# Solved selected problems of Introduction to the Theory of Computation by Michael Sipser.

Franco Zacco

## 3 The Church-Turing Thesis

### 3.1 Turing Machines

*Proof.* **3.1**

**a.**

$$q_1 0 \sqcup$$
$$\sqcup q_2 \sqcup$$
$$\sqcup \sqcup q_{\text{accept}}$$

**b.**

$q_1 00 \sqcup$ $\quad \sqcup q_2 x \sqcup$
$\sqcup q_2 0 \sqcup$ $\quad \sqcup x q_2 \sqcup$
$\sqcup x q_3 \sqcup$ $\quad \sqcup x \sqcup q_{\text{accept}}$
$\sqcup q_5 x \sqcup$
$q_5 \sqcup x \sqcup$

**c.**

$$q_1 000 \sqcup$$
$$\sqcup q_2 00 \sqcup$$
$$\sqcup x q_3 0 \sqcup$$
$$\sqcup x 0 q_4 \sqcup$$
$$\sqcup x 0 \sqcup q_{\text{reject}}$$

**d.**

$q_1 000000 \sqcup$ $\quad \sqcup x0x0q_5 x \sqcup$ $\quad \sqcup x q_2 0x0x \sqcup$
$\sqcup q_2 00000 \sqcup$ $\quad \sqcup x0x q_5 0x \sqcup$ $\quad \sqcup xx q_3 x0x \sqcup$
$\sqcup x q_3 0000 \sqcup$ $\quad \sqcup x0q_5 x0x \sqcup$ $\quad \sqcup xxx q_3 0x \sqcup$
$\sqcup x0q_4 000 \sqcup$ $\quad \sqcup x q_5 0x0x \sqcup$ $\quad \sqcup xxx0q_4 x \sqcup$
$\sqcup x0x q_3 00 \sqcup$ $\quad \sqcup q_5 x0x0x \sqcup$ $\quad \sqcup xxx0x q_4 \sqcup$
$\sqcup x0x0q_4 0 \sqcup$ $\quad q_5 \sqcup x0x0x \sqcup$ $\quad \sqcup xxx0x \sqcup q_{\text{reject}}$
$\sqcup x0x0x q_3 \sqcup$ $\quad \sqcup q_2 x0x0x \sqcup$

□

*Proof.* **3.2**

**a.**

$$q_1 11\sqcup$$
$$xq_3 1\sqcup$$
$$x1q_3\sqcup$$
$$x1\sqcup q_{\text{reject}}$$

**b.**

$q_1 1\#1\sqcup$ $xq_1\#x\sqcup$
$xq_3\#1\sqcup$ $x\#q_8x\sqcup$
$x\#q_5 1\sqcup$ $x\#xq_8\sqcup$
$xq_6\#x\sqcup$ $x\#x\sqcup q_{\text{accept}}$
$q_7x\#x\sqcup$

**c.**

$$q_1 1\#\#1\sqcup$$
$$xq_3\#\#1\sqcup$$
$$x\#q_5\#\sqcup$$
$$x\#\#q_{\text{reject}}\sqcup$$

**d.**

$q_1 10\#11\sqcup$ $xq_1 0\#x1\sqcup$
$xq_3 0\#11\sqcup$ $xxq_2\#x1\sqcup$
$x0q_3\#11\sqcup$ $xx\#q_4x1\sqcup$
$x0\#q_5 11\sqcup$ $xx\#xq_4 1\sqcup$
$x0q_6\#x1\sqcup$ $xx\#x1q_{\text{reject}}\sqcup$
$xq_7 0\#x1\sqcup$
$q_7x0\#x1\sqcup$

**e.**

$q_1 10\#10\sqcup$ $xq_1 0\#x0\sqcup$ $xxq_1\#xx\sqcup$
$xq_3 0\#10\sqcup$ $xxq_2\#x0\sqcup$ $xx\#q_8xx\sqcup$
$x0q_3\#10\sqcup$ $xx\#q_4x0\sqcup$ $xx\#xq_8x\sqcup$
$x0\#q_5 10\sqcup$ $xx\#xq_4 0\sqcup$ $xx\#xxq_8\sqcup$
$x0q_6\#x0\sqcup$ $xx\#q_6xx\sqcup$ $xx\#xx\sqcup q_{\text{accept}}$
$xq_7 0\#x0\sqcup$ $xxq_6\#xx\sqcup$
$q_7x0\#x0\sqcup$ $xq_7x\#xx\sqcup$

□

*Proof.* **3.3**

($\Rightarrow$) If a language $L$ is decidable then there is a deterministic Turing machine which decides it, but this machine is nondeterministic as well so there is a nondeterministic Turing machine which decides $L$.

($\Leftarrow$) We will use the proof to Theorem 3.16 to prove that every nondeterministic Turing machine that decides a language has an equivalent deterministic Turing machine that decides the language.

Suppose there is some nondeterministic Turing machine $N$ that decides a language $L$.

We need a multitape deterministic Turing machine $D$ where as we did in Theorem 3.16, the first tape (input tape) contains the input string, the second tape (simulation tape) contains a copy of $N$'s tape on some branch of its nondeterministic computation, and tape 3 (address tape) keeps track of $D$'s location in $N$'s nondeterministic finite computation tree.

Then $D$ is described as follows

1. Initially tape 1 contains the input $w$, and tapes 2 and 3 are empty.
2. Copy tape 1 to tape 2.
3. Use tape 2 to simulate $N$ with input $w$ on one branch of its nondeterministic computation. Before each step of $N$ consult the next symbol on tape 3 to determine which choice to make among those allowed by $N$'s transition function. If no more symbols remain on tape 3 or if this nondeterministic choice is invalid, abort this branch by going to stage 4. Also, go to stage 4 if a rejecting configuration is encountered. If an accepting configuration is encountered, accept the input.
4. Replace the string on tape 3 with the lexicographically next string. Simulate the next branch of $N$'s computation by going to stage 2.
5. If we tried every possible string on tape 3 and we didn't reach any accept configuration, then reject the input.

Therefore, since $D$ cannot loop forever because $N$'s tree is finite and hence we must reach an accept or reject state then we get a deterministic Turing machine that decides the language $L$ i.e. the language is decidable.

□

*Proof.* **3.4**
An **enumerator** is a 7-tuple, $(Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$, where $Q, \Sigma, \Gamma$ are all finite sets and

1. $Q$ is the set of states,
2. $\Sigma$ is the input alphabet,
3. $\Gamma$ is the printer alphabet,
4. $\delta : Q \times \Sigma \to Q \times \Sigma \times \mathcal{P}(\Gamma)$ is the transition function,
5. $q_0 \in Q$ is the start state,
6. $q_{accept} \in Q$ is the accept state, and
7. $q_{reject} \in Q$ id the reject state, where $q_{reject} \neq q_{accept}$.

An enumerated language $L$ is the language that an enumerator $E$ prints out hence $L \subset \mathcal{P}(\Gamma)$. □

*Proof.* **3.5**

**a.** Given that the transition function is defined as $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$ and $\textvisiblespace \in \Gamma$ then a Turing Machine can write $\textvisiblespace$ on its tape.

**b.** No, the tape alphabet $\Gamma$ cannot be the same as the input alphabet $\Sigma$ since by definition $\textvisiblespace \in \Gamma$ but $\textvisiblespace \notin \Sigma$.

**c.** Yes, this could happen when the head is on the leftmost square of the tape, then if the transtion function asks the head to go to the left the head will stay where it is hence the head will be in the same location in two successive steps.

**d.** No, a turing machine by definition contains at least $q_{reject}$ and $q_{start}$ where $q_{reject} \neq q_{accept}$.

□

*Proof.* **3.7** The machine $M_{bad}$ is not a legitimate Turing machine for two reasons.

First, given that the machine tries all possible settings of $x_1, ..., x_k$ to integer values, we must store these integer values somehow, but a Turing machine can only have finite alphabets $\Gamma$ and $\Sigma$.

Finally, this machine may not halt, because the polynomial may not be solvable, and hence we may loop forever. □

*Proof.* **3.8**

**a.** $M_1 = \{w \mid w \text{ contains an equal number of 0s and 1s}\}$
On input string $w$:

1. Scan the first element of the tape from left to right if it's a 0 then cross it off and continue scanning the tape until you find a 1 and cross it off. If the first element is a 1 then cross it off and continue scanning the tape until you find a 0 and cross it off.
2. If you reach the end of the tape (a symbol $\sqcup$) and you do not find either a 1 or a 0 depending on the case you are in, then reject.
3. Return to the second element and repeat the process described in **1.** and **2.** Do this for all slots in the tape. If you encounter a crossed-off element on the tape continue to the next slot.
4. If there are no more slots with 0s or 1s then accept, otherwise reject.

**b.** $M_2 = \{w \mid w \text{ contains twice as many 0s as 1s}\}$
On input string $w$:

1. Scan the tape from left to right until you find a 1. If there are no 1s but there are 0s, reject. Cross off the 1 you found.
2. Return to the head of the tape and scan from left to right until you find a 0 and cross it off, continue scanning until you find another 0 and cross it off. If while trying to execute this step there are no more 0s, reject.
3. Continue the process described in **1.** and **2.** for every 1 you find. If there are no more 1s but there are still 0s, reject. If while executing this step there are no more 0s and 1s. accept.

**c.** $M_2 = \{w \mid w \text{ does not contain twice as many 0s as 1s}\}$
On input string $w$:

1. Scan the tape from left to right until you find a 1. If there are no 1s but there are 0s, accept. Cross off the 1 you found.
2. Return to the head of the tape and scan from left to right until you find a 0 and cross it off, continue scanning until you find another 0 and cross it off. If while trying to execute this step there are no more 0s you can cross off, accept.
3. Continue the process described in **1.** and **2.** for every 1 you find. If there are no more 1s but there are still 0s, accept. If while executing this step there are no more 0s and 1s. reject.

□