15-453 Addendum to Lecture 34: Turing-Reducibility

Mihai Budiu

April 26

In Lecture 34 we mentioned the fact that the reducibility relation can be extended to deal with non-decision problems. Here are some definitions which make this notion clear:

Definition: a Turing Machine M with an oracle for the language O (written M^O) behaves like a regular Turing Machine, but can answer questions of the form $x \in O$ in unit time.

We can view M^O as M with a hard-wired subroutine O which works in unit time.

The following two definitions can be applied to problems which are *not* decision problems:

Definition: A language A is Turing-reducible to the language B if there is a Turing Machine with Oracle M^A which accepts B. We write $A \leq^T B$.

Definition: A lanugage A is polynomial-time Turing-reducible to B if there is a Turing-reduction between A and B which takes a polynomial time. We write $A \leq_p^T B$.

This is the needed notion of reduction. We can prove the following facts:

- $A \leq_p B \Longrightarrow A \leq_p^T B$ (polynomial-time reducibility is stronger than poly-time Turing reducibility).
- If $A \leq_p^T B$, A is NP-complete and B can be solved in polynomial time, them P=NP.
- If $A \leq_p^T B$, and $B \in$P, then A can be solved in polynomial time.

The notion of reduction we need is exactly polynomial-time Turing reduction.