Computer Science 351

Proofs of Undecidability — Examples I

Instructor: Wayne Eberly

Department of Computer Science
University of Calgary

Lecture #16

Goal for Today

- Another proof that a language is undecidable, using a many-one reduction, will be presented.

Decidable Languages

Recall that the following languages have been proved to be *decidable*:

- TM $\subseteq \Sigma_{\mathsf{TM}}^{\star}$: Valid encodings of Turing machines (whose start state is not a halting state)
- TM+I $\subseteq \Sigma_{\mathsf{TM}}^{\star}$: Valid encodings of Turing machines *M* and strings of symbols over the input alphabet for *M*.

Undecidable Languages

The following languages are *undecidable:*

- $\mathrm{A}_{\mathrm{TM}} \subseteq \mathrm{TM+I} \subseteq \Sigma_{\mathrm{TM}}^{*}$: Encodings of Turing machines *M* and strings ω of symbols over the input alphabet for *M* such that *M* accepts ω (see Lecture #13)
- $\mathrm{HALT}_{\mathrm{TM}} \subseteq \mathrm{TM+I} \subseteq \Sigma_{\mathrm{TM}}^{\star}$: Encodings of Turing machines *M* and strings ω of symbols over the input alphabet for *M* such that *M* halts when executed on input ω (see Lecture #15)

The Language All_{TM}

Let $\text{All}_{\text{TM}} \subseteq \Sigma^{\star}_{\text{TM}}$ be the set of encodings of Turing machines

$$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$$

(whose start state is not a halting state) such that $L(M) = \Sigma^{\star}$.

- $\text{All}_{\text{TM}} \subseteq \text{TM}$.
- We will prove that All_{TM} is ***undecidable*** by proving that $\text{A}_{\text{TM}} \preceq_{\text{M}} \text{All}_{\text{TM}}$.

A Reduction from A_{TM} to All_{TM}

What Do We Need to Do?

We must describe a total function $f : \Sigma_{\text{TM}}^{\star} \to \Sigma_{\text{TM}}^{\star}$ which satisfies the following properties:

- For every string $\mu \in \Sigma_{\text{TM}}^{\star}$,

  $\mu \in \text{A}_{\text{TM}}$ *if and only if* $f(\mu) \in \text{All}_{\text{TM}}$.
- The function f is computable.

A Reduction from A_{TM} to $\mathrm{All}_{\mathrm{TM}}$

Handling a Pesky Case

- Not all strings in $\Sigma_{\mathrm{TM}}^{\star}$ encode Turing machines and input strings for them — only strings in the ***decidable*** language TM+I do.
- If $\mu \in \Sigma_{\mathrm{TM}}^{\star}$ and $\mu \notin$ TM+I then $\mu \notin \mathrm{A}_{\mathrm{TM}}$, since $\mathrm{A}_{\mathrm{TM}} \subseteq$ TM+I. We want to define $f(\mu)$ so that $f(\mu) \notin \mathrm{All}_{\mathrm{TM}}$ in this case.
- Recall that $\mathrm{All}_{\mathrm{TM}} \subseteq \mathrm{TM}$, where TM is the language of encodings of Turing machines. If x_{No} is any string in $\Sigma_{\mathrm{TM}}^{\star}$ such that $x_{\mathrm{No}} \notin \mathrm{TM}$ then $x_{\mathrm{No}} \notin \mathrm{All}_{\mathrm{TM}}$ — so that setting $f(\mu)$ to be x_{No} ensures that $f(\mu) \notin \mathrm{All}_{\mathrm{TM}}$, as is needed here.
- Since $\lambda \notin \mathrm{TM}$ we can choose x_{No} to be λ for this problem.

A Reduction from A_{TM} to $\mathrm{All}_{\mathrm{TM}}$

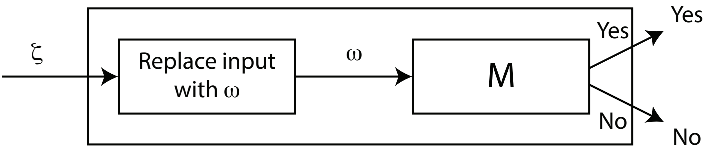
We are left with the problem of defining $f(\mu)$ when $\mu \in$ TM+I.

- In this case μ is the encoding of some Turing machine

  $$M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\mathrm{accept}}, q_{\mathrm{reject}})$$

  and some input string $\omega \in \Sigma^\star$ for the encoded Turing machine M.
- Suppose that we set $f(\mu)$ to be the encoding of another Turing machine, $\mathcal{M}_{\langle M, \omega \rangle}$, with the same input alphabet, Σ, as M, the same tape alphabet, Γ, as M, and such that $\mathcal{M}_{\langle M, \omega \rangle}$ has the structure shown on the following slide.

A Reduction from A_{TM} to $\mathrm{All}_{\mathrm{TM}}$



$\mathcal{M}_{\langle M, \omega \rangle}$

A Reduction from A_{TM} to $\mathrm{All}_{\mathrm{TM}}$

$\mathcal{M}_{\langle M, \omega \rangle}$ implements the following algorithm:

On input $\zeta \in \Sigma^\star$ {

1. Replace ζ with ω on the tape, and enter M's start state (so that M is in its initial configuration for input ω).
2. Run M (now, with input ω) — *accepting* if M eventually accepts ω, *rejecting* if M eventually rejects ω, and *looping* otherwise.

}

A Reduction from A_{TM} to $\mathrm{All}_{\mathrm{TM}}$

If $\omega = \lambda$ then step 1 can be expanded as follows — where $\sigma_1 \in \Sigma$ is as described in Lecture #12:

1a) Replace the symbol on the first cell of the tape with σ_1, moving right.

1b) Replace each non-blank symbol (after the copy of σ_1) with $\sqcup$, moving right. Move left when $\sqcup$ is seen without changing it.

1c) Move left past each copy of $\sqcup$ without changing it. When a non-blank symbol is seen replace this with $\sqcup$, moving left, and enter the start state for M.

This can be implemented using three states (which will be named q_0, q_1 and q_2).

A Reduction from A_{TM} to $\mathrm{All}_{\mathrm{TM}}$

If $|\omega| = 1$, so that $\omega = \alpha_1$ for some symbol $\alpha_1 \in \Sigma$, then step 1 can be expanded as follows, instead.

1a) Replace the symbol on the first cell of the tape with α_1, moving right.

1b) Replace the symbol on the second cell of the tape with $\sqcup$, moving right.

1c) Replace each non-blank symbol (after the second cell) with $\sqcup$, moving right. Move left when $\sqcup$ is seen without changing it.

1d) Move left over each copy of $\sqcup$ on the tape without changing it. When a non-blank symbol (which must be σ_1) is seen, move left without changing this symbol, and enter the start state for *M*.

This can be implemented using four states (named q_0, q_1, q_2 and q_3).

A Reduction from A_{TM} to All_{TM}

If $|\omega| = n \geq 2$, so that

$$\omega = \alpha_1 \alpha_2 \dots \alpha_n$$

for symbols $\alpha_1, \alpha_2, \dots, \alpha_n \in \Sigma$, then step 1 can be expanded as follows.

1a) Replace the symbol on the first cell of the tape with $\sqcup$, moving right.

1b) `for` $i = 2, 3, \dots, n$ $\{$
  Replace the symbol on the tape with α_i, moving right.
$\}$

1c) Replace the symbol now visible (at the $n+1^{\text{st}}$ cell) with $\sqcup$, moving right.

A Reduction from A_{TM} to All_{TM}

1d) Replace each non-blank symbol (after the $n+1^{\text{st}}$ cell) with $\sqcup$, moving right. Move left when $\sqcup$ is seen without changing it.

1e) Move left past each copy of $\sqcup$, without changing it. When a non-blank symbol is seen move left past it, without changing it either.

1f) Move left past each non-blank symbol without changing it. When $\sqcup$ is seen, replace this with α_1, moving left, and enter the start state for M.

This can be implemented using $n+3$ states (named $q_0, q_1, \ldots, q_{n+2}$).

A Reduction from A_{TM} to $\mathrm{All}_{\mathrm{TM}}$

- States must be renamed in the copy of M included in $\mathcal{M}_{\langle M, \omega \rangle}$: If M included the set of states

  $$Q = \{q_0, q_1, \dots, q_k, q_{\text{accept}}, q_{\text{reject}}\}$$

  for some non-negative integer k then, for each integer i such that $0 \le i \le k$, the name of state q_i should be changed to q_{i+n+3} (for $n = |\omega|$, as above).

A Reduction from A_{TM} to $\mathrm{All}_{\mathrm{TM}}$

Exercise:

- Confirm that if $\mathcal{M}_{\langle M, \omega \rangle}$ is produced from M and ω as described, above, then $\mathcal{M}_{\langle M, \omega \rangle}$ is a Turing machine with $n + k + 5$ states that implements the above algorithm.

If $\mu \in \text{A}_{\text{TM}}$ then $f(\mu) \in \text{All}_{\text{TM}}$

Claim #1: Let $\mu \in \Sigma_{\text{TM}}^{\star}$. If $\mu \in \text{A}_{\text{TM}}$ then $f(\mu) \in \text{All}_{\text{TM}}$.

Proof: Let $\mu \in \Sigma_{\text{TM}}^{\star}$ such that $\mu \in \text{A}_{\text{TM}}$

- Then μ is the encoding of a Turing machine M and input string ω, for M, such that M accepts ω.
- Consider the Turing machine $\mathcal{M}_{\langle M, \omega \rangle}$.
- $\mathcal{M}_{\langle M, \omega \rangle}$ replaces its input string, ζ, with ω — and then runs M. Since M eventually accepts ω, the input string ζ is eventually accepted by $\mathcal{M}_{\langle M, \omega \rangle}$.
- Thus the language of $\mathcal{M}_{\langle M, \omega \rangle}$ is $\Sigma^{\star}$.
- Thus this machine's encoding, $f(\omega)$, is in All_{TM}. $\square$

If $\mu \notin \text{A}_{\text{TM}}$ then $f(\mu) \notin \text{All}_{\text{TM}}$

Claim #2: Let $\mu \in \Sigma_{\text{TM}}^{\star}$. If $\mu \notin \text{A}_{\text{TM}}$ then $f(\mu) \notin \text{All}_{\text{TM}}$.

Proof: Let $\mu \in \Sigma_{\text{TM}}^{\star}$ such that $\mu \notin \text{A}_{\text{TM}}$. One of three cases holds:

1. $\mu \notin$ TM+I.
2. $\mu \in$ TM+I, but μ is the encoding of a Turing machine M and input string ω, for M, such that M rejects ω.
3. $\mu \in$ TM+I, but μ is the encoding of a Turing machine M and input string ω, for M, such that M loops on ω.

If $\mu \notin \text{A}_{\text{TM}}$ then $f(\mu) \notin \text{All}_{\text{TM}}$

Case: $\mu \notin$ TM+I.

- Then $f(\mu) = \lambda$.
- Since $\lambda \notin$ TM, $\lambda \notin \text{All}_{\text{TM}}$, as required.

If $\mu \notin \mathrm{A}_{\mathrm{TM}}$ then $f(\mu) \notin \mathrm{All}_{\mathrm{TM}}$

Case: $\mu \in$ TM+I, but μ is the encoding of a Turing machine *M* and input string ω, for *M*, such that *M* rejects ω.

- Consider the Turing machine $\mathcal{M}_{\langle M, \omega \rangle}$.
- $\mathcal{M}_{\langle M, \omega \rangle}$ ***rejects*** replaces its input string ζ with ω — and then runs *M*. Since *M* eventually rejects ω, the input string ζ is eventually rejected by $\mathcal{M}_{\langle M, \omega \rangle}$.
- Thus the language of $\mathcal{M}_{\langle M, \omega \rangle}$ is $\emptyset$.
- Thus this machine's encoding, $f(\mu)$, is *not* in $\mathrm{All}_{\mathrm{TM}}$, as required.

If $\mu \notin \text{A}_{\text{TM}}$ then $f(\mu) \notin \text{All}_{\text{TM}}$

Case: $\mu \in$ TM+I, but μ is the encoding of a Turing machine *M* and input string ω, for *M*, such that *M* loops on ω.

- Consider the Turing machine $\mathcal{M}_{\langle M, \omega \rangle}$.
- $\mathcal{M}_{\langle M, \omega \rangle}$ replaces its input string ζ with ω — and then runs *M*. Since *M* loops on ω, $\mathcal{M}_{\langle M, \omega \rangle}$ loops on its input string, ζ.
- Thus the language of $\mathcal{M}_{\langle M, \omega \rangle}$ is $\emptyset$.
- Thus this machine's encoding, $f(\mu)$, is *not* in All_{TM}, as required.

It has now been shown that $f(\mu) \notin \text{All}_{\text{TM}}$ in all cases, as needed to establish the claim. $\square$

f is Computable

Claim #3: The function f is computable.

Sketch of Proof: Recall that the language TM+I is ***decidable*** — so that it is possible to use a Turing machine to decide whether the input string, μ, belongs to TM+I.

- If $\mu \notin$ TM+I then $f(\mu)$ is the empty string — which is certainly easy to compute.
- Otherwise $f(\mu)$ is the *encoding* of a Turing machine, $\mathcal{M}_{\langle M, \omega \rangle}$, that is as described above. The proof of this claim can be completed by giving additional details about $\mathcal{M}_{\langle M, \omega \rangle}$, as needed to show how $f(\mu)$ is related is to μ and to see that $f(\mu)$ can be *computed* from μ.

  A supplemental document provides some of these details.
  $\square$

Finishing the Proof

- Since f is a well defined total function from $\Sigma_{\mathrm{TM}}^{\star}$ to $\Sigma_{\mathrm{TM}}^{\star}$, Claims #1, #2 and #3 imply that f is a ***many-one reduction*** from A_{TM} to $\mathrm{All}_{\mathrm{TM}}$.
- Thus $\mathrm{A}_{\mathrm{TM}} \preceq_{\mathrm{M}} \mathrm{All}_{\mathrm{TM}}$.
- Since A_{TM} is undecidable it now follows that $\mathrm{All}_{\mathrm{TM}}$ is undecidable, as well.