Introduction to Theoretical Computer Science

Exercise Sheet: Week 8 — Solutions

(1) The VERTEX-COVER problem: Given a graph $G = (V, E)$, a vertex cover is a subset $V' \subseteq V$ of the vertices such that every edge has at least one end in the cover V'. VERTEX-COVER is, given a graph G and an integer k, does G have a vertex cover of size k?

Give reductions from CLIQUE to VERTEX-COVER, and the other way round. (*Hint:* complementary graph.)

To reduce a clique problem to a vertex cover problem: given $G = (V, E)$, construct the complementary graph $\overline{G} = (V, (V \times V) \setminus E)$. If G has a k-clique C, then the vertices in C have no edges between them, and so every edge must touch $V \setminus C$, giving an $(n-k)$-cover. Conversely, if $\overline{G}$ has an $(n-k)$-cover C', then there can be no edge between two vertices in $V \setminus C'$, and so G has it a k-clique. The reduction also works the other way, of course.

(2) The following properties were mentioned in the lectures (with only brief explanations):

- PSpace $\supseteq$ PTime
- PSpace $\supseteq$ NPTime
- PSpace $\subseteq$ ExpTime

Write out explanations in enough detail to show how a proof would work.

A PTime *machine can clearly only consume P-much space, since in one instruction it can at most add one bit to the size of a register. For an* NPTime *machine, you can just loop through all the (exponentially many) possibilities, only using* PSpace *for each one, and polynomial space to keep track of the possibilities. Given a* PSpace *machine, if it runs for more than exponentially much time, it must be in a loop (since there are 2^{poly} possible configurations of the machine).*

(3) P^{NP} obviously includes all of NP and co-NP. So how does NP^{NP} differ from it – what else is there? (Assuming, that is, that $\text{P} \neq \text{NP}$.)

A naive way to put it is that with P^{NP}, you get to ask a polynomial number of NP *questions, but with NP^{NP} you can non-determistically ask 'exponentially many' such questions.*

However, exponentially many calls to an exponentially powerful oracle is still only exponential. So what gives?

The basic answer is that thinking of NP *as being exponentially powerful is only a* very crude *heuristic, and my 'explanation' is therefore very crude also. Remember: for all we know,* NP *might actually be* P*.*

Remember also that we know that the entire polynomial hierarchy is inside PSpace*, which is inside* ExpTime*. So actually, NP^{NP} is only exponential, and so is $\text{NP}^{\text{NP}^{\text{NP}}}$ and so on.*

So a more refined answer would be: We believe that NP *lets you do some amount of work, lets call it 'lots of' work, which is more*

than polynomially much work, but (much) less than exponentially much work. So P^{NP} *lets you make poly-many calls to lots of work; while* $\mathrm{NP}^{\mathrm{NP}}$ *lets you make lots of calls to lots of work. And it is reasonable that lots of lots is more than lots.*

(It needn't be – as far as we know, it is possible that NP *is more than* P*, but that* $\mathrm{NP}^{\mathrm{NP}}$ *is in fact the same as* P^{NP}*.)*