Formal Languages and the Theory of Computation

Topic 3. Turing Machines

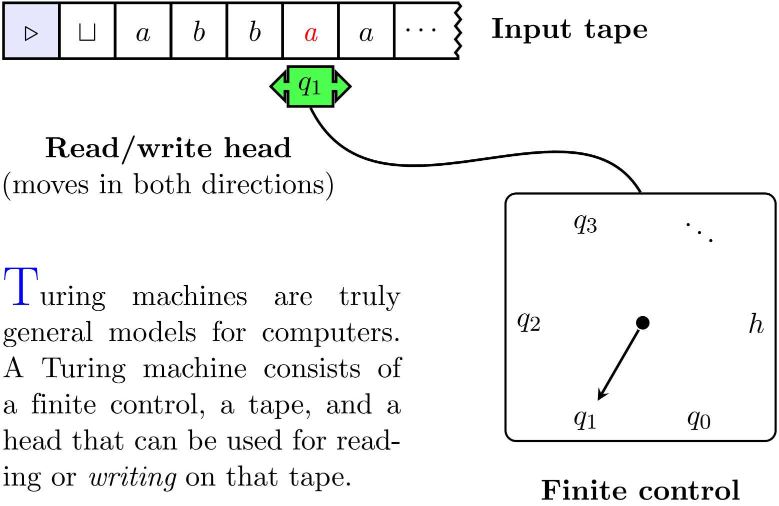
Han, Nguyen Dinh (han.nguyendinh@hust.edu.vn)

Faculty of Mathematics and Informatics
Hanoi University of Science and Technology

Outline

1

The schema of a Turing machine



Turing machines are truly general models for computers. A Turing machine consists of a finite control, a tape, and a head that can be used for reading or *writing* on that tape.

$\boxed{2}$

The definition of a Turing machine

Definition 3.1 A Turing machine is a 5-tuple $(Q, \Sigma, \delta, s, H)$, where

1. Q is a finite set of *states*,
2. Σ is an alphabet, containing the **blank symbol** $\sqcup$ and the **left end symbol** $\triangleright$, but not containing the symbols $\leftarrow$ and $\rightarrow$,
3. $s \in Q$ is the *start state*,
4. $H \subseteq Q$ is the set of **halting states**,
5. δ, the **transition function**, is a function from $(Q - H) \times \Sigma$ to $Q \times (\Sigma \cup \{\leftarrow, \rightarrow\})$ such that:
   (a) for all $q \in Q - H$, if $\delta(q, \triangleright) = (p, b)$ then $b = \rightarrow$,
   (b) for all $q \in Q - H$ and $a \in \Sigma$, if $\delta(q, a) = (p, b)$ then $b \neq \triangleright$.

Example 3.1 Consider the Turing machine $M = (Q, \Sigma, \delta, s, \{h\})$, where $Q = \{q_0, q_1, h\}$, $\Sigma = \{a, \sqcup, \triangleright\}$, $s = q_0$, and δ is given by the following table.

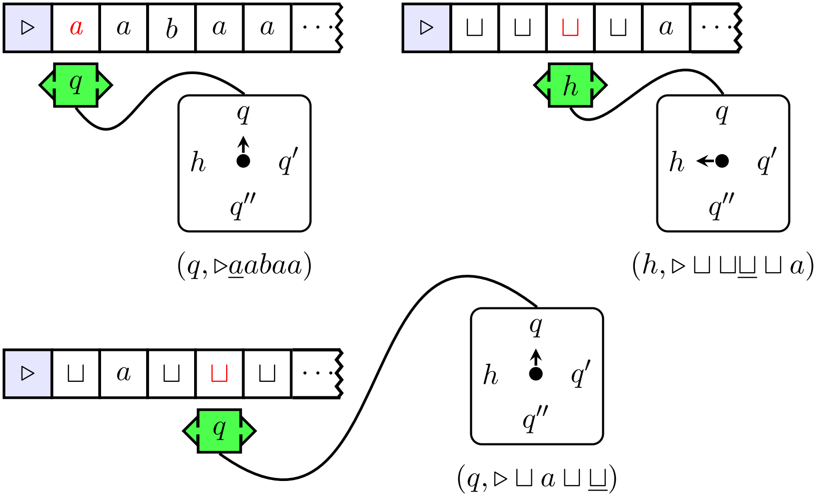
q,	σ	$\delta(q, \sigma)$
q_0	a	$(q_1, \sqcup)$
q_0	$\sqcup$	$(h, \sqcup)$
q_0	$\triangleright$	$(q_0, \rightarrow)$
q_1	a	(q_0, a)
q_1	$\sqcup$	$(q_0, \rightarrow)$
q_1	$\triangleright$	$(q_1, \rightarrow)$

Definition 3.2 Let $M = (Q, \Sigma, \delta, s, H)$ be a Turing machine. Then

1. A **configuration** of M is a member of $Q \times \triangleright\Sigma^* \times (\Sigma^*(\Sigma - \{\sqcup\}) \cup \{\varepsilon\})$.
2. A configuration whose state component is in H will be called a **halted configuration**.

NOTES $\rightsquigarrow$ All configurations are assumed to start with the left end symbol, and never end with a blank – unless the blank is currently scanned. Thus $(q, \triangleright a, aba), (h, \triangleright \sqcup \sqcup\sqcup, \sqcup a)$ and $(q, \triangleright \sqcup a \sqcup \sqcup, \varepsilon)$ are configurations, but $(q, \triangleright baa, a, bc\sqcup)$ and $(q, \sqcup aa, ba)$ are not. Also, we can write $(q, \omega a, u)$ as $(q, \omega \underline{a} u)$.

2. The definition of a Turing machine



Definition 3.3 Let $M = (Q, \Sigma, \delta, s, H)$ be a Turing machine and consider two configurations of M, $(q_1, \omega_1 \underline{a_1} u_1)$ and $(q_2, \omega_2 \underline{a_2} u_2)$, where $a_1, a_2 \in \Sigma$. Then

$$(q_1, \omega_1 \underline{a_1} u_1) \vdash_M (q_2, \omega_2 \underline{a_2} u_2)$$

if and only if for some $b \in \Sigma \cup \{\leftarrow, \rightarrow\}, \delta(q_1, a_1) = (q_2, b)$ and either

1. $b \in \Sigma, \omega_1 = \omega_2, u_1 = u_2$ and $a_2 = b$, or
2. $b = \leftarrow$, $\omega_1 = \omega_2 a_2$, and either
   - (a) $u_2 = a_1 u_1$ (if $a_1 \neq \sqcup$ or $u_1 \neq \varepsilon$), or
   - (b) $u_2 = \varepsilon$ (if $a_1 = \sqcup$ and $u_1 = \varepsilon$); or
3. $b = \rightarrow$, $\omega_2 = \omega_1 a_1$, and either
   - (a) $u_1 = a_2 u_2$, or
   - (b) $u_1 = u_2 = \varepsilon$ and $a_2 = \sqcup$.

Example 3.2 Consider the Turing machine M described in Example 3.1. If M is started in configuration $(q_1, \triangleright\underline{\sqcup}aaaa)$, its computation would be represented formally as follows.

$$
\begin{array}{llll}
(q_1, \triangleright\underline{\sqcup}aaaa) & \vdash_M (q_0, \triangleright\sqcup\underline{a}aaa) & \vdash_M (q_1, \triangleright\sqcup\underline{\sqcup}aaa) \\
& \vdash_M (q_0, \triangleright\sqcup\sqcup\underline{a}aa) & \vdash_M (q_1, \triangleright\sqcup\sqcup\underline{\sqcup}aa) \\
& \vdash_M (q_0, \triangleright\sqcup\sqcup\sqcup\underline{a}a) & \vdash_M (q_1, \triangleright\sqcup\sqcup\sqcup\underline{\sqcup}a) \\
& \vdash_M (q_0, \triangleright\sqcup\sqcup\sqcup\sqcup\underline{a}) & \vdash_M (q_1, \triangleright\sqcup\sqcup\sqcup\sqcup\underline{\sqcup}) \\
& \vdash_M (q_0, \triangleright\sqcup\sqcup\sqcup\sqcup\sqcup\underline{\sqcup}) & \\
& \vdash_M (h, \triangleright\sqcup\sqcup\sqcup\sqcup\sqcup\underline{\sqcup}) &
\end{array}
$$

Definition 3.4 Let $M = (Q, \Sigma, \delta, s, H)$ be a Turing machine and let $\vdash_M^*$ be the **reflexive, transitive closure** of $\vdash_M$. Then

1. We say that configuration C **yields** configuration C' if $C \vdash_M^* C'$.
2. A **computation** by M is a sequence of configurations $C_0, C_1, \ldots, C_n$, for some $n \geq 0$ such that

$$C_0 \vdash_M C_1 \vdash_M C_2 \vdash_M \cdots \vdash_M C_n.$$

3. We say that the computation is of **length** n or that it has n **steps**, and we write $C_0 \vdash_M^n C_n$.

Problem 3.1 Let M be the Turing machine $(Q, \Sigma, \delta, s, \{h\})$, where $Q = \{q_0, q_1, h\}$, $\Sigma = \{a, b, \sqcup, \triangleright\}$, $s = q_0$, and δ is given by the following table.

q,	σ	$\delta(q,\sigma)$	q,	σ	$\delta(q,\sigma)$
q_0	a	(q_1, b)	q_1	a	$(q_0, \rightarrow)$
q_0	b	(q_1, a)	q_1	b	$(q_0, \rightarrow)$
q_0	$\sqcup$	$(h, \sqcup)$	q_1	$\sqcup$	$(q_0, \rightarrow)$
q_0	$\triangleright$	$(q_0, \rightarrow)$	q_1	$\triangleright$	$(q_1, \rightarrow)$

(a) Trace the computation of M starting from $(q_0, \triangleright\underline{a}abbba)$.

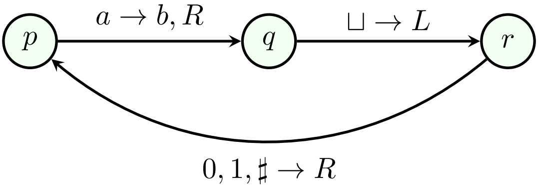
(b) Describe informally what M does when started in q_0 on any square of a tape.

Problem 3.2 Let M be the Turing machine $(Q, \Sigma, \delta, s, \{h\})$, where $Q = \{q_0, q_1, q_2, h\}$, $\Sigma = \{a, b, \sqcup, \triangleright\}$, $s = q_0$, and δ is given by the following table.

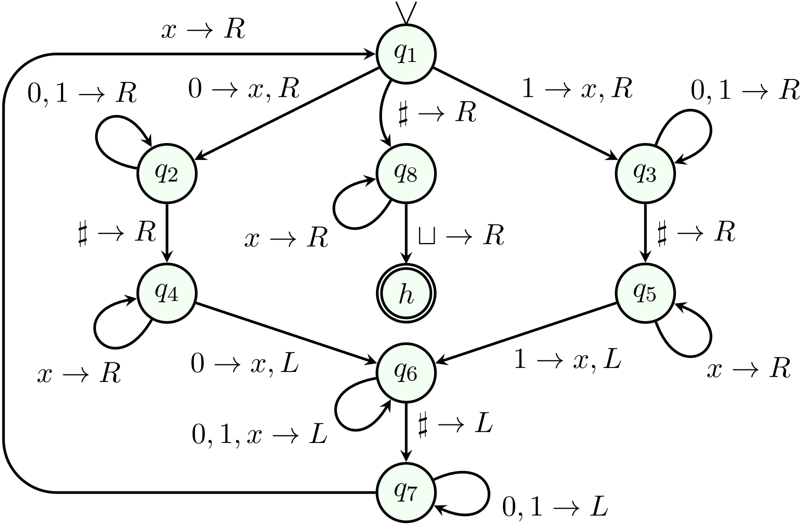
$q,$	σ	$\delta(q,\sigma)$	$q,$	σ	$\delta(q,\sigma)$
q_0	a	$(q_1, \leftarrow)$	q_1	$\sqcup$	$(q_1, \leftarrow)$
q_0	b	$(q_0, \rightarrow)$	q_2	a	$(q_2, \rightarrow)$
q_0	$\sqcup$	$(q_0, \rightarrow)$	q_2	b	$(q_2, \rightarrow)$
q_1	a	$(q_1, \leftarrow)$	q_2	$\sqcup$	$(h, \sqcup)$
q_1	b	$(q_2, \rightarrow)$			

Suppose that M starts from $(q_0, \triangleright a\underline{b}b \sqcup bb \sqcup \sqcup \sqcup aba)$. Repeat Problem 3.1 for M.

A transition function can be given in the form of a state diagram. However, we shall use two symbols L and R instead of $\leftarrow$ and $\rightarrow$. For example, the following state diagram is a pictorial representation for the transition function $\delta : (Q - H) \times \Sigma \longrightarrow Q \times \Sigma \times \{L, R\}$, $\delta(p, a) = (q, b, R)$, $\delta(q, \sqcup) = (r, \sqcup, L)$, $\delta(r, 0) = (p, 0, R)$, $\delta(r, 1) = (p, 1, R)$ and $\delta(r, \sharp) = (p, \sharp, R)$.



Can you explain the operation of this Turing machine?



Problem 3.3 Design and write out in full a Turing machine which finds the first blank in a given string.

Problem 3.4 Design and write out in full a Turing machine that checks the symmetry of strings on a given alphabet Σ (e.g. *eye, madam, mom* are symmetric words).

Problem 3.5 Design and write out in full a Turing machine that checks whether or not two binary strings are equal. The alphabet of the Turing machine should be $\{0, 1, \sharp, \sqcup, \triangleright\}$.

$\boxed{3}$

A notation for Turing machines

3. A notation for Turing machines

To represent Turing machines more graphic and transparent, at first we define *basic machines* as follows.

Let Σ be an alphabet. For each $a \in \Sigma \cup \{\rightarrow, \leftarrow\} - \{\triangleright\}$, we define a Turing $M_a = (\{s, h\}, \Sigma, \delta, s, \{h\})$, where for each $b \in \Sigma - \{\triangleright\}$, $\delta(s, b) = (h, a)$. Naturally, $\delta(s, \triangleright) = (s, \rightarrow)$. If $a \in \Sigma - \{\triangleright\}$, M_a is called the *symbol-writing machine*, and is called the *head-moving machine* otherwise.

Indeed, the only thing M_a does is to perform action a – writing symbol a if $a \in \Sigma - \{\triangleright\}$, moving in the direction indicated by a if $a \in \{\rightarrow, \leftarrow\}$ – and then to immediately halt.

3. A notation for Turing machines

By convention, the basic machines M_a, $M_\leftarrow$ and $M_\rightarrow$ will be denoted simply as a, L and R. Next we define *rules for combining machines* as follows.

Turing machines will be combined in a way suggestive of the structure of a finite automaton. Individual machines are like the states of a finite automaton, and the machines may be connected to each other in the way that the states of a finite automaton are connected together. *However, the connection from one machine to another is not pursued until the first machine halts; the other machine is then started from its initial state with the tape and head position as they were left by the first machine.*

3. A notation for Turing machines

Example 3.3 Suppose that M_1, M_2 and M_3 are Turing machines. Then, the combined Turing machine M and its operation are as follows.

$$\begin{tikzcd}
M_1 \arrow[r, "a"] \arrow[d, "b"] & M_2 \\
M_3 &
\end{tikzcd}$$

M starts in the initial state of M_1; operates as M_1 would operate until M_1 would halt. Then, if the currently scanned symbol is an a, initiates M_2 and operates as M_2 would operate; otherwise, if the currently scanned symbol is a b, then initiates M_3 and operates as M_3 would operate.

Example 3.3 (cont.) Formally, now suppose that $M_1 = (Q_1, \Sigma, \delta_1, s_1, H_1)$, $M_2 = (Q_2, \Sigma, \delta_2, s_2, H_2)$, $M_3 = (Q_3, \Sigma, \delta_3, s_3, H_3)$ and assume $\bigcap_{i=1}^{3} Q_i = \emptyset$. Then

$M = (Q, \Sigma, \delta, s, H)$, where $Q = \bigcup_{i=1}^{3} Q_i, s = s_1, H = H_2 \cup H_3$. For each $\sigma \in \Sigma$ and $q \in Q - H$, $\delta(q, \sigma)$ is as follows:

- (a) If $q \in Q_1 - H_1$, then $\delta(q, \sigma) = \delta_1(q, \sigma)$.
- (b) If $q \in Q_2 - H_2$, then $\delta(q, \sigma) = \delta_2(q, \sigma)$.
- (c) If $q \in Q_3 - H_3$, then $\delta(q, \sigma) = \delta_3(q, \sigma)$.
- (d) Finally, if $q \in H_1$ - the only case remaining - then $\delta(q, \sigma) = s_2$ if $\sigma = a$, $\delta(q, \sigma) = s_3$ if $\sigma = b$, and $\delta(q, \sigma) \in H$ otherwise.

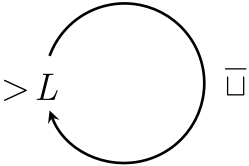
3. A notation for Turing machines

Let us find a convenient way to represent Turing machines by diagrams. The following two diagrams are for the same machine, but the right-hand side presentation is more convenient.

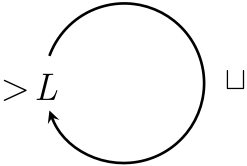
$$\begin{tikzcd}[column sep=huge] > R \arrow[r, bend left=50, "a"] \arrow[r, bend left=15, "b"] \arrow[r, bend right=15, "\triangleright"'] \arrow[r, bend right=50, "\sqcup"'] & R \end{tikzcd} \qquad > R \xrightarrow{a,\, b,\, \triangleright,\, \sqcup} R$$

Furthermore, if an arrow is labeled by all symbols in Σ, then the labels can be omitted entirely. For example, if $\Sigma = \{a, b, \triangleright, \sqcup\}$, then we can display the above machine as $R \to R$, or simply RR or even R^2.

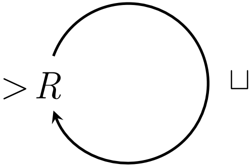
Example 3.4 TMs to find marked or unmarked squares.



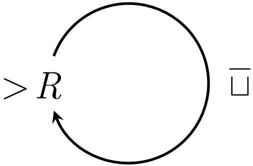
(a) $L_{\sqcup}$



(b) $L_{\overline{\sqcup}}$



(c) $R_{\overline{\sqcup}}$



(d) $R_{\sqcup}$

Problem 3.6 Describe informally the operation of the following Turing machine.

$$\begin{tikzcd} > R \arrow[r, "a \neq \sqcup"] \arrow[d, "\sqcup"] & \sqcup\, R_{\sqcup}^{2}\, a\, L_{\sqcup}^{2}\, a \arrow[l, bend right] \\ R_{\sqcup} & \end{tikzcd}$$

$\boxed{4}$

Computing with Turing machines

We explore how Turing machines are used to perform computational tasks. First, we adopt a policy for representing input to Turing machines. If $M = (Q, \Sigma, \delta, s, H)$ is a Turing machine and $\omega \in (\Sigma - \{\sqcup, \triangleright\})$, then t**he initial configuration of M on input ω** is

$$(s, \triangleright\underline{\sqcup}\omega).$$

That is, the input string, with no blank symbols in it, is written to the right of the leftmost symbol $\triangleright$, with a blank to its left, and blanks to its right. The head is positioned at the tape square containing the blank between the $\triangleright$ and the input and M starts operating in its start state.

$\boxed{4.1}$

Turing machines recognize languages

Definition 3.5 Let $M = (Q, \Sigma, \delta, s, H)$ be a Turing machine, such that $H = \{y, n\}$ consists of two distinguished halting states (y and n for "yes" and "no"). Then

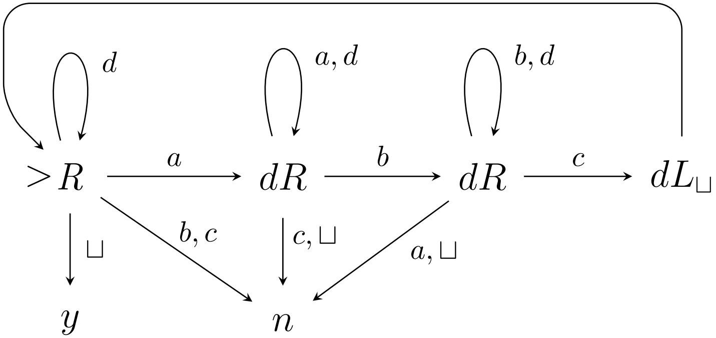
1. Any halting configuration whose state component is y is called an **accepting configuration**, while a halting configuration whose state component is n is called a **rejecting configuration**.
2. For an input $\omega \in (\Sigma - \{\sqcup, \triangleright\})^*$, we say that M **accepts** ω if $(s, \triangleright\underline{\sqcup}\omega)$ yields an accepting configuration and we say that M **rejects** ω if $(s, \triangleright\underline{\sqcup}\omega)$ yields a rejecting configuration.

Definition 3.5 (cont.) Let $\Sigma_0 \subseteq \Sigma - \{\sqcup, \triangleright\}$ to be an alphabet, called the **input alphabet** of M. By fixing Σ_0 to be a subset of $\Sigma - \{\sqcup, \triangleright\}$, we allow M to use extra symbols during its computation, besides those appearing in its inputs. Then

3. We say that M **decides** a language $L \subseteq \Sigma_0^*$ if for any string $\omega \in \Sigma_0^*$ the following is true:
   If $\omega \in L$ then M accepts ω; and if $\omega \notin L$ then M rejects ω.

4. We call a language L **recursive** if there is a Turing machine that decides it.

Example 3.5 The following Turing machine decides the language $L = \{a^n b^n c^n : n \geq 0\}$.



$\boxed{4.2}$

Turing machines compute functions

Definition 3.6 Let $M = (Q, \Sigma, \delta, s, \{h\})$, let $\Sigma_0 = \Sigma - \{\sqcup, \triangleright\}$ be an alphabet, and let $\omega \in \Sigma_0^*$. Suppose that M halts on input ω, and that $(s, \triangleright\underline{\sqcup}\omega) \vdash_M^* (h, \triangleright\underline{\sqcup}y)$ for some $y \in \Sigma_0^*$. Then y is called the **output of** M **on input** ω and is denoted $M(\omega)$. Notice that $M(\omega)$ is defined only if M halts on input ω, and in fact does so at a configuration of the form $(h, \triangleright\underline{\sqcup}y)$ with $y \in \Sigma_0^*$.

Let f be any function from Σ_0^* to Σ_0^*. We say that M computes function f if, for all $\omega \in \Sigma_0^*$, $M(\omega) = f(\omega)$. That is, for all $\omega \in \Sigma_0^*$, M eventually halts on input ω, and when it does halt, its tape contains the string $\triangleright\underline{\sqcup}f(\omega)$. A function f is called **recursive**, if there is a Turing machine M that computes f.

Problem 3.7 Check if the following Turing machine can compute the function $f : \Sigma^* \to \Sigma^*, f(\omega) = \omega\omega$?

$$
\begin{tikzcd}
> R \arrow[rr, "a \neq \sqcup"] \arrow[d, "\sqcup"] & & \sqcup\, R^2_{\sqcup}\, a\, L^2_{\sqcup}\, a \arrow[ll, bend right] \\
R_{\sqcup} & &
\end{tikzcd}
$$

$\boxed{5}$

Extensions of the Turing machine

THANK YOU VERY MUCH FOR YOUR ATTENTION!