

COMPUTATIONAL COMPLEXITY THEORY (COMS 30126)
COURSEWORK

Please hand in written solutions to the four questions overleaf.

DUE DATE:

Please hand in at lecture on WEDNESDAY 22 APRIL 2009
(or to MVB 3.22 anytime before that day).

Handwritten solutions are fine.
If you wish to submit an electronic version
please email it to richard@cs.bris.ac.uk

Please ensure that the solutions are your own work.
They will be marked and returned,
forming 30% of the assessment for this course.

COMS30126 COURSEWORK 2009

(1)

(a) If \mathcal{L} is any language let $\mathcal{L}^* = \{w : w = w_1w_2 \dots w_k \text{ for some } k \text{ and } w_1, w_2, \dots, w_k \in \mathcal{L}\}$. i.e. \mathcal{L}^* is the language of all concatenations of finitely many words from \mathcal{L} . If $\mathcal{L} \in \text{NP}$ show that \mathcal{L}^* is in NP too.

(b) Consider any standard enumeration of all Turing machines and introduce the language $\text{HALT} = \{(n, x) : \text{the } n^{\text{th}} \text{ Turing machine with input } x \text{ is a halting computation}\}$. Is $\text{HALT} \in \text{NP}$? Show that HALT is NP-hard. (3+6 marks)

(2)

In lecture notes we gave a definition of poly-time reduction which is sometimes also called poly-time *Karp* reduction. There is an alternative notion of reduction called Cook reduction: a language \mathcal{A} is *poly-time Cook reducible* to language \mathcal{B} if there is a poly-time oracle TM \mathcal{M} that, given an oracle for deciding \mathcal{B} , can decide \mathcal{A} .

(a) Show that if \mathcal{A} is poly-time Karp reducible to \mathcal{B} then \mathcal{A} is poly-time Cook reducible to \mathcal{B} .

(b) Let \mathcal{L} be any language in co-NP. Show that \mathcal{L} is poly-time Cook reducible to 3-SAT. Would you expect \mathcal{L} to be poly-time Karp reducible to 3-SAT? (2+4 marks)

(3)

Suppose languages \mathcal{L}_1 and \mathcal{L}_2 are in RP. Is $\mathcal{L} = \mathcal{L}_1 \cap \mathcal{L}_2$ in RP? Give a justification for your answer. (5 marks)

(4)

A language $\mathcal{L} \subseteq \{0, 1\}^*$ is called a *unary language* if any word w in \mathcal{L} is a string of 1's i.e. $w = 11 \dots 1$ of some length k (but not all strings of 1's need be in \mathcal{L}). Such languages need not be simple things – in lectures we will see that there are unary languages that are even undecidable. In this question we'll aim to show that nevertheless, if some unary language is NPC then P must equal NP.

(a) If $\phi(x_1, \dots, x_n)$ is any Boolean formula with n variables, consider the full binary tree $T(\phi)$ of depth n defined as follows. Each node of the tree is a formula. The root node is $\phi(x_1, \dots, x_n)$. Its children are $\psi_0(x_2, \dots, x_n) = \phi(0, x_2, \dots, x_n)$ and $\psi_1(x_2, \dots, x_n) = \phi(1, x_2, \dots, x_n)$. Similarly if the formula ξ is a node then its children are the two formulae obtained by setting the first variable of ξ to 0 and 1 respectively. Show that ϕ is satisfiable iff the tree $T(\phi)$ has a leaf node that is 1.

(b) Let \mathcal{L} be any unary language. Show that if \mathcal{L} is NPC then $\text{P} = \text{NP}$.

Hint: Let 1^k denote a string $11 \dots 1$ of k 1's. If f is any poly-time reduction from 3-SAT to \mathcal{L} with poly running time $\leq n^c$ then for any 3-cnf formula ϕ of length n , $f(\phi) = 1^k$ with $k \leq n^c$. Use this fact to perform some creative topiary¹ on the tree structure in (a) (via its associated f values) to get a poly-time decider for 3-SAT. (2+8 marks)

¹Pocket Oxford Dictionary of Current English has the entry:

tōp'iarȳ (*adj., n.*) of &c. the clipping of trees &c. into fantastic shapes.