Alex Hubers
CSC:341 Spring 2024

Exam #2 (Decidability)

Due: February 16, 2024

Academic Honesty & Collaboration. I expect you to work alone, and for your work to represent wholly your own efforts. You are, of course, allowed to ask your classmates (and me) general questions about the material. Please include the following information at the top of your submission, along with your name.

- Written sources used: (Include textbook(s), complete citations for web or other written sources. Write none if no sources used)
- Help obtained: (Include names of anyone other than the instructor.)

Extensions. I am happy to offer extensions, but I ask that you ask *in advance* and at least prior to *Friday, Apr 12.* Further, I expect you to be able to show either (i) proof of effort and some progress on the exam, or (ii) some minor proof of conflict in schedule. (Exceptions will be made, of course, in cases of sickness, injury, or emergency.) Finally, please talk to me *as soon as possible* if you know you have a conflict in schedule (e.g., another take-home exam in a separate course) and you need some help meeting this deadline.

The latest extension I can/will give is to Thursday, Apr 18.

Definitions

Definition 1 (triviality)**.** *A language A is trivial if $A = \emptyset$ or $A = \Sigma^*$. A is called nontrivial if it is not trivial.*

#1 (50 pts

Prove that the following language is decidable, where $|w|$ denotes the length of input string w.

$$A = \{w \mid |w| \text{ is a power of } 2\}$$

(N.b. 0 is even, so $\epsilon \in A$.)

Show that A is decidable by exhibiting a 2-tape $M = (Q, \{0,1\}, \{0,1,\square,\triangleright\}, \delta, q_0, q_A, q_R)$ that decides A. Let both the input tape and work tape of M have the start-character $\triangleright$ written on their first cell.

Your description of M will be partially high-level and partially formal. The strategy for deciding A is this: Recall that, if $|w|$ is a power of 2 (greater than 0), then $|w|$, as a binary number, will have exactly one 1 at the leftmost bit and then only zeros thereafter. So our strategy is to encode $|w|$ in binary and check if that pattern is matched. For full credit, I want you to describe this strategy at a high level, but describe the process of binary encoding at a formal level. In particular, please describe M according to the following steps.

1. (25 pts). Formally describe the states $\{q_{inc}, q_{carry}\} \subseteq Q$, where q_{inc} will increment a bit in the output (without a carry) and q_{carry} increments a bit with a carry.[1] Please describe how δ behaves on each of these states and inputs $\{1, 0, \triangleright\}$. So, fill in the following table. You may assume state q_{inc} is entered with the work tape head at its rightmost bit.

$$\begin{aligned}
&\forall x \in \Gamma, \\
\delta(q_{inc}, x, 0) &= (?, x, ?, ?) \\
\delta(q_{inc}, x, 1) &= (?, x, ?, ?) \\
\delta(q_{carry}, x, 0) &= (?, x, ?, ?) \\
\delta(q_{carry}, x, 1) &= (?, x, ?, ?) \\
\delta(q_{carry}, x, \triangleright) &= (?, x, ?, ?) \\
\delta(q_{inc}, x, \triangleright) &= (?, x, ?, ?)
\end{aligned}$$

   Some further clarifications.

   - Since you start from the rightmost bit, we will assume that $\square$ is never encountered on the work-tape in the q_{inc} and q_{carry} states. (So these equations are removed.)
   - Note that $x \in \Gamma$ is the character written on the input tape, which should have no effect on what q_{inc} and q_{carry} do.
   - You may also need some process of "shifting" the work tape right by 1. Call this state q_{shift}. You do not need to give a formal description of q_{shift}, but please describe its behavior.
   - To make things easy, assume that the TM transitions to state q_{end} when it is done incrementing the work tape. Again, you do not need to give any description of this state.

2. (25 pts). Describe the high-order steps taken by M. That is, fill in the following steps below.

[1]See wikipedia for how to count in binary. We use state q_{carry} to represent "carrying" a 1, which happens when we increment a 1-bit.

(a) M begins in state q_0 with w on its input tape.
(b) ...
(c) M accepts if ...; M rejects otherwise.

When invoking a particular state as a subroutine, make sure to tell me its name. (E.g., say that "M transitions to state q_{inc}" when M invokes one incrementation of the work tape.) You do not need to formally specify other subroutines that M might use—e.g, for checking if the final output has the right form. But please describe, in English, what this subroutine does.

#2 (50 pts)

Prove the following claim.

Claim 1. *Let A and B be two decidable, nontrivial languages. Then $A \leq_m B$.*

In your answer, let M_A decide A and M_B decide B. Then:

1. (25 pts). Describe the TM M_f that computes the mapping reduction $f : \Sigma^* \to \Sigma^*$ for which $A \leq_m B$.
2. (25 pts). Next, prove that, for all $w \in \Sigma^*$,
   (a) if $w \in A$ then $f(w) \in B$ and
   (b) if $w \notin A$ then $f(w) \notin B$.

#3 (50 pts)

Let

$$\mathsf{AMBIG}_{\mathsf{CFG}} = \{\langle G \rangle \mid G \text{ is an ambiguous CFG}\}.$$

A grammar G is called ambiguous if there is some string w generated by G that has two or more different left-most derivations (Sipser pg 108). In this problem we will show that $\mathsf{AMBIG}_{\mathsf{CFG}}$ is undecidable.

We will construct the following reduction from PCP. Given an instance of PCP,

$$P = \left\{ \left[\frac{t_1}{b_1}\right], \left[\frac{t_2}{b_2}\right], \ldots, \left[\frac{t_k}{b_k}\right] \right\},$$

construct a CFG G, with the following rules:

$$\begin{aligned} &S \to T \mid B \\ &T \to t_1 T a_1 \mid \cdots \mid t_k T a_k \mid t_1 a_1 \mid \ldots \mid t_k a_k \\ &B \to b_1 B a_1 \mid \cdots \mid b_k B a_k \mid b_1 a_1 \mid \ldots \mid b_k a_k, \end{aligned}$$

where $a_1, \ldots, a_k$ are new terminal symbols.

Formally, you can think of this reduction as the computable function f that takes an instance of P and turns it into an encoding of the grammar G described above.

$$f(\langle P \rangle) = \langle G \rangle$$

Recall that this is a mapping reduction if, whenever $f(\langle P \rangle) = \langle G \rangle$,

$$\langle P \rangle \in \text{PCP} \Leftrightarrow \langle G \rangle \in \mathsf{AMBIG}_{\mathsf{CFG}}$$

Your task is to prove that f is a mapping reduction. Please ignore the details of the encoding. I specifically just want you to prove both directions of the above biconditional. That is:

1. (25 pts). Prove that, if P has a solution $t_{i_1}t_{i_2}...t_{i_n} = b_{i_1}b_{i_2}...b_{i_n}$, then we can create a string that has two leftmost derivations in G.
2. (25 pts). Prove that, if G is ambiguous, then P has a solution. *Hint: Note that every string $w \in L(G)$ is of the form $xa_{i_m}a_{i_{m-1}}...a_{i_1}$ where x is a string composed entirely of symbols that appear on the dominos of P.*

Note. Recall that, for mapping reduction f from A to B, the condition that

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

is equivalent (and contrapositive) to

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

Respectively, the first one is "if G is ambiguous then P has a solution", and the second is "if P has no solution then G is unambiguous." I asked you to do the first one, above, but you may choose to prove either. **Please tell me, at the start of your solution, if you choose to prove the second.**

#4 (40 pts)

Let A and B be decidable languages, where M_A decides A and M_B decides B.

1. (25 pts). Describe (at a high-level) a TM M that decides the language $AB = \{ab \mid a \in A, b \in B\}$.
2. (10 pts). Briefly argue that your construction is correct: if $w \in AB$, what does M do? if $w \notin AB$, what does M do? Does M ever run forever?
3. (5 pts). You have just proven an important property of decidability. What is that property? (Fill in the blanks in this sentence: *Decidability is* ______ *under* ______.)

#5 (10 pts)

Suppose you have just been employed as a compiler designer on the Python Compiler Design team. To give you something to do, your employers ask that you add a step to the compiler's syntax parser that checks, for each python program p, if p will halt on all inputs. This feature should be able to run on all valid python programs—including the python compiler itself!

Remember that each python program has a string representation. Here's an example, in Python:

```
def good():
  print "accept"

good()
```

By now, your brain sees "code". But this code is just a string! Here's another example:

```
def bad():
  print bad()

bad()
```

Suppose that, for each string of python code p, we have an encoding $\langle p \rangle \in \{0,1\}^*$. Further, suppose we have a Python Interpreter TM M_p that, given an input $\langle p \rangle$, will interpret the python code p. We say that p halts if M_p runs the input $\langle p \rangle$ and halts. Then:

1. (5 pts). Describe the task you have been given as a language P over encoded python programs $\langle p \rangle$.
2. (5 pts). Describe what a reduction from $\mathsf{HALT}_{\mathrm{TM}}$ to P would look like. (Do not formalize or prove this reduction.) The reduction would transform inputs of what form (from $\mathsf{HALT}_{\mathrm{TM}}$) to inputs of what other form (for P)?

#6 Extra Credit. (10 pts)

Theorem 1. *Let P be any set of languages that satisfies the following conditions:*

1. *P **is nontrivial.** Let N denote the set of all TM encodings. Then $P \neq \emptyset$ and $P \neq N$. So P has some but not all TM descriptions.*
2. *P **is a property of the TM's language.** Whenever $L(M_1) = L(M_2)$, we have $\langle M_1 \rangle \in P$ iff $\langle M_2 \rangle \in P$.*

Rice's Theorem states that P is undecidable whenever these two conditions are met.

This is a very powerful result that more or less tells us: all "interesting" properties of Turing Machines are undecidable. Rice's theorem is also very easy to use. Here is an example:

Claim 2. *Let the language E consist of all Turing machines that accept the empty string:*

$$E = \{\langle M \rangle \mid M \textit{ accepts given an empty initial tape } \}$$

Then E is undecidable.

Proof. Let E be as defined above.

- Given an empty initial tape, the TM M_{ACCEPT} that accepts and halts on all inputs clearly will accept the empty string. So $M_{Accept} \in E$. Further, the TM M_{LOOP} that diverges on all inputs is not in E. So $M_{\mathrm{LOOP}} \notin E$, and E is **nontrivial.**
- Suppose M_1 and M_2 are TMs such that $L(M_1) = L(M_2)$. We must show that $\langle M_1 \rangle \in E \Leftrightarrow \langle M_2 \rangle \in E$. In one direction, suppose $\langle M_1 \rangle \in E$. Then clearly $\langle M_2 \rangle \in E$, as both M_1 and M_2 accept the empty string (and so M_2 should accept an empty initial tape). Further, if we suppose that $\langle M_2 \rangle \in E$ then clearly $\langle M_1 \rangle \in E$, for the same reason. Thus E is a **Property of the TM's language.**

By Rice's Theorem, E is undecidable. □

For extra credit, please answer the questions below about Rice's theorem.

1. (5 pts). Use Rice's theorem to prove that the following languages are undecidable.

$$\{\langle M \rangle \mid M \text{ is a TM and } 1011 \in L(M)\}$$

$$\{\langle M \rangle \mid M \text{ is a TM and } M \text{ accepts all palindromes}\}$$

2. (5 pts). Is the following language undecidable according to Rice's Theorem? Explain why.

$$L_2 = \{ \langle M \rangle \mid M \text{ is a TM and on some input, 010 exists on the tape at some point} \}.$$