

CS3230: Design and Analysis of Algorithms  
Semester 2, 2020-21, School of Computing, NUS

Practice Problem Set 5

April 18, 2022

**Question 1:** A 2-CNF formula is a CNF formula where each clause consists of at most 2 literals. The MAX-2-SAT problem is: given a 2-CNF formula  $\phi$  and a number  $k$ , is there an assignment that satisfies at least  $k$  clauses of  $\phi$ ? Show that MAX-2-SAT is NP-complete. [**Hint:** Try a reduction from 3-SAT. For each 3-CNF clause involving 3 variables, give a set of ten 2-CNF clauses on 4 variables such that if the 3-CNF clause is satisfiable, then 7 of the 2-CNF clauses can be satisfied whereas otherwise, only at most 6 of the 2-CNF clauses can.]

**Answer:** First we note that if a truth setting of the variable makes at least  $k$  clauses true, then by substituting these truth values we can efficiently check the truth or falsity clause by clause, keeping a counter to hold the number of clauses found to be true so far. Hence *MAX-2-SAT* is in NP.

The following is a reduction from 3SAT to *MAX-2-SAT*. That is, given an instance of 3SAT we construct an instance of MAX-2-SAT so that a satisfying truth assignment of 3SAT can be extended to a satisfying truth assignment of MAX-2-SAT. Note that a satisfying truth assignment is one that makes at least  $k$  clauses true in the MAX-2-SAT instance.

Let  $S$  be the instance of 3SAT where the clauses are  $C_1, \dots, C_m$  where  $C_i = \{x_i, y_i, z_i\}$ , where each  $x_i, y_i$ , and  $z_i$  represents either a variable or its negation and  $1 \leq i \leq m$ . From  $S$  we build an instance  $S'$  of MAX-2-SAT as follows. Each  $C_i$  in  $S$  corresponds to a clause group  $C'_i$  in  $S'$  where,

$$C'_i = \{(w_i), (x_i), (y_i), (z_i), (\overline{x_i} + \overline{y_i}), (\overline{y_i} + \overline{z_i}), (\overline{x_i} + \overline{z_i}), (\overline{w_i} + x_i), (\overline{w_i} + y_i), (\overline{w_i} + z_i)\}$$

where  $w_i$  is a new variable and  $1 \leq i \leq m$ . We set  $k = 7m$

Each clause in  $S'$  as constructed above has at most two literals. It can be seen that the clauses in  $S'$  can be efficiently generated from the clauses in  $S$  in polynomial time. We now argue that a satisfying truth assignment of  $S$  exists if and only if it can be extended to a satisfying truth assignment for  $S'$  appropriately.

Assume that  $S$  is satisfiable. Then in a typical clause  $C_i = \{x_i, y_i, z_i\}$  either one or two or all the three variables are true.

- Let  $x_i = T, y_i = F, z_i = F$ . With this truth assignment in  $S'$ , if  $w_i = T$ , six clauses of  $C'_i$  become true; if  $w_i = F$ , seven clauses of  $C'_i$  become true.
- Let  $x_i = T, y_i = T, z_i = F$ . This truth assignment in  $S'$  together with  $w_i = T$  or  $w_i = F$ , makes seven clauses of  $C'_i$  true.
- Let  $x_i = T, y_i = T, z_i = T$ . With this truth assignment in  $S'$ , if  $w_i = T$ , seven clauses of  $C'_i$  become true; if  $w_i = F$ , six clauses of  $C'_i$  become true.

In summary, a satisfying truth assignment of  $S$  can be extended to a satisfying truth assignment of  $S'$  where exactly seven clauses in each clause group get satisfied. Moreover no setting of  $w_i$  causes more than seven of the ten clauses to be true in each clause group in  $S'$ .

Now assume that  $S$  is not satisfiable. Then in at least one clause  $C_i = \{x_i, y_i, z_i\}$  in  $S$ , we have  $x_i = F, y_i = F, z_i = F$ . With this truth assignment in  $S'$ , if  $w_i = T$ , four clauses of  $C'_i$  become true; if  $w_i = F$ , six clauses of  $C'_i$  become true. That is, if  $S$  is not satisfiable, no truth setting can make at least seven clauses true in each clause group  $C'_i$ . Since 3SAT is NP-complete, we conclude that MAX-2-SAT is also NP-complete.

**Question 2:** Consider the *Max-Clique problem*: Given an undirected graph  $G = (V, E)$  and an integer  $k$  decide whether there exists a clique of size at least  $k$  in  $G$  (i.e., as a subgraph of  $G$ ). Show that Max-Clique problem is NP-complete. [**Hint:** Try a reduction from the Maximum Independent Set problem.]

**Answer:** It suffices to show (a) the Max-Clique problem is in NP, and (b) Maximum Independent Set problem( $IS$ )  $\leq_p$  Max-Clique problem( $CP$ ).

To show (a), given a graph  $G$  and a positive integer  $k$ , we should be able to verify the certificate in polynomial time. The certificate is a subset  $V'$  of the vertices, which comprises the vertices belonging to the clique. We can validate this solution by checking that each pair of vertices belonging to the solution are adjacent, by simply verifying that they share an edge with each other. This can be done in polynomial time.

To show (b), let  $G = (V, E)$  and  $(G, k) \in IS$  then,  
 $(G, k) \in IS \iff G$  as independent subset of size at least  $k$   
 $\iff \overline{G}$  has a clique of size at least  $k$   
 $\iff (\overline{G}, k) \in CP$

Therefore, any instance of the clique problem can be reduced to an instance of the Independent Set problem. Therefore, Max-Clique problem is NP-complete.

**Question 3:** The circuit satisfiability problem is: Given a circuit with boolean gates (AND, OR, and NOT) with several input wires and one output wire, decide if there is an input that makes the output be true.

Give a poly-time reduction from circuit satisfiability to CNF-SAT. [**Hint:** For each gate  $k$  introduce a new variable  $x_k$ . Then create a sub-formula for each gate. e.g. Suppose for  $k$ -th gate the input wires are from (output wire of)  $i$  and  $j$ -th gate. Then the sub-formula will be  $(\overline{x_k} \vee x_i) \wedge (\overline{x_k} \vee x_j) \wedge (x_k \vee \overline{x_i} \vee \overline{x_j})$ . Similarly define sub-formulas for OR and NOT gates.]

**Answer:** We will need to add new variables. Suppose the circuit  $C$  has  $m$  gates, including input gates, then we introduce new variables  $x_1, \dots, x_m$ , with the intended meaning that variable  $x_j$  corresponds to the output of gate  $j$ .

We make a formula  $F$  which is the AND of  $m + 1$  sub-expression. There is a sub-expression for every gate  $j$ , saying that the value of the variable for that gate is set in accordance to the value of the variables corresponding to inputs for gate  $j$ .

We also have a  $(m + 1)$ -th term that says that the output gate outputs 1. There is no sub-expression for the input gates.

For a gate  $j$ , which is a NOT applied to the output of gate  $i$ , we have the sub-expression  
 $(x_i \vee x_j) \wedge (\overline{x_i} \vee \overline{x_j})$

For a gate  $k$ , which is an AND applied to the output of gates  $i$  and  $j$ , we have the sub-expression  
 $(\overline{x_k} \vee x_i) \wedge (\overline{x_k} \vee x_j) \wedge (x_k \vee \overline{x_i} \vee \overline{x_j})$

We have a similar sub-expression for OR gate.

We now have to show that this reduction works. Suppose  $C$  is satisfiable, then consider setting  $x_j$  being equal to the output of the  $j$ -th gate of  $C$  when a satisfying set of values is given in input. Such a setting for  $x_1, \dots, x_m$  satisfies  $F$ .

Suppose  $F$  is satisfiable, and give in input to  $C$  the part of the assignment to  $F$  corresponding to input gates of  $C$ . We can prove by induction that the output of gate  $j$  in  $C$  is also equal to  $x_j$ , and therefore the output gate of  $C$  outputs 1.

So  $C$  is satisfiable if and only if  $F$  is satisfiable.

**Question 4:** Give a poly-time reduction from CNF-SAT to 3-SAT. [**Hint:** For the clauses with more than three literals try to write them as AND of several clauses by introducing new variables.]

**Answer:** Take a formula  $F$  of CNF-SAT. We transform it into a formula  $F'$  of 3-SAT such that  $F'$  is satisfiable if and only if  $F$  is satisfiable.

Each clause of  $F$  is transformed into a sub-expression of  $F'$ . Clauses of length 3 are left unchanged.

A clause of length 1, such as  $(x)$  is changed as follows,

$$(x \vee y_1 \vee y_2) \wedge (x \vee y_1 \vee \bar{y}_2) \wedge (x \vee \bar{y}_1 \vee y_2) \wedge (x \vee \bar{y}_1 \vee \bar{y}_2)$$

where  $y_1$  and  $y_2$  are two new variables added specifically for the transformation of that clause.

A clause of length 2, such as  $(x_1 \vee x_2)$  is changed as follows

$$(x_1 \vee x_2 \vee y) \wedge (x_1 \vee x_2 \vee \bar{y})$$

where  $y$  is a new variable added specifically for the transformation of that clause.

For a clause of length  $k \geq 4$ , such as  $(x_1 \vee \dots \vee x_k)$ , we change it as follows

$$(x_1 \vee x_2 \vee y_1) \wedge (\bar{y}_1 \vee x_3 \vee y_2) \wedge (\bar{y}_2 \vee x_4 \vee y_3) \wedge \dots \wedge (\bar{y}_{k-3} \vee x_{k-1} \vee x_k)$$

where  $y_1, \dots, y_{k-3}$  are new variables added specifically for the transformation of that clause.

We now have to prove the correctness of the reduction,

For reductions from length 1 and 2, we can construct the truth table and show that all instances where  $F$  is true the output of  $F'$  is true, and all instance of false are false.

For reductions from length 4 and above, When  $F$  is satisfiable, at least one literal in  $x_1, \dots, x_k$  is true. If any one of  $x_1$  or  $x_2$  is True; set all additional variables  $y_1, \dots, y_{k-3}$  to False. Now, first clause of all literals in  $F'$  becomes True, therefore  $F'$  has a satisfying assignment.

If any one of  $x_{k-1}$  or  $x_k$  is True, set all additional variables  $y_1, \dots, y_{k-3}$  to True. Now, last clause of all literals in  $F'$  becomes Trues, therefore  $F'$  has a satisfying assignment.

If any other literal is true,  $x_i$  where  $i \notin \{1, 2, k-1, k\}$ . Now if we make  $y_1, \dots, y_{i-2}$  True and make  $y_{i-1}, \dots, y_{k-3}$  False, we satisfy all the clauses in  $F'$ . Therefore, there is a satisfying assignment.

For an assignment which makes  $F$  false, all literals in  $x_1, \dots, x_k$  has to be false. In such a case there is no assignment of  $y_1, \dots, y_{k-3}$  which will make  $F'$  true. Therefore, all instances where  $F$  is false  $F'$  is also false.

Therefore,  $F'$  is satisfiable if and only if  $F$  is satisfiable.