Limits of Computation (CS:4340:0001 or 22C:131:001)
Kasturi Varadarajan

1 Proving Languages to be Undecidable using Reductions

Given a *language* $L \subseteq \{0,1\}^*$, we can define a corresponding function $f_L : \{0,1\}^* \to \{0,1\}$: let $f_L(x) = 1$ if $x \in L$ and $f_L(x) = 0$ if $x \notin L$.

Going the other way around, let $f : \{0,1\}^* \to \{0,1\}^*$ be a function. We can define the corresponding language $L_f = \{x \in \{0,1\}^* \mid f(x) = 1\}$.

We say that Turing machine M decides a language L to mean that M computes the corresponding function f_L.

Reducibility. Let L_1 and L_2 be two languages. We say that L_1 is many-one reducible to L_2 if there is an algorithm (i.e., a Turing Machine) A that halts on every input $x \in \{0,1\}^*$, and has the property that for any $x \in \{0,1\}^*$,

$$x \in L_1 \Leftrightarrow A(x) \in L_2.$$

This is similar to the Karp-reducibility that we use to show NP-completeness of problems, except that we do not require that the algorithm A run in polynomial time.

Claim 1. *Suppose that L_1 is many-one reducible to L_2, and that L_2 is decidable. Then L_1 is decidable as well.*

Proof. Let A be the algorithm (TM) that "reduces" L_1 to L_2 and M_{L_2} be the algorithm (TM) that decides L_2. We describe a TM M_{L_1} that decides L_1.

On input $x \in \{0,1\}^*$, M_{L_1} runs A on x to get $A(x)$ and then runs M_{L_2} on $A(x)$. It outputs $M_{L_2}(A(x))$. We observe:

$$x \in L_1 \implies A(x) \in L_2 \implies M_{L_2}(A(x)) = 1 \implies M_{L_1}(x) = 1;$$

$$x \notin L_2 \implies A(x) \notin L_2 \implies M_{L_2}(A(x)) = 0 \implies M_{L_1}(x) = 0.$$

Thus M_{L_1} decides L_1. □

`Hello-World`. We define a function `Hello-World` : $\{0,1\}^* \to \{0,1\}$. Let `Hello-World`$(\alpha) =$ 1 if M_α, when input the empty string, halts and outputs the string 10101010; let `Hello-World`$(\alpha) =$ 0 otherwise. Abusing notation slightly, let `Hello-World` denote the language corresponding to this function as well.

Claim 2. *`Hello-World` is undecidable.*

Proof. We show that `Halt` is many-one reducible to `Hello-World`. By Claim 1, this is all we need to show.

Recall that $\texttt{Halt} = \{\langle \alpha, x\rangle \mid M_\alpha \text{ halts on } x\}$. Given $\langle \alpha, x\rangle$, our reduction algorithm A constructs the encoding of the TM $M' = M'_{\alpha,x}$ that works as follows:

"On input y, (a) Write α, x on one of the tapes; (b) Use the universal TM U to simulate M_α on x; (c) If U halts, output 10101010 and halt."

Essentially, the transition function of M' resembles that of the universal TM. However, it also has two additional parts:

1. M' needs to write α and x on its tape before invoking the universal TM on α and x. The logic for this writing is hard-coded into the transition function of M'. Note that α and x are not inputs to M'.

2. After the universal TM halts, M' needs to write 10101010 on its output tape. This is again hard-coded onto the transition function of M'.

What is the behavior of M'? If $\langle \alpha, x\rangle \in \texttt{Halt}$, then $M'(y) = 10101010$ for *every* y and in particular when y is the empty string. Thus $\lfloor M' \rfloor \in \texttt{Hello-World}$ in this case.

If $\langle \alpha, x\rangle \notin \texttt{Halt}$, then M' does not halt on any input. Thus $\lfloor M' \rfloor \notin \texttt{Hello-World}$ in this case.

So we have shown that `Halt` is many-one reducible to `Hello-World` as desired. □

`AAS`. We define a function $\texttt{AAS} : \{0,1\}^* \to \{0,1\}$. Let $\texttt{AAS}(\alpha) = 1$ if $M_\alpha(y) = 1$ for every $y \in \{0,1\}^*$; let $\texttt{AAS}(\alpha) = 0$ otherwise. `AAS` is an abbreviation for "Accepts All Strings". Let `AAS` also denote the corresponding language.

Claim 3. *`AAS` is undecidable.*

Proof. It suffices to show that `Halt` is reducible to `AAS`.

Given $\langle \alpha, x\rangle$, our reduction algorithm A constructs the encoding of the TM $M' = M'_{\alpha,x}$ that works as follows:

"On input y, (a) Write α, x on one of the tapes; (b) Use the universal TM U to simulate M_α on x; (c) If U halts, output 1 and halt."

What is the behavior of M'? If $\langle \alpha, x\rangle \in \texttt{Halt}$, then $M'(y) = 1$ for *every* y. Thus $\lfloor M' \rfloor \in \texttt{AAS}$ in this case.

If $\langle \alpha, x\rangle \notin \texttt{Halt}$, then M' does not halt on any input y. Thus $\lfloor M' \rfloor \notin \texttt{AAS}$ in this case. □

You can see a general pattern in the arguments that `Hello-World` and `AAS` are undecidable. Informally, any question about the run-time behavior of Turing machines is undecidable.[1] One very general result in this direction is Rice's Theorem, see Exercise 1.12 of the textbook [1] and the book by Sipser [2].

In contrast, consider the language

$$\texttt{Has-Ten-States} = \{\alpha \in \{0,1\}^* | M_\alpha \text{ has ten states}\}.$$

[1] Update: This sentence is an example of something I wrote without thinking things through. Perhaps it is better to say that many such questions are undecidable.

To be more precise, for a string α to be in the language, α must really be the encoding of a TM according to our scheme, and this TM must have 10 states. This language is of course decidable. Deciding this language is about answering a question about the transition function of the TM, and not its run-time behavior.

References

[1] S. Arora and B. Barak. Computational Complexity, A Modern Approach. Cambridge University Press.

[2] M. Sipser. Introduction to the Theory of Computation. PWS Publishing Company.