CIS 511: Theory of Computation **Feb 7, 2017**

Lecture 8: More on Reducibility

Professor Sampath Kannan *Zach Schutzman*

NB: *These notes are from CIS511 at Penn. The course followed Michael Sipser's Introduction to the Theory of Computation (3ed) text.*

Mapping Reducibility

Consider languages $A \subseteq \Sigma^*$, and $B \subseteq \Sigma^*$.

Recall the idea of a reduction is A reduces to B means that we can use B to solve A. If A is hard, then B must be (at least as) hard. If B is easy, then A must be easier.

Definition 8.1 *A Turing machine* ***computes*** *a function $f : \Sigma^* \to \Sigma^*$ if on all inputs x it halts with exactly $f(x)$ on its output tape.*

Definition 8.2 *A function f is* ***Turing computable*** *if there exists a Turing machine which computes it.*

Definition 8.3 *A* ***mapping reduction*** *is a Turing computable function $f : \Sigma^* \xrightarrow{TC} \Sigma^*$ such that if $x \in A$, then $f(x) \in B$. Similarly, if $y \notin A$, then $f(y) \notin B$.*

Recall last time we constructed a Turing reduction from $A_{TM} = \{\langle M, w\rangle | M \text{ accepts } w\}$ to $E_{TM}\{\langle M\rangle | L(M) = \emptyset\}$. Note that this was not a mapping reduction as we had to take the complement of the solver for E_{TM} at the end to properly complete the reduction.

Claim 8.4 *If we have a mapping reduction from A to B and A is not Turing recognizable, then B is not Turing recognizable.*

Proof: *If B is Turing recognizable, let $B = L(M)$. Then we construct a recognizer for A by using the mapping function on the input for A and running M on it.*

■

Example: *let $L_{AE} = \{\langle M\rangle | \epsilon \in L(M)\}$. We want to find a mapping reduction from A_{TM} to L_{AE}.*

Proof: *A typical input to A_{TM} is of the form $\langle M, w\rangle$. Our function f should map $\langle M, w\rangle$ to $\langle M'\rangle$. M', on any input, prints w on its input tape and runs M.*

We can see that M' either accepts everything or nothing, but it accepts if and only if $\langle M, w\rangle \in A_{TM}$. Because A_{TM} is undecidable, we must have L_{AE} undecidable as well. ■

Claim 8.5 *If $A \preceq_m B$, then $\bar{A} \preceq_m \bar{B}$.*

Proof:

We can use the same reduction as the original for the complement, because the function f preserves membership in sets.

■

Example: *let $EQ_{TM} = \{\langle M_1, M_2\rangle | L(M_1) = L(M_2)\}$. We will show this is unrecognizable by a mapping reduction.*

Proof: *We will reduce from $\overline{A_{TM}}$. $\overline{A_{TM}}$ takes input of the form $\langle M, w\rangle$. We want to make a construction that builds two equivalent TMs if M accepts w and two inequivalent TMs if M does not accept w.*

Let $L(M_1) = \emptyset$ and M_2 be a machine that on any input, runs M on w. This machine either accepts every string or no strings, depending on whether M accepts w.

$M_1 \sim M_2$ if and only if M rejects w, which is exactly what we wanted to show. Therefore, EQ_{TM} is unrecognizable. ■

Example: *let's also prove that $\overline{EQ_{TM}}$ is also not recognizable, via mapping reduction.*

Proof:

If we can show a mapping reduction from A_{TM} to EQ_{TM} then we know that $\overline{A_{TM}}$ has a mapping reduction to $\overline{EQ_{TM}}$, and EQ_{TM} is therefore unrecognizable.

Let's make $L(M_1) = \Sigma^*$ and M_2 works exactly as before. The same process shows that $M_1 \sim M_2$ if and only if M accepts w. Therefore, by examining the complements of these languages, we have a reduction from $\overline{A_{TM}}$ to $\overline{EQ_{TM}}$, so neither EQ_{TM} nor its complement are recognizable. ■

Definition 8.6 *A **property** of Turing machines is a function from TM descriptions to $\{0, 1\}$*

Definition 8.7 *A property of Turing machines is a **language property** if only depends on the language recognized by the TM and not on the description of the TM itself. That is, the property is true for any machine M recognizing a language L.*

Let $L_P = \{\langle M\rangle | M \text{ has property } P\}$

Definition 8.8 *A property is **non-trivial** if there is some Turing machine that has the property and some that does not. That is, L_P is not the set of all TMs nor is it empty.*

Claim 8.9 *(Rice's Theorem) L_P for any non-trivial language property of Turing machines is undecidable.*

Proof:

Let P be a non-trivial language property and L_P be the language of the property. We will show, by constructing a mapping reduction, that L_P is undecidable.

Consider A_{TM}. Assume that a Turing machine that accepts the empty language does not have property P (this is without loss of generality, because we can always consider $\overline{P}$ instead). On input $\langle M, w\rangle$, create a machine M' which, on any input x, first runs M on w. If M rejects, then M' rejects. Otherwise, then, runs some Turing machine T with property P on input x.

If M does not accept w, it may be because it rejects, or runs forever. In both cases, M' rejects. In the first, it explicitly does so. In the second, it never gets to the second step, and therefore rejects. So $L(M') = \emptyset$, which does not have property P.

If M accepts w, then $L(M') = L(T)$. T has property P, but since P is a language property, M' has the property as well. Therefore, the reduction results in M' having property P if and only if M accepts w. Since A_{TM} is undecidable, L_P must be as well.

■