Schedule S1, (Computer Science, CS and Philosophy, Maths and CS)
Schedule B, MSc in Computer Science

Hilary Term 2019

Computational Complexity

Exercise class 2: P, NP, polynomial-time reductions

1. Given undirected graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ a homomorphism from G_1 to G_2 is a total function $h : V_1 \mapsto V_2$ satisfying the following property: for every edge $\{u, v\} \in E_1$ we have that $\{h(u), h(v)\} \in E_2$. The language Homomorphism is defined as follows:

$$\textsc{Homomorphism} = \{\langle G_1, G_2\rangle \mid \text{ there is a homomorphism from } G_1 \text{ to } G_2\}$$

A *vertex colouring* of a graph G with k colours is a function

$$c : V(G) \longrightarrow \{1, \ldots, k\}$$

such that adjacent nodes in G have different colours, i.e., $\{u, v\} \in E(G)$ implies $c(u) \neq c(v)$. k-Colouring is the problem of determining if a given graph G has a vertex colouring with k colours. The language k-Colouring is defined as follows:

$$k\text{-}\textsc{Colouring} = \{G \mid G \text{ has a vertex colouring with } k \text{ colours}\}.$$

Let G be an undirected graph and let k be an integer. G contains a clique of order k if there exists some subset $S \subseteq V(G)$ with $|S| = k$ such that there exists an edge $\{x, y\}$ for every pair of distinct vertices $x, y \in S$. The language Clique is then defined as follows:

$$\textsc{Clique} = \{\langle G, k\rangle \mid \ G \text{ is an undirected graph containing a clique of order } \geq k\}$$

Do the following:

(a) Show that k-Colouring is polynomially reducible to Homomorphism.

(b) Assuming that Clique is NP-complete, show that Homomorphism is NP-hard.

(c) Assume that we could find a polynomial time computable reduction from $\overline{3\text{-}\textsc{Colourability}}$ to Homomorphism. Knowing that 3-Colourability is NP-complete, discuss the complexity-theoretic implications that such a surprising finding would have.

2. Show that if Ptime = NP then

(a) NP is closed under complementation (i.e. if $\mathcal{L} \in$ NP then so is $\{w \in \Sigma^* : w \notin \mathcal{L}\}$)

(b) Every language in NP except $\emptyset$ and Σ^* is NP-complete.

3. Show that the following problem is NP-complete.

NonTotalSat	
Input:	Formula ϕ in CNF such that that the assignment that maps all variables in φ to true satisfies φ
Problem:	Does ϕ have a satisfying assignment mapping some variable to false?

4. Show that the following problem is NP-complete:

Double Satisfiability	
Input:	Formula ϕ in CNF
Problem:	Does ϕ have at least 2 different satisfying assignments?

5. Show that the following problem is NP-complete:

Bounded Halting	
Input:	A non-deterministic Turing machine M, a string w, and a string $\underbrace{1\dots1}_{t \text{ times}}$ of t symbols 1.
Problem:	Does M accept w in at most t steps?