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

Hilary Term 2022

Computational Complexity

Exercise class 3: more reductions, the class DP

1. The class DP is defined as follows. We say that a language $\mathcal{L}$ belongs to DP if there exist languages $\mathcal{L}_1 \in$ NP and $\mathcal{L}_2 \in$ co-NP such that $\mathcal{L} = \mathcal{L}_1 \cap \mathcal{L}_2$.

   Let G be an undirected graph and let k be an integer. G contains a clique of size 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:

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

   We define the following languages:

   $$\begin{aligned} \textsc{UniqueSat} &= \{\phi \ : \ \phi \text{ is a propositional formula with exactly one satisfying assignment}\} \\ \textsc{Sat} &= \{\phi \ : \ \phi \text{ is a satisfiable propositional formula}\} \\ \textsc{Sat-Unsat} &= \{\langle \phi, \psi\rangle \ : \ \phi \text{ and } \psi \text{ propositional formulas}, \phi \text{ satisfiable}, \psi \text{ unsatisfiable}\} \\ \textsc{ExactClique} &= \{\langle G, k\rangle \ : \ \text{The largest clique in } G \text{ is of size exactly } k\} \end{aligned}$$

   Answer the following questions:

   (a) Show that NP $\cup$ co-NP $\subseteq$ DP.

   (b) Show that UniqueSAT and ExactClique are in DP.

   (c) Show that Sat-Unsat is DP-complete.

   (d) Show that DP is contained in EXPTIME.

   (e) Suppose that we could find a polynomial time computable reduction from Sat-Unsat to $\overline{\textsc{Sat}}$. Discuss the complexity-theoretic implications that such a finding would have.

2. Given two Boolean formulae ϕ and ψ, we say that ϕ is *equivalent* to ψ if ϕ and ψ have the same set of variables, and for any truth assignment β to those variables, β makes ϕ true if and only if it makes ψ true. We say that a Boolean formula ϕ is *minimal* if there is no shorter formula ψ such that ϕ is equivalent to ψ.

   The language MF is the language of minimal Boolean formulae, i.e.:

   $$\text{MF} = \{\langle \phi\rangle \ : \ \phi \text{ is a minimal Boolean formula}\}.$$

   (a) Show that MF $\in$ PSPACE.

   (b) Explain the fallacy in the following argument: If $\phi \notin$ MF, then there exists a smaller equivalent formula. An NTM can verify that $\phi \in \overline{\text{MF}}$ by guessing a smaller formula ψ and checking if ϕ is equivalent to ψ. Therefore, MF $\in$ co-NP.

3. Recall that in lectures we introduced the notion of an oracle-based complexity class A^{B}, where A and B are complexity classes. A detailed definition is below.

   What relationships can you identify between DP, P^{NP}, and $\mathrm{P}^{\text{co}-\mathrm{NP}}$?

   Definition. An *oracle Turing Machine* (OTM) is a Turing Machine M that has a special read-write tape (the machine's oracle tape) and three special states: q_{query}, q_{yes}, q_{no}. To execute M, we specify in addition a language $\mathcal{O} \subseteq \{0,1\}^*$ that is used as the oracle for M. Whenever during the execution M enters the state q_{query}, the machine moves to the state q_{yes} if $w \in \mathcal{O}$ and q_{no} if $w \notin \mathcal{O}$, where w denotes the contents of the special oracle tape. Regardless of the choice of $\mathcal{O}$, a membership query to $\mathcal{O}$ counts only as a *single computation step.* For every $\mathcal{O} \in \{0,1\}^*$, $\mathrm{P}^{\mathcal{O}}$ is the class of problems that can be decided by a polynomial-time deterministic TM with oracle access to $\mathcal{O}$. The corresponding class of problems based on non-deterministic Turing machines is denoted as $\mathrm{NP}^{\mathcal{O}}$.