16/12/2013
E 10.1

Exercise 2:

Show that CSAT remains NP-complete even if it is restricted to instances in which each variable appears at most three times. Let's call this variant 3VAR CSAT

Solution: We show how to reduce CSAT to 3VARCSAT.

Given a formula F in CNF, we construct a formula E in CNF where each variable appears at most three times and such that E is satisfiable iff F is satisfiable.

Let x be a variable appearing in F k times, with $k > 3$. Then we construct from F a formula F_x as follows:

1) We replace the i-th occurrence of x in F with x_i, for $i \in \{1, \dots, k\}$, where each x_i is a fresh variable.
2) We add the following clauses, ensuring that all variables $x_1, \dots, x_k$ are assigned the same truth value

$$(\overline{x}_1 \vee x_2) \wedge (\overline{x}_2 \vee x_3) \wedge \dots \wedge (\overline{x}_{k-1} \vee x_k) \wedge (\overline{x}_k \vee x_1)$$

We have that F_x can be constructed from F in polynomial time, and F_x is satisfiable iff F is satisfiable.

Then, the formula E is obtained from F by repeating the above transformation for each variable x occurring in F more than three times.

We have that:

1) Each variable appears in E at most three times
2) E can be constructed from F in polynomial time
3) E is satisfiable iff F is satisfiable

□

Exercise 3

Consider 2VARCSAT, i.e. the variant of CSAT in which each variable appears at most two times.

What is the complexity of 2VARCSAT.?

Solution: 2VARCSAT is in P.

Let F be an instance of 2VARCSAT.

Notice that for each variable x, we have one of 3 cases:

1) x appears only positively in F (one or two times)
2) x " " negatively in F (" ")
3) x appears one time positively and one time negatively in F

From this, we obtain the following algorithm to decide the satisfiability of F:

Input: set F of clauses over variables $x_1, \dots, x_n$, with each x_i appearing in at most two clauses

Output: YES, if F is satisfiable, NO otherwise

For each variable $x \in \{x_1, \dots, x_n\}$

if x appears only positively in F (case 1), or
x appears only negatively in F (case 2), or
x appears only in one clause of F
then remove from F the clause(s) in which x_i appears
else let $C = x \vee C_{rest}$ and $C' = \bar{x} \vee C'_{rest}$ (case 3)
be the two clauses of F in which x appears
if C_{rest} and C'_{rest} are both empty
then answer NO
else remove from F both C and C', and
replace them with the clause $C_{rest} \vee C'_{rest}$

Answer YES

E 10.3

The algorithm runs in polynomial time, since the for-loop is executed n times, and each iteration is at most linear.

Note that a variable x_i might be removed from F before it is considered in the i-th iteration of the for loop. In this case, F is not changed in that iteration.

Moreover:

- For cases (1) and (2) the clauses containing x can be trivially satisfied by making x true/false.
- For case (3), the algorithm applies a resolution step to x, and replaces the clauses C and C' with their resolvant.
- By applying a resolution step to a variable x_i, for another variable x_j that has not yet been considered (i.e. $j \geq i$) and that occurred positively and negatively in two different clauses, the two occurrences of x_j might be merged into a single clause $C_{rest} \vee C'_{rest}$.
  This clause can be removed, since it is trivially satisfied by every truth assignment.

Exercise: 2SAT is in P

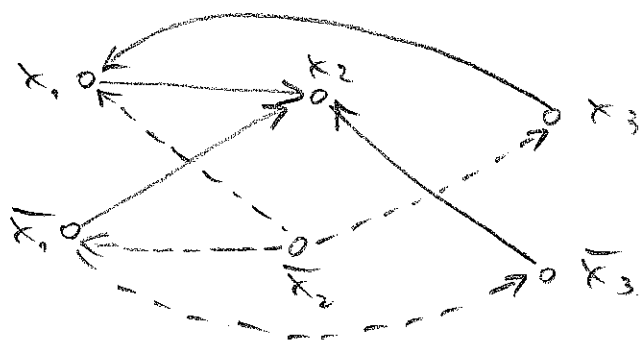
Idea: we show that 2SAT can be encoded as a graph reachability problem, and then use an algorithm for graph reachability

1) Encoding of 2SAT as a directed graph reachability problem

Let ϕ be an instance of 2SAT. We define a graph $G(\phi)$ as follows:
- one node for each variable and for each negated variable
- for each clause $\alpha \vee \beta$ two edges $\bar{\alpha} \to \beta$, $\bar{\beta} \to \alpha$

(note: $\alpha \vee \beta \equiv \bar{\alpha} \to \beta \equiv \bar{\beta} \vee \alpha$)

Example: $(x_1 + x_2) \cdot (x_1 + \bar{x}_3) \cdot (\bar{x}_1 + x_2) \cdot (x_2 + x_3)$



Then ϕ is unsatisfiable iff there is a variable x such that $G(\phi)$ contains two paths $x \to \cdots \to \bar{x}$ and $\bar{x} \to \cdots \to x$.

"$\Leftarrow$" Suppose that ϕ has a satisfying truth assignment T. Assume that $T(x) = \text{true}$ (a similar argument holds for $T(x) = \text{false}$)

Since $T(x) = \text{true}$ and $T(\bar{x}) = \text{false}$, and there is a path $x \to \cdots \to \bar{x}$, there must be an edge $\alpha \to \beta$ along this path with $T(\alpha) = \text{true}$ and $T(\beta) = \text{false}$.

However, since $\alpha \to \beta$ is an edge of $G(\phi)$, it follows that $\bar{\alpha} + \beta$ is a clause of ϕ. This clause is not satisfied by T_T which is a contradiction.

"$\Rightarrow$" Let $G(\phi)$ be a graph that does not contain any node x with $x \to \dots \to \bar{x}$ and $\bar{x} \to \dots \to x$.

We construct from such a graph a satisfying truth assignment T

Repeat the following steps as often as possible:

Choose a node α such that
- $T(\alpha)$ is not yet defined, and
- there is no path $\alpha \xrightarrow{*} \bar{\alpha}$

For every node β that is reachable from α

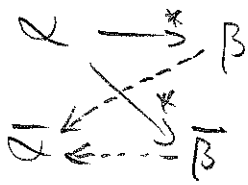
1) set $T(\beta) = \text{true}$
2) set $T(\bar{\beta}) = \text{false}$

(Note: 2) means to assign false to all predecessors of $\bar{\alpha}$) $\alpha \xrightarrow{*} \beta$, $\bar{\alpha} \xleftarrow{*} \bar{\beta}$

Observe: 1) the truth assignment T is well defined, i.e., we never have both $T(\beta) = \text{true}$ and $T(\bar{\beta}) = \text{true}$ or $T(\beta) = \text{false}$ and $T(\bar{\beta}) = \text{false}$

T would not be well defined, if we had both $\alpha \xrightarrow{*} \beta$, $\alpha \xrightarrow{*} \bar{\beta}$ (for some β)

But this cannot happen, since we would have



Hence, we would have $\alpha \xrightarrow{*} \bar{\alpha}$

2) We assign to all nodes a truth value, since there is no path $\alpha \xrightarrow{*} \bar{\alpha}$ and $\bar{\alpha} \xrightarrow{*} \alpha$

3) The truth assignment satisfies all clauses of F, since each clause corresponds to an implication, and there is no $\alpha \leftrightarrow \bar{\alpha}$.

E 10.6

Exercise 4

Consider the following problems:

1) Vertex-cover (VC)

Given an undirected graph $G=(V,E)$ and an integer $k \geq 2$, is there a subset C of V with $|C| \leq k$ such that C covers all edges of G (i.e., for each edge $\{v_i, v_j\} \in E$ with $v_i \neq v_j$, $\{v_i, v_j\} \cap C \neq \emptyset$).

2) Independent-set (IS)

Given an undirected graph $G=(V,E)$ and an integer $k \geq 2$, is there a subset C of V with $|C| \geq k$ such that for all $v_i, v_j \in C$ with $v_i \neq v_j$, $\{v_i, v_j\} \notin E$

3) Clique

Given an undirected graph $G=(V,E)$ and an integer $k \geq 2$, is there a subset C of V with $|C| \geq k$ such that for all $v_i, v_j \in C$ with $v_i \neq v_j$, $\{v_i, v_j\} \in E$.

Show that VC, IS, and Clique can be reduced to each other in polynomial time.

N.B. In the definitions of VC, IS, and Clique we have ignored self-loops (since we required $v_i \neq v_j$).

IS $\leq_{poly}$ Clique

Given an instance (G, k) of IS, we construct an instance (G', k') of Clique as follows:

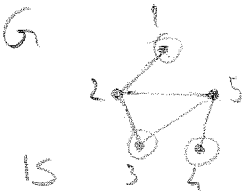
$k' = k$

Let $G = (V, E)$.

Then $G' = (V, E')$, where $E' = V \times V \setminus E$

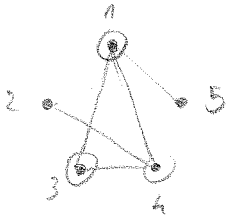
(i.e., the edges of E' are obtained by connecting all pairs of nodes that are not connected in E.)

E.g. G



IS

G'



Clique

The reduction works because the maximum independent set of G is precisely the maximum clique in the complement graph of G.

VC $\leq_{poly}$ IS

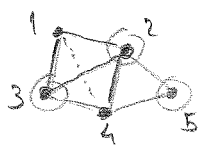
Given an instance (G, k) of VC, we construct an instance (G', k') of IS as follows:

$k' = |V| - k$

$G' = G$

The reduction works because the vertexes in a vertex-cover C cover all edges of G. Hence the set $V \setminus C$ must have no edges between its elements, and is thus an independent set

E.g.



The marked nodes $\{2,3,5\}$ are a VC of size 3. Hence, there cannot be an edge $\{1,4\}$

E 10.8

Clique $\leq_{poly}$ VC

Given an instance (G, k) of Clique, we construct an instance (G', k') of VC as follows:

$k' = |V| - k$

Let $G = (V, E)$.

Then $G' = (V, E')$, where $E' = V \times V \setminus E$