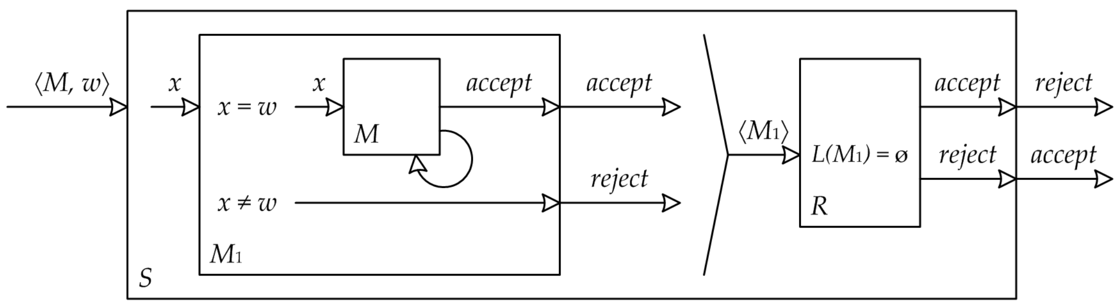
ECS120 FALL 2006
Discussion Notes
November 28, 2006

The Emptiness Problem Revisited

The emptiness problem, $E_{\mathsf{TM}} = \{ \langle M \rangle \mid M \text{ is a } \mathsf{TM} \text{ and } L(M) = \emptyset \}$, is undecidable.

We showed an informal proof of this during last discussion:



Informally, we assume R is a decider for E_{TM}. Then we build S to decide A_{TM} by building the Turing machine M_1 and feeding it to R. Finally, S outputs the opposite result of R.

In fact, what we have done here is reduce the problem of A_{TM} to the **complement** of E_{TM}. More formally, we are showing that if $A_{\mathsf{TM}} \leq_{\mathrm{m}} \overline{E_{\mathsf{TM}}}$ and A_{TM} is undecidable, then $\overline{E_{\mathsf{TM}}}$ is undecidable (corollary 5.23 on page 208).

Lets do this reduction more formally now, and give a *computable function* that shows $A_{\mathsf{TM}} \leq_{\mathrm{m}} \overline{E_{\mathsf{TM}}}$.

First, we need to figure out what the input and output of our function needs to be. Since elements of A_{TM} are in the form $\langle M, w \rangle$, this will be the *input* of our function. Since the elements of E_{TM} are in the form $\langle M \rangle$, this will be the output of our function. This gives:

$F =$ "On input $\langle M, w \rangle$:

1. ...
2. Output $\langle M' \rangle$."

Second, we need to figure out what we want to actually show. Remember, for mapping reducibility we need the relationship where $\langle M, w \rangle \in A_{\mathsf{TM}} \Leftrightarrow \langle M' \rangle \in \overline{E_{\mathsf{TM}}}$, or equivalently, $\langle M, w \rangle \in A_{\mathsf{TM}} \Leftrightarrow$

$\langle M' \rangle \notin E_{\mathsf{TM}}$ (definition 5.20 on page 207). This means, we want to construct a Turing machine M' such that when M accepts w, M' is not empty. This gives:

$F =$ "On input $\langle M, w \rangle$:

1. Construct M' as follows:
   $M' =$ "On input ??:
   - If M accepts w … (accept something).
   - If M does not accept w … (accept nothing)."
2. Output $\langle M' \rangle$."

We are getting closer. However, we still have some gaps to fill in. First, lets think about M' some more. Our aim is to build a Turing machine M' such that $L(M') \neq \emptyset$ if M accepts w and $L(M') = \emptyset$ if M rejects w. We only care about the **language** of this Turing machine, not the **simulation** of it. Also, this Turing machine is created for a specific M and w pair. However, it may accept input like any other Turing machine. Therefore we have:

$F =$ "On input $\langle M, w \rangle$:

1. Construct M' as follows:
   $M' =$ "On input x:
   - If M accepts w … (accept something).
   - If M does not accept w … (accept nothing)."
2. Output $\langle M' \rangle$."

Now we must decide what to do with the input of M'. Remember, we want $L(M')$ to be empty when M rejects w. So lets start by rejecting all input not equal to w:

$F =$ "On input $\langle M, w \rangle$:

1. Construct M' as follows:
   $M' =$ "On input x:
   (a) If $x \neq w$, reject.
   - If M accepts w … (accept something)."
2. Output $\langle M' \rangle$."

Finally, if $x = w$ we want to accept only if M accepts w. We determine this by simulating M on w. If M accepts w, we must accept x:

$F =$ "On input $\langle M, w \rangle$:

1. Construct M' as follows:
   $M' =$ "On input x:
   (a) If $x \neq w$, reject.
   (b) If $x = w$, simulate M on w.
   (c) If M accepts w, accept.
2. Output $\langle M' \rangle$."

This gives us our Turing-computable function F. However, we are not quite done. We need to show that $\langle M, w\rangle \in A_{\mathsf{TM}} \Leftrightarrow \langle M'\rangle \notin E_{\mathsf{TM}}$ holds.

Notice that if $\langle M, w\rangle \in A_{\mathsf{TM}}$, then M' will accept a single string $x = w$. Therefore, $L(M') \neq \emptyset$. This gives $\langle M, w\rangle \in A_{\mathsf{TM}} \Rightarrow \langle M'\rangle \notin E_{\mathsf{TM}}$.

If $\langle M'\rangle \notin E_{\mathsf{TM}}$, then we know $L(M') \neq \emptyset$. The only string M' will ever accept is $x = w$, and this happens only when M accepts w. Therefore, we have $\langle M'\rangle \notin E_{\mathsf{TM}} \Rightarrow \langle M, w\rangle \in A_{\mathsf{TM}}$.

Showing that $\langle M, w\rangle \in A_{\mathsf{TM}} \Leftrightarrow \langle M'\rangle \notin E_{\mathsf{TM}}$ holds may not take a lot of work, but is **necessary** in showing that $A_{\mathsf{TM}} \leq_{\mathrm{m}} \overline{E_{\mathsf{TM}}}$.

So now, we have proven that $\overline{E_{\mathsf{TM}}}$ is undecidable. What about E_{TM}? (Think about Theorem 4.22 on page 181.)

The Equivalence Problem

The equivalence problem, $EQ_{\mathsf{TM}} = \{\, \langle M_1, M_2 \rangle \mid M_1 \text{ and } M_2 \text{ are TMs and } L(M_1) = L(M_2) \,\}$, is undecidable. We will show this by showing that $E_{\mathsf{TM}} \leq_{\mathrm{m}} EQ_{\mathsf{TM}}$ and using Corollary 5.23.

First, we need to figure out what the input and output of our function needs to be. Since elements of E_{TM} are in the form $\langle M\rangle$, this will be the *input* of our function. Since the elements of EQ_{TM} are in the form $\langle M_1, M_2\rangle$, this will be the output of our function. This gives:

$F =$ "On input $\langle M\rangle$:

1. ...
2. Output $\langle M, M'\rangle$."

Second, we need to figure out what we want to actually show. We want the situation where if $L(M)$ is empty, then $L(M) = L(M')$. Since $L(M)$ is empty, we have $L(M) = L(M')$ only when $L(M')$ is also empty. Therefore, we get:

$F =$ "On input $\langle M\rangle$:

1. Construct M' as follows:
   $M' =$ "On input x: reject."
2. Output $\langle M, M'\rangle$.

Now, we must show that $\langle M\rangle \in E_{\mathsf{TM}} \Leftrightarrow \langle M, M'\rangle \in EQ_{\mathsf{TM}}$ holds.

If $L(M)$ is empty, then $L(M) = L(M')$ since $L(M')$ is empty. This gives $\langle M\rangle \in E_{\mathsf{TM}} \Rightarrow \langle M, M'\rangle \in EQ_{\mathsf{TM}}$. If $L(M) = L(M')$, then $L(M)$ is empty since $L(M')$ is empty. This gives $\langle M, M'\rangle \in EQ_{\mathsf{TM}} \Rightarrow \langle M\rangle \in E_{\mathsf{TM}}$.

Again, these statements seem apparent, but are necessary in completing our proof.

Guide To Classifying Languages

Claim: L is decidable.

There are three methods you may use to prove this is true. The easiest is to use definition 3.6 (page 142). This states that a language is decidable if some Turing machine decides it. Therefore, you may provide a decider Turing machine M such that $L(M) = L$ to prove L is decidable.

Alternatively, you may use theorem 4.22 (page 181). This states that a language is decidable iff it is Turing-recognizable and co-Turing recognizable. If you show that L is both recognizable and co-recognizable, you prove that L is decidable. How to prove a language is Turing-recognizable or co-Turing-recognizable is covered in the following sections.

Finally, you may use theorem 5.22 (page 208). This states that if $A \leq_{\mathrm{m}} B$ and B is decidable, then A is decidable. If you show that $L \leq_{\mathrm{m}} D$ where D is already proven to be decidable, then you prove that L is also decidable.

Claim: L is Turing-recognizable (or acceptable).

The easiest method is to use definition 3.5 (page 142). This states that a language is Turing-recognizable if some Turing machine recognizes it. Therefore, you may provide a Turing machine M such that $L(M) = L$ to prove L is recognizable.

You may also use theorem 3.21 (page 153). This states that a language is Turing-recognizable if and only if some enumerator enumerates it. Therefore, if you provide an enumerator M such that $L(M) = L$, then you prove L is Turing-recognizable.

We also know that every decidable language is Turing-recognizable (page 142). Therefore, if you already know L is decidable, then you know L is also Turing-recognizable.

Finally, you may use theorem 5.28 (page 209). This states that if $A \leq_{\mathrm{m}} B$ and B is Turing-recognizable, then A is Turing-recognizable. If you show that $L \leq_{\mathrm{m}} R$ where R is recognizable, you prove that L is also Turing-recognizable.

However, if you want to prove that L is **just** Turing-recognizable and not also decidable, you must prove that L is undecidable. How to do this is given in the following sections.

Claim: L is co-Turing-recognizable.

This is done by showing that the complement of L is Turing-recognizable. Use the methods from above to show this.

Claim: L is undecidable.

You may use theorem 4.22 (page 181). This states that a language is decidable iff it is Turing-recognizable and co-Turing recognizable. Therefore, if L is not Turing-recognizable or co-Turing recognizable, then L is not decidable. How to show this is provided in the following sections.

Finally, you may use corollary 5.23 (page 208). This states that if $A \leq_{\mathrm{m}} B$ and A is undecidable, then B is undecidable. Therefore, you must show that $U \leq_{\mathrm{m}} L$ for some undecidable language U.

Claim: *L* is not Turing-recognizable.

You may again use theorem 4.22 (page 181). This states that a language is decidable iff it is Turing-recognizable and co-Turing recognizable. Therefore, if you know that L is undecidable and $\overline{L}$ is recognizable, then L may not also be recognizable. This method was used on corollary 4.23 (page 182).

Finally, you may use corollary 5.29 (page 210). This states that if $A \leq_{\mathrm{m}} B$ and A is not Turing-recognizable, then B is not Turing-recognizable. Therefore, you must show that $S \leq_{\mathrm{m}} L$ for some language S which is not Turing-recognizable.

Claim: *L* is not co-Turing-recognizable.

This is done by showing that the complement of L is not Turing-recognizable. For example, you could use theorem 4.22 and show that L is undecidable and recognizable, meaning $\overline{L}$ must not also be recognizable.

Summary:

I've tried to summarize all the methods we have covered in the following table. Please let me know if anything is missing!

Claim:	**Method:**	**Thm:**	**Pg:**
L is decidable.	Give a decider M such that $L(M) = L$.	3.6	142
	Show L is recognizable and co-recognizable.	4.22	181
	Show $L \leq_{\mathrm{m}} B$ for a decidable language B.	5.22	208
L is recognizable.	Give a Turing machine M such that $L(M) = L$.	3.5	142
	Give an enumerator M such that $L(M) = L$.	3.21	153
	Show $L \leq_{\mathrm{m}} B$ for a recognizable language B.	5.28	209
L is co-recognizable.	Show that $\overline{L}$ is recognizable.	–	181
L is undecidable.	Show L is not recognizable.	4.22	181
	Show L is not co-recognizable.	4.22	181
	Show $A \leq_{\mathrm{m}} L$ for some A which is undecidable.	5.23	208
L is not recognizable.	Show L is undecidable & co-recognizable.	4.22	181
	Show $A \leq_{\mathrm{m}} L$ for some A which isn't recognizable.	5.29	210
L is not co-recognizable.	Show L is undecidable & recognizable.	4.22	181
	Show that $\overline{L}$ is not recognizable.	5.29	210