NOTES ON THE THEORY OF COMPUTATION

YANNAN MAO

16th September 2022

Contents

1 Automata and Formal Languages 1
1.1 Finite-State Automata and Regular Languages 1
1.1.1 Deterministic Finite-State Automata 1
1.1.2 Nondeterministic Finite-State Automata 2
1.1.3 Regular Expressions and Regular Languages 3
1.1.4 Nonregular Languages 7
1.2 Pushdown Automata and Context-Free Languages 8
1.2.1 Context-Free Grammars and Context-Free Languages 8

Automata and Formal Languages

An **alphabet** is a finite set Σ, and a **word over the alphabet** Σ is a finite sequence of the elements of Σ. If a word w is the sequence $(w_0, \dots, w_n)$ for some $n \in \mathbb{N}$, we may write the word as $w_0 \cdots w_n$. The empty word is denoted by ε. The set of all words over Σ is Σ^*[1]. A **formal language over the alphabet** Σ is a subset of Σ^*.

An **automaton** is an ordered sequence that **accepts** some words over an alphabet. The set of words an automaton accepts forms a language, which is unique, in which case we say the automaton **recognises** the language. Given an automaton M, we may speak of the unique language recognised by M as the **language of the automaton** M. An automaton may accept no string, in which case the language thereof is $\emptyset$.

1.1 Finite-State Automata and Regular Languages

1.1.1 Deterministic Finite-State Automata

Definition 1. A **deterministic finite-state automaton** is an ordered quintuple $(\Sigma, S, \delta, s_0, F)$ wherein

(a) Σ is an alphabet,

(b) S is a finite set of **states**,

(c) $\delta : S \times \Sigma \to S$ is the **transition function**,

(d) $s_0 \in S$ is the **initial state**, and

(e) $F \subseteq S$ is the set of **final states** or **accepting states**.

Let $M = (\Sigma, S, \delta, s_0, F)$ be a deterministic finite-state automaton and let $w = w_0 \cdots w_n$ wherein $n \in \mathbb{N}$ be a word over Σ. Then M accepts w if there exists a finite sequence of states $(r_0, \dots, r_{n+1})$ in S such that

(a) $r_0 = s_0$,

(b) $\delta(r_i, w_i) = r_{i+1}$ for $i = 0, \dots, n$, and

(c) $r_{n+1} \in F$.

[1] * is the unary operator of Kleene star, defined as $A^* = \{\varepsilon\} \cup \{a_0 \cdots a_n : n \in \mathbb{N} \wedge \forall i \in \mathbb{N}_{<n+1} (a_i \in A)\}n$.

1.1.2 Nondeterministic Finite-State Automata

Definition 2. A **nondeterministic finite-state automaton** is an ordered quintuple $(\Sigma, S, \delta, s_0, F)$ wherein

(a) Σ is an alphabet,

(b) S is a finite set of states,

(c) $\delta : S \times (\Sigma \cup \varepsilon) \to \mathscr{P}(S)$ is the transition function,

(d) $s_0 \in S$ is the initial state, and

(e) $F \subseteq S$ is the set of final or accepting states.

Let $M = (\Sigma, S, \delta, s_0, F)$ be a nondeterministic finite-state automaton and let w be a word over Σ. Then M accepts w if $w = w_0 \cdots w_n$ wherein $n \in \mathbb{N}$ such that each $w_i \in \Sigma_\varepsilon$, $i \in \mathbb{N}_{<n+1}$, and that there exists a finite sequence of states $(r_0, \dots, r_{n+1})$ in S such that

(a) $r_0 = s_0$,

(b) $r_{i+1} \in \delta(r_i, w_i)$ for $i = 0, \dots, n$, and

(c) $r_{n+1} \in F$

We say that two automata are equivalent if they recognise the same language.

Theorem 1. *Every nondeterministic finite-state automaton has an equivalent deterministic finite-state automaton.*

Proof. Let $N = (\Sigma, S, \delta, s_0, F)$ be the nondeterministic finite-state automaton recognising some language A. We construct a deterministic finite-state automaton $M = (\Sigma, S', \delta', s_0', F')$ recognising A.

We first see that $S' = \mathscr{P}(S)$ and that $F' = \{R \in S' : R \cap F \neq \emptyset\}$.

Let $\delta_0 : S \times \{\varepsilon\} \to \mathscr{P}(S)$ be defined as $\delta_0(s, \varepsilon) = \delta(s, \varepsilon)$ for each $s \in S$. Assume first that, thus induced, $\delta_0 = \emptyset$ for N. For each $R \in S'$ and for each $a \in \Sigma$, let $\delta'(R, a) = \{s \in S : \exists r \in R \left(s \in \delta(r, a)\right)\}$. Equivalently,

$$\delta'(R, a) = \bigcup_{r \in R} \delta(r, a).$$

Also let $s_0' = \{s_0'\}$. We then see that $M = (\Sigma, S', \delta', s_0', F')$ recognises A.

Assume then that $\delta_0 \neq \emptyset$ for N. For each $R \subseteq S$, let

$$E(R) = \{s \in S : \exists n \in \mathbb{N}\, \exists r \in R\, (s = \delta^n(r, \varepsilon))\}.$$

We then let

$$\delta'(R, a) = \{s \in S : \exists r \in R\, s \in E(\delta(r, a))\}$$

and let $s_0' = E(\{s_0\})$. We similarly see that $M = (\Sigma, S', \delta', s_0', F')$ recognises A.

Therefore, the theorem holds. □

1.1.3 Regular Expressions and Regular Languages

Definition 3. Let Σ be an alphabet, and let $a \in \Sigma$. Then some $R \subseteq \Sigma^*$ is a **regular language** if

(a) $R = \emptyset$,

(b) $R = \{\varepsilon\}$,

(c) $R = \{a\}$,

(d) $R = R_1 \cup R_2$ wherein R_1 and R_2 are regular languages over Σ,

(e) $R = R_1 R_2$[2] wherein R_1 and R_2 are regular languages over Σ, or

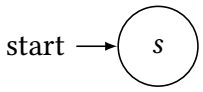
(f) $R = R_1^*$ wherein R_1 is a regular language over Σ.

An expression used to describe a regular language is a **regular expression**. If R is a regular expression, we denote the regular language it describes by $L(R)$.

Lemma 1. *If a language is regular, then some nondeterministic finite-state automaton recognises it.*

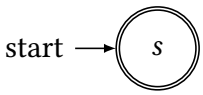
Proof. Let Σ be an alphabet, and let R be a regular language over Σ. If $R = \emptyset$, then the nondeterministic finite-state automaton characterised by the following diagram recognises it.

[2] $R_1 R_2$ is the concatenation of R_1 and R_2, defined as $R_1 R_2 = \{xy : x \in R_1 \wedge y \in R_2\}$



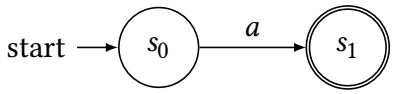
Equivalently, $N = (\Sigma, \{s\}, \delta, s, \emptyset)$ wherein $\delta(r, b) = \emptyset$ for any $r \in \{s\}$ and for any $b \in \Sigma$ recognises R.

If $R = \{\varepsilon\}$, then the nondeterministic finite-state automaton characterised by the following diagram recognises it.



Equivalently, $N = (\Sigma, \{s\}, \delta, s, \{s\})$ wherein $\delta(r, b) = \emptyset$ for any $r \in \{s\}$ and for any $b \in \Sigma$ recognises R.

If $R = \{a\}$ for some $a \in \Sigma$, then the nondeterministic finite-state automaton characterised by the following diagram recognises it.



Equivalently, $N = (\Sigma, \{s_0, s_1\}, \delta, s_0, \{s_1\})$ wherein $\delta(s_0, a) = \{s_1\}$ and $\delta(r, b) = \emptyset$ if $r \neq s_0$ or $b \neq a$.

Assume that R_1 and R_2 are regular languages over Σ, that $N_1 = (\Sigma, S_1, \delta_1, s_1, F_1)$ is a nondeterministic finite-state automaton recognising R_1, and that $N_2 = (\Sigma, S_2, \delta_2, s_2, F_2)$ is a nondeterministic finite-state automaton recognising R_2.

If $R = R_1 \cup R_2$, let s_0 be a state not in S_1 or S_2, let $S = \{s_0\} \cup S_1 \cup S_2$, and let $F = F_1 \cup F_2$. Define $\delta : S \times \Sigma \to \mathscr{P}(S)$ so that for each $r \in S$ and for each $b \in \Sigma$ we have

$$\delta(r, b) = \begin{cases} \delta_1(r, b) & \text{if } r \in S_1, \\ \delta_2(r, b) & \text{if } r \in S_2, \\ \{s_1, s_2\} & \text{if } r = s_0 \wedge b = \varepsilon, \text{ and} \\ \emptyset & \text{otherwise.} \end{cases}$$

We see that $N = (\Sigma, S, \delta, s_0, F)$ is a nondeterministic finite-state automaton recognising R.

If $R = R_1R_2$, let $S = S_1 \cup S_2$. Define $\delta : S \times \Sigma \to \mathscr{P}(S)$ so that for each $r \in S$ and for each $b \in \Sigma$ we have

$$\delta(r,b) = \begin{cases} \delta_1(r,b) & \text{if } (r \in S_1 \wedge r \notin F_1) \vee (r \in F_1 \wedge b \neq \varepsilon), \\ \delta_1(r,b) \cup \{s_2\} & \text{if } r \in F_1 \wedge b = \varepsilon, \text{ and} \\ \delta_2(r,b) & \text{otherwise.} \end{cases}$$

We see that $N = (\Sigma, S, \delta, s_1, F_2)$ is a nondeterministic finite-state automaton recognising R.

If $R = R_1^*$, let s_0 be a state not in S_1, let $S = \{s_0\} \cup S_1$, and let $F = \{s_0\} \cup F_1$. Define $\delta : S \times \Sigma \to \mathscr{P}(S)$ so that for each $r \in S$ and for each $b \in \Sigma$ we have

$$\delta(r,b) = \begin{cases} \delta_1(r,b) & \text{if } (r \in S_1 \wedge r \notin F_1) \vee (r \in F_1 \wedge b \neq \varepsilon), \\ \delta_1(r,b) \cup \{s_1\} & \text{if } r \in F_1 \wedge b = \varepsilon, \\ \{s_1\} & \text{if } r = s_0 \wedge b = \varepsilon, \text{ and} \\ \varnothing & \text{otherwise.} \end{cases}$$

We see that $N = (\Sigma, S, \delta, s_0, F)$ is a nondeterministic finite-state automaton recognising R.

Therefore, the lemma holds by the principle of induction. □

Definition 4. A **generalised nondeterministic finite-state automaton** is an ordered quintuple $(\Sigma, S, \delta, s_0, s_1)$ wherein

(a) Σ is an alphabet,

(b) S is a finite set of states,

(c) $\delta : (S \setminus \{s_1\}) \times (S \setminus \{s_0\}) \to \mathscr{R}$ wherein $\mathscr{R}$ is the set of all regular expressions over Σ is the transition function,

(d) $s_0 \in S$ is the initial state, and

(e) $s_1 \neq s_0 \in S$ is the final or accepting state.

Let $M = (\Sigma, S, \delta, s_0, s_1)$ be a generalised nondeterministic finite-state automaton and let w be a word over Σ. Then M accepts w if $w = w_0 \cdots w_n$ wherein $n \in \mathbb{N}$ such that each $w_i \in \Sigma^*$, $i \in \mathbb{N}_{<n+1}$, and that there exists a finite sequence of states $(r_0, \ldots, r_{n+1})$ such that

(a) $r_0 = s_0$,

(b) $r_{n+1} = s_1$, and

(c) for each i we have $w_i \in L(R_i)$ wherein $R_i = \delta(r_i, r_{i+1})$.

Lemma 2. *If a nondeterministic finite-state automaton recognises a language, then it is regular.*

Proof. Let $N = (\Sigma, S, \delta, s_0, F)$ be a nondeterministic finite-state automaton recognising the language A over Σ. We prove that A is described by some regular expression R over Σ.

Let $G = (\Sigma, S', \delta', s_0', s_1')$ be a generalised nondeterministic finite-state automaton such that

(a) $s_0' \notin S$,

(b) $s_1' \notin S$,

(c) $S' = \{s_0', s_1'\} \cup S$, and

(d) for each $r_0 \in \{s_0'\} \cup S$ and for each $r_1 \in \{s_1'\} \cup S$ we have

$$\delta'(r_0, r_1) = \begin{cases} \varepsilon & \text{if } (r_0 = s_0' \wedge r_1 = s_0) \vee (r_0 \in F \wedge r_1 = s_1'), \\ R' & \text{if } r_0 \in S \wedge r_1 \in S \wedge \forall r \in L(R') \left(r_1 \in \delta(r_0, r)\right), \text{ and} \\ \varnothing & \text{otherwise.} \end{cases}$$

We see that G also recognises A. We shall then convert G into regular expression R.

Let $k = |S'|$. If $k = 2$, then $S' = \{s_0', s_1'\}$, and so $R = \delta'(s_0', s_1')$ is the regular expression.

If $k > 2$, let $s \in S'$ be distinct from s_0' and s_1', and let $G' = (\Sigma, S'', \delta'', s_0', s_1')$ be a generalised nondeterministic finite-state automaton such that

(a) $S'' = S' \setminus \{s\}$,

(b) for any $r_0 \in S'' \setminus \{s_0'\}$ and for any $r_1 \in S'' \setminus \{s_1'\}$ we have

$$\delta''(r_0, r_1) = R_0 R_1^* R_2 \cup R_3$$

wherein $R_0 = \delta'(r_0, s)$, $R_1 = \delta'(s, s)$, $R_2 = \delta'(s, r_1)$, and $R_3 = \delta'(r_0, r_1)$.

We see that G' is equivalent to G.

Because G' has one fewer state than G, by the principle of induction, there exists a regular expression R converted from G for any generalised nondeterministic finite-state automaton.

Therefore, the lemma holds. □

Theorem 2. *A language is regular if and only if some nondeterministic finite-state automaton recognises it.*

Proof. The theorem holds by Lemma 1 and Lemma 2. □

1.1.4 Nonregular Languages

Theorem 3 (Pumping lemma). *If R is a regular language over Σ, then there is a number $p \in \mathbb{Z}_{>0}$, the* **pumping length**, *such that if $r \in R$ is of length at least p, then there exist $x, y, z \in \Sigma^*$ such that $r = xyz$ which satisfy that*

(a) $xy^iz \in R$ *for each* $i \in \mathbb{N}$,

(b) $|y| > 0$, *and*

(c) $|xy| \leq p$.

Proof. Let $N = (\Sigma, S, \delta, s_0, F)$ be a nondeterministic finite-state automaton recognising R and let $p = |S|$.

Let $r = r_0 \cdots r_n$ wherein $n \in \mathbb{N}$ be a word in R of length $n+1$ which satisfies $n+1 \geq p$. Let $s_0, \ldots, s_{n+1}$ be the sequence of states that N enters when accepting r. This sequence has length $n+2$, which must be at least $p+1$. Among the first $p+1$ elements in the sequence, two must be the same state by the pigeonhole principle. Let the first of these be s_i and the second s_j. We note that $i \leq p$. Now let $x = r_0 \cdots r_{i-1}$, $y = r_i \cdots r_{j-1}$, and $z = r_j \cdots r_n$.

Thus induced, $r = xyz$ satisfies the pumping lemma. □

We may use the pumping lemma to prove that a language is not regular. To do so, first assume that the language is regular, and then use the pumping lemma to guarantee the existence of a pumping length p such that all words of length p or greater in the language can be pumped. Next, find a word in the language with length p or greater but cannot be pumped. Finally, demonstrate that this word cannot be pumped by considering all ways of diving it

into three such substrings described by the pumping lemma. The existence of this word contradicts the pumping lemme assuming the language is regular. Hence, the language is not regular.

1.2 Pushdown Automata and Context-Free Languages

1.2.1 Context-Free Grammars and Context-Free Languages