Lecture Notes
Finite Automata and Büchi Automata

Assaf Kfoury

January 20, 2018

Section 1 in this handout is a brief review of *finite automata* and *regular languages*; this is material that students have normally seen in one or more undergraduate courses in the standard computer-science curriculum. I include it here because it is a good background for Section 2, which is a quick introduction to so-called *Büchi automata* and *ω-regular languages*. The latter material is rarely, if ever, covered in undergraduate courses.

As much as possible, I follow the notational conventions of the textbook [PMC]. Throughout, I mention facts without their proofs; I include references to the latter from [PMC] whenever appropriate.

1 Review: Regular Expressions and Finite Automata

In contrast to [PMC], I use two different script versions of the letter L: $\mathcal{L}$ and $\mathscr{L}$. The first script $\mathcal{L}$ is a metavariable (appropriately decorated) ranging over languages, the second script $\mathscr{L}$ is an operator (mapping regular expressions or automata to languages).

The next definition is [PMC, Definition 4.1, page 151].

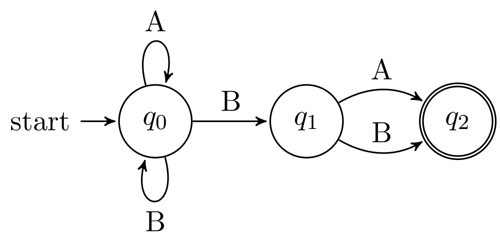
Definition 1 (**Nondeterministic Finite Automata**)**.** A *nondeterministic finite automaton* (NFA) is a 5-tuple $\mathcal{A} = (Q, \Sigma, \delta, Q_0, F)$ where:

- Q is a finite set of *states*,
- Σ is the *alphabet*,
- $\delta : Q \times \Sigma \to 2^Q$ is the *transition function*,
- $Q_0 \subseteq Q$ is the subset of *initial states*,
- $F \subseteq Q$ is the subset of *final* or *accept states*.

The language *accepted/recognized* by a NFA $\mathcal{A}$ is denoted $\mathscr{L}(\mathcal{A})$. The transition function δ can be identified with the ternary relation $\to\, \subseteq Q \times \Sigma \times Q$ defined by $q \xrightarrow{A} q'$ iff $q' \in \delta(q, A)$. $\square$

The following example is part of [PMC, Example 4.2, pages 152-153].

Example 2. The following NFA, $\mathcal{A}_1$, recognizes the set of words/strings defined by the regular expression $(A + B)^*B(A + B)$.



□

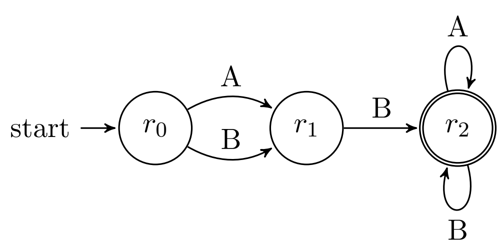
The definitions of *regular expressions* and *regular sets* (also called *regular languages*) are given in one of the appendices of the book, [PMC, pages 914-915]. An important fact to keep in mind: Every *regular expression* defines a *regular set*, and every *regular set* is defined by a *regular expression*.

Fact 3. *If $\mathcal{A}$ is a NFA over alphabet Σ, then the set of finite words recognized/accepted by $\mathcal{A}$ is a regular set over Σ.*

Fact 4. *If X is a regular set over alphabet Σ, then we can construct a NFA $\mathcal{A}$ which recognizes/accepts the set X.*

The two preceding facts show that *NFA's* and *regular expressions* are two equivalent ways of defining regular languages/regular sets.

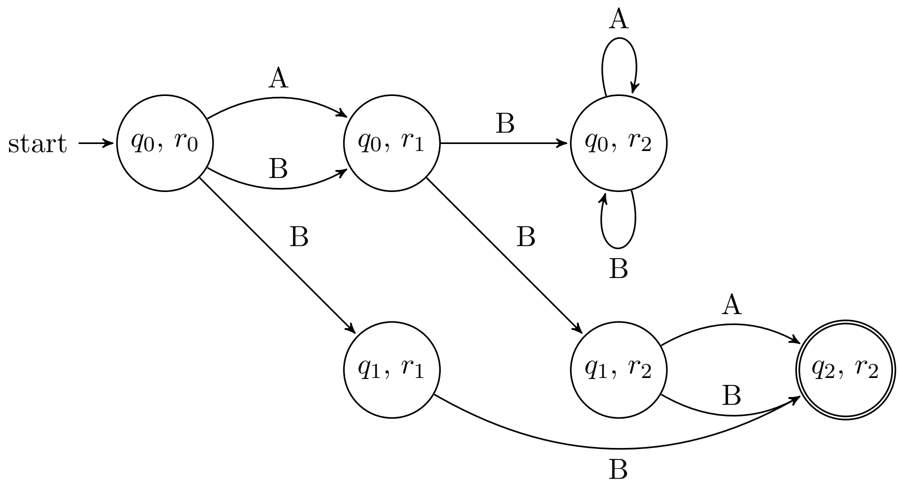
Example 5. The following NFA, $\mathcal{A}_2$, recognizes the set of words/strings defined by the regular expression $(A+B)B(A+B)^*$.



□

A formal definition of the *synchronous product* of two NFA's is given in [PMC, Definition 4.8, page 156]. It is illustrated in the next example.

Example 6. The synchronous product $\mathcal{A}_1 \otimes \mathcal{A}_2$ of $\mathcal{A}_1$ and $\mathcal{A}_2$ is



$\mathcal{A}_1 \otimes \mathcal{A}_2$ recognizes the set of words defined by the regular expression $(A + B)\, B\, (A + B) + B\, B$. $\square$

Fact 7. *Given NFA $\mathcal{A}_1$ and NFA $\mathcal{A}_2$, it holds that $\mathscr{L}(\mathcal{A}_1) \cap \mathscr{L}(\mathcal{A}_2) = \varnothing$ if and only if $\mathscr{L}(\mathcal{A}_1 \otimes \mathcal{A}_2) = \varnothing$.*

For the next fact, *equivalence of NFA's* is defined in [PMC, Definition 4.6, page 155]. The definition of *deterministic finite automaton* (DFA) is given in [PMC, Definition 4.9, page 156], which is a particular NFA satisfying two conditions:

- $|Q_0| \leqslant 1$, and
- $|\delta(q, A)| \leqslant 1$ for every pair $(q, A) \in Q \times \Sigma$.

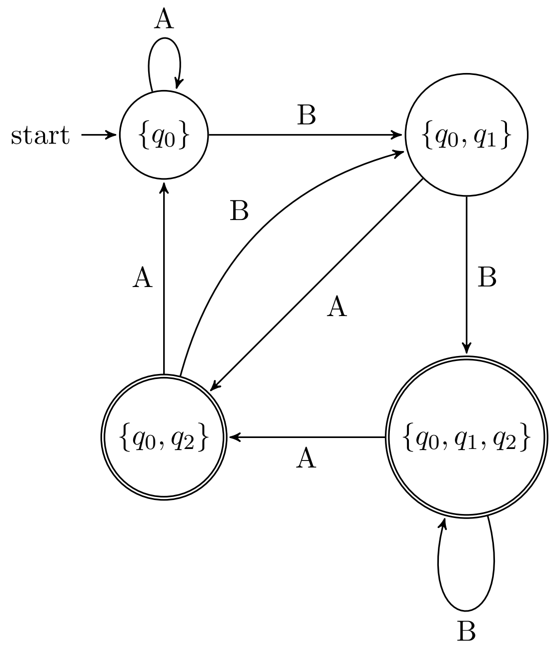
If the equality holds in these two conditions, the DFA is called *total*.[1]

Fact 8. *If $\mathcal{A}$ is a NFA, then we can construct a DFA $\mathcal{B}$ equivalent to $\mathcal{A}$, i.e., such that $\mathscr{L}(\mathcal{A}) = \mathscr{L}(\mathcal{B})$.*

The proof of the preceding fact uses what is called the *powerset construction*, sometimes called the *subset construction*, which is described in [PMC, page 157]. The next example is [PMC, Example 4.10, page 158] which is obtained using the powerset construction.

Example 9. The DFA $\mathcal{A}_3$, given below, is total deterministic and equivalent to $\mathcal{A}_1$

[1] In many other textbooks, they require that the equality holds in these two conditions and they do not distinguish between DFA's and *total* DFA's.



The language recognized/accepted by $\mathcal{A}_3$ is $\mathscr{L}\big(\mathcal{A}_3\big) = \mathscr{L}\big((A+B)^*B(A+B)\big)$. $\square$

More material, with numerous examples, on *finite automata* is in [PMC, Section 4.1, pages 151-158].

2 ω-Regular Expressions and Büchi Automata

As in Section 1 and in contrast to [PMC], I use two two different script versions of the letter L: $\mathcal{L}$ and $\mathscr{L}$, the first as a metavariable (appropriately decorated) ranging over languages and the second as an operator on ω-regular expressions and Büchi automata, as defined below – as well as an operator on regular expressions and finite automata.

The following is from [PMC, Definition 4.23, page 171].

Definition 10 (**ω-Regular Expressions**)**.** An *ω-regular expression* G over the alphabet Σ has the form

$$G = E_1 \cdot F_1^\omega + \cdots + E_n \cdot F_n^\omega \qquad \text{(sometimes written as } E_1\, F_1^\omega + \cdots + E_n\, F_n^\omega \text{ more compactly)}$$

where $E_1, \ldots, E_n, F_1, \ldots F_n$ are regular expressions over Σ, with $n \geqslant 1$, and the empty string ε is not in $\mathscr{L}\big(F_i\big)$ for every $1 \leqslant i \leqslant n$. $\square$

Example 11. $(A + B)^*A(AAB + C)^\omega$ and $A(B + C)^*A^\omega + B(A + C)^\omega$ are examples of ω-regular expressions over the alphabet $\Sigma = \{A, B, C\}$. $\square$

For a language $\mathcal{L} \subseteq \Sigma^*$, let $\mathcal{L}^\omega$ be the set of words in $\Sigma^* \cup \Sigma^\omega$ that arise from the infinite concatenation of (arbitrary) words in Σ, *i.e.*,

$$\mathcal{L}^\omega \triangleq \{\, w_1 w_2 w_3 \cdots \mid w_i \in \mathcal{L} \text{ and } i \geqslant 1 \,\}.$$

If $\mathcal{L}$ does not contain the empty word ε, then $\mathcal{L}^\omega \cap \Sigma^* = \varnothing$ and $\mathcal{L}^\omega$ is an ω-language, *i.e.*, every word in $\mathcal{L}^\omega$ is infinite. In this case, we also have that $\mathcal{L}^\omega \subseteq \Sigma^\omega$ (do you see why?). Note carefully that the superscripts here are operators:

- ( )* and ( )$^\omega$ are operators, *i.e.*, the "$*$" and the "ω" are not part of their argument names.[2]

Let $\mathcal{L}_1 \subseteq \Sigma^*$ and $\mathcal{L}_2 \subseteq \Sigma^\omega$, *i.e.*, $\mathcal{L}_1$ is a language of finite words and $\mathcal{L}_2$ a language of infinite words. We write $\mathcal{L}_1 \cdot \mathcal{L}_2$ to denote the set of all the *concatenations* of two words, one from $\mathcal{L}_1$ and one from $\mathcal{L}_2$, which is an ω-language:

$$\mathcal{L}_1 \cdot \mathcal{L}_2 \triangleq \{ w_1 w_2 \mid w_1 \in \mathcal{L}_1 \text{ and } w_2 \in \mathcal{L}_2 \}.$$

The next definition is [PMC, Definition 4.24, page 172], written a little differently (there is no need to introduce an additional operator denoted "$\mathcal{L}(\ )_\omega$" as in [PMC], which may be a little confusing in the presence of several uses of ω in superscript position as an operator and in subscript position as part of an operator name).

Definition 12 (**ω-Regular Languages**)**.** An ω-language $\mathcal{L} \subseteq \Sigma^\omega$ is called an *ω-regular language* if there an ω-regular expression G such that:

$$\mathcal{L} = \mathscr{L}\big(G\big) \triangleq \mathscr{L}\big(E_1\big) \cdot \big(\mathscr{L}\big(F_1\big)\big)^\omega \cup \cdots \cup \mathscr{L}\big(E_n\big) \cdot \big(\mathscr{L}\big(F_n\big)\big)^\omega \qquad \square$$

The following example is taken from the first paragraph after [PMC, Definition 4.24, page 172].

Example 13.

1. All infinite words over the alphabet $\{A, B\}$ that contain infinitely A's is ω-regular. This ω-regular language is induced by the ω-regular expression $(B^* A)^\omega$.
2. All infinite words over $\{A, B\}$ that contain finitely many A's is ω-regular. This ω-regular language is induced by the ω-regular expression $(A + B)^* B^\omega$.
3. The empty set is an ω-regular language induced by the ω-regular expression $\varnothing^\omega$. $\square$

The following definition is part of [PMC, Definition 4.23, page 171].

Definition 14 (**Equivalence of ω-Regular Expressions**)**.** Two ω-regular expressions G_1 and G_2 are *equivalent*, denoted $G_1 \equiv G_2$, if and only if $\mathscr{L}\big(G_1\big) = \mathscr{L}\big(G_2\big)$. $\square$

Notation 15. If $\varepsilon \notin \mathscr{L}\big(E\big)$ where E is a regular expression, then we can view E^ω as an ω-regular expression, since E^ω can be identified with $E \cdot E^\omega$ or also with $\varepsilon \cdot E^\omega$.

The following fact is taken from the second paragraph after [PMC, Definition 4.24, page 172].

Fact 16. *ω-regular languages, just like regular languages, are closed under: (i) union, (ii) intersection and (iii) complementation.*

In Fact 16, the proof of (*i*) is easy (left to you); the proof of (*ii*) is a consequence of [PMC, Corollary 4.60, page 198], which is proved after introducing a variant of NBA's called *generalized nondeterministic Büchi automata* (GNBA) [PMC, Definition 4.52, page 193]; and the proof of (*iii*) is more complicated and omitted in [PMC] (though references to the literature for this result are included).

The next definition is [PMC, Definition 4.27, page 174].

[2] To be absolutely clear, we could write $(\Sigma)^*$, $(\mathcal{L})^*$, $(\Sigma)^\omega$, and $(\mathcal{L})^\omega$, instead of Σ^*, $\mathcal{L}^*$, Σ^ω, and $\mathcal{L}^\omega$, respectively. But it is a common practice to omit the parenthesis pairs and to remember that "$*$" and "ω" are operators and not parts of their argument names. Note also that "$*$" appearing in regular expressions, and "$*$" and "ω" appearing in ω-regular expressions, are not operators; they are just symbols, like the symbols "$+$" and "$\cdot$", and these four symbols {"$*$","ω", "$+$","$\cdot$"} become operators when regular expressions and ω-regular expressions are interpreted as regular languages and ω-regular languages.

Definition 17 (**Nondeterministic Büchi Automata**)**.** A *nondeterministic Büchi automaton* (NBA) $\mathcal{A}_B$ is a 5-tuple $\mathcal{A}_B = (Q, \Sigma, \delta, Q_0, F)$ where:

- Q is a finite set of *states*,
- Σ is the *alphabet*,
- $\delta : Q \times \Sigma \to 2^Q$ is the *transition function*,
- $Q_0 \subseteq Q$ is the subset of *initial states*, and
- $F \subseteq Q$ is the subset of *accept* (or *final*) states, called the *acceptance set.*

Note that a NBA is defined just like a NFA, except that acceptance/recognition of words is defined differently, as we explain next.[3] A *run* σ for NBA $\mathcal{A}_B$ is an ω-sequence over Σ, say,

$$\sigma = A_0\, A_1\, A_2\ \cdots$$

which induces an ω-sequence of states, say,

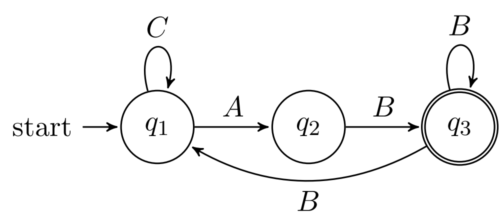
$$q_0\ q_1\ q_2\ \cdots$$

such that $q_i \xrightarrow{A_i} q_{i+1}$, $i.e.\delta(q_i, A_i) = q_{i+1}$, for every $i \geqslant 0$. The run σ is an *accepting run* if $q_i \in F$ for infinitely many i's. The *language accepted* (or *recognized*) by $\mathcal{A}_B$ is denoted $\mathscr{L}(\mathcal{A}_B)$ and defined by:

$$\mathscr{L}(\mathcal{A}_B) \triangleq \{\, \sigma \in \Sigma^\omega \mid \text{ there is an accepting run for } \sigma \text{ in } \mathcal{A}_B \,\} \qquad \square$$

The following is [PMC, Example 4.28, page 175].

Example 18. The following is a NBA $\mathcal{A}_B$ over the alphabet $\Sigma = \{A, B, C\}$:



We can view the NBA $\mathcal{A}_B$ as a NFA $\mathcal{A}$. As a NBA, $\mathcal{A}_B$ recognizes the ω-regular language corresponding to the ω-regular expression $C^* A\, B\, (B + B\, C^* A\, B)^\omega$. As a NFA, $\mathcal{A}$ recognizes the regular language corresponding to the regular expression $C^* A\, B\, (B + B\, C^* A\, B)^*$. [4] $\square$

The next result is [PMC, Theorem 4.32, page 178].

Fact 19. *The languages accepted by NBA's are exactly the ω-regular languages.*

The proof of the preceding fact is rather long [PMC, pages 178-184], but it includes many helpful examples.

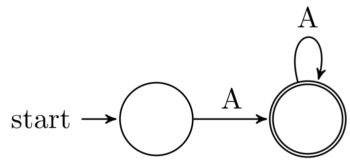
Example 20. We should be careful when we compare the behaviours of NFA's and NBA's.

1. The following finite automata $\mathcal{A}_1$ and $\mathcal{A}_2$ accept the same finite words:

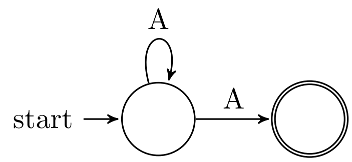
   Namely, $\mathscr{L}(\mathcal{A}_1) = \mathscr{L}(\mathcal{A}_2) = \{A^n | n \geqslant 1\}$. As Büchi automata, however, $\mathscr{L}(\mathcal{A}_{1,B}) = \{A^\omega\}$ and $\mathscr{L}(\mathcal{A}_{2,B}) = \varnothing$.

[3] This is why I use "$\mathcal{A}_B$" as a name for a Büchi automaton: The subscript "B" indicates that the automaton is used as a Büchi automaton, not as a finite automaton. As a rule, if a NFA is called $\mathcal{A}$, I will call $\mathcal{A}_B$ its counterpart as a NBA; and if a NBA is called $\mathcal{A}_B$, I will call $\mathcal{A}$ its counterpart as a NFA.

[4] In [PMC, Example 4.28, page 175], the ω-regular expression defined by $\mathcal{A}_B$ is given as $C^* A\, B\, (B^+ + B\, C^* A\, B)^\omega$ – note the "+" on the second occurrence of "B", but this "+" is not necessary (can you see why?).

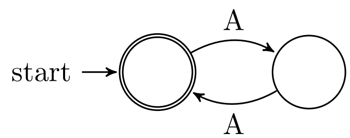


(a) $\mathcal{A}_1$

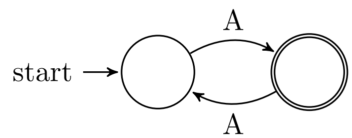


(b) $\mathcal{A}_2$

2. The following Büchi automata $\mathcal{A}_{1,B}$ and $\mathcal{A}_{2,B}$ accept the same infinite words:



(c) $\mathcal{A}_{1,B}$



(d) $\mathcal{A}_{2,B}$

Namely, $\mathscr{L}(\mathcal{A}_{1,B}) = \mathscr{L}(\mathcal{A}_{2,B}) = \{A^\omega\}$. As finite automata, however, $\mathscr{L}(\mathcal{A}_1) = \{A^{2n} | n \geqslant 0\}$ and $\mathscr{L}(\mathcal{A}_2) = \{A^{2n+1} | n \geqslant 0\}$. □

The definition of a *deterministic Büchi automaton* (DBA) is from [PMC, Definition 4.48, page 188] and is the same as that of a *deterministic finite automaton* (DFA) given before Fact 8 above. Specifically, we say the Büchi automaton $\mathcal{A}_B = (Q, \Sigma, \delta, Q_0, F)$ is a DBA iff:

- $|Q_0| \leqslant 1$, and
- $|\delta(q, A)| \leqslant 1$ for every pair $(q, A) \in Q \times \Sigma$.

If the equality holds in these two conditions, the DBA is called *total*.

The following result is mentioned at the end of [PMC, page 186]; its proof is not too difficult.

Fact 21. *Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be DFA's, and $\mathcal{A}_{1,B}$ and $\mathcal{A}_{2,B}$ their counterparts as DBA's. If $\mathscr{L}(\mathcal{A}_1) = \mathscr{L}(\mathcal{A}_2)$, then $\mathscr{L}(\mathcal{A}_{1,B}) = \mathscr{L}(\mathcal{A}_{2,B})$.*

Two important comments regarding Fact 21:

- If we lift the restriction to *deterministic* automata, then Fact 21 does not hold.
  (This is readily shown using Fact 22 below.)
- The converse implication in Fact 21 is not true; a counter-example is part 2 in Example 20.

The following result is [PMC, Theorem 4.50, page 190], in sharp contrast with finite automata, where NFA's and DFA's are equally expressive.

Fact 22. *NBA's are more powerful than DBA's. Specifically, there does not exist a DBA $\mathcal{A}_B$ such that $\mathscr{L}(\mathcal{A}_B) = \mathscr{L}((A + B)^* B^\omega)$ (the proof is not trivial, given in [PMC, pages 190-191]).*