Tutorial 4: NP-Completeness

CSC 463

March 6, 2020

1. The Cook-Levin theorem shows that there is a polynomial time reduction reduction $A \leq_p$ **3SAT** for any language $A \in$ **NP**. Sometimes this reduction can be described more easily than the reduction presented in the Cook-Levin theorem.

   Show that there is a polynomial time reduction **3COL** $\leq_\mathbf{p}$ **3SAT**, where **3COL** is the problem of deciding if a graph G is three-colourable.

2. Let $k > 3$ be an integer. Show that the problem of deciding if a graph has a k-colouring is **NP**-Complete assuming that **3COL** is **NP**-Complete.

3. In class, we stated that a Boolean formula ϕ in conjunctive normal form can be converted into a Boolean formula ϕ' with at most 3 literals per clause that is satisfiable iff ϕ is, thus showing a reduction **SAT** $\leq_\mathbf{p}$ **3SAT**. One way of this doing is converting breaking up any clause $l_1 \vee \cdots \vee l_n$ in ϕ with $n > 3$ literals into two clauses $(l_1 \vee l_2 \vee z_1) \wedge (\overline{z_1} \vee l_3 \vee \cdots \vee l_n)$, where z_1 is a new variable and then recursively applying this procedure to the longer clause, until each clause has at most three variables.

   Prove that ϕ is satisfiable if and only if ϕ' produced by this algorithm from ϕ is satisfiable.

4. Recall in the proof of the Cook-Levin theorem we used 2x3 sized windows to check if each row in the computational tableau follows from the previous one according to the transition function of a non-deterministic Turing machine N.

   Show that the proof would have failed if we tried to use 2x2 sized windows by giving an example of an illegal window that is consistent with the transition function of N across its 2x2 subwindows.

5. An interesting property of **NP**-Complete problems is that their search and decision versions are polynomially equivalent, while this is not believed to be true for an arbitrary problem in **NP**.

   Prove that for the case of **VERTEX − COVER** by showing that if you have an algorithm A that checks if a graph has a vertex cover of size k and only outputs a yes or no answer, then you use A to find a vertex covering of G of size k if one exists.