ICCS310: Assignment 6

Possawat Sanorkam
possawat2017@hotmail.com
March 25, 2021

1: The Meaning of Things

(1) Class NP is the problems that can be verify within polynomial time. Basically, there is no efficient algorithm to solve the problem. But, we can verify it pretty quick.

(2) Without loss of generality, we can assume that there is a certificate and a verifier that check the certificate to verify whether it is a "yes" or "no" given that the verifier answer within polynomial time.

(3) NP-complete is the problem that can only be solved within polynomial time using a NFA. Besides, we can solve it in polynomial time using a machine that compute all possibilities at once.

(4) We can find a problem that is NP, then we find a polynomial time algorithm to change the solution from one problem to the problem that we want to show that it is NP-complete.

2: Closure of NP

(i) Disprove that $A \cap B$ must be in NP

Proof:

We want to show that $A \cap B$ must not be in NP. Let $a \in A$ and $b \in B$. We were given that $A \in \mathsf{NP}$ and $B \in \mathsf{NP}$. So, $(\forall a \in A)[a \in \mathsf{NP}]$ and $(\forall b \in B)[b \in \mathsf{NP}]$. However, if $(\forall a \in A)[a \notin B]$, then we got two disjoint sets of A and B. Then, $A \cap B = \emptyset$. We have that $\emptyset$ is a special language that is undecidable and there is no machine that can decide it, because we will not be able to find "yes" instance from nothingness. Therefore, $A \cap B$ cannot be in NP. $\square$

(ii) Disprove that $A \cup B$ must be in NP

Proof:

We want to show that $A \cup B$ must not be in NP. Let $a \in A$ and $b \in B$. We were given that $A \in \mathsf{NP}$ and $B \in \mathsf{NP}$. So, $(\forall a \in A)[a \in \mathsf{NP}]$ and $(\forall b \in B)[b \in \mathsf{NP}]$. However, if $(\forall a \in A)[a \notin B]$, then we got two disjoint sets of A and B. Then, $A \cup B = \Sigma^*$. We have that Σ^* is a special language that is undecidable and there is no machine that can decide it, because we will not be able to find "no" instance from everything. Therefore, $A \cup B$ cannot be in NP. $\square$

3: This is NP

Prove that $\mathsf{5COLOR} \in \mathsf{NP}$

Proof:

We want to show that $\mathsf{5COLOR} \in \mathsf{NP}$. So, we want to claim that there exist a polynomially-bounded certificate and a polynomially-bounded verifier.

Claim: There exists such polynomially-bounded certificate.

We claim that the certificate that is a yes instance is the list of vertices describing the color in which we colored them. The length of such a certificate is the same as the number of vertices given. So, we got the certificate.

Claim: There exists such polynomially-bounded verifier.

We claim that the verifier that verify the certificate will check through the list of vertices describing the color in which we colored them. If there exists an edge that contain same color on both ends, we have that the certificate is a "no" instance. Otherwise, it is a "yes" instance. The time complexity of this algorithm is $O(|E|)$ where E is the list of edges. So, we got the verifier.

Hence, we showed that there exist a polynomially-bounded certificate and a polynomially-bounded verifier. Therefore, $\mathsf{5COLOR} \in \mathsf{NP}$.

4: NP-Complete

(1) Show that

$$\mathsf{TOTAL} = \{\langle M\rangle|\text{ M is a Turing machine that halts on every input}\}$$

is undecidable

Proof:

(2) Show that

$$\mathsf{TOTAL} = \{\langle M\rangle|\text{ M is a Turing machine that halts on every input}\}$$

is undecidable

Proof:

5: Silver Lining If P = NP

Prove that if $\mathsf{P} = \mathsf{NP}$, then $\mathsf{SPC} \in \mathsf{P}$

Proof:

Suppose that $\mathsf{P} = \mathsf{NP}$, then $\mathsf{NP} = \mathsf{coNP}$. Let $\overline{\mathsf{SPC}}$ be a problem to check whether C is not the smallest-possible circuit with the exact same behavior as C. We want to show that $\overline{\mathsf{SPC}} = \mathsf{coNP}$.

We want to show that $\overline{\mathsf{SPC}} = \mathsf{coNP}$. So, we want to claim that there exist a polynomially-bounded certificate and a polynomially-bounded verifier.

Claim: There exists such polynomially-bounded certificate.

We claim that the certificate that is a "yes" instance is the circuit smaller gates than C and input values. The length of such a certificate is the less than the number of gates in C. So, we got the certificate.

Claim: There exists such polynomially-bounded verifier.

We claim that the verifier that verify the certificate will run the circuit. If the output is the same to C, it is "yes" instance. Otherwise, it is a "no" instance. The time complexity of this algorithm is $O(n)$ where n is the size of input values. So, we got the verifier.

Hence, we showed that there exist a polynomially-bounded certificate and a polynomially-bounded verifier. Therefore, $\overline{\mathsf{SPC}} = \mathsf{coNP}$.

Hence, if $\overline{\mathsf{SPC}} = \mathsf{coNP}$, then $\mathsf{SPC} = \mathsf{NP}$ because $\mathsf{P} = \mathsf{NP}$ and $\mathsf{P} = \mathsf{coNP}$.

Therefore, if $\mathsf{P} = \mathsf{NP}$, then $\mathsf{SPC} \in \mathsf{P}$. $\square$