

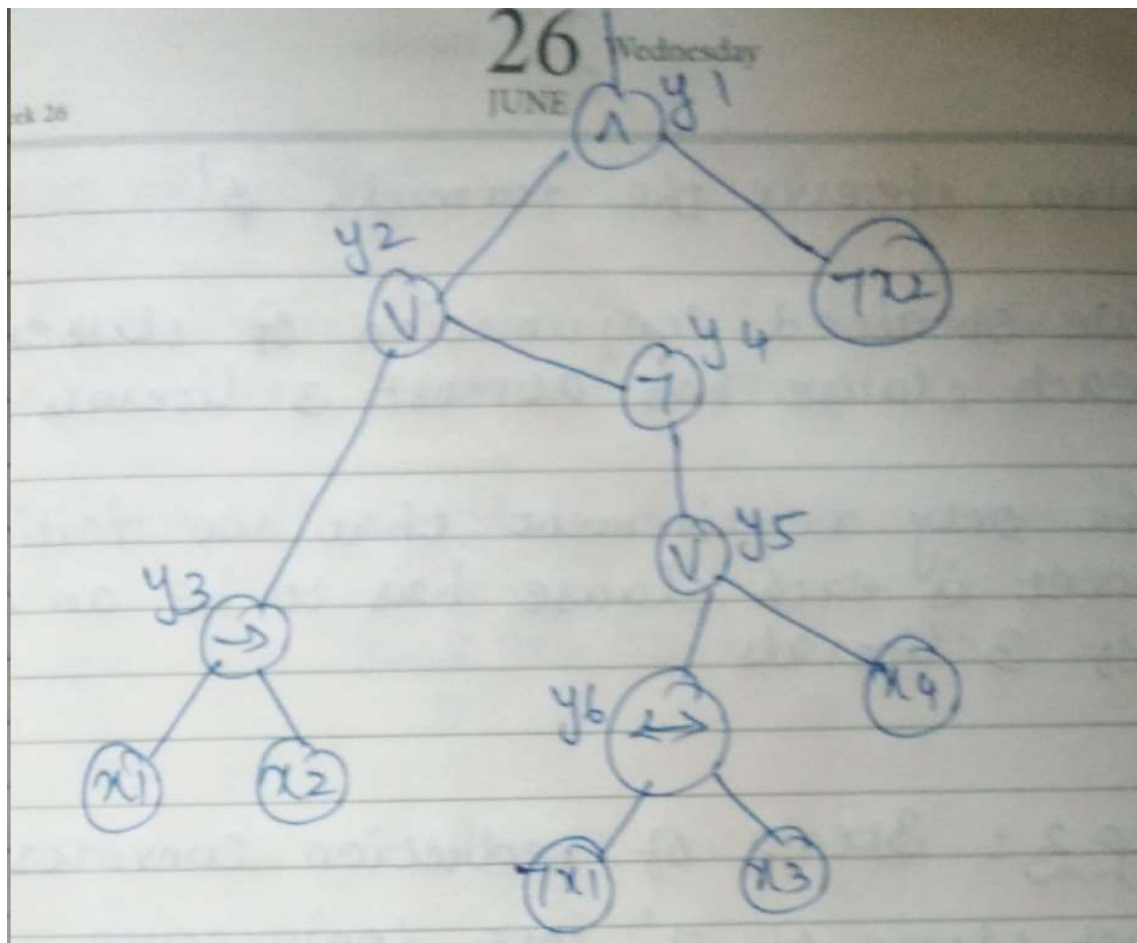
Reduction of SAT to 3-CNF-SAT:

SAT $\leq_p$ 3-CNF-SAT procedure:

step 1: SAT $\Rightarrow$ BT (expression tree).
poly time.

$\phi = ((x_1 \rightarrow x_2) \vee \neg((\neg x_1 \leftrightarrow x_3) \vee x_4)) \wedge \neg x_2$.

Construct expression tree for ϕ and assign labels to each internal node.



Then, rewrite the formula ϕ as the AND of root variable and a conjunction of clauses describing the operation of each node.

$$\begin{aligned}\phi' = & y_1 \wedge (y_1 \leftrightarrow (y_2 \wedge \neg x_2)) \\ & \wedge (y_2 \leftrightarrow (y_3 \vee y_4)) \\ & \wedge (y_3 \leftrightarrow (x_1 \rightarrow x_2)) \\ & \wedge (y_4 \leftrightarrow \neg y_5) \\ & \wedge (y_5 \leftrightarrow (y_6 \vee x_4)) \\ & \wedge (y_6 \leftrightarrow \neg x_1 \leftrightarrow x_3)\end{aligned}$$

Now, observe the formula ϕ' .

We obtained conjunction of clauses and each clause has at most 3 literals.

The only requirement that we fail to meet is each clause has to be an OR of 3 literals.

Step 2: Step 2 of reduction converts each clause of ϕ' into conjunctive normal form.

Step 2a: Build formula in Disjunctive normal form (DNF)



OR of ANDs.

Step 2b: Using Demorgan's law we negate DNF formula and convert into CNF.

Step 2a: DNF



Construct truth table for clauses with more than 1 literal.

eg $\phi'_1 = (y_1 \leftrightarrow (y_2 \wedge \neg x_2)) \rightarrow$ first clause in ϕ'

y_1	y_2	x_2	$y_2 \wedge \neg x_2$	$y_1 \leftrightarrow (y_2 \wedge \neg x_2)$
1	1	1	0	0
1	1	0	1	1
1	0	1	0	0
1	0	0	0	0
0	1	1	0	1
0	1	0	1	0
0	0	1	0	1
0	0	0	0	1

Now, using truth table entries that evaluate to 0, we build formula in DNF.

$$\text{DNF}(\phi'_1) = (y_1 \wedge y_2 \wedge x_2) \vee (y_1 \wedge \neg y_2 \wedge x_2) \vee (y_1 \wedge \neg y_2 \wedge \neg x_2) \vee (\neg y_1 \wedge y_2 \wedge \neg x_2)$$

Step 2b:

29 Saturday
JUNE

180/185 Week 26

Using Demorgans law Convert DNF to CNF.

~~$\neg(a \wedge b) = \neg a \vee \neg b$~~

$$\neg(a \wedge b) = \neg a \vee \neg b$$

$$\neg(a \vee b) = \neg a \wedge \neg b$$

Demorgan's law $\rightarrow$ change AND's into OR's and change OR's into AND's.

CNF of ϕ_1' is ϕ_1''

$$\phi_1'' = \neg(y_1 \wedge y_2 \wedge x_2) \wedge \neg(y_1 \wedge \neg y_2 \wedge x_2) \wedge \neg(y_1 \wedge \neg y_2 \wedge \neg x_2) \wedge \neg(\neg y_1 \wedge y_2 \wedge \neg x_2)$$

$$\phi_1'' = (\neg y_1 \vee \neg y_2 \vee \neg x_2) \wedge (\neg y_1 \vee y_2 \vee \neg x_2) \wedge (\neg y_1 \vee y_2 \vee x_2) \wedge (y_1 \vee \neg y_2 \vee x_2)$$

$\downarrow$

equivalent to original clause ϕ_1'

Likewise we have to convert each clause into CNF formula ϕ_i''

$$\phi'' = \phi_1'' \wedge \phi_2'' \wedge \phi_3'' \dots \phi_n''$$

2013

JUNE 2013						
S	M	T	W	T	F	S
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

Step 3: transforming formula into 3-CNF

As a result of Step 2, we get ϕ'' for all clauses in ϕ' (CNF)

In step 3, we construct final 3-CNF formula ϕ''' from clauses of ϕ''

ϕ''' is constructed as follows:

- * If C_i has 3 distinct literals, then simply include C_i as a clause of ϕ'''
- * If C_i has 2 distinct literals (i.e. $C_i = (l_1 \vee l_2)$ then include $(l_1 \vee l_2 \vee p) \wedge (l_1 \vee l_2 \vee \neg p)$ as clauses of ϕ''' .

Here, p is the auxiliary literal.

whether $p=0$ (or) $p=1$, one of the clauses is equivalent to $(l_1 \vee l_2)$ and other evaluates to 1, which is ~~identity~~ ^{always} for AND

- * If C_i has just 1 ^{distinct} literal, then include $(l \vee p \vee q) \wedge (l \vee p \vee \neg q) \wedge (l \vee \neg p \vee q) \wedge (l \vee \neg p \vee \neg q)$ as clauses of ϕ''' .

Regardless of the values of p and q , one of the 4 clauses is equivalent to l and other 3 evaluates to 1.

Reduction of 3-CNF-SAT to Clique:

NOTE: In the reduction of 3-CNF-SAT to Clique, size of clique will be number of clauses (k) in the given CNF Boolean formula.

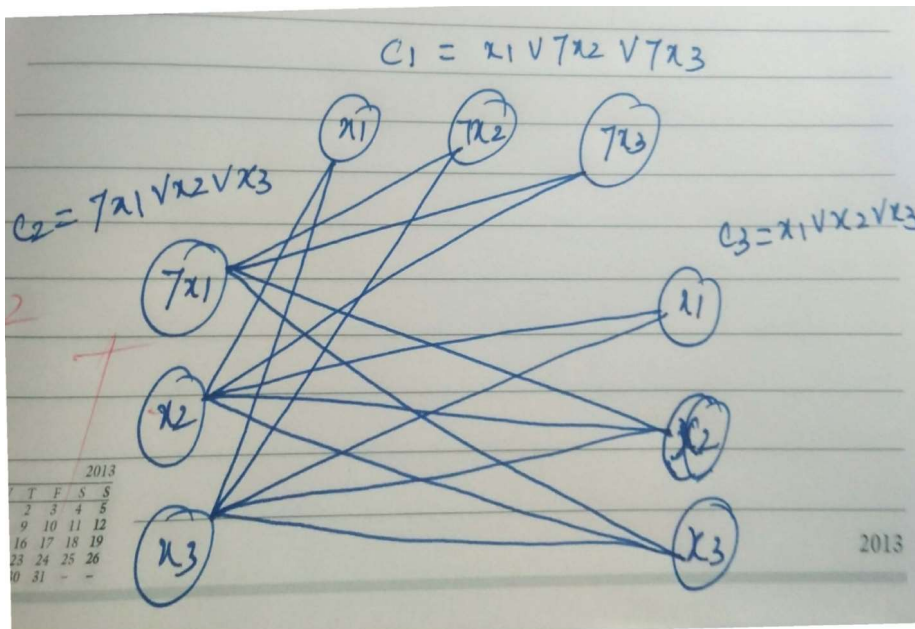
1. Construct a graph G of k clusters with 3 nodes in each cluster.
2. Each cluster corresponds to a clause in ϕ .
3. Each node in a cluster corresponds to a literal in that particular clause.
Each node in the cluster is labeled with a literal from the clause.
4. An edge is put between all pairs of nodes in different clusters except for pairs of the form $(x, \neg x)$.
5. No edge is put between any pair of nodes in the same cluster.
6. G has k -clique iff ϕ is satisfiable.
7. If the graph G has k -clique,
 - a. Clique has exactly one node from each cluster.
 - b. Corresponding literals can be assigned True.
 - c. Hence, ϕ is satisfiable.

Example

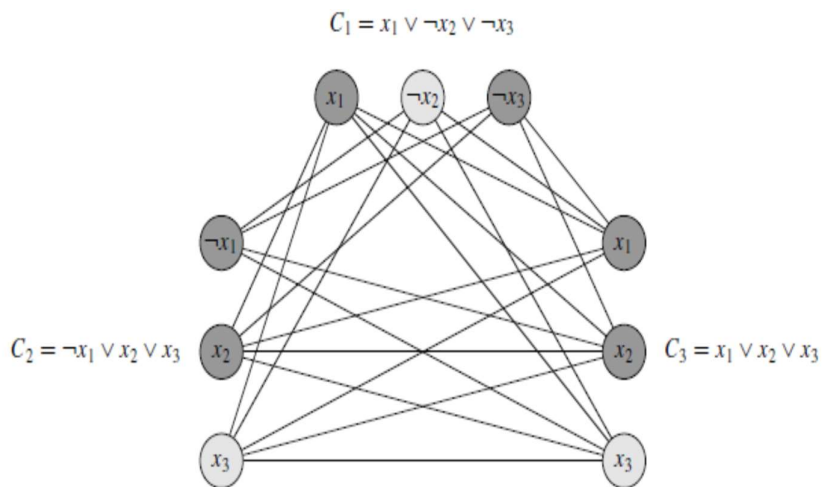
1. Reduce the Boolean 3-CNF formula $\phi = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee x_3)$ to clique.

Solution

- Construct graph G with 3 clusters and 3 nodes in each cluster. Add edges between all pairs of nodes in different cluster.
- Below figure shows the graph with edges added between pair of nodes in (cluster 2, cluster 1) and (cluster 2, cluster 3).



- Likewise, when you add edges between pair of nodes in (cluster 1, cluster 3), we get below graph.



- Select one node from each cluster and check whether it forms a clique.
- In the above graph, nodes highlighted with light grey have been selected. It forms a clique.
- The size of clique is 3 as we have 3 clauses in the given Boolean formula.
- Corresponding literals can be assigned true value. The satisfying assignment of the formula has $x_2 = 0$ and $x_3 = 1$. x_1 may be either 0 or 1.

- A corresponding clique of size $k = 3$ consists of the vertices corresponding to $\neg x_2$ from the first clause, x_3 from the second clause, and x_3 from the third clause.
- Because the clique contains no vertices corresponding to either x_1 or $\neg x_1$, we can set x_1 to either 0 or 1 in this satisfying assignment.

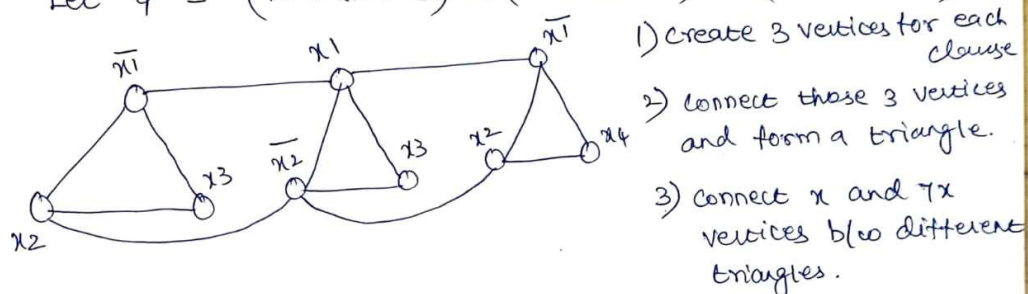
Reduction of 3-CNF-SAT to Independent Set:

$$3\text{-SAT} \leq_p \text{Independent Set.}$$

Given an instance ϕ of 3-SAT, construct an instance $\langle G, k \rangle$ of independent set such that G has independent set of size at least k iff ϕ is satisfiable.

Construction of G :

$$\text{Let } \phi = (\bar{x}_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \bar{x}_2 \vee x_3) \wedge (\bar{x}_1 \vee x_2 \vee x_4)$$



↳ we can pick at most one vertex from each triangle to make independent set. we cannot pick any 2 vertices that are connected by an edge in different triangles.

$$\therefore \text{Independent Set} = \{x_2, x_3, x_4\}$$

Satisfying assignment $\Rightarrow$ x_2 - True
 x_3 - True
 x_4 - True.