CS244 Theory of Computation
Homework 2 Solution

Let TM be **deterministic** Turing machine if not otherwise specified with non-deterministic.

Problem 1

Say that a non-terminal A in CFG G is ***usable*** if it appears in some derivation of some string $w \in L(G)$. Given a CFG G and a non-terminal A, consider the problem of testing whether A is usable. Formulate this problem as a language and show that it is decidable.

Solution

The problem can be formulated as

$$U = \{\langle G, A\rangle \mid A \text{ is a usable non-terminal in } \mathsf{CFG}\ G\}$$

Proof. Let $G = (\mathcal{N}, \Sigma, \mathcal{P}, S)$. To be usable, A has to be

a) reachable from S, and

b) able to derive a string of terminals.

We can construct a TM M deciding U:

M on $\langle G, A\rangle$:

1. Construct a CFG $G_1 = (\mathcal{N}_1, \Sigma_1, \mathcal{P}_1, S)$ which is the same as G except that A is considered as a terminal, i.e., $\mathcal{N}_1 = \mathcal{N} - \{A\}$, $\Sigma_1 = \Sigma \cup \{A\}$, and all rules with A on the left hand side are removed in $\mathcal{P}_1$.
2. Construct a CFG G_2 with $L(G_2) = L(G_1) \cap \Sigma_1^* A \Sigma_1^*$ using the procedure for showing the closure of CFLs under intersection with regular languages.
3. Run the E_{CFG} decider on $\langle G_2\rangle$. *Reject* if $\langle G_2\rangle$ is accepted.
4. Construct CFG G_3 which is the same as G except that A is the start symbol, i.e., $G_3 = (\mathcal{N}, \Sigma, \mathcal{P}, A)$.
5. Run the E_{CFG} decider on $\langle G_3\rangle$. *Reject* if $\langle G_3\rangle$ is accepted. *Accept* otherwise.

Thus U is decidable. $\square$

Problem 2

A ***queue automaton*** is like a push-down automaton except that the stack is replaced by a queue. A ***queue*** is a tape allowing symbols to be written only on the left-hand end and read only at the right-hand end. Each write operation (we'll call it a *push*) adds a symbol to the left-hand end of the queue and each read operation (we'll call it a *pull*) reads and removes a symbol at the right-hand end. As with a PDA, the input is placed on a separate read-only input tape, and the head on the input tape can move only from left to right. The input tape contains a cell with a blank symbol following the input, so that the end of the input can be detected. A queue automaton accepts its input by entering a special accept state at any time. Show that a language can be recognized by a deterministic queue automaton iff the language is Turing-recognizable.

Problem 2 continued on next page...

Solution

($\rightarrow$) Any deterministic queue automaton Q can be simulated by a two-tape TM M. The first tape of M holds the input, and the second tape simulates the queue.

1. To simulate reading Q's next input symbol, M reads the symbol under the first head and moves it to the right.
2. To simulate a push of a, M writes a on the leftmost blank square of the second tape.
3. To simulate a pull, M reads the leftmost symbol on the second tape and shifts that tape one symbol leftward.
4. If Q enters the accept state, M *accepts*.

Thus if a language can be recognized by a deterministic queue automaton, it is Turing-recognizable.

($\leftarrow$) Any single-tape, deterministic TM M can be simulated by a queue automaton Q. Automaton Q simulates M by maintaining a copy of M's tape in the queue, and record M's current state in its control. Q also holds each tape symbol for one step in the control before push.

1. Q starts by reading the symbols from the input tape and pushing them into the queue, until the first blank symbol is encountered.
2. For each symbol c of M's tape alphabet, the queue alphabet of Q has two symbols, c and $\dot{c}$. We use $\dot{c}$ to denote c with M's head over it. In addition, the queue alphabet has an end-of-tape marker symbol \$.
3. Q can effectively scan the tape from right to left by pulling symbols from the queue and pushing them back onto the queue, until the \$ is seen.
4. When the dotted symbol is encountered, Q can determine M's next move. Instead of pushing the old symbol back onto the queue, push the new symbol given by the transition function.
5. If M's tape head moves rightward, the dot in the queue should move leftward. So pull another symbol from the queue, and push the dotted version of it onto the queue.
6. If M's tape head moves leftward, the dot should move rightward (in a cyclic manner). Retrieve from the control the symbol that should be pushed in the last step, push the dotted version of it, and then proceed with the current symbol.

If a language is Turing-recognizable, it can be recognized by a single-tape, deterministic TM. Thus it can also be recognized by a deterministic queue automaton.

Problem 3

Show that a language is decidable iff some enumerator enumerates the language in the string order. (***String order*** is the standard length-increasing, lexicographic order. See page 14 of the textbook.)

Solution

($\rightarrow$) If A is decidable, then there is a TM R deciding A. We can construct the enumerator E:

1. Go through each possible w in the string order and run R on w.
2. If R accepts w, print w.

E enumerates the language in the string order.

($\leftarrow$) If enumerator E enumerates A in string order, we break the problem into two cases.

(a) If A is infinite, we can construct a TM deciding A:

M on w:

1. Run E to enumerate all strings in string order until the output appears after w in string order.
2. If w has appeared in the enumeration, *accept*. Otherwise, *reject*.

(b) If A is finite, the enumerator E may loop without producing any additional output, so the above M is not a decider. However, all finite languages are decidable.

Problem 4

Let C be a language. Prove that C is Turing-recognizable iff a decidable language D exists such that $C = \{x \mid \exists y \in \{0,1\}^* \ (\langle x, y \rangle \in D)\}$. (Hint: You must prove both directions of the "iff". The ($\leftarrow$) direction is easier. For the ($\rightarrow$) direction, think of y as providing additional information that allows you to confirm when $x \in C$, but without the possibility of looping.)

Solution

($\leftarrow$) Assume that a decidable language D exists such that $C = \{x \mid \exists y \in \{0,1\}^* \ (\langle x, y \rangle \in D)\}$. Then there is a TM M deciding D. We can construct a TM N recognizing C:

N on x:

1. Go through each possible $y \in \{0,1\}^*$ in string order (ε, 0, 1, 00, 01, 10, 11, 000, ...) and run M on $\langle x, y \rangle$.
2. If M accepts, *accept*.

Thus C is Turing-recognizable.

($\rightarrow$) Assume that C is Turing-recognizable. Then there is a TM M recognizing C. Let $D = \{\langle x, y \rangle \mid M$ accepts x within y steps (y as a binary number)$\}$. We can construct a TM N deciding D:

N on $\langle x, y \rangle$:

1. Run M on x for y steps.
2. If M accepts x, *accept*. Otherwise, *reject*.

Thus D is decidable.

$x \in C \Rightarrow M$ accepts x within some number of steps $\Rightarrow \langle x, y \rangle \in D$ for any sufficiently long y

$x \notin C \Rightarrow \langle x, y \rangle \notin D$ for any y

Thus $C = \{x \mid \exists y \in \{0,1\}^* \ (\langle x, y \rangle \in D)\}$.

Problem 5

Consider the problem of determining whether a single-tape Turing machine ever writes a blank symbol over a nonblank symbol during the course of its computation on any input string. Formulate this problem as a language and show that it is undecidable.

Solution

The problem can be formulated as $E = \{\langle M\rangle \mid M$ is a single-tape TM and there exists a string w such that M ever writes a blank symbol over a nonblank symbol during computation on $w\}$.

Proof. Assume for contradiction that E is decidable, then there is a TM R deciding E. We can construct a TM S deciding A_{TM}:

S on $\langle M, w\rangle$:

1. Construct the following TM $T_{M,w}$:

   $T_{M,w}$ on x:
   1. Run M on w. Use a new symbol $\sqcup'$ instead of blank when writing, and treat it like a blank when reading it.
   2. If M accepts, write a true blank symbol $\sqcup$ over a nonblank symbol."
2. Run R on $\langle T_{M,w}\rangle$.
3. If R accepts, M accepts w, therefore *accept*. Otherwise, *reject*."

However, we know that A_{TM} is undecidable. Thus E is undecidable. ☐

Alternatively, we can construct a TM S deciding E_{TM}:

S on $\langle M\rangle$:

1. Construct the following TM T_M:

   T_M on x:
   1. Run M on x. Use a new symbol $\sqcup'$ instead of blank when writing, and treat it like a blank when reading it.
   2. If M accepts, write a true blank symbol $\sqcup$ over a nonblank symbol."
2. Run R on $\langle T_M\rangle$.
3. If R accepts, $L(M)$ is not empty, therefore *reject*. Otherwise, *accept*."

Problem 6

Let A be a language.

(a) Show that A is Turing-recognizable iff $A \leq_{\mathrm{m}} A_{\mathsf{TM}}$.

(b) Show that A is decidable iff $A \leq_{\mathrm{m}} \mathtt{0}^*\mathtt{1}^*$.

Solution

(a) *Proof.* ($\leftarrow$) A_{TM} is Turing-recognizable. Thus if $A \leq_{\mathrm{m}} A_{\mathsf{TM}}$ then A is Turing-recognizable.

($\rightarrow$) If A is Turing-recognizable, then there is a TM M recognizing it. We can reduce A to A_{TM} with the function computed by the following TM F:

F on input w:

Output $\langle M, w\rangle$.

$w \in A \iff M$ accepts $w \iff \langle M, w\rangle \in A_{\mathsf{TM}}$. Thus $A \leq_{\mathrm{m}} A_{\mathsf{TM}}$.

□

(b) *Proof.* ($\leftarrow$) $\mathtt{0}^*\mathtt{1}^*$ is a regular language, thus is decidable. If $A \leq_{\mathrm{m}} \mathtt{0}^*\mathtt{1}^*$ then A is decidable.

($\rightarrow$) If A is decidable, then there is a TM M deciding it. We can reduce A to $\mathtt{0}^*\mathtt{1}^*$ with the function f computed by the following TM F:

F on input w:

1. Run M on w to test if $w \in A$.
2. If $w \in A$ then output $\mathtt{01}$. If $w \notin A$ then output $\mathtt{10}$.

$w \in A \iff f(w) \in \mathtt{0}^*\mathtt{1}^*$. Thus $A \leq_{\mathrm{m}} \mathtt{0}^*\mathtt{1}^*$.

□

Problem 7

(a) Let $J = \{w \mid \text{either } w = 0x \text{ for some } x \in A_{\mathsf{TM}}, \text{ or } w = 1y \text{ for some } y \in \overline{A_{\mathsf{TM}}}\}$. Use mapping reductions to show that neither J nor $\overline{J}$ is Turing-recognizable.

(b) Let $\mathit{FINITE}_{\mathsf{TM}} = \{\langle T\rangle \mid T \text{ is a } \mathsf{TM} \text{ and } L(T) \text{ is a finite language}\}$. Show that $A_{\mathsf{TM}} \leq_{\mathrm{m}} \overline{\mathit{FINITE}_{\mathsf{TM}}}$ to prove that $\mathit{FINITE}_{\mathsf{TM}}$ is not T-recognizable.

(c) (harder) Show that $A_{\mathsf{TM}} \leq_m \mathit{FINITE}_{\mathsf{TM}}$ to prove that $\overline{\mathit{FINITE}_{\mathsf{TM}}}$ is not T-recognizable.

Solution

(a) *Proof.* We can reduce $\overline{A_{\mathsf{TM}}}$ to J with the function computed by the following TM F:

F on input y:
  Output $1y$.

$y \in \overline{A_{\mathsf{TM}}} \iff 1y \in J$. Thus $\overline{A_{\mathsf{TM}}} \leq_{\mathrm{m}} J$.

We can reduce A_{TM} to J with the function computed by the following TM F:

F on input x:
  Output $0x$.

$x \in A_{\mathsf{TM}} \iff 0x \in J$. Thus $A_{\mathsf{TM}} \leq_{\mathrm{m}} J$, $\overline{A_{\mathsf{TM}}} \leq_{\mathrm{m}} \overline{J}$.

$\overline{A_{\mathsf{TM}}}$ is not Turing-recognizable, thus neither J nor $\overline{J}$ is Turing-recognizable. ☐

(b) *Proof.* We can reduce A_{TM} to $\overline{\mathit{FINITE}_{\mathsf{TM}}}$ with the function computed by the following TM F:

F on input $\langle M, w\rangle$:
1. Construct the following TM T:
   T on x:
     Simulate M on w.
2. Output $\langle T\rangle$.

$\langle M, w\rangle \in A_{\mathsf{TM}} \Rightarrow M$ accepts $w \Rightarrow T$ accepts all $x \Rightarrow L(T) = \Sigma^*$, which is infinite.

$\langle M, w\rangle \notin A_{\mathsf{TM}} \Rightarrow M$ does not accept $w \Rightarrow T$ does not accept any $x \Rightarrow L(T) = \emptyset$, which is finite.

Thus $\langle M, w\rangle \in A_{\mathsf{TM}} \iff \langle T\rangle \in \overline{\mathit{FINITE}_{\mathsf{TM}}}$, $A_{\mathsf{TM}} \leq_{\mathrm{m}} \overline{\mathit{FINITE}_{\mathsf{TM}}}$ and equivalently $\overline{A_{\mathsf{TM}}} \leq_{\mathrm{m}} \mathit{FINITE}_{\mathsf{TM}}$. $\overline{A_{\mathsf{TM}}}$ is not T-recognizable, thus $\mathit{FINITE}_{\mathsf{TM}}$ is not T-recognizable. ☐

(c) *Proof.* We can reduce A_{TM} to $\mathit{FINITE}_{\mathsf{TM}}$ with the function computed by the following TM F:

F on input $\langle M, w\rangle$:
1. Construct the following TM T:
   T on x:
   1. Simulate M on w for $|x|$ steps.
   2. If M accepts w, *reject*. Otherwise, *accept*.
2. Output $\langle T\rangle$.

Let k be the number of steps that M on w runs until it halts, $\langle M, w\rangle \in A_{\mathsf{TM}} \Rightarrow M$ accepts w in k steps $\Rightarrow T$ accepts exactly those x with $|x| < k \Rightarrow L(T)$ is finite.

$\langle M, w\rangle \notin A_{\mathsf{TM}} \Rightarrow M$ does not accept $w \Rightarrow T$ accepts all $x \Rightarrow L(T) = \Sigma^*$, which is infinite.

Thus $\langle M, w\rangle \in A_{\mathsf{TM}} \iff \langle T\rangle \in \mathit{FINITE}_{\mathsf{TM}}$, $A_{\mathsf{TM}} \leq_{\mathrm{m}} \mathit{FINITE}_{\mathsf{TM}}$ and equivalently $\overline{A_{\mathsf{TM}}} \leq_{\mathrm{m}} \overline{\mathit{FINITE}_{\mathsf{TM}}}$. $\overline{A_{\mathsf{TM}}}$ is not T-recognizable, thus $\overline{\mathit{FINITE}_{\mathsf{TM}}}$ is not T-recognizable. ☐

Problem 8

Define a ***two headed finite automaton*** (2HFA) to be a deterministic finite automaton that has two read-only heads that each can move independently *from left to right* on the input tape. The transition function δ of a 2HFA has the form: $\delta : Q \times \Sigma \times \Sigma \to Q \times \{\text{S,R}\} \times \{\text{S,R}\}$. Thus, at each step, the 2HFA can read the symbols under each of its two heads and move each head independently either right (an R move) or let it stay in place (an S move). We also assume that a 2HFA accepts its input by entering one of its designated accept states, regardless of its head positions. Furthermore, a 2HFA can detect when either of its heads are at the right end of the input tape. For example, a 2HFA can recognize the language $\{\mathsf{a}^n\mathsf{b}^n\mathsf{c}^n \mid n \geq 1\}$ as follows:

1. Head 1 skips across any a's.
2. Head 1 reads b's while Head 2 reads a's.
3. Head 1 reads c's while Head 2 reads b's.
4. *Reject* if in Stages 2 or 3, Head 1 doesn't finish exactly when Head 2 finishes, or if symbols are ever encountered out of the order: a's, b's then c's. Otherwise, *accept*.

Let $E_{\mathsf{2HFA}} = \{\langle B \rangle \mid B \text{ is a 2HFA which recognizes the empty language}\}$. Sketch a proof that E_{2HFA} is not decidable. (Give enough detail to show how your proof depends on the 2HFA model.)

Solution

Assume for contradiction that E_{2HFA} is not decidable, then there is a TM R deciding E_{2HFA}. We can construct a TM S deciding A_{TM}:

S on $\langle M, w \rangle$:

1. Construct a 2HFA $B_{M,w}$ to test whether its input is an accepting computation history for M on w:

   $B_{M,w}$ on x:
   1. Head 1 reads input up to first #. *Reject* if it is not the starting configuration for M on w.
   2. Repeat till Head 1 reaches the right end of the tape:
   3. Head 1 and Head 2 each read input until they reach #s. *Reject* if the configuration that Head 2 just read doesn't legally yield the configuration that Head 1 just read.
   4. *Reject* if the last configuration that Head 1 read did not contain q_{accept}.
   5. *Accept.*
2. Run R on $\langle B_{M,w} \rangle$. If R accepts, $B_{M,w}$ recognizes the empty language, which means there is no accepting computation history for M on w, M rejects w, thus *reject*. Otherwise, *accept*.

However, we know that A_{TM} is undecidable. Thus E_{2HFA} is undecidable.