Assignment 5

Aleksandr Salo

Due October 21, 2014

1 Textbook exercises

5.4 If $A \leq_m B$ and B is a regular language, does that imply that A is a regular language? Why or why not?

Proof (by contradicting example).

1. Consider the languages $A = \{a^n b^n | n \geq 0\}$ and $B = \{b\}$ over alphabet $\Sigma = \{a, b\}$.
2. $A \leq_m B$ means that there is a computable function $f : \Sigma^* \rightarrow \Sigma^*$, where for every w : $w \in A \Leftrightarrow f(w) \in B$ *(Sipser Def. 5.20)*.
3. Let us define f as:

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

   Note that in f:

   (a) A is a CFL, that implies that A is also a Turing-decidable.

   (b) By *(a) and Sipser Def. 6.16* f must be a computable function.

   (c) $f(w) \in B$ iff $w \in A$.

   (d) Considering 1-3 we se that f correctly allows A to be mapping reducible to B.

4. Now we observe that:

   (a) Language B is finite and thus regular.

   (b) Language A is **not** regular *(Sipser example 1.73)*.

5. Thus the fact that $A \leq_m B$ and B is a regular language does **NOT** imply that A is a regular language.

5.30 (a) Using reduction, prove the undecidability of the language $INFINITE_{TM} = \{< M > | M$ is a TM and L(M) is an infinite language$\}$.

Proof (by contradiction; reduction from A_{TM}; $A_{TM} \leq INFINITE_{TM}$)

1. Note the following facts about $INFINITE_{TM}$:
   (a) It contains **some**, but not **all** TM descriptions.
   (b) It doesn't contain descriptions of the machines that accept empty languages.
2. Assume I decides $INFINITE_{TM}$ (for the sake of showing that this assumption allows us to construct a TM S that decides A_{TM}.)
3. Let N be a TM, where $N \in INFINITE_{TM}$.
4. For any TM M, string w, let:
   $C =$ "on input x:
   (a) Simulate M on w
   (b) If M rejects w, REJECT.
   (c) Run N on x and do what it does"

   Note the following about the language of the machine C:

M, w	$L(C)$
M accepts w	$L(C) = L(N)$
M rejects w	$L(C) = \emptyset$
M loops w	$L(C) = \emptyset$

5. To decide A_{TM}, construct:
   $S =$ "on input $< M, w >$, where M is a TM and w is a string
   (a) Create C with M and w.
   (b) Run I on $< C >$ and do what it does.

   Note, that if I accepts $< C >$ hence M accepts w, otherwise M rejects or loops on w.
6. Thus, our assumption about the existence of a decider for $INFINITE_{TM}$ led us to **contradiction**. Hence that assumption is wrong, and the $INFINITE_{TM}$ is **undecidable** language.

Proof (by mapping reduction $A_{TM} \leq_m INFINITE_{TM}$)

1. Language A is **mapping reducible** to language B, written $A \leq_m B$, if there is a computable function $f : \Sigma^* \to \Sigma^*$, where for every $w : w \in A \Leftrightarrow f(w) \in B$ *(Sipser Def. 5.20).*
2. The following machine F computes a reduction f that takes input of the form $< M, w >$ and returns output of the form $< M' >$.
   $F =$ "on input $< M, w >$,
   (a) Construct the following machine M':
   $M' =$ "On input x:
   1. Run M on w
   2. If M halts and accepts, ACCEPT
   3. If M halts and rejects, LOOP."

   (b) Output $< M' >$."
   Note the following about the language of the machine M':

M, w	$L(M')$
M accepts w	$L(M') = \Sigma^*$
M rejects w	$L(M') = \emptyset$
M loops w	$L(M') = \emptyset$

Thus, only if M' accept anything (essentially infinite language) M would accept w. M' accepts nothing if M rejects or loops. Formally: $< M, w > \in A_{TM}$ iff $< M' > \in INFINITE_{TM}$

3. The existence of a valid computable (trivially we can construct and simulate TM) reduction function proves that A_{TM} is mapping reducible to $INFINITE_{TM}$. That in turn proves that $INFINITE_{TM}$ is **undecidable** language. *(Sipser corollary 5.23)*.

5.30 (c) Prove the undecidability of the language $ALL_{TM} = \{< M > | M \text{ is a TM and } L(M) = \Sigma^*\}$. You should use Rices theorem for this proof. You may follow the structure of the books sample solution for 5.30 (a), but make sure you show each sub-part carefully.

Proof (using Rice's Theorem)

1. By definition, ALL_{TM} is a language of TM descriptions.
2. Note, that it satisfies the two conditions of Rices theorem:
   (a) Subset of TM description, given in ALL_{TM}, is nontrivial because:
      i. It is not empty. For example, machine M_1 is in ALL_{TM}, where M_1:
      $M_1 =$ "on input w, ACCEPT"
      ii. It is a proper subset of the set of all recursively enumerable languages. For example, T.R. machine $M_2 \notin ALL_{TM}$, where M_2:
      $M_2 =$ "on input w,
      1. If $w = aaa$, REJECT
      2. Otherwise ACCEPT"
   (b) Membership in ALL_{TM} depends only on the Turing machines language, i.e. if $L(M_1) = L(M_3)$ then $< M_1 > \in L \Leftrightarrow < M_3 > \in L$. For example, machine M_3 is in ALL_{TM}, despite its description differs from M_1, where M_3:
   $M_3 =$ "on input w, if $w = aaa$, ACCEPT, otherwise ACCEPT.
3. Consequently, Rices theorem implies that ALL_{TM} is undecidable.

5.30 co-TR Is $INFINITE_{TM}$ co-Turing-recognizable? Prove your answer.

Lemma: If $A \leq_m B$ and B is co-Turing-recognizable, then A is co-T.R.

1. We say that a language is co-Turing-recognizable if it is the complement of a Turing-recognizable language
2. Let M be a TM that recognizes $\overline{B}$ and let f be the reduction from $\overline{A}$ to $\overline{B}$.
3. Let us construct TM N that recognizes $\overline{A}$:
   N = "On input w,
   (a) Compute $f(w)$
   (b) Run M on input $f(w)$ and do what M does.
4. Knowing that $\overline{A}$ is T.R. we derive that A is co-T.R.

Corollary: If $A \leq_m B$ and A is not co-Turing-recognizable, then B is not co-T.R.

1. Assume that B is co-T.R. despite A is not co-T.R.
2. That leads to a logical contradiction to the lemma, which states that in that case A must be co-T.R.
3. That contradiction proves assumption to be wrong and corollary to be correct.

Proof 1 (by mapping reduction)

1. A_{TM} by *Sipser, Thm. 4.11*:
   (a) is undecidable;
   (b) Turing-recognizable language.
2. That implies *(by Sipser, Thm. 4.22)* that A_{TM} must be **not** co-Turing-recognizable;
3. We already showed a mapping reduction $A_{TM} \leq_m INFINITE_{TM}$.
4. Thus we have: $A_{TM} \leq_m INFINITE_{TM}$ and A_{TM} is **not** co-T.R., hence $INFINITE_{TM}$ must be **not** co-Turing-Recognizable by the corollary mentioned above.

5.30 extra Is $INFINITE_{TM}$ Turing-recognizable? Prove your answer.

Lemma: $HALT_{TM}$ is mapping reducible to a $\overline{INFINITE_{TM}}$

1. Language A is **mapping reducible** to language B, written $A \leq_m B$, if there is a computable function $f : \Sigma^* \rightarrow \Sigma^*$, where for every $w : w \in A \Leftrightarrow f(w) \in B$ *(Sipser Def. 5.20)*.
2. Let us construct reduction function f as following:
   F = "On input $< M, w >$ where M is a TM and w is a string,
   (a) Construct the TM M':
   $M_1 =$ "on input x
   1. Run M on w for $|x|$ steps
   2. if M has not halted, ACCEPT
   (b) Output $< M' >$."

   Note the following about the $L(M')$:

M, w	$L(M')$
M halts w	$finite$
M loops w	Σ^*

3. Thus M halts w iff $L(M')$ is finite, because:
   (a) If M halts on w then M' accepts fixed-length strings only, hence $L(M')$ is finite.
   (b) If M does not halt, that is loops, on w, then M' accept everything, naturally infinite language.
4. Note, that f is computable function because it's described via TM.
5. Thereby we proved that $HALT_{TM} \leq_m \overline{INFINITE_{TM}}$.

Proof (by mapping reduction $HALT_{TM} \leq_m \overline{INFINITE_{TM}}$)

1. A_{TM} is T.R.
2. $\overline{A_{TM}}$ must be **not** Turing-recognizable *from Sipser Corollary 4.23.*
3. We know that $HALT_{TM} \leq_m A_{TM}$ *(Sipser example 5.24)*.
4. The definition of mapping reducibility implies that $A \leq_m B$ means the same as $\overline{A} \leq_m \overline{B}$.
5. Thus $\overline{HALT_{TM}} \leq_m \overline{A_{TM}}$
6. If $A \leq_m B$ and A is not Turing-recognizable, then B is not Turing-recognizable *(Sipser 5.29)*.
7. Thus $\overline{HALT_{TM}}$ is **not** T.R.
8. We already proved the lemma that $HALT_{TM} \leq_m \overline{INFINITE_{TM}}$.
9. Thus $\overline{HALT_{TM}} \leq_m INFINITE_{TM}$.
10. Hence $INFINITE_{TM}$ must be **not** Turing-recognizable as well.