CS 7910 Computational Complexity Homework 1

Gopal Menon

January 23, 2016

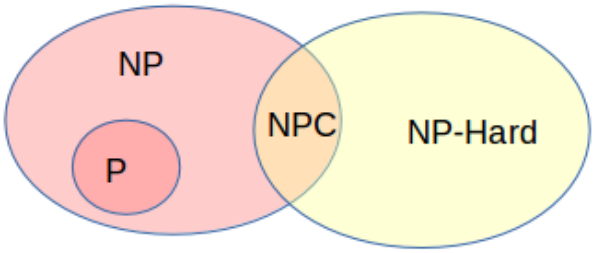

Figure 1: P, NP, NP-Complete (NPC) and NP-Hard problems

1. *For each of the following statements, please tell whether it is true and briefly explain your answer.*

   a) *If there exists a polynomial time algorithm that can solve the circuit satisfiability problem, then every problem in NPC can be solved in polynomial time.*

   The circuit satisfiability ($CSAT$) problem is NP-Complete. This means that

   i. $CSAT \in NP$ and

   ii. for all problems $Y \in NP$, $Y \leq_P CSAT$

   This means that every problem Y in NP can be reduced to $CSAT$. Or in other words, $CSAT$ is at least as hard as the hardest problem in NP. The set $NP-Complete$ is a subset of NP as shown in figure 1. So $CSAT$ is at least as hard as any $NP-Complete$ problem. This means that if if a polynomial time algorithm can be used to solve $CSAT$, then every other $NP-Complete$ problem (which will be in NP) can be solved in polynomial time since it would be easier or at most as difficult as $CSAT$. So the statement is true.

b) *If no polynomial time algorithm can solve the SAT problem, then no polynomial time algorithm can solve the circuit satisfiability problem.*

The $CSAT$ problem can be reduced to the SAT problem as we saw in class. This means that $CSAT \leq_P SAT$. i.e. the SAT problem is at least as hard if not harder than $CSAT$. If no polynomial time algorithm can be used to solve the SAT problem, it does not follow from this that no polynomial time algorithm can be used to solve the $CSAT$ problem. However since all NP problems can be reduced to an $NP-Complete$ problem, and since we know that $CSAT$ is an $NP-Complete$ problem, it follows that $SAT \leq_P CSAT$. Since no polynomial time algorithm can be used to solve the SAT problem, it follows that no polynomial time algorithm can be used to solve the $CSAT$ problem which is at least as hard as the SAT problem. So the statement is true.

c) *NP is the union of P and NPC.*

If $P = NP$ and since $NPC \subset NP$, in this case $NP = P \cup NPC$. But it is unlikely that $P = NP$. Since NPC problems are the hardest NP problems, the relation $NP = P \cup NPC$ can only be satisfied if $NP - P = NPC$ (as in figure 2). i.e. if each NP problem that cannot be solved in polynomial time can be reduced to any other NP problem in polynomial time. This seems unlikely as it would mean that the NP problems that cannot be solved is polynomial time are all NP complete. So the statement is not true.

d) *If there is an NPC problem A that is also in P, then NP is the union of P and NPC.*

Since the NPC problem A is in NP and is as hard or harder than any problem in NP, if A is in P, then every problem in NP can be solved in polynomial time. In other words $P = NP$. Since $NPC \subset NP$, it follows that $NP = P \cup NPC$. So the statement is true.

e) *If no problem of NPC is in P, then NP is the union of P and NPC.*

If no problem in NPC is in P, it just means that these are two disjoint sets. Since NPC problems are those that are in NP and can be reduced from all NP problems in polynomial time, there could be some NP problems that cannot be reduced from other NP problems in polynomial time. In this case NP cannot be the union of P and NPC, since this union will not include those NP problems that cannot be reduced from other NP problems in polynomial time. This is the situation shown in figure 1. So the statement is not true.

f) *If NP is the union of P and NPC, then $P = NP$.*

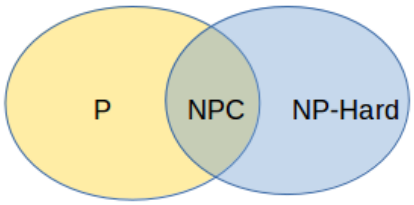

Figure 2: When $NP = P \cup NPC$

If $NP = P \cup NPC$, then it means that each NP problem that is not in P, can be converted to any other NP problem that is not in P. The situation can be the one as shown in figure 2. Here NP is the entire yellow ellipse and P is the part of the

ellipse that does not include NPC. So in this case $P \neq NP$. So the statement is not true.

2. *In class we studied a polynomial time problem reduction from the circuit satisfiability problem to the SAT problem. The following figure shows an instance of the circuit satisfiability problem. Give the SAT problem instance constructed from it based on the problem reduction we studied in class, i.e., give the Boolean variables and the clauses for the SAT problem instance.*

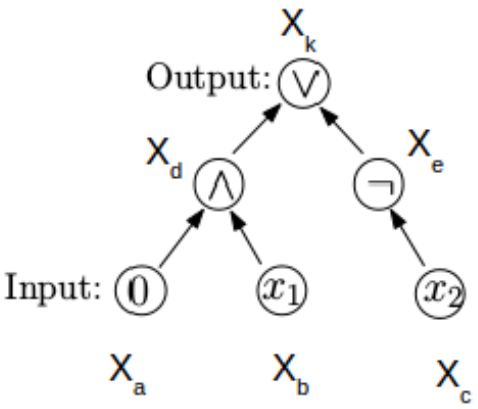

Figure 3: An instance of the circuit satisfiability problem

The inputs to the nodes have been marked X_a, X_b, X_c, X_d and X_e. The final output is marked X_k as shown in figure 3.

Given this instance of $CSAT$, we need to construct an equivalent instance of SAT.

The output of the AND operation will always be 0 as one of the inputs is 0. The equivalent clause will be

$$C_{AND} = \overline{X_d}$$

The clauses for the NOT operation will be

$$C_{NOT_1} = X_e \vee X_c$$

$$C_{NOT_2} = \overline{X_e} \vee \overline{X_c}$$

The clauses for the OR operation are

$$C_{OR_1} = X_k \vee \overline{X_d}$$

$$C_{OR_2} = X_k \vee \overline{X_e}$$

$$C_{OR_3} = \overline{X_k} \vee X_d \vee X_e$$

The clause for guaranteeing that the output is 1 will be

$$C_{OUTPUT} = X_k$$

The clause for the constant input is

$$C_{CONSTANT.INPUT} = \overline{X_a}$$

These clauses will cause the following conjunction to evaluate to 1.

$$C_{AND} \wedge C_{NOT_1} \wedge C_{NOT_2} \wedge C_{OR_1} \wedge C_{OR_2} \wedge C_{OR_3} \wedge C_{OUTPUT} \wedge C_{CONSTANT.INPUT}$$