Theory of Computation: Assignment 9 Solutions

Arjun Chandrasekhar

1. $\text{INF}_{\text{TM}} = \{\langle M \rangle | L(M) \text{ is finite}\}$. AFSOC INF_{TM} is decidable by some machine M_I. We'll construct a machine M_A that decides A_{TM} as follows:

   1. M_A receives $\langle M, w \rangle$ as input
   2. Create a machine P that does the following:
      a. P takes a string s as input
      b. Ignore s, and run M on w (these are both hard-coded constants)
   3. Run M_I on $\langle P \rangle$ - that is, check if P recognizes an infinite language
   4. If M_I accepts $\langle P \rangle$, then M_A accepts $\langle M, w \rangle$.
   5. Otherwise if M_I rejects $\langle P \rangle$, then M_A rejects $\langle M, w \rangle$

   Let's walk through why this works. Suppose M accepts w. Then P, which always runs M on w, will accept *every* string - so $L(P)$ is infinite. However, if M doesn't accept w, then P will never accept anything - so $L(P)$ is finite. Thus, if we can determine if $L(P)$ is finite, then we can determine if M accepts w.

   However, we know that A_{TM} is undecidable! This is a contradiction; thus, we conclude that M_I does not exist, and INF_{TM} is undecidable.

   Figure 1 gives a diagram of this reduction.

2. $\text{DIS}_{\text{TM}} = \{\langle M_1, M_2 \rangle | L(M_1) \cap L(M_2) = \emptyset\}\}$. AFSOC DIS_{TM} is decidable by some machine M_D. We'll construct a machine M_E that decides E_{TM} as follows:

   1. M_E receives $\langle M \rangle$ as input
   2. Create a machine P that recognizes Σ^*
   3. Run M_D on $\langle M, P \rangle$ - that is, check if M and P are disjoint
   4. If M_D accepts $\langle M, P \rangle$, then M_E accepts $\langle M \rangle$
   5. Otherwisse, if M_D rejects $\langle M, P \rangle$, then M_E rejects $\langle M \rangle$

   Let's walk through why this works. The machine P recognizes Σ^*. This means it has something in common with every language - *except the empty set!* The only that $L(M) \cap L(P) = \emptyset$ is if $L(M) = \emptyset$. Thus, if we can determine whether $L(M)$ and $L(P)$ are disjoint, we can determine if $L(M)$ is empty.

   However, we know that E_{TM} is undecidable! This is a contradiction; thus we conclude that M_D does not exist, and DIS_{TM} is undecidable.

   Figure 2 gives a diagram of this reduction.

3. $L = \{\langle M, D \rangle | M \text{ is a TM, } D \text{ is a DFA, } L(M) = L(D)\}$. AFSOC L is decidable by some machine M_L. We'll design a machine M_A that decides ALL_{TM} as follows:

   1. M_A receives $\langle M \rangle$ as input
   2. Create a DFA D that recognizes Σ^* (a regular language)
   3. Run M_L on $\langle M, D \rangle$ - that is, check if $L(M) = L(D) = \Sigma^*$
   4. If M_L accepts $\langle M, D \rangle$, then M_A accepts $\langle M \rangle$
   5. Otherwise, if M_L rejects $\langle M, D \rangle$, then M_A rejects $\langle M \rangle$

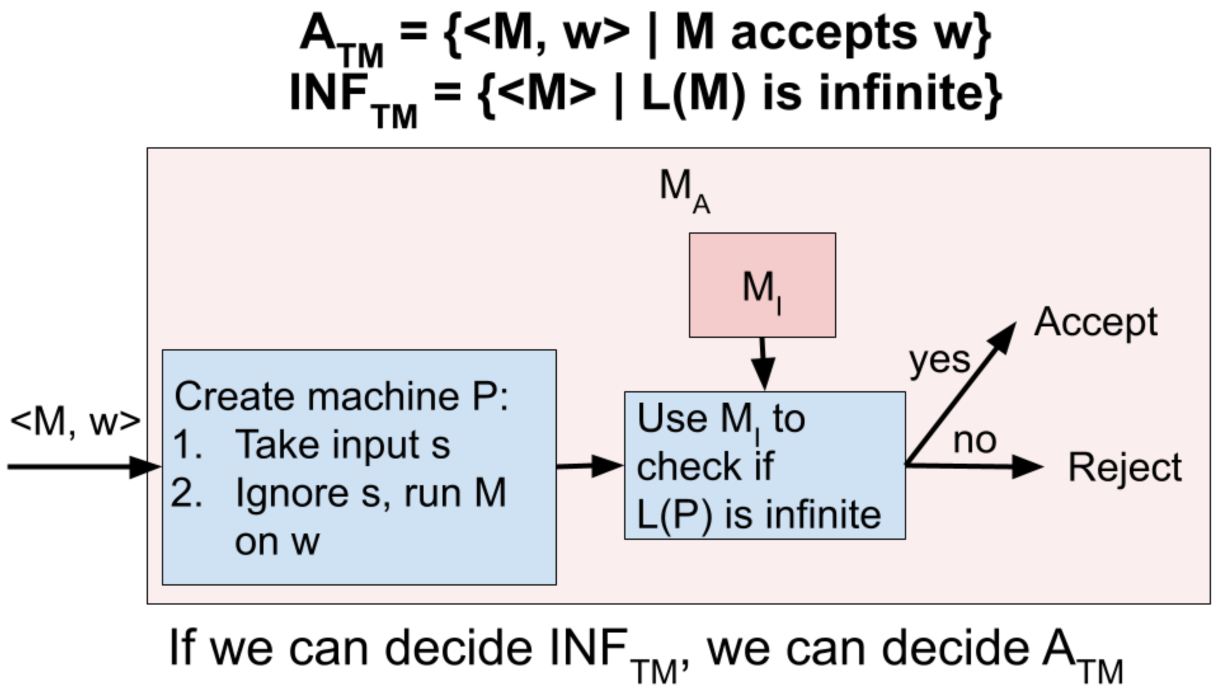

Figure 1: $\mathrm{A_{TM}}$ is reducible to $\mathrm{INF_{TM}}$

Let's walk through why this works. We know that Σ^* is a regular language, so we can create a DfA to recognize it. We do just that in creating D. From there, it follows that $L(M) = \Sigma^* \Leftrightarrow L(M) = L(D)$. Thus if we can check if $L(M) = L(D)$, the we can check if $L(M) = \Sigma^*$.

However, we know that $\mathrm{ALL_{TM}}$ is undecidable! This is a contradiction; we conclude that M_L does not exist, and M_L is undecidable.

Figure 3 gives a diagram of this reduction.

4. First, suppose $L \leq_m \mathrm{A_{TM}}$. We know that $\mathrm{A_{TM}}$ is Turing-recognizable, thus L is also Turing-recognizable since it reduces to a recognizable language.

   Next, we will show that if L is Turing-recognizable, then $L \leq_m \mathrm{A_{TM}}$. If L is Turing-recognizable, then there is a machine M_L that recognizes L. This means that

$$w \in L \Leftrightarrow M \text{ accepts w } \Leftrightarrow \langle M, w\rangle \in \mathrm{A_{TM}}$$

   Thus, the reduction $f(w) = \langle M, w\rangle$ is a mapping reduction from L to $\mathrm{A_{TM}}$

5. (a) If $L \leq_m \overline{L}$ then there is a computable function $f(w)$ such that $w \in L \Leftrightarrow f(w) \in \overline{L}$. We then see that

$$w \in \overline{L} \Leftrightarrow w \notin L \Leftrightarrow f(w) \notin \overline{L} \Leftrightarrow f(w) \in L$$

   Thus, the same function $f(w)$ is also a mapping reduction from $\overline{L}$ to L.
   Note that you may also simply appeal to the result from class that if $A \leq_m B$ then $\overline{A} \leq_m \overline{B}$. The proof of that result is almost identical to the proof given above.

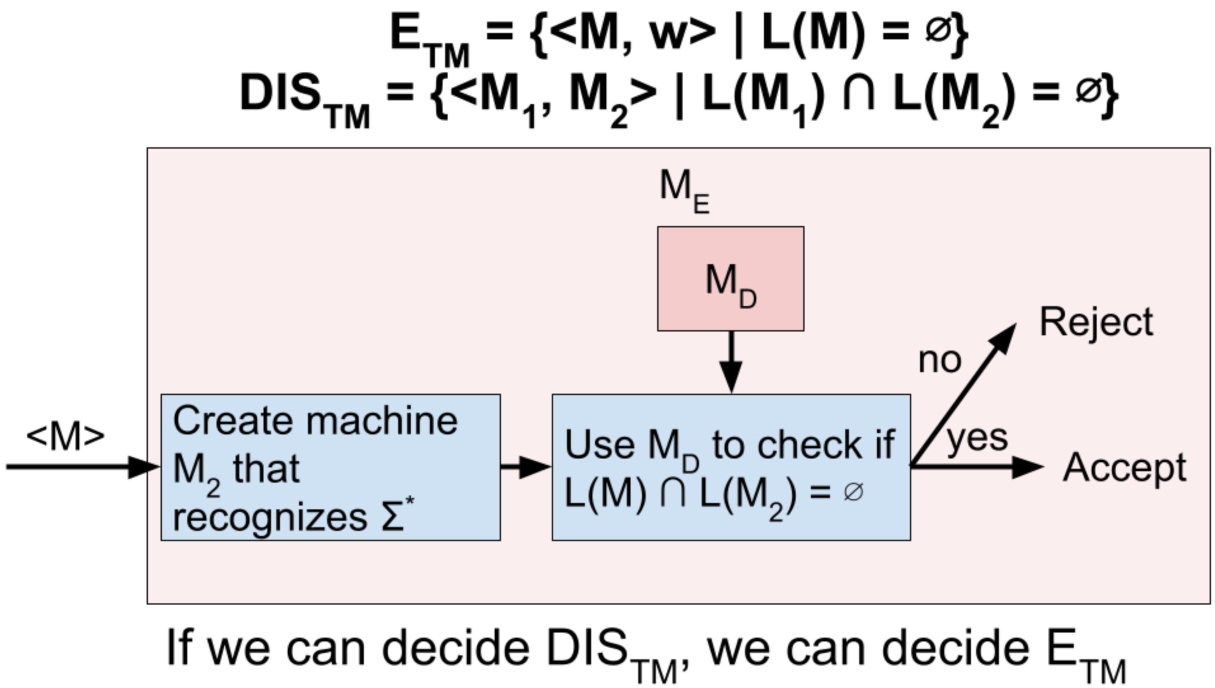

Figure 2: $\mathrm{E_{TM}}$ is reducible to $\mathrm{DIS_{TM}}$

(b) We are given that L is Turing-recognizable. We are also given that $L \leq_m \overline{L}$. As we showed in part (a), this implies that $\overline{L} \leq_m L$, meaning $\overline{L}$ is also Turing-recognizable. If L and $\overline{L}$ are both recognizable, then L is decidable.

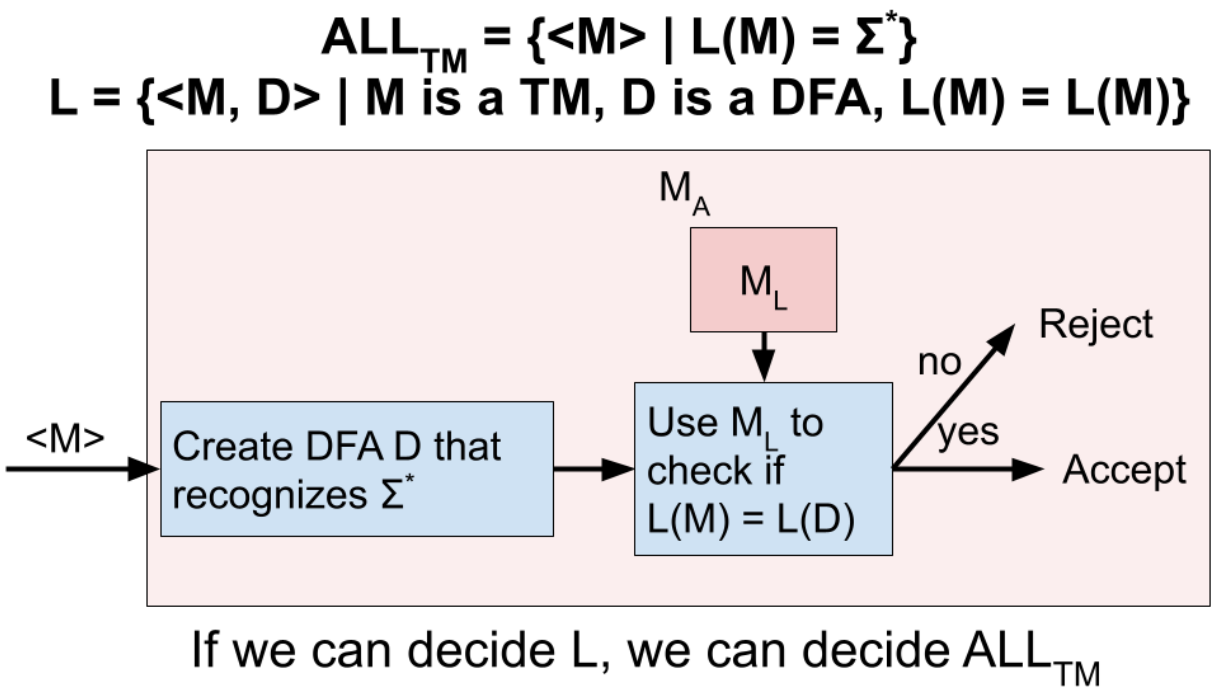

Figure 3: $\mathrm{ALL}_{\mathrm{TM}}$ is reducible to L