Assignment 10 - Turing Machines

CS 234

Daniel Lee

Part 2 Reducing with Turing Machines

Prove that the following problems are undecidable using reductions. For each problem, write a sentence explaining in words what the implication of the undecidability of the language is for computer programs.

14.10 $HASEMPTY = \{i : \lambda \in L(M_i)\}$

Proof. This property can be shown via a reduction from $HALT$ to $HASEMPTY$. Suppose for the sake of contradiction that $HASEMPTY$ is decidable. Then there is some total Turing machine S that decides $HASEMPTY$. Now use S to define the machine H as given by the following pseudocode:

```
1  H(i, x) =
2    let Q(y) = run M_i(x) then accept  in
3    let q = index(Q) in
4    S(q)
```

Now note the following facts about H:
Firstly, H is total. Values are only defined without interesting computation until line 4. Then in line 4, S is run, but S is total by assumption so this process will terminate.

Secondly, $\mathcal{L}(H) = HALT$. This fact follows from the following two cases:

· Suppose $(i, x) \in HALT$. Then the following chain of implications holds:

$$\begin{aligned}
(i,x) \in HALT &\implies M_i(x) \text{ halts} && \text{HALT def} \\
&\implies \forall y. Q(y) \text{ accepts} && \text{Q def} \\
&\implies \mathcal{L}(Q) = \Sigma^* && \mathcal{L} \text{ def} \\
&\implies \lambda \in \mathcal{L}(Q) && \lambda \in \Sigma^* \\
&\implies \lambda \in \mathcal{L}(M_q) && q \text{ def} \\
&\implies S(q) \text{ accepts} && S \text{ def} \\
&\implies (i,x) \in \mathcal{L}(H) && H \text{ def}
\end{aligned}$$

· Suppose $(i,x) \notin HALT$. Then the following chain of implications holds:

$$\begin{aligned}
(i,x) \notin HALT &\implies M_i(x) \text{ loops} && \text{HALT def}\\
&\implies \forall y.Q(y) \text{ loops} && \text{Q def}\\
&\implies \mathcal{L}(Q) = \emptyset && \mathcal{L} \text{ def}\\
&\implies \lambda \notin \mathcal{L}(Q) && \lambda \notin \emptyset\\
&\implies \lambda \notin \mathcal{L}(M_q) && q \text{ def}\\
&\implies S(q) \text{ rejects} && S \text{ def}\\
&\implies (i,x) \notin \mathcal{L}(H) && H \text{ def}
\end{aligned}$$

Thus we know that $\mathcal{L}(H) = HALT$ and H is total. In other words, H decides $HALT$. However, $HALT$ is known to be undecidable (see Theorem 14.7 from the textbook). Thus, there is a contradiction, and the assumption that $HASEMPTY$ is decidable must be false. Therefore, $HASEMPTY$ is in fact undecidable.

$\square$

14.12 $SUBSET = \{(i,j) : L(M_i) \subseteq L(M_j)\}$

Proof. This property can be shown via a reduction from $HALT$ to $SUBSET$. Suppose for the sake of contradiction that $SUBSET$ is decidable. Then there is some total Turing machine S that decides $SUBSET$.
Now use S to define the machine H as given by the following pseudocode:

```
1  H(i', x) =
2     let I(y) = accept in
3     let i = index(I) in
4     let J(z) = run M_{i'}(x) then accept in
5     let j = index(J) in
6     S(i, j)
```

Now note the following facts about H:
Firstly, H is total. Values are only defined without interesting computation until line 6. Then in line 6, S is run, but S is total by assumption so this process will terminate.
Secondly, $\mathcal{L}(H) = HALT$. This fact follows from the following two cases:
· Suppose $(i', x) \in HALT$. Then the following chain of implications holds:

$$\begin{aligned}
(i',x) \in HALT &\Longrightarrow \mathrm{M}_{i'}(x) \text{ halts} && \text{HALT def} \\
&\Longrightarrow \forall z.J(z) \text{ accepts} && \text{J def} \\
&\Longrightarrow \mathcal{L}(J) = \Sigma^* && \mathcal{L} \text{ def} \\
&\Longrightarrow \mathcal{L}(I) \subseteq \mathcal{L}(J) && \mathcal{L}(I) = \Sigma^* \\
&\Longrightarrow \mathcal{L}(M_i) \subseteq \mathcal{L}(M_j) && i,j \text{ def} \\
&\Longrightarrow S(i,j) \text{ accepts} && S \text{ def} \\
&\Longrightarrow (i',x) \in \mathcal{L}(H) && H \text{ def}
\end{aligned}$$

· Suppose $(i', x) \notin HALT$. Then the following chain of implications holds:

$$\begin{aligned}
(i',x) \notin HALT &\Longrightarrow \mathrm{M}_{i'}(x) \text{ loops} && \text{HALT def} \\
&\Longrightarrow \forall z.J(z) \text{ loops} && \text{J def} \\
&\Longrightarrow \mathcal{L}(J) = \emptyset && \mathcal{L} \text{ def} \\
&\Longrightarrow \mathcal{L}(I) \nsubseteq \mathcal{L}(J) && \mathcal{L}(I) = \Sigma^* \\
&\Longrightarrow \mathcal{L}(M_i) \nsubseteq \mathcal{L}(M_j) && i,j \text{ def} \\
&\Longrightarrow S(i,j) \text{ rejects} && S \text{ def} \\
&\Longrightarrow (i',x) \notin \mathcal{L}(H) && H \text{ def}
\end{aligned}$$

Thus we know that $\mathcal{L}(H) = HALT$ and H is total. In other words, H decides $HALT$. However, HALT is known to be undecidable (see Theorem 14.7 from the textbook). Thus, there is a contradiction, and the assumption that $SUBSET$ is decidable must be false. Therefore, $SUBSET$ is in fact undecidable.

□

14.13 $DIFFERENT = \{(i, j) : L(M_i) \neq L(M_j)\}$

Proof. This property can be shown via a reduction from $HALT$ to $DIFFERENT$. Suppose for the sake of contradiction that $DIFFERENT$ is decidable. Then there is some total Turing machine S that decides $DIFFERENT$. Now use S to define the machine H as given by the following pseudocode:

```
1  H(i', x) =
2    let I(y) = run M_i'(x) then accept in
3    let i = index(I) in
4    let J(z) = loop in
5    let j = index(J) in
6    S(i, j)
```

Now note the following facts about H:
Firstly, H is total. Values are only defined without interesting computation until line 6. Then in line 6, S is run, but S is total by assumption so this process will terminate.
Secondly, $\mathcal{L}(H) = HALT$. This fact follows from the following two cases:
· Suppose $(i', x) \in HALT$. Then the following chain of implications holds:

$$\begin{array}{rll}
(i', x) \in HALT & \Longrightarrow \mathrm{M}_{i'}(x) \text{ halts} & \text{HALT def} \\
& \Longrightarrow \forall y. I(y) \text{ accepts} & \text{I def} \\
& \Longrightarrow \mathcal{L}(I) = \Sigma^* & \mathcal{L} \text{ def} \\
& \Longrightarrow \mathcal{L}(I) \neq \mathcal{L}(J) & \mathcal{L}(J) = \emptyset \\
& \Longrightarrow \mathcal{L}(M_i) \neq \mathcal{L}(M_j) & i, j \text{ def} \\
& \Longrightarrow S(i, j) \text{ accepts} & S \text{ def} \\
& \Longrightarrow (i', x) \in \mathcal{L}(H) & H \text{ def}
\end{array}$$

· Suppose $(i', x) \notin HALT$. Then the following chain of implications holds:

$$\begin{array}{rll}
(i', x) \notin HALT & \Longrightarrow \mathrm{M}_{i'}(x) \text{ loops} & \text{HALT def} \\
& \Longrightarrow \forall y. I(y) \text{ loops} & \text{I def} \\
& \Longrightarrow \mathcal{L}(I) = \emptyset & \mathcal{L} \text{ def} \\
& \Longrightarrow \mathcal{L}(I) = \mathcal{L}(J) & \mathcal{L}(J) = \emptyset \\
& \Longrightarrow \mathcal{L}(M_i) = \mathcal{L}(M_j) & i, j \text{ def} \\
& \Longrightarrow S(i, j) \text{ rejects} & S \text{ def} \\
& \Longrightarrow (i', x) \notin \mathcal{L}(H) & H \text{ def}
\end{array}$$

Thus we know that $\mathcal{L}(H) = HALT$ and H is total. In other words, H decides $HALT$. However, HALT is known to be undecidable (see Theorem 14.7 from the textbook). Thus, there is a contradiction, and the assumption that $DIFFERENT$ is decidable must be false. Therefore, $DIFFERENT$ is in fact undecidable.

□

14.14 $OVERLAP = \{(i,j) : L(M_i) \cap L(M_j) \neq \emptyset\}$

Proof. This property can be shown via a reduction from $HALT$ to $OVERLAP$. Suppose for the sake of contradiction that $OVERLAP$ is decidable. Then there is some total Turing machine S that decides $OVERLAP$.
Now use S to define the machine H as given by the following pseudocode:

```
1  H(i', x) =
2    let I(y) = run M_i'(x) then accept in
3    let i = index(I) in
4    let J(z) = accept in
5    let j = index(J) in
6    S(i, j)
```

Now note the following facts about H:
Firstly, H is total. Values are only defined without interesting computation until line 6. Then in line 6, S is run, but S is total by assumption so this process will terminate.
Secondly, $\mathcal{L}(H) = HALT$. This fact follows from the following two cases:
· Suppose $(i', x) \in HALT$. Then the following chain of implications holds:

$$
\begin{aligned}
(i', x) \in HALT &\implies \mathrm{M}_{i'}(x) \text{ halts} && \text{HALT def} \\
&\implies \forall y. I(y) \text{ accepts} && \text{I def} \\
&\implies \mathcal{L}(I) = \Sigma^* && \mathcal{L} \text{ def} \\
&\implies \mathcal{L}(I) \cap \mathcal{L}(J) \neq \emptyset && \mathcal{L}(J) = \Sigma^* \\
&\implies \mathcal{L}(M_i) \cap \mathcal{L}(M_j) \neq \emptyset && i, j \text{ def} \\
&\implies S(i,j) \text{ accepts} && S \text{ def} \\
&\implies (i', x) \in \mathcal{L}(H) && H \text{ def}
\end{aligned}
$$

· Suppose $(i', x) \notin HALT$. Then the following chain of implications holds:

$$
\begin{aligned}
(i', x) \notin HALT &\implies \mathrm{M}_{i'}(x) \text{ loops} && \text{HALT def} \\
&\implies \forall y. I(y) \text{ loops} && \text{I def} \\
&\implies \mathcal{L}(I) = \emptyset && \mathcal{L} \text{ def} \\
&\implies \mathcal{L}(I) \cap \mathcal{L}(J) = \emptyset && \mathcal{L}(J) = \Sigma^* \\
&\implies \mathcal{L}(M_i) \cap \mathcal{L}(M_j) = \emptyset && i, j \text{ def} \\
&\implies S(i,j) \text{ rejects} && S \text{ def} \\
&\implies (i', x) \notin \mathcal{L}(H) && H \text{ def}
\end{aligned}
$$

Thus we know that $\mathcal{L}(H) = HALT$ and H is total. In other words, H decides $HALT$. However, HALT is known to be undecidable (see Theorem 14.7 from the textbook). Thus, there is a contradiction, and the assumption that $OVERLAP$ is decidable must be false. Therefore, $OVERLAP$ is in fact undecidable.

□