Computability and Complexity

Recitation 1, Tuesday August 1, 2023

Ari Feiglin

Definition:

If G is a graph, a set of vertices S is **independent** if for every $u, v \in S$, $(u, v) \notin S$.

Exercise 1.1:

We define the following search problem

$$R_{\text{BigIS}} = \{(G, S) \mid S \text{ is independent and } |S| \geq |V| - 20\}$$

Our goal is to show $R_{\text{BigIS}} \in \mathbf{PF}$.

Firstly, R_{BigIS} is polynomially bound since if $(G, S) \in R_{\text{BigIS}}$ then $|S| \leq |G|$. And a simple algorithm for this is to iterate over every set of vertices of size ≤ 20, and look at their complement and check if it is independent. If it is, return it. Otherwise if we have finished iterating, return $\perp$.

The number of subsets of G of size 20 is

$$\binom{|G|}{20} = \binom{n}{20} = \frac{n!}{(n-20)! \cdot 20!} \in \Theta(n^{20})$$

And the time it takes to iterate over each set is $O(|V|^2 \cdot |E|)$ (iterate over every pair u and v, and check if they are neighbors). Thus this algorithm runs in $O(n^{23})$ time, as required.

Exercise 1.2:

Let us define

$$R_{\text{IS}} = \left\{ ((G, k), S) \;\middle|\; S \text{ is independent in } G \text{ and larger than } k \right\}$$

Is $R_{\text{IS}} \in \mathbf{PF}$? Is $R_{\text{IS}} \in \mathbf{PC}$?

(1) Notice that the algorithm from the previous question iterates over $\binom{|V|}{k}$ sets. But if $k = \frac{|V|}{2}$, the time complexity of the algorithm is greater than $2^{\frac{|V|}{2}}$, which is exponential. Thus the naïve algorithm will not work here. But this does not prove that $R_{\text{IS}} \notin \mathbf{PF}$, because there may be another algorithm which does solve R_{IS}. The question of the existence of such an algorithm is an open problem.

(2) Obviously if we are given $((G, k), S)$ then we can check if S is independent and larger than k, which will run in polynomial time, and this verifies the solution. Thus $R_{\text{IS}} \in \mathbf{PC}$.

Exercise 1.3:

Let us define the following decision problem:

$$\textit{composite} = \{n \in \mathbb{N} \mid n \text{ is reducible (not prime)}\}$$

Show that $\textit{composite} \in \mathbf{NP}$.

This is quite simple, let us define $V(x, y)$ to return 1 if and only if y divides x and $y \neq 1$ and $y \neq x$. Thus there exists a y such that $V(x, y) = 1$ if and only if x is not prime, so V is a verifier for *composite*. It runs in polynomial time (worst case if you don't want to think about how to define division, just iterate over $i \leq x$ and check if $iy = x$). And if $V(x, y) = 1$ then $y \leq x$, so V is polynomially bound. Thus V is a polynomial proof system for *composite*, and therefore *composite* is in **NP**.

Alternatively we could define $V(n, (k_1, k_2))$ which returns 1 if and only if $1 < k_1, k_2 < n$ and $n = k_1 k_2$. Then V verifies *composite*, and it runs in polynomial time, and if $V(n, (k_1, k_2)) = 1$ then $|(k_1, k_2)| = k_1 + k_2 \leq 2n$ (storing values as unary), so it is polynomially bound. Thus we have defined another polynomial proof system for *composite*.

Exercise 1.4:

Suppose $S_1, S_2 \in \mathbf{NP}$ and V_1 and V_2 are polynomial proof systems of S_1 and S_2 respectively. Prove or disprove:

(1) We define an algorithm V where given an input (x, y), it runs $V_1(x, y)$ and $V_2(x, y)$ and returns 1 if and only if one returns 1. Then V is a polynomial proof system for $S_1 \cup S_2$.

(2) We define an algorithm V where given an input (x, y), it runs $V_1(x, y)$ and $V_2(x, y)$ and returns 1 if and only if both return 1. Then V is a polynomial proof system for $S_1 \cap S_2$.

(1) This is true. V obviously is a verifier for $S_1 \cup S_2$ since if $x \in S_1 \cup S_2$ then x is in S_1 or S_2, and so there exists a y such that $V_1(x, y) = 1$ or $V_2(x, y) = 1$. Thus $V(x, y) = 1$. Otherwise $x \notin S_1 \cup S_2$ so x is not in S_1 or in S_2 and so for every y, $V_1(x, y) = 0$ and $V_2(x, y) = 0$ so $V(x, y) = 0$. So V is indeed a verifier for $S_1 \cup S_2$.

Now, V runs in polynomial time since suppose the time complexity of V_1 is bound by p_1 and the time complexity of V_2 is bound by p_2, then V is bound by $p_1 + p_2 + c$ (where c is the number of transitions to compare the results of V_1 and V_2 which will be constant). This is a polynomial, so V runs in polynomial time.

V is also polynomially bound, since if $V(x, y) = 1$ then $V_1(x, y) = 1$ or $V_2(x, y) = 1$ and so either $|y| \leq q_1(|x|)$ or $|y| \leq q_2(|x|)$ for the polynomial bounds q_1 and q_2 for V_1 and V_2. And so $|y| \leq q_1(|x|) + q_2(|x|)$ which is a polynomial. Thus V is polynomially bound.

So V is a polynomial proof system for $S_1 \cup S_2$ as required.

(2) This is false, take V_1 and V_2 to be the two verifiers for *composite* that we defined above. Then $S_1 = S_2 =$ *composite* so $S_1 \cap S_2 =$ *composite*. But V would not accept anything, since the types of inputs they accept are different (the first accepts a number, the second accepts a pair of numbers as witnesses). So V does not verify $S_1 \cap S_2$.

Another counter example can be constructed as followed: we define a new problem $S_1 = S_2 = \{0, 1\}^*$ (the set of all the binary strings) and $V_1(x, y) = 1$ if and only if $y = 1$ and $V_2(x, y) = 1$ if and only if $y = 0$. Then V_1 and V_2 are polynomial proof systems for $S_1 = S_2$, but V would similarly not accept anything and therefore not be a verifier for $S_1 \cap S_2$.

Exercise 1.5:

Show that if $S \in \mathbf{NP}$, then there exist two verifiers (I will use verifier here to mean polynomial proof system) V_1 and V_2 such that

$$V_1(x, y) = 1 \implies V_2(x, y) = 0$$

and

$$V_2(x, y) = 1 \implies V_1(x, y) = 0$$

Notice that $V_1(x, y) = 1 \implies V_2(x, y) = 0$ is equivalent to its contrapositive, which is $V_2(x, y) = 1 \implies V_1(x, y) = 0$. So we need only to prove $V_1(x, y) = 1 \implies V_2(x, y) = 0$.

We know there exists a verifier V for S, then let us define V_1 which takes as input $(x, (b, y))$ and returns 1 if and only if $b = 1$ and $V(x, y) = 1$. And we defin V_2 to take as input $(x, (b, y))$ and returns 1 if and only if $b = 0$ and $V(x, y) = 1$.

Then if $x \in S$ then there exists a y such that $V(x, y) = 1$ and so $V_1(x, (1, y)) = V_2(x, (0, y)) = 1$. And if $x \notin S$ then for every y, $V(x, y) = 0$ so $V_1(x, (b, y)) = V_2(x, (b, y)) = 0$. So these verify S, and are also obviously polynomially bound (since $|(b, y)| = |y| + 1 \leq p(|x|) + 1$) and run in polynomial time.

Now, if $V_1(x, (b, y)) = 1$ then $b = 1$ and so $V_2(x, (b, y)) = 0$ as required.