Lecture #9: Proving Undecidability — Part Two
Key Concepts

Many-One Reductions

Let Σ_1 and Σ_2 be two alphabets (possibly the same) and let $L_1 \subseteq \Sigma_1^\star$ and $L_2 \subseteq \Sigma_2^\star$ be two languages over these alphabets.

Definition 1. A ***many-one reduction*** from L_1 to L_2 is a ***total*** function

$$f : \Sigma_1^\star \to \Sigma_2^\star$$

such that the following properties are satisfied.

(a) For every string $\omega \in \Sigma_1^\star$, $\omega \in L_1$ *if and only if* $f(\omega) \in L_2$.

(b) The function f is computable.

To prove that a language $L_1 \subseteq \Sigma_1^\star$ is many-one reducible to a language $L_2 \subseteq \Sigma_2^\star$,

1. Clearly and precisely describe the ***total*** function $f : \Sigma_1^\star \to \Sigma_2^\star$ that will be shown to be a "many-one reduction."
2. ***Prove*** that if $\omega \in L_1$ then $f(\omega) \in L_2$ for every string $\omega \in \Sigma^\star$.
3. ***Prove*** that if $\omega \notin L_1$ then $f(\omega) \notin L_2$ for every string $\omega \in \Sigma^\star$.
4. ***Sketch a Proof*** that f is computable — including enough detail for it to be reasonably clear that you really *could* write a Python or Java program that computes this function from strings to strings.

Decision Problems

Definition 2. A ***decision problem*** is a computational problem that asks a question — so that it has a "Yes/No" answer.

When an *alphabet* Σ and an *encoding* — mapping instances of the problem to strings in $\Sigma^\star$ — is chosen, *three* languages can be associated with the decision problem:

- *Language of Instances:* The set of strings in $\Sigma^\star$ that are encodings of instances of the decision problem.
- *Language of Yes-Instances:* The set of strings in $\Sigma^\star$ that are encodings of instances of the decision problem for which the answer is "Yes".
- *Language of No-Instances:* The set of strings in $\Sigma^\star$ that are encodings of instances of the decision problem for which the answer is "No".

The Usual Situation:

- Every instance of the decision problem is encoded by some string in $\Sigma^\star$, and no string in $\Sigma^\star$ is an encoding for two or more instances of the decision problem.
- We are, generally, interested in decision problems whose languages of instances are ***decidable***.
- When we say that we are "reducing one decision problem to another" — *in this course* — we generally mean that we are giving a ***many-one reduction*** from the ***language of Yes-instances for the first decision problem*** to the ***language of Yes-instances of the second decision problem***.

Suggested Process for Reductions with Decision Problems — for the "Usual Situation":

The main new idea: When you can, start by working at the high level (working with decision problems instead of languages of instances); get things right at that level; and then add detail, as needed to prove a corresponding reduction from one *language* to another.

1. Start by describing a mapping, φ, from instances of the first problem to instances of the second problem (which can be turned into the function, from strings to strings, that we will eventually need).
2. Show that the mapping φ maps each Yes-instance of the first problem to a Yes-instance of the second problem, and that φ maps each No-instance of the first problem to a No-instances of the second problem.

3. Now consider encoding schemes: One, e_1, maps instances of the first problem to strings over an alphabet Σ_1, and another, e_2, maps instances of the second problem to strings over an alphabet Σ_2.

   These will generally be given for you, if you are completing a reduction to solve a problem on an assignment.

4. Let $L_{I_1} \subseteq \Sigma_1^\star$ be the language of encodings of instances of the first problem, and let $L_{I_2} \subseteq \Sigma_2^\star$ be the language of encodings of instances of the second problem.

5. Confirm the ***Usual Situation:*** Prove (often, by stating a previously established result) that L_{I_1} is decidable, and confirm that $L_{I_2} \neq \Sigma_2^\star$ — so that there exists a string $\mu_{\text{Junk}} \in \Sigma_2^\star$ such that $\mu_{\text{Junk}} \notin L_{I_2}$. ***Say, clearly, what string μ_{Junk} you are going to use.***

6. Let $f : \Sigma_1^\star \to \Sigma_2^\star$ be defined as follows for every string $\mu \in \Sigma_1^\star$.
   - If $\mu \in L_{I_1}$ — so that μ is the encoding of some instances, α, of the first problem, then $f(\mu)$ is the encoding of the correspondence instance $f(\alpha)$ of the second problem — where φ is the mapping from instances of the first problem to instances of the second problem, chosen at the beginning of this process.
   - If $\mu \notin L_{I_2}$ then $f(\mu) = \mu_{\text{Junk}}$ — for the string $\mu_{\text{Junk}} \in \Sigma_2^\star$ chosen during the step before this one.

   Note: If you have completed all the above steps then $f : \Sigma_1^\star \to \Sigma_2^\star$ such that $\omega \in L_1$ if and only if $f(\omega) \in L_2$, for every string $\omega \in \Sigma_1^\star$.

7. Give an algorithm to compute the function f and prove that it is correct.
   - It is often advisable to start at the high level, using the mapping φ and, and the encoding schemes for problems, to give an algorithm to compute f that can be described using pseudocode.
   - Details can be added, as needed, to confirm that there is a multi-tape Turing that computes the function f, as well.

Properties

Closure Properties and Their Applications

Claim 1. *Suppose that $L_1 \subseteq \Sigma_1^\star$ and $L_2 \subseteq \Sigma_2^\star$ (for alphabets Σ_1 and Σ_2) are languages such that $L_1 \preceq_M L_2$. If L_2 is decidable then L_1 is decidable too.*

Claim 2. *Suppose that $L_1 \subseteq \Sigma_1^\star$ and $L_2 \subseteq \Sigma_2^\star$ (for alphabets Σ_1 and Σ_2) are languages such that $L_1 \preceq_M L_2$. If L_2 is recognizable then L_1 is recognizable too.*

The following are corollaries of the above claims.

Claim 3. *Suppose that $L_1 \subseteq \Sigma_1^\star$ and $L_2 \subseteq \Sigma_2^\star$ (for alphabets Σ_1 and Σ_2) are languages such that $L_1 \preceq_M L_2$. If L_1 is undecidable then L_2 is undecidable too.*

Claim 4. *Suppose that $L_1 \subseteq \Sigma_1^\star$ and $L_2 \subseteq \Sigma_2^\star$ (for alphabets Σ_1 and Σ_2) are languages such that $L_1 \preceq_M L_2$. If L_1 is unrecognizable then L_2 is unrecognizable too.*

Another process to prove that a language $L \subseteq \Sigma^\star$ is undecidable:

- Choose another language $\widehat{L} \subseteq \widehat{\Sigma}^\star$ (over some alphabet $\widehat{\Sigma}$) such that $\widehat{L}$ is undecidable.
- Prove that $\widehat{L} \preceq_{\mathsf{M}} L$.
- Conclude, by Corollary #3, above, that L must be undecidable too.

A process to prove that a language $L \subseteq \Sigma^\star$ is unrecognizable:

- Choose another language $\widehat{L} \subseteq \widehat{\Sigma}^\star$ (over some alphabet $\widehat{\Sigma}$) such that $\widehat{L}$ is unrecognizable.
- Prove that $\widehat{L} \preceq_{\mathsf{M}} L$.
- Conclude, by Corollary #4, above, that L must be unrecognizable too.

Relationships Between Reducibilities

Claim 5. *The set of many-one reductions forms a reducibility.*

Claim 6. *Let $L_1 \subseteq \Sigma_1^\star$ and let $L_2 \subseteq \Sigma_2^\star$. If $L_1 \preceq_M L_2$ then $L_1 \preceq_O L_2$.*

Claim 7. *There exist languages $L_1 \subseteq \Sigma_1^\star$ and $L_2 \subseteq \Sigma_2^\star$ (for alphabets Σ_1 and Σ_2) such that $L_1 \preceq_O L_2$ but $L_1 \not\preceq_M L_2$.*