Lecture #17: Proofs of Undecidability — Examples II
What Will Happen During the Lecture

Remember... You Had Homework!

Students were asked to work through the following set of lecture notes before this lecture.

- Lecture Notes — "Proofs of Undecidability — Examples II".

As always, you may attend the lecture presentation if you have not worked through this material ahead of time — but it will not be repeated for you, and you might get a little bit lost, during the presentation, if you haven't worked through this.

Problems To Be Solved

When using many-one reductions to prove that a language is undecidable, ***you do not have to use $\mathbf{A_{TM}}$ as the undecidable language used in your reduction.***

With that noted, let $\Sigma_{\text{2TM}} = \Sigma_{\text{TM}} \cup \{\#\}$. A *pair* of Turing machines M_1 and M_2 can be encoded as a string $\alpha\#\beta \in \Sigma^\star_{\text{2TM}}$ where $\alpha \in \text{TM} \subseteq \Sigma^\star_{\text{TM}}$ is the encoding for M_1 and $\beta \in \text{TM} \subseteq \Sigma^\star_{\text{TM}}$ is the encoding for M_2.

1. Let $\text{Pair}_{\text{TM}} \subseteq \Sigma^\star_{\text{2TM}}$ be the language of encodings of pairs of Turing machines

   $$M_1 = (Q_1, \Sigma, \Gamma_1, \delta_1, q_{0,1}, q_{A,1}, q_{R,1})$$

   and

   $$M_2 = (Q_2, \Sigma, \Gamma_2, \delta_2, q_{0,2}, q_{A,2}, q_{R,2})$$

   with the same input alphabet Σ. The lecture presentation will include a sketch of a proof that the language Pair_{TM} is ***decidable***.

2. Now let

$$\mathsf{E}_{\mathsf{TM}} \subseteq \mathsf{Pair}_{\mathsf{TM}} \subseteq \Sigma^{\star}_{\mathsf{2TM}}$$

be the language including encodings of pairs of Turing machines M_1 and M_2, with the same input alphabet Σ, such that $L(M_1) = L(M_2)$. During the lecture presentation, it will be shown, using a many-one reduction, that the language E_{TM} is ***undecidable***.

As noted above, ***you do not have to use $\mathbf{A}_{\mathbf{TM}}$ as the undecidable language used in your reduction.*** The lecture presentation will, ideally, suggest that it can be *much* easier to use a many-one reduction to prove undecidability, if the undecidable language, that you start with, is chosen with care.