

CS341 Automata Theory (Summer 2006)  
Final Review Sheet - Selected Solutions

**Notes:**

1. Recall that the terms “recursive” and “decidable” are synonyms. Also the terms “semi-decidable” and “recursively enumerable” are also synonyms. You should know the definitions of these terms.
  2. You must know how to use Rice’s Theorem to prove a language is or is not decidable. However, note that Rice’s Theorem does not give you any information about whether or not an undecidable language is semi-decidable. There will likely be questions on the final exam on which you are not allowed to use Rice’s Theorem in your proof, so you still need to know how to do reduction proofs!
  3. A former 341 TA, Luay Nakhleh, wrote this incredibly comprehensive review sheet. I will post his solution set also, though his reduction proof style is not the same as the proof style we used in class, and it is NOT the style of proof I expect to see on the final exam. When you do a reduction proof, I expect you to do a proof by contradiction and to follow the format we discussed in class. The example solutions below are good examples of that style.
1.  $L_1 = \{ \langle M \rangle \mid M \text{ is a TM and there exists an input on which } M \text{ halts in less than } |\langle M \rangle| \text{ steps} \}$  is decidable.

We will define TM T that decides  $L_1$ .

TM T on input  $\langle M \rangle$ , where M is a TM:

1. Determine the length  $n$  of TM M.
2. Generate in canonical order all strings over M’s input alphabet which are of length less than or equal to  $n$ . (This set of strings is finite).
3. Run M on each of the strings generated in step 2 for  $n = |\langle M \rangle|$  steps. If M halts on any of these strings, accept. Otherwise, reject.

Note that if a TM M accepts some string in fewer than  $n$  steps, then it must accept some string of length at most  $n$  in fewer than  $n$  steps, since M can only examine (at most) the first  $n$  symbols in an input in  $n-1$  steps. So T decides  $L_1$ , and so  $L_1$  is decidable.  $\square$

2.  $L_2$  is not semi-decidable. Intuition: Given a TM M, if you want to know whether the size of M’s language is at most 3, you might run M on various inputs. For example, you might list the strings over M’s alphabet in canonical order, and for every positive number  $i = 1, 2, 3, \dots$ , run M on the first  $i$  strings for  $i$  steps. You will not be able to stop after a finite number of steps and say that M accepts at most 3 strings, because if you ran M longer, you might find that it accepts 4 strings..

3.  $L_3$  is semi-decidable, but not decidable. We proved in class that this language was not decidable, so I only prove here that it is semi-decidable.

We will define a TM  $T$  that recognizes  $L_3$ .

TM  $T$  on input  $\langle M \rangle$ , where  $M$  is a TM:

1. For  $i = 1, 2, 3, \dots$

- 1a. Run  $M$  on  $s_1, \dots, s_i$  for  $i$  steps, where  $s_1, s_2, \dots, s_i$  are the first  $i$  strings in canonical order over  $M$ 's input alphabet. If at any point  $M$  has accepted 3 strings, accept.

So  $T$  recognizes  $L_3$ , and so  $L_3$  is semi-decidable.  $\square$

4.  $L_4$  is not semi-decidable.
5.  $L_5$  is not semi-decidable.
6.  $L_6$  is not semi-decidable.
7.  $L_7$  is decidable.
8.  $L_8$  is decidable. This language is the empty set, since there are no TMs that have an uncountable language. I leave it to you to write the decider.
9.  $L_9$  is semi-decidable but not decidable.

First we show that  $L_9$  is semi-decidable by constructing a TM  $T$  that recognizes it.

TM  $T$  on input  $\langle M_1, M_2 \rangle$  where  $M_1, M_2$  are TMs:

1. Run  $M_1$  and  $M_2$  alternately on  $\varepsilon$ , step by step, and accept if one of them accepts.

$T$  recognizes  $L_9$ .

Now we show that  $L_9$  is not decidable. Assume BWOC that  $L_9$  is decidable, and that TM  $T$  decides it. Using  $T$ , we will construct a TM  $S$  that decides  $HALT_{TM}$ .

First we show how, given an input  $\langle M, w \rangle$  for  $S$ , to construct a helper TM  $M_1$  which will be an input for  $T$ . We construct  $M_1$  so that

$\langle M, w \rangle \in HALT_{TM}$  if and only if  $\langle M_1, M_1 \rangle \in L_9$ , that is,  $M$  halts on  $w$  if and only if  $\varepsilon \in L(M_1) \cup L(M_1) = L(M_1)$ .

TM  $M_1$  on input  $x$ :

1. Run  $M$  on  $w$ . If  $M$  halts on  $w$ , accept.

Note that if  $M$  halts on  $w$ ,  $M_1$  accepts every string, including  $\varepsilon$ , and if  $M$  does not halt on  $w$ ,  $M_1$  accepts no strings.

TM  $S$  on input  $\langle M, w \rangle$  where  $M$  is a TM and  $w$  is a string:

1. Use  $M$  and  $w$  to construct helper TM  $M_1$ .
2. Run  $T$  on  $\langle M_1, M_1 \rangle$ . If  $T$  accepts, then  $\varepsilon \in L(M_1) \Rightarrow M$  halts on  $w$ . So accept. If  $T$  rejects, then  $\varepsilon \notin L(M_1) \cup L(M_1) = L(M_1)$ , and so  $M$  does not halt on  $w$ . So reject.

So  $S$  decides  $HALT_{TM}$ . Contradiction, since  $HALT_{TM}$  is not decidable. So  $L_9$  is not decidable.  $\square$

10.  $L_{10}$  is semi-decidable but not decidable.
11.  $L_{11}$  is not semi-decidable. Intuition: Given 2 TMs  $M$  and  $N$ , we cannot decide after a finite number of steps that the empty string is NOT in  $L(N)$ . After any finite number of steps, even if  $N$  has not accepted the empty string, it might accept if we ran it a bit longer on  $\varepsilon$ .
12.  $L_{12}$  is semi-decidable but not decidable.

First we show it's semi-decidable by constructing a recognizer  $T$ .

TM  $T$  on input  $\langle M \rangle$  where  $M$  is a TM:

1. Run  $M$  on  $M_0$ . If  $M$  halts and accepts, accept.

So  $T$  recognizes  $L_{12}$ . Therefore  $L_{12}$  is semi-decidable.

We now show that  $L_{12}$  is not decidable. Assume BWOC that  $L_{12}$  is decidable and that TM  $T$  decides it. Using  $T$ , we will construct a decider  $S$  for  $A_{TM}$ .

First we show, given an input  $\langle M, w \rangle$  for  $S$ , how to construct an input  $N$  for  $T$ . We construct  $N$  so that  $\langle M, w \rangle \in A_{TM} \Leftrightarrow \langle N \rangle \in L_{12}$ . That is,  $M$  accepts  $w$  if and only if  $N$  accepts  $M_0$ .

TM  $N$  on input  $x$ :

1. If  $x \neq M_0$ , reject.
2. If  $x = M_0$ , run  $M$  on  $w$ . If  $M$  accepts  $w$ , accept.

Note that  $N$  accepts  $M_0$  if and only if  $M$  accepts  $w$ .

TM  $S$  on input  $\langle M, w \rangle$ , where  $M$  is a TM and  $w$  is a string:

1. Use  $M$  and  $w$  to construct helper TM  $N$ .

2. Run T on  $\langle N \rangle$ . If T accepts  $\langle N \rangle$ , then N accepts  $M_0$ , and so M accepts w. So accept. If T rejects  $\langle N \rangle$ , then N does not accept  $M_0$ , and so M does not accept w. So reject.

So S is a decider for  $A_{TM}$ . Contradiction, since  $A_{TM}$  is not decidable. So  $L_{12}$  is not decidable either.  $\square$

13.  $L_{13}$  is decidable.

We define a TM T that decides  $L_{13}$ .

TM T on input  $\langle M \rangle$ , where M is a TM:

1. Run  $M_0$  on M. Since  $M_0$  halts on all inputs, it either accepts or rejects M. If  $M_0$  accepts, accept. If  $M_0$  rejects M, reject.

So T decides  $L_{13}$ . So  $L_{13}$  is decidable.  $\square$

14.  $L_{14}$  is deciable.

We construct a TM T that decides  $L_{14}$ .

TM T on input  $\langle M, x \rangle$  where M is a TM and x is a string:

1. accept

Let  $M'$  be a TM that rejects all inputs. Then  $x \notin L(M) \cap L(M')$ , since x is not in  $L(M')$ . So T decides  $L_{14}$ .

15.  $L_{15}$  is decidable. The proof is similar to the proof for  $L_1$ .

16.  $L_{16}$  is semi-decidable but not decidable.

First we show the language is semi-decidable by constructing a TM T that recognizes it.

TM T on  $\langle M \rangle$ , where M is a TM:

1. Generate all strings in M's input alphabet of length less than 100. Alternately run M on each of these strings, one step at a time. If M halts on any of the strings accept.

T recognizes  $L_{16}$ . So  $L_{16}$  is semi-decidable.

Now we show that  $L_{16}$  is not decidable. Assume BWOC that it is decidable, and that TM  $T$  decides it. Using  $T$ , we will construct a decider  $S$  for  $HALT_{TM}$ .

First we show how, given an arbitrary input  $\langle M, w \rangle$  for  $S$ , to construct an input  $\langle N \rangle$  for  $T$ . We construct  $N$  so that  $\langle M, w \rangle \in HALT_{TM}$  if and only if  $\langle N \rangle \in L_{16}$ , that is,  $M$  halts on  $w$  if and only if there is some input of length less than 100 on which  $N$  halts.

TM  $N$  on input  $x$ :

1. Run  $M$  on  $w$ . If  $M$  halts on  $w$ , accept.

Note that  $N$  accepts all inputs, including for example  $\varepsilon$  which has length less than 100, if  $M$  halts on  $w$ . If  $M$  does not halt on  $w$ ,  $N$  does not accept any string.

TM  $S$  on  $\langle M, w \rangle$ , where  $M$  is a TM and  $w$  is a string:

1. Use  $M$  and  $w$  to construct helper TM  $N$ .
2. Run  $T$  on  $\langle N \rangle$ . If  $T$  accepts  $\langle N \rangle$ , then  $N$  halts on some input of length less than 100. So  $M$  halts on  $w$ , so accept. If  $T$  rejects  $\langle N \rangle$ , then  $N$  does not accept any inputs of length less than 100, and so  $M$  does not halt on  $w$ . So reject.

So  $S$  decides  $HALT_{TM}$ . Contradiction, since  $HALT_{TM}$  is not decidable. So  $L_{16}$  is not decidable.  $\square$

17.  $L_{17}$  is decidable (it's the empty set).