Complexity exercise set #4

for the tutorial on
May 12, 2022

Exercises marked with an asterisk ($*$) may be handed in for grading and can earn you a small bonus[1] on the exam, provided you submit your solutions via Brightspace in PDF before **15:15 on Monday May 16**.

Exercise 1$*$ (30 points) Prove the following

1. If $A \in \mathbf{NP}$ and $B \in \mathbf{NP}$, then $A \cup B \in \mathbf{NP}$
2. If $A \in \mathbf{NP}$ and $B \in \mathbf{NP}$, then $A \cap B \in \mathbf{NP}$
3. If $A \in \mathbf{NP}$ and $B \in \mathbf{NP}$, then $A \cdot B \in \mathbf{NP}$

Solution. For each item, suppose we have $A, B \in \mathbf{NP}$, we also call the verification algorithms A and B.

- We define a verifier for $A \cup B$ as follows:

  $(A \cup B)(x, y) :=$
  if $A(x, y) = 1$ **then return** *1*;
  else if $B(x, y) = 1$ **then return** *1*;
  else return *0*;

  If $x \in A \cup B$ then there is some y with $|y| = O(|x|^d)$ such that $A(x, y) = 1$ or $B(x, y) = 1$, if $x \notin A \cup B$ there can be no such certificate by definition of the verifiers for A and B, so we indeed have a verifier with certificate size polynomial in $|x|$. It is a polynomial-time verifier because we run two polynomial-time algorithms consecutively followed by a constant time operation i.e. $O(|(x, y)|^c + |(x, y)|^d + 1) = O(|(x, y)|^{\max\{c,d\}})$.

- We define a similar verifier for $A \cap B$:

  $(A \cap B)(x, y) :=$
  if $A(x, y.1) = 0$ **then return** *0*;
  else if $B(x, y.2) = 0$ **then return** *0*;
  else return *1*;

  Where we interpret y as a pair $(y.1, y.2)$.

  For $x \in A \cap B$, there are certificates y_1, y_2 such that $A(x, y_1) = 1$ and $B(x, y_2) = 1$, so taking y to be an encoding (which we can decode in

[1] For more details, see `https://cs.ru.nl/~awesterb/teaching/2022/complexity.html`.

polynomial time) of the pair (y_1, y_2), we have $(A \cap B)(x, y) = 1$. If $x \notin A \cap B$, then $x \notin A$ or $x \notin B$, so there can be no certificates for both A and B, so also no certificate for x. We therefore have a verifier with certificate size polynomial in $|x|$ for $x \in A \cap B$. The verifier runs in polynomial time by the same argument as above.

- For $A \cdot B$, we take the verifier:

  $(A \cdot B)(x, y) :=$
  if $x \neq y.1 \cdot y.2$ **then return** *0*;
  if $A(y.1, y.3) = 0$ **then return** *0*;
  else if $B(y.2, y.4) = 0$ **then return** *0*;
  else return *1*;

  where we now interpret the input y as a tuple $(y.1, y.2, y.3, y.4)$. Then, by definition, $x \in A \cdot B$ if and only if there are $y_1 \in A$ and $y_2 \in B$ such that $x = y_1 \cdot y_2$. By assumption we have certificates y_3, y_4 for y_1 and y_2 with $|y_3| = O(|y_1|^c)$ and $|y_4| = O(|y_2|^d)$ if and only if $y_1 \in A$ and $y_2 \in B$. For $x \in A \cdot B$, clearly $|y_1| = O(|x|) = |y_2|$, so the size of the tuple we construct is polynomial in $|x|$. Thus, we can provide a certificate of polynomial size if and only if $x \in A \cdot B$. Note that we can check the equality in the first step in $O(|x|)$ steps, so the algorithm is polynomial by a similar argument to the above.

■

Grading. For each item **5 points** are given for the verification algorithm and **5 points** are given for the explanation.

Exercise 2∗ (40 points) In this exercise, A and B are arbitrary decision problems. Prove the following

1. If $A \leq_P B$ and B is in **NP**, then $A \in$ **NP**
2. Suppose that we have a set B and two elements $x, y \in \{0, 1\}^*$ such that $x \in B$ and $y \notin B$. If $A \in$ **P**, then $A \leq_P B$
3. $A \leq_P \overline{B}$ if and only if $\overline{A} \leq_P B$

Solution.

- Suppose f is the function witnessing the reduction $A \leq_P B$ and consider the following algorithm:

  $A(x, y) :=$
  return $B(f(x), y)$

  As $B \in$ **NP**, for $f(x) \in B$ there is a y with $|y| = O(|f(x)|^d)$ such that $B(f(x), y) = 1$ and by assumption on f, $f(x) \in B \iff x \in A$. For $f(x) \notin B \iff x \notin A$ there is no such y, so we have a verifier for A. Note also that $|f(x)| = O(|x|^c)$ as f is polynomial-time computable, so $|y|$ is also polynomial in $|x|$. The verifier for A is then polynomial-time computable as f and B are by definition/assumption. Thus, $A \in$ **NP**.

- We require a polynomial-time computable function $f\colon \{0,1\}^* \to \{0,1\}^*$ such that $s \in A \iff f(s) \in B$. Consider the following algorithm:

  $f(s) :=$
  if $A(s) = 1$ **then return** x;
  else return y;

  As $A \in \mathbf{P}$, we can check $A(s) = 1$ in time polynomial in $|s|$, all other steps can be performed in constant time. Further, we have

  $$s \in A \implies A(s) = 1 \implies f(s) = x \implies f(s) \in B$$

  and

  $$s \notin A \implies A(s) = 0 \implies f(s) = y \implies f(s) \notin B.$$

  Thus, f is of the required form giving $A \leq_P B$.

- Suppose $A \leq_P \overline{B}$. Then there is a polynomial-time computable function $f\colon \{0,1\}^* \to \{0,1\}^*$ such that $x \in A \iff f(x) \in \overline{B}$. For the same f, we have

  $$x \in \overline{A} \iff x \notin A \iff f(x) \notin \overline{B} \iff f(x) \in B.$$

  So, $\overline{A} \leq_P B$.

  Now suppose $\overline{A} \leq_P B$ and let $g\colon \{0,1\}^* \to \{0,1\}^*$ be the reducing function. Then

  $$x \in A \iff x \notin \overline{A} \iff g(x) \notin B \iff g(x) \in \overline{B}.$$

  So, $A \leq_P \overline{B}$.

  We can also show either implication from the other. For example, if we have shown $A \leq_P \overline{B} \implies \overline{A} \leq_P B$, then we can show the converse as follows:

  Suppose $\overline{A} \leq_P B$, then also $\overline{A} \leq_P \overline{\overline{B}} = B$, so by the above implication we have $A = \overline{\overline{A}} \leq_P \overline{B}$.

∎

Grading. For the first two items, **5 points** are given for a correct algorithm and **5 points** for explanation.

For the third item, **10 points** are given for each direction of the implication, with **5 points** for choosing the correct algorithm (the assumed computable function) and **5 points** for the explanation in each case. If the second implication is proven from the first, the full **10 points** is given for a correct explanation.

For each item: if the answer is imprecise, then at most **5 points** are given.

Exercise 3$*$ (10 points) Suppose, $B \in \mathbf{NP}$ and $f : \{0,1\}^* \to \{0,1\}^*$ is computable in polynomial time. Show that $\{a \in \{0,1\}^* \mid f(a) \in B\} \in \mathbf{NP}$.

Solution. Define $X := \{a \in \{0,1\}^* \mid f(a) \in B\}$ and consider the following algorithm:

$X(x, y) :=$
return $B(f(x), y)$

Now $x \in X$ if and only if $f(x) \in B$ by definition. Then, as $B \in \mathbf{NP}$, $B(f(x), y)$ is polynomial-time computable and there is y such that $X(x, y) = B(f(x), y) = 1$ if and only if $f(x) \in B$. We have therefore constructed a verifier for X, so $X \in \mathbf{NP}$. ■

Grading. Again, **5 points** for a correct algorithm and **5 points** for a correct explanation.

If the answer is imprecise, then at most **5 points** are given

Exercise 4∗ (20 points) Put the following formulas in CNF

1. $A \to ((C \vee \neg B) \wedge A)$
2. $\neg((C \wedge A) \vee (B \wedge C))$
3. $(A \vee (B \wedge \neg(B \to \neg A))) \leftrightarrow \neg A$

Which of these formulas are satisfiable?

Solution.

- An equivalent CNF formula is: $(\neg A \vee \neg B \vee C)$. This is satisfiable (by any assignment making A or B false, or C true).
- An equivalent CNF formula is: $(\neg C \vee \neg A) \wedge (\neg B \vee \neg C)$, which is satisfiable (for example, by any assignment making C false).
- An equivalent CNF formula is: $A \wedge \neg A$. This formula is not satisfiable.

■

Grading. For each formula:

- **6 points** are given if the correct conjunctive normal form is given.
- **0 points** are given if the incorrect or no conjunctive normal form is given
- If the answer is not in conjunctive normal form, then **0 points** are given.

For the satisfiable formulas:

- **2 points** are given if **all** the correct formulas are denoted as satisfiable.
- **0 points** are given otherwise