Mid-term II review

• 2017/10/24

• Chapter 3 in Sipser
Examine the formal definition of a Turing machine to answer the following questions, and explain your reasoning.

a. Can a Turing machine ever write the blank symbol $\square$ on its tape?

b. Can the tape alphabet $\Gamma$ be the same as the input alphabet $\Sigma$?

c. Can a Turing machine’s head *ever* be in the same location in two successive steps?

d. Can a Turing machine contain just a single state?
TM questions

The collection of Turing-decidable languages is closed under the complement operation.
TM questions

The collection of **Turing-decidable** languages is closed under the complement operation

☐ True
The collection of Turing-recognizable languages is closed under the complement operation.
TM questions

The collection of Turing-recognizable languages is closed under the complement operation

False
If a language is **Turing-decidable** then all of its subsets are Turing-decidable.
If a language is \textbf{Turing-decidable} then all of its subsets are Turing-decidable.

\[ \square \text{False} \]
If a language is *Turing-recognizable* then all of its subsets are Turing-recognizable.
TM questions

If a language is **Turing-recognizable** then all of its subsets are Turing-recognizable.

- False
TM questions

Any undecidable language is a subset of a decidable language
Any undecidable language is a subset of a decidable language

☐ True
3.8 Give implementation-level descriptions of Turing machines that decide the following languages over the alphabet \{0,1\}.

\(a\). \(\{w| w \text{ contains an equal number of 0s and 1s}\}\)

\(b\). \(\{w| w \text{ contains twice as many 0s as 1s}\}\)

\(c\). \(\{w| w \text{ does not contain twice as many 0s as 1s}\}\)
3.8 Give implementation-level descriptions of Turing machines that decide the following languages over the alphabet \( \{0,1\} \).

\[ a. \, \{ w \mid w \text{ contains an equal number of } 0\text{s and } 1\text{s} \} \]
\[ b. \, \{ w \mid w \text{ contains twice as many } 0\text{s as } 1\text{s} \} \]
\[ c. \, \{ w \mid w \text{ does not contain twice as many } 0\text{s as } 1\text{s} \} \]

M on input \( w \):
1. Mark the first symbol as the starting symbol
2. Move head from left to right, cross off the first two 0s and one 1 then go to 3;
   - if no more 1s or 0s, accept;
   - if missing any symbols, reject
   (crossing off the starting symbol leaves a \( x \) marked as starting as well)
3. Go to the left most position and iterate 2.
*3.18* Show that a language is decidable iff some enumerator enumerates the language in the standard string order.

**Proof** First we show that if we have an enumerator $E$ that enumerates a language $A$, a TM $M$ recognizes $A$. The TM $M$ works in the following way.

$M =$ “On input $w$:
1. Run $E$. Every time that $E$ outputs a string, compare it with $w$.
2. If $w$ ever appears in the output of $E$, accept.”

Clearly, $M$ accepts those strings that appear on $E$’s list.

Now we do the other direction. If TM $M$ recognizes a language $A$, we can construct the following enumerator $E$ for $A$. Say that $s_1, s_2, s_3, \ldots$ is a list of all possible strings in $\Sigma^*$.

$E =$ “Ignore the input.
1. Repeat the following for $i = 1, 2, 3, \ldots$.
2. Run $M$ for $i$ steps on each input, $s_1, s_2, \ldots, s_i$.
3. If any computations accept, print out the corresponding $s_j$.”
TM questions

*3.18* Show that a language is decidable iff some enumerator enumerates the language in the standard string order.

**Proof** First we show that if we have an enumerator $E$ that enumerates a language $A$, a $\text{TM} M$ recognizes $A$. The $\text{TM} M$ works in the following way.

$M =$ “On input $w$:
1. Run $E$. Every time that $E$ outputs a string, compare it with $w$.
2. If $w$ ever appears in the output of $E$, accept.”

Clearly, $M$ accepts those strings that appear on $E$’s list.

3. Compare $w$ with output of $e$. If $w < e$, reject

Now we do the other direction. If $\text{TM} M$ recognizes a language $A$, we can construct the following enumerator $E$ for $A$. Say that $s_1, s_2, s_3, \ldots$ is a list of all possible strings in $\Sigma^*$.

$E =$ “Ignore the input.
1. Repeat the following for $i = 1, 2, 3, \ldots$.
2. Run $M$ for $i$ steps on each input, $s_1, s_2, \ldots, s_i$.
3. If any computations accept, print out the corresponding $s_j$.”

How do we modify this proof for Turing recognizable?
3.14 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.
3.14 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.

Direction 1: simulate Q with M, M on input w:

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>pull</th>
<th>X</th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>_</th>
<th>_</th>
<th>X</th>
<th>X</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>_</th>
<th>_</th>
</tr>
</thead>
<tbody>
<tr>
<td>push e</td>
<td>X</td>
<td>a</td>
<td>b</td>
<td>c</td>
<td>d</td>
<td>_</td>
<td>_</td>
<td>X</td>
<td>a</td>
<td>b</td>
<td>c</td>
<td>d</td>
<td>e</td>
<td>_</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
TM questions

3.14 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.

Direction 2: simulate M with Q, Q on input w:

- **current state**: $s \rightarrow s', R$
- **desirable state**: $b c d s'$

Note: we need to know which one is the last symbol. How?
TM questions

3.14 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.

Direction 2: simulate M with Q, Q on input w: current state

head on the left most symbol

head on a symbol

In between

head on the last symbol that is blank

Note: we need to know which one is first and last symbol. How?