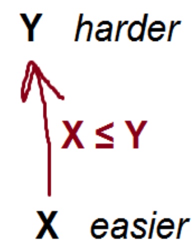
Mathematical Insights in Computing: Day 7
Chelsea Voss (csvoss@mit.edu)

Reductions, Oracles, and the Halting Problem



"Reducing Problem **X** to Problem **Y**" means showing that Problem **Y** is *at least as hard as* (or harder than!) Problem **X**. We do this via Turing reductions: if we can build a Turing machine that solves Problem **X** using Problem **Y** as a tool, it means that Problem **Y** is at least as powerful as Problem **X**. To say "Problem **X** is Turing-reducible to Problem **Y**", we write:

$$\mathbf{X} \leq_T \mathbf{Y}$$

Problem 1.

The "language of acceptance", A_L =

{ <M>, x | M(x) accepts}

Reduce A_L to $HALT_L$.

> *Hint: Create a machine for A_L – call it A_M – that uses a magic blackbox machine for $HALT_L$ – call it $HALT_M$ – as a component.*

Code for A_M(<M>, x):

This only proves that HALT is harder than A. Next, we'll prove that A is undecidable, by proving that A is harder than HALT.

Problem 2.

Go the other way. Reduce $HALT_L$ to A_L. (This proves A_L is undecidable.)

Code for $HALT_M$(<M>, x):

Problem 3.

The "language of emptiness", E_L =

$$\{ <M> \mid L(M) \text{ is } \varnothing \}$$

L(M) means "the language of M" – that is, the language that M decides. (*not* recognizes)

Reduce A_L to E_L.

Code for A_M(<M>, x):

Problem 4.

The "language of regular-language-accepting machines", $REGULAR_L$ =
{ <M> | L(M) is regular }

Show that $REGULAR_L$ is undecidable by reducing one of the above undecidable languages (A_L, E_L, or $HALT_L$) to $REGULAR_L$. Write your choice among those three in the blank.

Code for ______________, using $REGULAR_L$ as a magic blackbox:

Problem 5.

The "language of equality", EQ_L =

$$\{ \langle M_1 \rangle, \langle M_2 \rangle \mid L(M_1) = L(M_2) \}$$

Show that EQ_L is undecidable by reducing one of the above undecidable languages (A_L, E_L, $HALT_L$, or $REGULAR_L$) to EQ_L. Write your choice among those four in the blank.

Code for ______________, using EQ_L as a magic blackbox: