CSE 105 Homework 5

Due March 12

- You may (and are encouraged to) discuss the problems with other students, but what you hand in must be your own work.
- If you do not understand a problem or find a mistake, please email William at wgmatthews@cs.ucsd.edu.
- In addition to these problems, we recommend doing Sipser problems 4.18, 4.19, 4.28, 5.1, 5.9, 5.20, 5.22, 5.23, 5.33 for more practice. However, you should not hand in your solutions to these problems.
- On any problems where you are constructing TMs, make sure it is very clear why the TM solves the problem. This can either be because it is obvious from the description of the TM, or from additional explanation.
- Typed homework solutions will receive 5% extra credit.

1 Decidable Languages

Let $A, B, C$ be recognizable languages over an alphabet $\Sigma$, such that $A \cup B \cup C = \Sigma^*$ and $A \cap B = \emptyset$, $A \cap C = \emptyset$, and $B \cap C = \emptyset$. Show that $A$ is not only recognizable, but also decidable.

1.1 Solution

Let $M_A, M_B,$ and $M_C$ be TMs that recognize languages $A, B,$ and $C$ respectively. From the definition of $A, B,$ and $C$, we know that every string will be accepted by exactly one of these machines. We will use this idea to construct a TM that decides $A$:

"On input $w$:

1. Simulate $M_A$ on $w$, $M_B$ on $w$, and $M_C$ on $w$ in parallel.

2. As soon as $M_A$ accepts, accept.

3. As soon as $M_B$ or $M_C$ accept, reject."

The TM we constructed is guaranteed to always halt because one of the three machines ($M_A, M_B,$ or $M_C$) is guaranteed to accept and halt. The TM given accepts iff $w \in A$, since it accepts when $M_A$ accepts, and $w \in B$ or $w \in C$ implies $w \notin A$.

2 Another Decidable Language

Show that the following language is decidable.

$L = \{ \langle M \rangle \mid M \text{ is a DFA and there exists some string } w \text{ such that both } w \text{ and } w^R \text{ are in } L(M) \}$
2.1 Solution

We know that regular languages are closed under both intersection and reverse. A description of a DFA \( \langle M \rangle \) is in \( L \) iff \( L(M) \cap L(M)^R \neq \emptyset \) because the intersection is exactly the set of strings \( w \) such that both \( w \) and \( w^R \) are in \( L(M) \). We can use these properties to construct a TM to decide \( L \):

Let \( T \) be a TM that decides \( E_{DFA} \).

"On input \( w \):

1. If \( w \) is not of the form \( \langle M \rangle \), reject. Otherwise extract \( M \) from the input.
2. Construct a DFA \( N \) such that \( L(N) = L(M)^R \).
3. Construct a DFA \( P \) such that \( L(P) = L(M) \cap L(N) \).
4. Run \( T \) on \( \langle P \rangle \).
5. Accept if \( T \) rejects, and reject if \( T \) accepts."

We know that this TM always halts because \( T \) always halts and we can do the DFA transformations using the algorithms from Chapter 1. This TM decides \( L \) because it accepts iff \( L(M) \cap L(M)^R \neq \emptyset \).

3 Undecidable Languages

Prove that each of the following languages is undecidable. (This problem will be worth twice the points of the others.)

a) \( NEVERLOOP_{TM} = \{ \langle M \rangle \mid M \text{ is a TM and } M \text{ halts on all inputs.} \} \)

b) \( VERIFY_{TM} = \{ \langle M, w, r \rangle \mid M \text{ is a TM, } r \in \{ A, R, L \}, \text{ and } r = A \text{ if } M \text{ accepts } w; r = R \text{ if } M \text{ rejects } w; \text{ and } r = L \text{ if } M \text{ loops on } w. \} \)

c) \( KSTRINGS_{TM} = \{ \langle M, k \rangle \mid M \text{ is a TM and } M \text{ accepts exactly } k \text{ strings.} \} \)

3.1 Solution

For each of the subproblems, I will give a map reduction. For each of the subproblems, either a map reduction or a Turing reduction works to prove that the languages are undecidable. The map reductions I give may be turned into Turing reductions using the following template (where \( f \) is the mapping function and \( A \) is a TM that we are assuming decides the language that we’re reducing to): “On input \( w \)

1. Run \( f \) on \( w \) to get \( x \).
2. Run \( A \) on \( x \).
3. If \( A \) accepts then accept. Otherwise reject.”

a) We will show \( HALT_{TM} \leq_m NEVERLOOP_{TM} \) by constructing the following mapping function \( f \). Since \( HALT_{TM} \) is undecidable then \( NEVERLOOP_{TM} \) is undecidable.

Let \( z \) by a string that does not describe a TM. Define \( f \) as “On input \( w \):

1. If \( w \) is not of the form \( \langle M, x \rangle \), where \( M \) is the description of a TM, then output \( z \). Otherwise, parse \( w \) to get \( M \) and \( x \).
2. Construct the following TM $N$: “On input $s$:
   a. Run $M$ on $x$.”
3. Output $\langle N \rangle$.”

To prove that this is a map reduction from $HALT_{TM}$ to $NEVERLOOP_{TM}$, we need to show that $w \in HALT_{TM}$ iff $f(w) \in NEVERLOOP_{TM}$ and that $f$ always halts. The second part, that $f$ always halts is straightforward since it only ever does simple transformations of the descriptions of TMs. If $w \in HALT_{TM}$, then $w = \langle M, x \rangle$ such that $M$ halts on $x$. Thus, $f(w)$ will be $\langle N \rangle$ where $N$ always halts, since $N$ simply runs $M$ on $x$. If $w \notin HALT_{TM}$ then either $w$ is not of the form $\langle M, x \rangle$, in which case we output $z$ which will be rejected by $NEVERLOOP_{TM}$ since it is not of the right form. If $w \notin HALT_{TM}$ and $w$ is of the form $\langle M, x \rangle$, then it means that $M$ must loop on $w$. In this case, when we output $\langle N \rangle$, it will always loop, and therefore not be in $NEVERLOOP_{TM}$.

b) We will show $A_{TM} \leq_m VERIFY_{TM}$ by constructing the following mapping function $f$. Since $A_{TM}$ is undecidable then $VERIFY_{TM}$ is undecidable.

Let $z$ be a string that does not describe a TM. Define $f$ as “On input $w$:

1. If $w$ is not of the form $\langle M, x \rangle$, where $M$ is the description of a TM, then output $\langle z, w, A \rangle$. Otherwise, parse $w$ to get $M$ and $x$.
2. Output $\langle M, w, A \rangle$.”

If $w$ is not of the form $\langle M, w \rangle$, we output a string where the first part is not a valid TM, and therefore not in $VERIFY_{TM}$. If $w = \langle M, x \rangle$ and $w \in A_{TM}$ then $M$ must accept when run on $x$. We output $\langle M, x, A \rangle$ which is in $VERIFY_{TM}$ since $M$ accepts $x$. Finally, if $w(M, x)$ and $w \notin A_{TM}$ then $M$ must not accept $x$. We output $\langle M, x, A \rangle \notin VERIFY_{TM}$ since $M$ does not accept $x$.

c) We will show $E_{TM} \leq_m KSTRINGS_{TM}$ by constructing the following mapping function $f$. Since $E_{TM}$ is undecidable then $KSTRINGS_{TM}$ is undecidable.

Let $z$ be a string that does not describe a TM. Define $f$ as “On input $w$:

1. If $w$ is not of the form $\langle M \rangle$, where $M$ is the description of a TM, then output $\langle z, 0 \rangle$. Otherwise, parse $w$ to get $M$.
2. Output $\langle M, 0 \rangle$.”

If $w$ is not of the form $\langle M \rangle$ then we output a string where the first part is not a valid TM and therefore not in $KSTRINGS_{TM}$. If $w(M) \in E_{TM}$ then $|L(M)| = 0$ so the output $\langle M, 0 \rangle \notin KSTRINGS_{TM}$. If $w = \langle M \rangle \notin E_{TM}$ then $|L(M)| \neq 0$ so the output $\langle M, 0 \rangle \notin KSTRINGS_{TM}$.

4 Miscellanea

Prove or disprove each of the following statements.

a) Every recognizable language can be recognized by a TM that either accepts or loops, but never rejects.

b) If a language is decidable, then every proper subset of that language is decidable.

c) If $A \leq_m B$ and $B$ is a regular language then $A$ is a regular language.
4.1 Solution

a) This statement is true. Proof: For any recognizable language $L$, we have a TM $M$ that recognizes it. We can construct a new TM $M'$ that also recognizes $L$ but never rejects:

“On input $w$:

1. Run $M$ on $w$.
2. If $M$ accepts then accept.
3. If $M$ rejects then loop.”

This TM recognizes the same language as $M$ because it accepts iff $M$ accepts. It never rejects because instead of rejecting it loops.

b) This statement is false. Counterexample: We know that $\text{ATM}$ is undecidable. Let $\Sigma$ be the alphabet that $\text{ATM}$ is defined over. $\text{ATM} \subseteq \Sigma^*$ and $\Sigma^*$ is regular and therefore decidable.

c) This statement is false. Counterexample: Let $A = \{0^n1^n \mid n \geq 0\}$ and $B = 0^*1^*$. $B$ is regular because it is defined by a regular expression, and $A$ is one of our canonical examples of a non-regular language. The following is a mapping function from $A$ to $B$:

“On input $w$:

1. If $w$ is in $A$ then output $01$.
2. Otherwise output $10$.”

This is a computable function because testing whether a string is in a context-free language is decidable. If $w \in A$ then $f(w) = 01 \in B$ and if $w \notin A$ then $f(w) = 10 \notin B$. 