CS525 Winter 2012 \ Chapter #5 Preparation

Ariel Stolerman

Problems

5.9)

Let $T = \{\langle M \rangle \mid M \text{ is a TM that accepts } w^R \text{ whenever it accepts } w\}$. Following is a proof that T is undecidable. Assume that T is decidable, then there is a Turing machine R that decides it, so we can construct a TM S that will decide A_{TM} as follows.

We first define a Turing machine M' for any $\langle M, w \rangle$ as follows:

"On input x:

- If $x \neq w$ and $x \neq w^R$, *reject*.
- If $x = w^R$, *accept*.
- Otherwise ($x = w$), run M on input x and *accept* if M does."

Now we define the TM S as follows:

"On input $\langle M, w \rangle$:

- Construct M' as described above, with respect to the input M, w.
- Run R on M'. If it accepts, *accept*. Otherwise, *reject*."

Clearly M' is defined to accept only the string w^R, and also w only in the case M accepts w. Therefore if we have a TM that decides T, it will accept M' if and only if M accepts w, and reject otherwise (will always halt). Thus the construction of S above decides A_{TM}.

5.12)

5.13)

Let $US_{TM} = \{\langle M \rangle \mid M \text{ is a TM with a useless state}\}$. Following is a proof that US_{TM} is undecidable. Assume that US_{TM} is decidable, then there is a Turing machine R that decides it, so we can construct a TM S that will decide A_{TM} as follows.

We first define a Turing machine M' for any $\langle M, w \rangle$, with an additional symbol in the alphabet a and additional states are introduced, specifically $q_{useless}$ among them. In addition, any previous transition to q_{accept} shall be through $q_{useless}$ EXCEPT if the input symbol is the new a symbol. M' is defined as follows:

"On input x:

- If $x = a$, go through all states except $q_{useless}$ and then go to q_{accept} (accept).
- Otherwise, if $x \neq w$, go to q_{reject} (reject).
- Otherwise simulate M on $x = w$ (using the same states of M)."

Now we define the TM S as follows:

"On input $\langle M, w \rangle$:

- Construct M' from $\langle M, w \rangle$ as described above.
- Run R on M'. If it accepts, *reject*. Otherwise, *accept*.

First, note that M' will enter all states except for $q_{useless}$ in any case, as for the input a it goes through all but $q_{useless}, q_{reject}$ and for any input other than a, w it goes through q_{reject}. Now, for M' to be in US_{TM} it has to avoid going through $q_{useless}$ for the input w, and that will happen iff $\langle M, w \rangle \notin A_{TM}$. Therefore we can simulate R on the generated TM M' and be sure that if it accepts, it has a useless state $q_{useless}$, meaning M does not accept w (and the other way as well).

5.14)

Let

$A_{LEFT} =$

$\{\langle M, w \rangle \mid M \text{ is a Turing machine that for the input } w \text{ attempts to move its head left when it is on the leftmost tape cell}\}$.

Following is a proof that A_{LEFT} is undecidable. Assume A_{LEFT} is decidable, then there is a Turing machine R that decides it, so we can construct a TM S that decides A_{TM} as follows.

First we define a TM M' for any $\langle M \rangle$ as follows:

- Before doing anything, mark the first character. The first character in the tape will keep being marked every time it is changed by some state.
- Simulate M on the given input step by step, only whenever you land on a the marked character (which indicates the first position), and M dictates a left step, go right and left again instead, doing the state transition as originally dictated, and the character swap keeping the mark:

  In M: $\delta(q, a) = (r, b, L) \Rightarrow$ in M':
  - Not at first cell (unmarked character): $\delta'(q, a) = (r, b, L)$
  - At first cell (marked character): $\delta'(q, \hat{a}) = \left(q', \hat{b}, R\right), \forall c \in \Sigma: \delta'(q', c) = (r, c, L)$, where ˆ indicates the mark and q' is a new state in M'.
- Any original transition M has to its accept state, swap with a series of transitions that go to the leftmost cell of the tape, try to do one more left step disregarding any mark (the only states that will not do the right-step-left-step bypass as described previously), and then go to the accept state.

Now we define S as follows:

"On input $\langle M, w \rangle$:

- Construct M' as described above, with respect to the input M, w.
- Run R on $\langle M', w \rangle$. If it accepts, *accept*. Otherwise, *reject*."

The construction of M' for any given M makes sure that any attempts to go left from the leftmost tape cell are removed, by doing the original transition by going right and back left to the first cell again. Keeping the first cell marked makes sure there will be no attempts to go left from the leftmost cell during the simulation (until an accept state). Only when M accepts w, M' will attempt a left step from the leftmost tape cell. Therefore M accepts w if and only if M' will attempt to do a left step from the leftmost tape cell on the input w.

5.15)

5.16)

5.24)

Let $J = \{w \mid either\ w = 0x\ for\ some\ x \in A_{TM}, or\ w = 1y\ for\ some\ y \in \overline{A_{TM}}\}$. Following is a proof that both J and $\bar{J}$ are not Turing-recognizable.

To show J is not Turing-recognizable we show a mapping $\overline{A_{TM}} \leq_m J$: for any given input y we map it to $1y$. Clearly $y \in \overline{A_{TM}} \Leftrightarrow 1y \in \overline{A_{TM}}$ and this mapping is Turing-computable, therefore J is not Turing-recognizable.

To show $\bar{J}$ is not Turing-recognizable it is sufficient to show a mapping $A_{TM} \leq_m J$, as $A_{TM} \leq_m J \Leftrightarrow \overline{A_{TM}} \leq_m \bar{J}$: for any given input x we map it to $0x$. Clearly $x \in A_{TM} \Leftrightarrow 0x \in J$ and this mapping is Turing-computable, therefore $A_{TM} \leq_m J$ and so $\overline{A_{TM}} \leq_m \bar{J}$, thus $\bar{J}$ is not Turing-recognizable as well.

5.29)

5.30)

b.

c.

5.31)

Let H be a TM that decides A_{TM}. We will construct a TM T that answers the $3x + 1$ problem.

For the rest of the solution we assume any number representation is given over $\Sigma = \{0,1\}$ (binary representation).

Define the language:

$$L = \{x \mid x \in \mathbb{N},\ \ there\ exists\ a\ finite\ sequence\ t_0, \dots, t_k\ s.t.\ \forall i: t_i = f^i(x)\ and\ f^k(x) = 1\}$$

where $f^i(x)$ is $\underbrace{f(f(\dots f(x) \dots))}_{i\ times}$. Clearly if $L \equiv \mathbb{N}$ then all positive starting points end up at 1, otherwise there is at least one positive integer for which it does not occur. Either way, finding out whether $L \equiv N$ answers the $3x + 1$ problem.

We define a TM M_{REC} that **recognizes** L as follows:

"For input x:

- While $x \neq 1$ do:
  - $x = \begin{cases} 3x + 1, & x \in \mathbb{N}_{odd} \\ x/2, & x \in \mathbb{N}_{even} \end{cases}$

- If got to this point, *accept*."

Clearly M_{DEC} recognizes L, as for any $x \in L$ it will get to 1 at some point and accept (and if $x \notin L$ it will loop forever). Now we define a TM M_{DEC} that **decides** L as follows:

"For input x:

- Run H on $\langle M_{REC}, x \rangle$. If it accepts, *accept*. If it rejects, *reject*."

Clearly M_{DEC} decides L, as if $x \in L$ then M_{REC} will accept x and so H will accept $\langle M_{REC}, x \rangle$ thus M_{DEC} will accept, and if $x \notin L$ then M_{REC} on x loops forever, so H will reject $\langle M_{REC}, x \rangle$, thus M_{DEC} will reject.

We define N to be a TM that decides $\mathbb{N}$ (i.e. accepts all strings that represent a positive integer; it should be noted that using a binary representation and an alphabet of $\{0,1\}$, $\mathbb{N} \equiv \{0,1\}^+$).

Now we show a reduction from A_{TM} to EQ_{TM}: for a given input $\langle M, x \rangle$, construct a TM M_1 that ignores all inputs and simulates M on x, and a TM M_2 that accepts all inputs. Clearly $\langle M, x \rangle \in A_{TM} \Leftrightarrow \langle M_1, M_2 \rangle \in EQ_{TM}$ and this reduction is Turing-computable, thus $A_{TM} \leq_m EQ_{TM}$. Since we are given a decider for A_{TM} then it is decidable, thus EQ_{TM} is decidable as well, so there exists some TM E that decides EQ_{TM}.

We now construct a TM $TXP1$ that solves the $3x + 1$ problem, such that if all positive starting points end up at 1 it will accept, otherwise it will reject:

- "Ignore any input if given.
- Run E on the input TM pair $\langle M_{DEC}, N \rangle$. If it accepts, *accept*. Otherwise, *reject*."

Only in the case that **all** positive (integer) starting points have a finite sequence as defined for the $3x + 1$ problem that ends at 1, will $L = L(M_{DEC})$ be equivalent to $L(N) = \mathbb{N}$. Thus in that case only $TXP1$ will accept, otherwise there is some positive integer for which there exists no such sequence, and $TXP1$ will reject, as required.