COMP 330 - Theoretical Aspects of Computer Science
Solutions for Assignment 5

Guillaume Saulnier

November 29, 2010

1. a) $H_{TM} \leq_m L_a$

The reduction will be as follow:
Given a turing machine M with input w, we create a new turing machine

N:
  on input x
  if $x \notin L(110\{0,1\}^*)$ **then**
    REJECT x
  else
    Simulate M on w
    if M halts **then**
      ACCEPT x
    end if
  end if

Note that N only accepts $x \in L(110\{0,1\}^*)$ if M halts on w. If $x \notin L(110\{0,1\}^*)$ then it is rejected. If $x \in L(110\{0,1\}^*)$, but M does not halt on w, then N does not halt on x and thus $x \notin L(N)$. We can check if $x \in L(110\{0,1\}^*)$ or not because it is a regular language and we know we can decide those using DFAs.

We claim that N is a reduction of H_{TM} to L_a:

Proof.

$$< M, w > \in H_{TM} \Rightarrow M \textit{ halts on } w \Rightarrow L(N) = L(110\{0,1\}^*) \Rightarrow N \in L_a$$

$$< M, w > \notin H_{TM} \Rightarrow M \textit{ does not halt on } w \Rightarrow L(N) = \emptyset \Rightarrow N \notin L_a$$

□

Therefore, if we had a decider for L_a, we could decide H_{TM}. This is impossible since we know H_{TM} is undecidable by diagonalization.

b) $H_{TM}^C \leq_m L_b$

The reduction will be as follow:
Given a turing machine M with input w, we create a new turing machine

N:
  on input x
  Simulate M on w

if M halts **then**
  ACCEPT x
end if

We have that:

$$L(N) = \begin{cases} \Sigma^* & \textit{if } M \textit{ halts on } w \\ \emptyset & \textit{if } M \textit{ does not halt on } w \end{cases}$$

We claim that N is a reduction of H_{TM}^C to L_b:

Proof.

$$< M, w > \in H_{TM}^C \Rightarrow M \textit{ does not halt on } w \Rightarrow L(N) = \emptyset \Rightarrow L(N) \textit{ is finite} \Rightarrow N \in L_b$$

$$< M, w > \notin H_{TM}^C \Rightarrow M \textit{ halts on } w \Rightarrow L(N) = \Sigma^* \Rightarrow L(N) \textit{ is infinite} \Rightarrow N \notin L_b$$

□

We know that H_{TM}^C is not recursively enumerable, therefore L_b is not recursively enumerable. Thus L_b is not decidable.

(The question could also be solved by creating a decider for the halting problem, which would create a contradiction with the fact that it is undecidable).

c) $H_{TM}^C \leq_m L_c$

The reduction will be as follow:
Given a turing machine M with input w, we create a new turing machine N (as follow) with a new state q (not accepting nor rejecting) which is never visited during the simulation of M on w

N:
  on input x
  if $x \neq w$ **then**
    Visit all states of N **except** q.
  else
    Simulate M on w
    if M halts **then**
      Visit q.
    end if
  end if

The machine visits all of its states on input $x \neq w$ to prevent the existence of other non accepting nor rejecting states that would not be visited while simulating M on w. If such states would exists, then even if q was visited, then the machine would still be in L_c.

We claim that N is a reduction of H_{TM}^C to L_c:

Proof.

$$< M, w > \in H_{TM}^C \Rightarrow M \textit{ does not halt on } w \Rightarrow q \textit{ is never visited on any input} \Rightarrow N \in L_c$$

$$< M, w > \notin H_{TM}^C \Rightarrow M \textit{ halts on } w \Rightarrow q \textit{ is visited on input } x{=}w \Rightarrow N \notin L_c$$

□

We know that H_{TM}^C is not recursively enumerable, therefore L_c is not recursively enumerable. Thus L_c is not decidable.

(The question could also be solved by creating a decider for the halting problem, which would create a contradiction with the fact that it is undecidable).

2. First we will show that $A_{TM}^C \leq_m L$.

The reduction will be as follow:

$f(< M, w >)$:

- Create the following turing machine N:
  if $x \neq w$ **then**
    ACCEPT x
  else
    Simulate M on w
    if M accepts w **then**
      ACCEPT x
    end if
  end if

- Let $v \in \Sigma^*$ such that $v \neq w$.
- Return $< N, v, w >$.

We claim that $f(< M, w >)$ is a reduction from A_{TM}^C to L.

Proof.

$$\begin{aligned}< M, w > \in A_{TM}^C &\Rightarrow M \textit{ does not accept } w \\ &\Rightarrow N \textit{ does not accept } w \\ &\Rightarrow N \textit{ accepts } v \textit{ and does not accept } w \\ &\Rightarrow < N, v, w > \in L\end{aligned}$$

$$\begin{aligned}< M, w > \notin A_{TM}^C &\Rightarrow M \textit{ accept } w \\ &\Rightarrow N \textit{ accepts } w \\ &\Rightarrow N \textit{ accepts } v \textit{ and accepts } w \\ &\Rightarrow < N, v, w > \notin L\end{aligned}$$

□

We know that A_{TM}^C is not recursively enumerable, therefore L is not recursively enumerable.

$A_{TM}^C \leq_m L^C$.

Note that:

$$L^C = \{< M, v, w > \mid M \textit{ is a TM and (M does not accept } v \textbf{\textit{ or }} M \textit{ accepts } w)\}$$

The reduction will be as follow:

$f(< M, v >)$:

- Create the following turing machine N:

  if $x \neq v$ **then**
    REJECT x
  else
    Simulate M on v
    if M accepts v **then**
      ACCEPT x
    end if
  end if

- Let $w \in \Sigma^*$ such that $w \neq v$.
- Return $< N, v, w >$.

We claim that $f(< M, v >)$ is a reduction from A_{TM}^C to L^C.

Proof.

$$\begin{aligned} < M, v > \in A_{TM}^C &\Rightarrow M \textit{ does not accept } v \\ &\Rightarrow N \textit{ does not accept } v \\ &\Rightarrow < N, v, w > \in L^C \end{aligned}$$

$$\begin{aligned} < M, v > \notin A_{TM}^C &\Rightarrow M \textit{ accept } v \\ &\Rightarrow N \textit{ accepts } v \\ &\Rightarrow N \textit{ accepts } v \textit{ and } N \textit{ does not accept } w \\ &\Rightarrow < N, v, w > \notin L^C \end{aligned}$$

□

We know that A_{TM}^C is not recursively enumerable, therefore L^C is not recursively enumerable.

3. a) A LBA has a total of qng^n possible tape configurations where

   - q is the number of states.
   - n is the length of the string (which determines the length of the tape).
   - g is the size of the tape alphabet.

   If we run a LBA for $qng^n + 1$ steps and it did not halt, it must be in a loop. Since $qng^n + 1 > qng^n$, one tape configuration must have been visited at least twice by the pigeonhole principle. Call this tape configuration t_i. There must exist a tape sequence $T = \{t_i, t_{i+1}, t_{i+2}, \ldots, t_{i+n}, t_i\}$. Since the tape configuration defines all of the machine's behavior, it will repeat the sequence T forever. Simulating a LBA for qng^n steps is enough as if it did not halt after that the next step will surely make it enter a loop.

   Let S be the set of all strings of length less than 100. Note that S is finite as $|S| = \sum_{i=0}^{99} |\Sigma|^i$. Take any ordering $s_0, s_1, s_2, \ldots, s_n$ of the strings in S.

   We can create a decider D_a for L_a in the following way.

   D_a:
     on input $< M >$
     for $i = 1$ **to** n **do**
       for $j = 1$ **to** qng^n **do**
         Simulate M on s_i for j steps

if M accepts s_i **then**
ACCEPT $< M >$
end if
end for
end for
REJECT $< M >$

We claim that D_a is a decider for L_a:

Proof. D_a will always halt as the set of strings is finite and we run M on each of them for a finite number of steps.

$$\begin{aligned} D_a \textit{ accepts } < M > &\Rightarrow M \textit{ accepts a string } s_j \in S \\ &\Rightarrow M \textit{ accepts a string with length less than 100} \\ &\Rightarrow < M > \in L_a \end{aligned}$$

$$\begin{aligned} D_a \textit{ rejects } < M > &\Rightarrow M \textit{ does not accept any string } \in S \\ &\Rightarrow M \textit{ does not accept a string with length less than 100} \\ &\Rightarrow < M > \notin L_a \end{aligned}$$

□

Therefore D_a is a decider for L_a which implies that L_a is decidable.

b) Note that there is a total of $q2|w|g^{2|w|}$ possible tape configurations if we have $2|w|$ cells available. Therefore, we will use the loop detection scheme defined in a) to limit the number of steps in the simulations.

We define a decider D_b for L_b as follow:

D_b:
on input $< M, w >$
for $j = 1$ **to** $q2|w|g^{2|w|}$ **do**
Simulate M on w for j steps.
if the head is beyond the first $2|w|$ cells of the tape **then**
REJECT $< M, w >$
end if
if M has halted **then**
ACCEPT $< M, w >$
end if
end for
ACCEPT $< M, w >$

We claim that D_b is a decider for L_b:

Proof. D_b will always halt as D_b simulate M for a finite number of steps and it accept afterwards (if it did not halt during the simulation).

$$\begin{aligned} D_b \textit{ accepts } < M, w > \Rightarrow & M \textit{ halted } \textbf{\textit{or}} \; M \textit{ is in a loop,} \\ & \textit{both without the head moving past the first } 2|w| \textit{ cells of the tape.} \\ & \Rightarrow < M, w > \in L_b \end{aligned}$$

If M halted without having the head move past the first $2|w|$ cells of the tape, then clearly $< M, w > \in L_b$. If M is in a loop, then $\exists$ a sequence $T = \{t_i, t_{i+1}, t_{i+2}, \ldots, t_{i+n}, t_i\}$ that M will repeat. Since $< M, w >$ was not rejected on the first pass of the sequence, then M will never move the head beyond the first $2|w|$ cells of the tape while repeating T and it will repeat T forever. Therefore $< M, w > \in L_b$.

$$D_b \text{ rejects } < M, w > \Rightarrow \textit{The head moved past the first } 2|w| \textit{ cells of the tape while simulating } M \textit{ on } w$$
$$\Rightarrow < M, w > \notin L_b$$

□

Therefore D_b is a decider for L_b which implies that L_b is decidable.

4. Since $B \neq \emptyset$, $\exists a \in B$ and since $B \neq \Sigma^*$, $\exists r \notin B$. We create the following reduction from A to B:

$$f(w) = \begin{cases} a & \textit{if } w \in A \\ r & \textit{if } w \notin A \end{cases}$$

We claim that f is a reduction:

Proof. f is total (it always halts) because A is decidable. Thus, we can verify if $w \in A$ or not in finite time.

$$w \in A \Rightarrow f(w) = a \Rightarrow a \in B \Rightarrow f(w) \in B$$

$$w \notin A \Rightarrow f(w) = r \Rightarrow r \notin B \Rightarrow f(w) \notin B$$

Therefore, $w \in A \Leftrightarrow f(w) \in B$. □

f is thus a valid reduction from A to B ($A \leq_m B$).

5. First, we show the following lemma.

$$A \leq_m B \Leftrightarrow A^C \leq_m B^C$$

Proof. A is reducible to B ($A \leq_m B$) if and only if $w \in A \Leftrightarrow f(w) \in B$, which can be rewritten as $w \in A \Rightarrow f(w) \in B$ and $f(w) \in B \Rightarrow w \in A$. Then,

$$\begin{aligned} f(w) \in B &\Rightarrow w \in A \\ w \notin A &\Rightarrow f(w) \notin B && \textit{using transposition} \\ w \in A^C &\Rightarrow f(w) \in B^C \end{aligned}$$

$$\begin{aligned} w \in A &\Rightarrow f(w) \in B \\ f(w) \notin B &\Rightarrow w \notin A && \textit{using transposition} \\ f(w) \in B^C &\Rightarrow w \in A^C \end{aligned}$$

Therefore, $(w \in A \Leftrightarrow f(w) \in B) \Rightarrow (w \in A^C \Leftrightarrow w \in B^C)$. We can do $(w \in A^C \Leftrightarrow w \in B^C) \Rightarrow (w \in A \Leftrightarrow f(w) \in B)$ similarly. □

We know that $A^C \leq_m A$ using the lemma. Since A is recursively enumerable, then so is A^C. We have that both A and A^C are recursively enumerable, therefore A is decidable.

6. Let $D_\cup$ and $D_\cap$ be the decider for $L_1 \cup L_2$ and $L_1 \cap L_2$ respectively. Let R_1 and R_2 be the recognizers for L_1 and L_2 respectively. Then we can create a decider D_1 for L_1 as follow:

D_1:

on input x
Run $D_\cup$ on x
if $D_\cup$ rejects **then**
  REJECT x
end if
Run $D_\cap$ on x
if $D_\cap$ accepts **then**
  ACCEPT x
else
  Run R_1 and R_2 on x simultaneously.
  if R_1 accepts **then**
    ACCEPT x
  end if
  if R_2 accepts **then**
    REJECT x
  end if
end if

- If $x \notin L_1 \cup L_2 \Rightarrow x \notin L_1$ and D_1 will reject when running $D_\cup$ on x.
- If $x \in L_1 \cap L_2 \Rightarrow x \in L_1$, then D_1 will not reject when running $D_\cup$ and will accept when running $D_\cap$ on x.
- If $x \in L_1$ but $x \notin L_1 \cap L_2$ then D_1 will not reject when running $D_\cup$, not accept when running $D_\cap$ and will accept when R_1 accepts (since x is not in the intersection, R_2 will not accept it).
- If $x \notin L_1$ but $x \in L_1 \cup L_2$ then D_1 will not reject when running $D_\cup$, not accept when running $D_\cap$ and will reject when R_2 accepts (since x is not in the intersection, R_1 will not accept it).

Since D_1 always halts and correctly decide every input, D_1 is a decider for L_1.

7. a) We will prove that K is undecidable by contradiction. Assume K is decidable, then there exists a decider D_K for it. We will create a decider D_L for L using D_K:

D_L:
  on input p
  Let $q = p^2 - 1$
  Run D_K on q
  if D_K accepts **then**
    REJECT p
  else
    ACCEPT p
  end if

We claim that D_L is decider for L

Proof. D_L always halt, because D_K always halt.

$$
\begin{aligned}
p \in L &\Rightarrow \exists x \textit{ such that } p(x) = 0. \\
&\Rightarrow p^2(x) = 0 && \textit{squaring does not change the roots of a polynomial} \\
&\Rightarrow p^2(x) - 1 < 0 \\
&\Rightarrow D_K \textit{ rejects } p^2 - 1 \\
&\Rightarrow D_L \textit{ accepts } p
\end{aligned}
$$

$$
\begin{aligned}
p \notin L &\Rightarrow \nexists x \textit{ such that } p(x) = 0 \\
&\Rightarrow \nexists x \textit{ such that } p^2(x) = 0 && \textit{squaring does not change the roots of a polynomial} \\
&\Rightarrow p^2 > 0 \\
&\Rightarrow p^2 \geq 1 && \cdot, +, - \textit{on integers yields an integer} \\
&\Rightarrow p^2 - 1 \geq 0 \\
&\Rightarrow D_K \textit{ accepts } p^2 - 1 \\
&\Rightarrow D_L \textit{ rejects } p
\end{aligned}
$$

□

Therefore, D_L is a decider for L. We know L is undecidable, therefore D_L cannot exists and thus the assumption that K is decidable is false. We can conclude that K is undecidable.

b) We will prove that K^+ is undecidable by contradiction. Assume K^+ is decidable, then there exists a decider D_{K^+} for it. We will create a decider D_K for K using D_{K^+}:

First we define $f(p, S)$ where p is a polynomial and S is a subset of its variables V to return a new polynomial q such that:

$$\forall \textit{ occurences of } s \in S \textit{ in p, we replace s by (-s) in q, otherwise q stays the same as p}$$

If we let $x \in \mathbb{N}^n$, we can see $f(p, S)$ as simulating a negative input for the variables in S. More formally, let $x = \{x_0, x_1, \ldots, x_n\} \in \mathbb{Z}^n$ and define $S = \{x_i | x_i < 0\}$. Then $p(x) = f(p, S)(x')$ where $x' = \{|x_0|, |x_1|, \ldots, |x_n|\}$ because $x = sign(x)|x|$.

D_K:
  on input p
  Let V be the set of variables in p
  for all $S \subseteq V$ **do**
    Run D_{K^+} on $f(p, S)$
    if D_{K^+} rejects **then**
      REJECT p
    end if
  end for
  ACCEPT p

We claim that D_K is decider for K

Proof. D_K always halts since the number of subsets of a finite set is finite and D_{K^+} always halts.

$$
\begin{aligned}
p \in K \Rightarrow & \textit{ p is positive on integers} \\
&\Rightarrow \forall f(p, S),\ D_{K^+} \textit{ will not reject.} && \textit{see explanations below} \\
&\Rightarrow D_K \textit{accepts } p
\end{aligned}
$$

If D_{K^+} rejects $f(p, S)$ then $\exists x' \in \mathbb{N}^n$ such that $f(p, S)(x') < 0$. But we can create $x \in \mathbb{Z}^n$ such that $x = x'$ and for each $x_i \in S$ we negate the entry in x. Then, $p(x) < 0$ which is a contradiction with the fact that $p \in K$.

$$
\begin{aligned}
p \notin K &\Rightarrow \exists x \in \mathbb{Z}^n \textit{ such that } p(x) < 0 \\
&\Rightarrow \exists S \subseteq V \textit{ such that } \forall x_i \in S, x_i < 0 \textit{ and } x' = \{|x_0|, |x_1|, \ldots, |x_n|\} \in \mathbb{N}^n \\
&\Rightarrow f(p, S)(x') < 0 \\
&\Rightarrow D_{K^+} \textit{ rejects this f(p, S)} \\
&\Rightarrow D_K \textit{ rejects } p
\end{aligned}
$$

□

Therefore, D_K is a decider for K, but we have seen in a) that K is undecidable. We have a contradiction, so our assumption that K^+ is decidable was false and thus K^+ is undecidable.