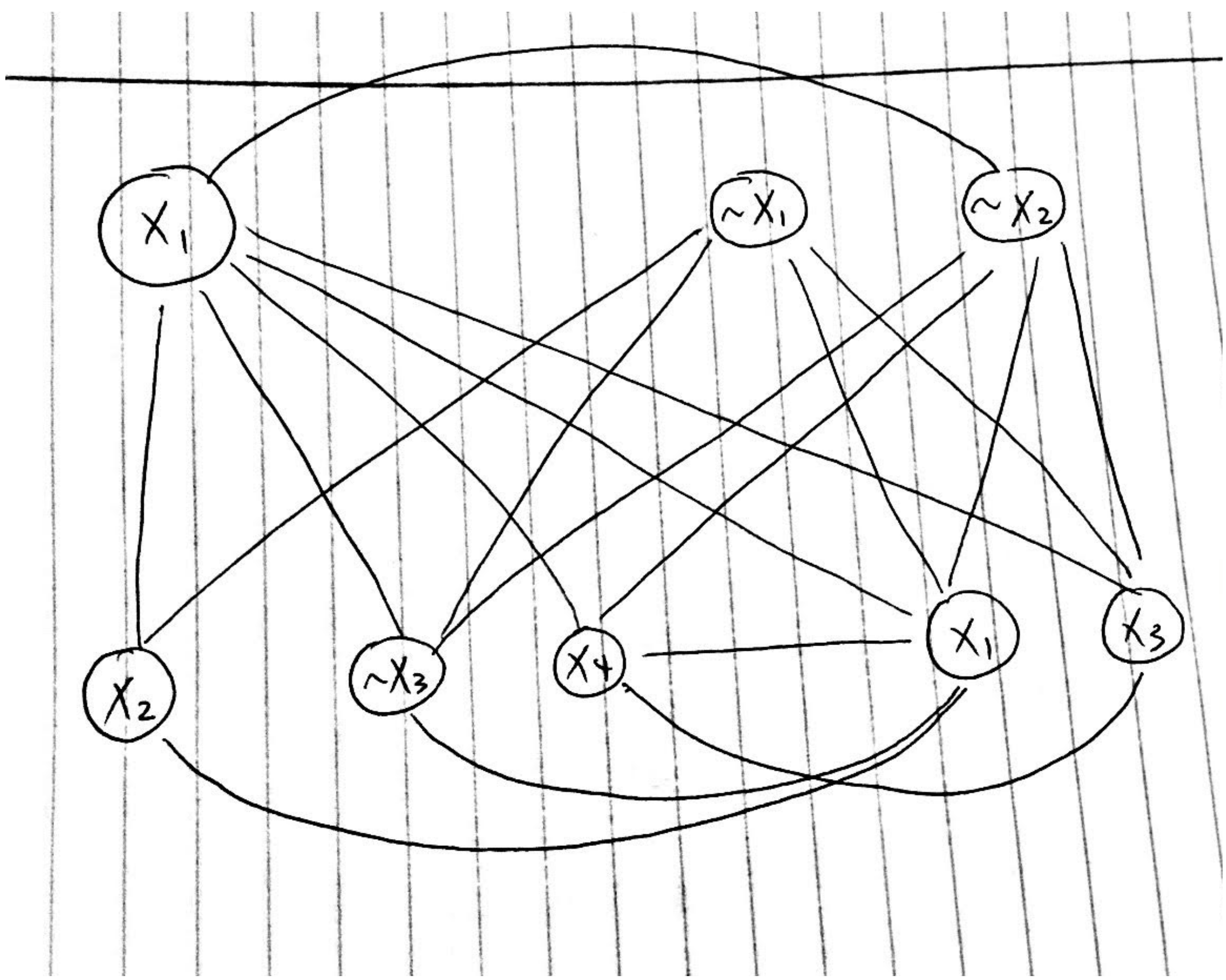
Mar 8

Theorem: CLIQUE is NP-hard

- Reduce SAT to CLIQUE
- Transform a formula ϕ into a graph G and int k such that:
  - G contain a clique of size k iff ϕ is satisfiable
  - transformation is in poly time
- Transformation Steps:
  1. Create a vertex for every literal in every clause.
  2. Connect a vertex from ith clause to vertex from jth clause (i =/= j) unless they are negation of each other.
  3. Let k= # clause in ϕ

Example: $\phi=(x1)\wedge(\sim x1 \vee \sim x2)\wedge(x1 \vee x3)\wedge(x2 \vee \sim x3 \vee x4)$ k=4



Proof:
=>) if ϕ has a truth assignment => every clause has at least one true literal => choose one from each clause => you will have a clique of size k

<=) if G has a k_clique Q => exactly one vertex from each clause is in Q => assign one to each literal vertex => ϕ is satisfied

Mar 9

Vertex cover problem: Given a undirected graph G = (V,E) and integer k, does G have a vertex cover of size k?

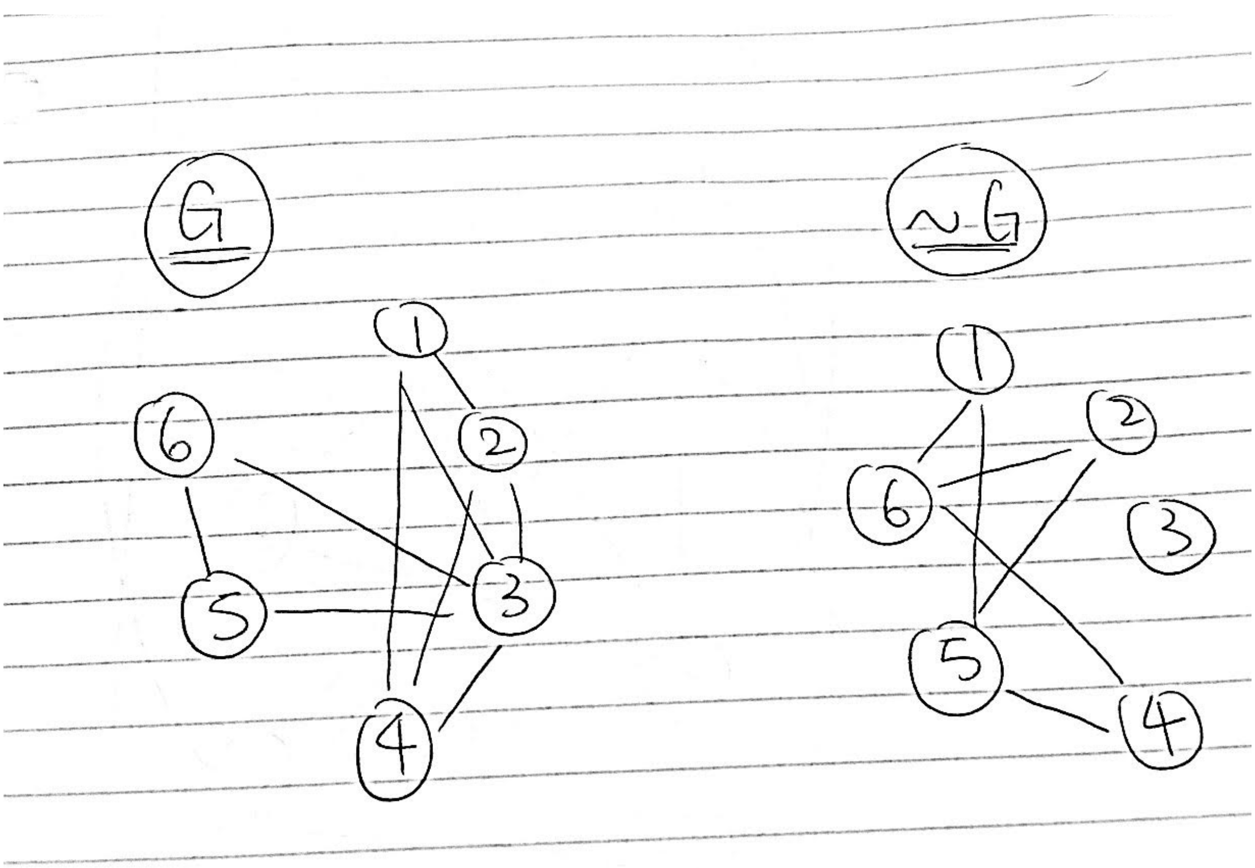
Definition: Vertex cover is a set of vertices S ⊆ V such that all edges in G have at least one end point in S. VC={ <G,k> | G has a Vertex Cover of size k}

Theorem: VC is NP-complete

Proof: 1. VC ∈ NP : certificate (witness) a VC S ⊆ V of size k. The verifier checks that |S| = k and check for each (u,v) ∈ E that u ∈ S or v ∈ S.

2. CLIQUE → VC (reduction from CLIQUE to VC)

~G : the complement of G : ~G = (V,~E) where ~E = {(u,v)|(u,v) is not in E}.



Claim: G has a clique of size k iff G′ = ~G has VC of size k′ =|V|−k

Proof:

⇒) if G has a clique S∈V, then V−S is a VC for ~G. Consider if (u,v) ∈ ~G, then (u,v) is

not in G, which also implies either u or v is not in S and either u or v is in V −S. Therefore, (u, v) is covered.

⇐) let R⊆V be a vertex cover of ~G⇒ This means that every edge in ~G that is incident

to a vertex in V-R must have its other endpoint in R. Thus no edge in ~G connects two vertices in V-R, which means V-R is a clique in G.

Theorem: 3 SAT is in NP-complete

3−SAT ={ϕ | each clause has at most 3 literals and ϕ is in SAT}

Idea of proof: We show how to transform a formula ϕ for SAT (which might contain clauses with more than 3 literals) into a formula ϕ' for 3SAT (in which every clause has at most 3 literals), so that the new formula ϕ' is satisfiable if and only if the original formula ϕ is satisfiable. So for every clause in ϕ with more than three literals, (a_1 or a_2 or ... or a_k) where k>3 and a_i is a literal (a variable or its negation), create k-2 clauses for ϕ': (a_1 or a_2 or y_1) and ($\sim y_1$ or a_3 or y_2) and ($\sim y_2$ or a_4 or y_3) and ... and ($\sim y_{k-3}$ or a_{k-1} or a_k).

Claim: ϕ is satisfiable iff ϕ′ is satisfiable.

Proof:

⇒)If ϕ is satisfiable, then there is a truth assignment so that each clause has a true literal. Let ai

be a true literal for clause $C=(a_1 \vee a_2 \vee \cdots \vee a_k)$,then set $y_1,y_2,...,y_{i-2}$ to true and $y_{i-1},y_i,...,y_{k-3}$ to false. In this way, every one of the 3-SAT clauses derived from C is satisfied.

⇐)If ϕ′ is satisfiable, then every one of the 3-SAT clauses derived from $C = (a_1 \vee a_2 \vee \cdots \vee a_k)$ has a true literal. At least one of the ai’s must be true. Otherwise, if all a_i’s are false, y_1 must be true (to satisfy the first 3-SAT clause) which implies y_2 must be true (to satisfy the second 3-SAT clause) which implies, eventually, that y_{k-3} must be true which implies that the last 3-SAT clause is false: a contradiction. Since at least one of the ai’s is true, the clause C is satisfied