Howard Straubing

Finite Automata, Formal Logic, and Circuit Complexity
To my parents
Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Preface</td>
<td>vii</td>
</tr>
<tr>
<td>I</td>
<td>Mathematical Preliminaries</td>
<td>1</td>
</tr>
<tr>
<td>I.1</td>
<td>Words and Languages</td>
<td>1</td>
</tr>
<tr>
<td>I.2</td>
<td>Automata and Regular Languages</td>
<td>2</td>
</tr>
<tr>
<td>I.3</td>
<td>Semigroups and Homomorphisms</td>
<td>6</td>
</tr>
<tr>
<td>II</td>
<td>Formal Languages and Formal Logic</td>
<td>9</td>
</tr>
<tr>
<td>II.1</td>
<td>Examples</td>
<td>9</td>
</tr>
<tr>
<td>II.2</td>
<td>Definitions</td>
<td>11</td>
</tr>
<tr>
<td>III</td>
<td>Finite Automata</td>
<td>21</td>
</tr>
<tr>
<td>III.1</td>
<td>Monadic Second-Order Sentences and Regular Languages</td>
<td>21</td>
</tr>
<tr>
<td>III.2</td>
<td>Regular Numerical Predicates</td>
<td>25</td>
</tr>
<tr>
<td>III.3</td>
<td>Infinite Words and Decidable Theories</td>
<td>28</td>
</tr>
<tr>
<td>IV</td>
<td>Model-Theoretic Games</td>
<td>39</td>
</tr>
<tr>
<td>IV.1</td>
<td>The Ehrenfeucht-Fraïssé Game</td>
<td>39</td>
</tr>
<tr>
<td>IV.2</td>
<td>Application to $FO[&lt;]$</td>
<td>44</td>
</tr>
<tr>
<td>IV.3</td>
<td>Application to $FO[+1]$</td>
<td>46</td>
</tr>
<tr>
<td>V</td>
<td>Finite Semigroups</td>
<td>53</td>
</tr>
<tr>
<td>V.1</td>
<td>The Syntactic Monoid</td>
<td>53</td>
</tr>
<tr>
<td>V.2</td>
<td>Calculation of the Syntactic Monoid</td>
<td>57</td>
</tr>
<tr>
<td>V.3</td>
<td>Application to $FO[&lt;]$</td>
<td>59</td>
</tr>
<tr>
<td>V.4</td>
<td>Semidirect Products</td>
<td>61</td>
</tr>
<tr>
<td>V.5</td>
<td>Categories and Path Conditions</td>
<td>66</td>
</tr>
<tr>
<td>V.6</td>
<td>Pseudovarieties</td>
<td>71</td>
</tr>
<tr>
<td>CONTENTS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>VI First-Order Logic</td>
<td>79</td>
<td></td>
</tr>
<tr>
<td>VI.1 Characterization of $FO[\leq]$</td>
<td>79</td>
<td></td>
</tr>
<tr>
<td>VI.2 A Hierarchy in $FO[\leq]$</td>
<td>84</td>
<td></td>
</tr>
<tr>
<td>VI.3 Another Characterization of $FO[+1]$</td>
<td>89</td>
<td></td>
</tr>
<tr>
<td>VI.4 Sentences with Regular Numerical Predicates</td>
<td>93</td>
<td></td>
</tr>
<tr>
<td>VII Modular Quantifiers</td>
<td>99</td>
<td></td>
</tr>
<tr>
<td>VII.1 Definition and Examples</td>
<td>99</td>
<td></td>
</tr>
<tr>
<td>VII.2 Languages in $(FO + MOD(P))[\leq]$</td>
<td>102</td>
<td></td>
</tr>
<tr>
<td>VII.3 Languages in $(FO + MOD)[+1]$</td>
<td>107</td>
<td></td>
</tr>
<tr>
<td>VII.4 Languages in $(FO + MOD)[Reg]$</td>
<td>117</td>
<td></td>
</tr>
<tr>
<td>VII.5 Summary</td>
<td>121</td>
<td></td>
</tr>
<tr>
<td>VIII Circuit Complexity</td>
<td>127</td>
<td></td>
</tr>
<tr>
<td>VIII.1 Examples of Circuits</td>
<td>127</td>
<td></td>
</tr>
<tr>
<td>VIII.2 Circuits and Circuit Complexity Classes</td>
<td>135</td>
<td></td>
</tr>
<tr>
<td>VIII.3 Lower Bounds</td>
<td>143</td>
<td></td>
</tr>
<tr>
<td>IX Regular Languages and Circuit Complexity</td>
<td>155</td>
<td></td>
</tr>
<tr>
<td>IX.1 Regular Languages in $NC^1$</td>
<td>155</td>
<td></td>
</tr>
<tr>
<td>IX.2 Formulas with Arbitrary Numerical Predicates</td>
<td>161</td>
<td></td>
</tr>
<tr>
<td>IX.3 Regular Languages and Nonregular Numerical Predicates</td>
<td>166</td>
<td></td>
</tr>
<tr>
<td>IX.4 Special Cases of the Central Conjecture</td>
<td>170</td>
<td></td>
</tr>
<tr>
<td>A Proof of the Krohn-Rhodes Theorem</td>
<td>179</td>
<td></td>
</tr>
<tr>
<td>A.1 Ideal Structure of Finite Semigroups</td>
<td>179</td>
<td></td>
</tr>
<tr>
<td>A.2 Transformation Semigroups and Wreath Products</td>
<td>182</td>
<td></td>
</tr>
<tr>
<td>A.3 Decomposition of Transformation Semigroups</td>
<td>187</td>
<td></td>
</tr>
<tr>
<td>A.4 Completion of the Proof</td>
<td>191</td>
<td></td>
</tr>
<tr>
<td>B Proofs of the Category Theorems</td>
<td>199</td>
<td></td>
</tr>
<tr>
<td>B.1 Proof of Theorem V.5.2</td>
<td>199</td>
<td></td>
</tr>
<tr>
<td>B.2 Aperiodic and Group Categories</td>
<td>203</td>
<td></td>
</tr>
<tr>
<td>B.3 Proof of Theorem V.5.5</td>
<td>205</td>
<td></td>
</tr>
<tr>
<td>Bibliography</td>
<td>217</td>
<td></td>
</tr>
<tr>
<td>Index</td>
<td>223</td>
<td></td>
</tr>
</tbody>
</table>
Preface

The study of the connections between mathematical automata and formal logic is as old as theoretical computer science itself. In the founding paper of the subject, published in 1936, Turing showed how to describe the behavior of a universal computing machine with a formula of first-order predicate logic, and thereby concluded that there is no algorithm for deciding the validity of sentences in this logic. Research on the logical aspects of the theory of finite-state automata, which is the subject of this book, began in the early 1960's with the work of J. Richard Büchi on monadic second-order logic.

Büchi's investigations were extended in several directions. One of these, explored by McNaughton and Papert in their 1971 monograph Counter-free Automata, was the characterization of automata that admit first-order behavioral descriptions, in terms of the semigroup-theoretic approach to automata that had recently been developed in the work of Krohn and Rhodes and of Schützenberger. In the more than twenty years that have passed since the appearance of McNaughton and Papert's book, the underlying semigroup theory has grown enormously, permitting a considerable extension of their results. During the same period, however, fundamental investigations in the theory of finite automata by and large fell out of fashion in the theoretical computer science community, which moved to other concerns. The recent work of Barrington and Thérien, establishing parallels with the complexity theory of small-depth circuits, has stimulated a renewed interest in the algebraic theory of automata, and shown that this theory has an important role to play in the study of computational complexity.

The present book, intended for researchers and advanced students in theoretical computer science and mathematics, is situated at the juncture of automata theory, logic, semigroup theory and computational complexity. The first seven chapters are devoted to the algebraic characterization of the regular languages definable in many different
logical theories, obtained by varying both the kinds of quantification and the atomic formulas that are admitted. This includes, to be sure, the results of Büchi and of McNaughton and Papert, as well as more recent developments that are scattered throughout research journals and conference proceedings, but that have never before been given a coherent treatment in book form. The reader who wishes to see what this is all about should take a look at the two tables at the end of Chapter VII, where most of the important results of this first part of the book are summarized. Chapter VIII, which has a quite different character from the other chapters, is a brief account of the complexity theory of small-depth families of boolean circuits. In Chapter IX all the threads are tied together: It is shown that questions about the structure of complexity classes of small-depth circuits are precisely equivalent to questions about the definability of regular languages in various versions of first-order logic. The book ends with a conjecture; the effort that I put into writing it will be more than amply rewarded if some reader finds in my book the beginnings of an answer to the questions that it poses.

I have kept the focus of this book relatively narrow; any temptation that I might have had to write an encyclopedic work was more than offset by my desire to finish in a reasonable amount of time. I have thus concentrated on automata over finite words and on the interpretation of logical formulas in finite word models, leaving aside considerations of automata on trees, graphs, infinite strings, etc. The one exception that I made to this rule is the presentation, in Chapter III, of Büchi's theorem on the equivalence of monadic second-order logic and nondeterministic automata on infinite words. I have written nothing about temporal logic, and very little about operations on regular languages or uniform circuit families, all worthy subjects that one might expect to find in a book such as this. Small-depth circuits are introduced as devices for efficiently performing the fundamental arithmetic computations, and their role in the wider context of computational complexity theory is described only briefly. The citations at the end of each chapter should provide a useful guide for the reader who wishes to pursue the topics that are merely mentioned in passing in the text.

On the other hand, I have tried to give a complete treatment of the topics that are included, so that—apart from the application of well-known results from other parts of mathematics—I never appeal to a theorem that is not proved in this book.
Much of the technical content of this work is concerned with proving necessary conditions for a property of words to be expressible in a particular logical formalism. Two general techniques for accomplishing this are presented. The first, the method of Ehrenfeucht-Fraissé games, is described in Chapter IV. This method is very pretty, completely general, and easy to grasp. However it is the second method, the semigroup-theoretic analysis of logical formulas, that dominates the book. While this technique apparently has a more limited range of application than the model-theoretic games, for the problems considered here it is more powerful and always gives a more satisfying result. The drawback to the algebraic method is that readers who are unfamiliar with it often find it obscure, and are frequently more comfortable with direct syntactic arguments, even if these turn out to be extremely complicated. I make no apologies, though, for the inclusion of semigroup theory. Apart from being a powerful tool of analysis, it is very much a part of the story that this book has to tell, as it reveals a deep mathematical structure in both the automata classes and circuit complexity classes considered here.

Aware of the difficulties that this approach may pose, I have placed the proofs of the hardest theorems—the Krohn-Rhodes theorem and some of the results on finite categories—in the two appendices. The reader should be able to learn what these theorems say and how to apply them without having to attend to the details of their proofs.

Several exercises appear at the end of each chapter. Some of these are entirely routine, while others... Unable to avoid the situation when I very much wished to include a result but was too lazy to write its proof in detail, I shamelessly indulged in the old math teacher’s dodge, and left it as an exercise for the reader.

Most of this book was written during a year-long sabbatical. I am grateful to Boston College for its support throughout this extended leave of absence, and to Pierre McKenzie at the University of Montreal, Denis Thérien at McGill University, Aldo De Luca at the University of Rome, and Wolfgang Thomas at the University of Kiel for their hospitality during my visits to them. Much of what appears in this book grew out of my research conducted over a period of many years, and I am fortunate in having had many able collaborators: Kevin Compton, Neil Immerman, Jean-Eric Pin, Wolfgang Thomas, Pascal Weil, and especially David Mix Barrington and Denis Thérien. I have also benefited from many conversations with Peter Clote, Peter Kugel, Stu-
art Margolis, Pierre McKenzie, Pierre Péladeau, Andreas Pothoff, Bret Tilson and Thomas Wilke. For the past several years, I have received generous research support from the National Science Foundation.

I owe a particular debt to John Rhodes, who was my thesis adviser at Berkeley fifteen years ago, and who has continued to be a most valued colleague. He has probably contributed more than any other individual to the development of the algebraic machinery that drives this book, which bears the mark of his influence throughout its length.

Finally, I must express my gratitude to Ron Book, who enthusiastically encouraged me to undertake this project, and to the staff of Birkhäuser Boston for their assistance in its completion. Most of all, to Emma, Nicolas and Nancy, who put up with my frequent absences, both physical and mental, while I wrote an obscure math book that none of them is likely to read.

Howard Straubing
Boston, Massachusetts
August, 1993
Chapter I

Mathematical Preliminaries

In this book we frequently use elementary group theory. We apply Ramsey’s Theorem a couple of times (in Sections III.3 and IX.3), and use a bit of basic number theory and the fundamentals of finite fields in Chapter VIII. The Prime Number Theorem and the Central Limit Theorem are brought on stage for some very brief appearances, but the book can easily be read without knowing a thing about them; relatively little is lost by skipping the proofs in which they appear.

For the most part, however, the mathematics we use here concerns formal logic, formal languages, finite automata, and finite semigroups. Our particular approach to logic is the subject of the next chapter; in this one we present the basics of what the reader will need to know about the other three topics.

I.1 Words and Languages

Throughout this book the letter $A$ denotes a finite set of symbols, which we call an alphabet; we call the elements of $A$ letters. We use this symbol generically, so that any statement involving $A$ is to be understood as beginning with an implicit “For every finite alphabet $A$...”. When we need, as occasionally happens, to discuss several different finite alphabets at once, we will introduce new names, and be explicit about which alphabet we mean.

A word over $A$ is a finite sequence of elements of $A$. We use word and string interchangeably. We write a word as

$$a_1a_2\cdots a_k,$$
where \( a_1, \ldots, a_k \in A \) are the successive elements of the sequence. In our notation we rarely need to be careful about the distinction between a letter \( a \) of \( A \) and the sequence of length 1 whose only element is \( a \), so we write them identically. If \( w = a_1 \cdots a_k \) is a word, then we denote by \(|w|\) the length \( k \) of the sequence. If \( a \in A \), then \(|w|_a\) denotes the number of times \( a \) occurs in \( w \).

We allow the empty sequence as a word, called the empty word, which we denote \( 1 \). Other symbols (\( \lambda \), or \( \Lambda \), or \( \epsilon \)) are more commonly used to denote the empty word, but our notation is more consistent with the algebraic approach of the book. Observe that \(|1| = 0\).

Two sequences can be concatenated to form a longer sequence. If \( u \) and \( v \) are words then we denote the concatenation by \( uv \), or sometimes \( u \cdot v \). Obviously, for all words \( u \) and \( v \) and all \( a \in A \) we have

\[
|uv| = |u| + |v|,
\]

and

\[
|uv|_a = |u|_a + |v|_a.
\]

We also have

\[ u \cdot 1 = 1 \cdot u = u \]

for all words \( u \). We write \( u^2 \) for \( uu \), \( u^3 \) for \( uuu \), and even at times \( u^0 \) for 1.

If \( v \) and \( w \) are words, then \( v \) is a prefix of \( w \) if there exists a word \( x \) such that \( w = vx \), \( v \) is a suffix of \( w \) if there exists a word \( x \) such that \( w = xv \), and \( v \) is a factor of \( w \) if there exist words \( x \) and \( y \) such that \( w = xyv \). The set of all words over \( A \) is denoted \( A^* \); the set of all nonempty words over \( A \) is denoted \( A^+ \). A subset of \( A^* \) is called a language.

### I.2 Automata and Regular Languages

A nondeterministic finite automaton is a quadruple

\[ M = (Q, i, F, E), \]

where \( Q \) is a finite set, \( i \in Q \), \( F \subseteq Q \), and and \( E \subseteq Q \times A \times Q \). We call \( Q \) the set of states of the automaton, \( i \) the initial state, \( F \) the set of final states, and \( E \) the set of edges. A sequence of edges

\[(q_0, a_1, q_1)(q_1, a_2, q_2) \cdots (q_{k-1}, a_k, q_k)\]
is called a path in $\mathcal{M}$ from $q_0$ to $q_k$, and the word

$$a_1 \cdots a_k$$

is called the label of the path. We make the convention that for all $q \in Q$ there is a path from $q$ to $q$ whose label is 1.

A word $w \in A^*$ is accepted by the automaton if there is a path labelled $w$ from $i$ to some $q \in F$. The set of all words accepted by $\mathcal{M}$ is called the language recognized by $\mathcal{M}$. A language $L \subseteq A^*$ is said to be regular if it is recognized by some nondeterministic finite automaton.

As our terminology indicates, we view an automaton as a directed graph whose vertices are the states and whose edges are labelled by elements of $A$. We will use this graphical interpretation in the figures in this book: We indicate the vertices by circles, and the edges by labelled arrows. We will use double circles to indicate which states are the final states, and indicate the initial state with an entering arrow.

A deterministic finite automaton is a quadruple

$$\mathcal{M} = (Q, i, F, \lambda),$$

where $Q$, $i$, and $F$ are as above, and $\lambda$ is a map from $Q \times A$ into $A$ (called the next state function.) If $q \in Q$ and $a \in A$ we will usually write $qa$ or $q \cdot a$ in place of $\lambda(q, a)$. We then define $qw$ for $q \in A$ and $w \in A^*$ by induction on $|w|$ as follows:

$$q \cdot 1 = q,$$

$$q \cdot (wa) = (qw) \cdot a,$$

for all $a \in A$. It is easy to see that for all $w_1, w_2 \in A^*$,

$$q(w_1w_2) = (qw_1)w_2.$$

A word $w$ is accepted by $\mathcal{M}$ if and only if $iw \in F$. We again define the language recognized by $\mathcal{M}$ as the set of all words that $\mathcal{M}$ accepts. The deterministic automaton is the same thing as the nondeterministic automaton

$$(Q, i, F, \{(q, a, qa) : q \in A, a \in A\}),$$

and thus the language it recognizes is regular. Observe that in this view, the deterministic automata are characterized by the property that for all $q \in Q$ and $w \in A^*$, there is exactly one path labelled $w$ that begins at $q$. 
Conversely, every regular language is recognized by a deterministic finite automaton, for if $L$ is recognized by a nondeterministic automaton $(Q, i, F, E)$, then it is recognized by the deterministic automaton

$$(2^Q, \{i\}, \{X \subseteq Q : X \cap F \neq \emptyset\}, \lambda),$$

where for all $X \subseteq Q$, $a \in A$, $X \cdot a = \bigcup_{p \in X} \{q : (p, a, q) \in E\}$.

Thus from the standpoint of the class of languages recognized by such devices, there is no difference between deterministic and nondeterministic finite automata.

Let $M = (Q, i, F, \lambda)$ be a deterministic finite automaton, and let $L \subseteq A^*$ be the language it recognizes. We set

$$Q' = \{iw : w \in A^*\},$$

and define an equivalence relation $\sim$ on $Q'$ by letting

$$q_1 \sim q_2$$

if and only if

$$\{w : q_1w \in F\} = \{w : q_2w \in F\}.$$

It is easy to show that if $q_1 \sim q_2$ and $a \in A$, then $q_1a \sim q_2a$. We thus have a well-defined map

$$\lambda : Q' \times A \to Q'$$

defined by setting

$$\lambda([q], a) = [qa],$$

where $[q]$ denotes the $\sim$-class of $q \in Q$. The deterministic finite automaton

$$\overline{M} = (Q'/\sim, [i], \{[t] : t \in F\}, \lambda)$$

recognizes $L$. Moreover, it is possible to show that the structure of $\overline{M}$ depends only on $L$, and not on which one of the infinitely many automata that recognize $L$ we take for $M$. That is, if $M'$ also recognizes $L$, then $\overline{M}$ and $\overline{M'}$ are isomorphic. $\overline{M}$ is thus called the minimal automaton of $L$.

Let $L_1, L_2 \subseteq A^*$ be a regular language. Then the following languages are also regular:
1.2. AUTOMATA AND REGULAR LANGUAGES

$L_1 \cup L_2$

$A^* \setminus L_1$

$L_1 \cap L_2$

$L_1L_2 = \{w_1w_2 : w_1 \in L_1, w_2 \in L_2\}$

$L_1^* = \{1\} \cup L_1 \cup L_1L_1 \cup L_1L_1L_1 \cup \ldots$

In fact, the regular languages constitute the smallest class of languages that contains all the finite subsets of $A^*$ and is closed under the operations

$$(L_1, L_2) \mapsto L_1 \cup L_2,$$

$$(L_1, L_2) \mapsto L_1L_2,$$

$L \mapsto L^*$.

This fact is called Kleene's Theorem. We can thus specify regular languages by expressions containing the letters of $A$ and these three operations. For example,

$$(aa \cup ab \cup ba \cup bb)^*$$

denotes the set of words of even length over the alphabet $\{a, b\}$. (One often sees '∪' written in place of '∪'.) These are called regular expressions.

It is worthwhile to keep in mind a few examples of languages that are not regular. Our standard example of this is

$L = \{a^n b^n : n \geq 0\}$

over the alphabet $A = \{a, b\}$. If this were regular, it would be recognized by a deterministic finite automaton $M = (Q, i, F, \lambda)$. There would thus exist $0 < m < n$ such that $q = ia^m = ia^n$. This implies

$ia^mb^m = qb^m = ia^mb^m \in F$,

so that $a^n b^n \in L$, a contradiction. We can readily conclude from this that some other languages are not regular. For example, the set of all $w \in \{a, b\}^*$ such that $|w|_a = |w|_b$ is not regular, because its intersection with the regular language $a^*b^*$ is the nonregular language $L$ defined above.
I.3 Semigroups and Homomorphisms

A semigroup is a set with an associative multiplication. Let $S$ and $T$ be semigroups. A map $\phi : S \rightarrow T$ is a homomorphism if for all $s_1, s_2 \in A$,

$$\phi(s_1 s_2) = \phi(s_1) \phi(s_2).$$

The equivalence relation $\equiv_\phi$ on $S$, defined by setting

$$s \equiv_\phi s'$$

if and only if

$$\phi(s) = \phi(s'),$$

is compatible with the multiplication in $S$, that is, if

$$s_1 \equiv_\phi t_1 \quad \text{and} \quad s_2 \equiv_\phi t_2,$$

then

$$s_1 s_2 \equiv_\phi t_1 t_2.$$

Such an equivalence relation is said to be a congruence on $S$. Conversely, if $S$ is a semigroup and $\equiv$ is a congruence on $S$, then there is a well-defined multiplication on the quotient set $S/ \equiv$, given by

$$[s_1][s_2] = [s_1 s_2],$$

where $[s]$ denotes the $\equiv$-class of $s \in S$. Thus $S/ \equiv$ is a semigroup, called the quotient semigroup of $S$ by $\equiv$. The projection map from $S$ onto $S/ \equiv$, which maps each $s \in S$ to its equivalence class, is a homomorphism.

Let $\phi : S \rightarrow T_1$, $\psi : S \rightarrow T_2$ be homomorphisms. If for all $s_1, s_2 \in S$,

$$\phi(s_1) = \phi(s_2) \implies \psi(s_1) = \psi(s_2),$$

then we say that $\psi$ factors through $\phi$, because in such an instance there is a homomorphism $\theta : \phi(S) \rightarrow T_2$ such that $\theta \circ \phi = \psi$.

An element $e$ of a semigroup $S$ is idempotent if $e^2 = e$. In this book we are mostly interested in finite semigroups, and finite semigroups always contain idempotents. In fact, if $S$ is a finite semigroup and $s \in S$, then there exists $k > 0$ such that $s^k$ is idempotent. To see why this is so, observe that the sequence

$$s, s^2, s^3, \ldots$$
contains only finitely many distinct elements. Thus there exist \( p, q > 0 \)
such that \( s^p = s^{p+r} \). Choose \( r \geq 0 \) so that \( p + r \equiv 0 \) (mod \( q \)). Then for
some \( m \geq 0 \) we have
\[
(s^{p+r})^2 = s^{p+mq+r} = s^{p+r},
\]
so that \( s^{p+r} \) is idempotent.

A **monoid** is a semigroup with an identity element; we denote the
identity element by \( 1 \). If \( M \) and \( N \) are monoids, then a map \( \phi : M \to N \)
is a **monoid homomorphism** if \( \phi \) is a homomorphism of semigroups, and
maps the identity of \( M \) to that of \( N \). When we are discussing monoids,
"homomorphism" will always be used to mean a homomorphism of
monoids.

A group, of course, is a special kind of monoid. Groups have the
property that the congruences are exactly the relations of coset equiv-
ance with respect to the normal subgroups.

If \( A \) is a finite alphabet, then \( A^+ \) is a semigroup, with concatenation
of words as multiplication. \( A^+ \) is the **free semigroup** with basis \( A \). This
means that for every semigroup \( S \), if \( \phi : A \to S \) is a map then there ex-
ists a unique homomorphism from \( A^+ \) into \( S \) that extends \( \phi \). Similarly,
\( A^* \) is the **free monoid** generated by \( A \), for if \( M \) is a monoid, then every
map \( \phi : A \to M \) has a unique extension to a monoid homomorphism
from \( A^* \) into \( M \).

We mention one last closure property of regular languages: Let \( A \)
and \( B \) be finite alphabets, and let \( \phi : A^* \to B^* \) be a homomorphism.
If \( L \subseteq A^* \) is a regular language, then \( \phi(L) \subseteq B^* \) is a regular language.

### Chapter Notes

Mathematical automata were introduced by Turing [68]. The Tur-
ing machine manipulates an unbounded storage tape, and thus is not,
strictly speaking, a finite-state device. Finite-state automata in the
narrow sense of the term (essentially the deterministic finite automata
defined here) began to be studied in the 1950's, motivated in part by
a practical interest (the design of sequential logic circuits) and a more
speculative one (the modeling of human neural activity). The mini-
mization of deterministic finite automata is due to Huffman [32], and
the equivalence of automata and regular expressions to Kleene [34].
The fundamentals of the theory of automata are now presented in a

The term "regular language" is a bit unfortunate. Papers influenced by Eilenberg's monograph often use either the term "recognizable language", which refers to the behavior of automata, or "rational language", which refers to important analogies between regular expressions and rational power series. (In fact, Eilenberg defines rational and recognizable subsets of arbitrary monoids; the two notions do not, in general, coincide.) This terminology, while better motivated, never really caught on, and "regular language" is used almost universally.

The earliest systematic study of semigroup theory is probably in Suschkewitsch [59]. All the structure theory of semigroups that we need here is presented, with complete proofs, in Appendix A. Other references that present the relevant parts of semigroup theory are Arbib [2], Lallement [37], and Pin [46].
Chapter II

Formal Languages and Formal Logic

Throughout this book we use sentences of formal logic to describe properties of words over a finite alphabet \( A \). A sentence will thus define a language \( L \subseteq A^* \); \( L \) is the set of all words that have the property described by the sentence.

II.1 Examples

In Section II.2 we will give all the formal definitions. The examples here are intended to illustrate the general idea.

II.1.a Example.

\[ \exists x \exists y (x < y). \]

*How to read the formula.* The letters \( x, y \) denote positions in a word; that is, integers in the range \( 1, \ldots, n \), where \( n \) is the length of the word. Thus the formula says that there exist at least two distinct positions in the word. The language defined is

\[ L = \{ w \in A^* : |w| \geq 2 \}. \]

II.1.b Example. Let \( A = \{ a, b \} \). Consider the formula

\[ \exists x \exists y (\forall z (z \geq x) \land Q_a x \land \forall z (z \leq y) \land Q_b y). \]
How to read the formula. The new elements are the symbols $Q_a$ and $Q_b$. $Q_ax$ means "the $x^{th}$ letter of the word is $a$". The formula $\forall z(z \geq x)$ tells us that $x$ is the first position of the word. Thus the formula says, "the first letter is $a$ and the last letter is $b$", so the language defined by this formula is the regular language $aA^*b$.

II.1.c Example.

$$\exists X(\forall z(z \geq x) \rightarrow X(x)) \land \forall x(\forall z(z \leq x) \rightarrow \neg X(x)) \land \forall x\forall y(((x < y) \land \forall z(x + z \geq y)) \rightarrow (X(x) \leftrightarrow \neg X(y))).$$

How to read the formula. The upper-case letter $X$ represents a property or a predicate of the position in the word. In this case the predicate takes a single argument—it's a monadic predicate. One can also say that $X$ represents a set of positions, namely the set of positions $x$ that have the property. The formula thus says, "there exists a set $X$ of positions such that...".

We already know how to interpret the rest of the formula. The first clause says that the first position belongs to $X$, the second says that the last position does not belong to $X$, and the third says if $x$ and $y$ are two consecutive positions, then one of these positions belongs to $X$ and the other does not. It follows that the formula defines the set of strings of even length.

Observe that the formula in the preceding example is satisfied by the empty word. In fact every formula of the form $\forall x \phi$, or $\exists x \forall x \phi$, is satisfied by the empty word. The empty word has even length, so our interpretation of this sentence as defining the set of words of even length is still correct.

A formula in which all the quantifiers act on individual positions is said to be a first-order formula. If we allow quantification over relations on positions as well as individual positions we have a second-order formula. The formula in Example II.1.c is a monadic second-order formula.
II.2. DEFINITIONS

II.2 Definitions

First-order formulas

are built from variables, numerical predicates, and atomic formulas.

A variable is one of the symbols

\[ x, y, z, x_1, x_2, \ldots, y_1, y_2, \ldots, z_1, z_2, \ldots. \]

These are names of specific variables, but we shall also use them informally as generic variable names. For example, we will write “a formula of the form \( x < y \)” to mean a formula of the form \( v_1 < v_2 \), where \( v_1 \) and \( v_2 \) are variables.

A numerical predicate is one of the symbols

\[ R^i_j, (i > 0, j \geq 0). \]

A bit later we will describe how these symbols are interpreted. From an informal point of view, a numerical predicate represents a relation on the positions in a word, like “\( x < y \)” in the examples in the preceding section, that does not depend on the letters that appear in these positions.

There are two types of atomic formulas. If \( x \) is a variable and \( a \in A \), then

\[ Q_a x \]

is an atomic formula.

If \( x_1, \ldots, x_j \) are variables and \( i > 0 \), then

\[ R^i_j(x_1, \ldots, x_j) \]

is an atomic formula.

Formulas are now defined recursively, according to the following rules:

Every atomic formula is a formula.

If \( \phi \) and \( \psi \) are formulas, then

\[ (\phi \land \psi) \]

is a formula.
12 CHAPTER II. FORMAL LANGUAGES AND FORMAL LOGIC

If $\phi$ is a formula, then

$$\neg \phi$$

is a formula.

If $\phi$ is a formula and $x$ is a variable, then

$$\exists x \phi$$

is a formula.

An occurrence of a variable $x$ in a formula is said to be free if it is not quantified; that is, if the occurrence is not in a subformula to which $\exists x$ has been applied. Otherwise, the occurrence is bound.

In the formula below, the free occurrences are in bold face.

$$\exists x (R^2_1(x, z) \land \neg R^2_2(x, y)) \land \exists y R^1_1(y).$$

To define monadic second-order formulas, we add several rules to those given above:

$$X, Y, Z, X_1, \ldots, Y_1, \ldots, Z_1, \ldots$$

are second-order variables.

If $X$ is a second-order variable and $x$ is a first-order variable, then

$$X(x)$$

is an atomic formula.

If $\phi$ is a formula and $X$ is a second-order variable, then

$$\exists X \phi$$

is a formula.
II.2. DEFINITIONS

In this book we consider only monadic second-order formulas. Of course one can define second-order variables of arbitrary arity, and thus obtain the general definition of second-order formulas. (See Exercise 3.)

We now define the semantics of first-order formulas. We will write

\[ w \models \phi \]

to mean, roughly speaking, that the formula \( \phi \) says something true about the word \( w \). This requires that we have some interpretation of the meaning of the numerical predicate symbols that occur in \( \phi \). We first fix this interpretation: A \( k \)-ary numerical relation associates to each \( n \geq 0 \) a subset of \( \{1, \ldots, n\}^k \).

An interpretation \( I \) associates to each numerical predicate \( R^j_i \) a \( j \)-ary numerical relation. For example, the relation \( < \) associates to each \( n \geq 0 \) the usual ordering on \( \{1, \ldots, n\} \). Of course, in this case, the relations for different values of \( n \) are compatible, and one can simply talk about a \( k \)-ary relation on the positive integers. Not every numerical relation has this property, however. For example, we interpret \( \text{last}(x) \) to mean that \( x \) is the last position in a string. Thus the unary numerical relation \( \text{last} \) associates to each positive integer \( n \) the one-element set \( \{n\} \), so whether \( \text{last}(x) \) is true depends on the length of the string in which we interpret it.

In practice, it is usually unnecessary and a bit annoying to have to keep in mind the distinction between a numerical predicate, which is just a symbol with an associated arity, and the numerical relation by which it is interpreted. Soon we will start being sloppy and confound the two. It is only in this chapter, where we need to present definitions that work, that we will be careful about the difference between the purely syntactic notion and the semantic one.

Generally speaking, the formulas that define properties of words are formulas that contain no free variables (sentences). For example,

\[ \exists x Q_aox \]

is supposed to mean "some letter of the word is \( a \)". But what property of words is defined by

\[ Q_aox? \]

Since sentences are built from formulas with free variables, we will need to assign some sort of meaning to formulas with free variables. In this book we view formulas as expressing properties of words over
an extended alphabet. To define this precisely, let $\mathcal{V}$ be a finite set of first-order variables. A $\mathcal{V}$-structure over $A$ is a word

$$(a_1, U_1) \cdots (a_r, U_r)$$

over the alphabet $A \times 2^\mathcal{V}$, such that

$$U_i \cap U_j = \emptyset,$$

if $i \neq j$, and

$$\bigcup_{i=1}^r U_i = \mathcal{V}.$$  

Let $w$ be a $\mathcal{V}$-structure, and suppose that $\phi$ is a first-order formula that satisfies the following conditions. (We will explain the reason for these conditions shortly.)

(i) If $x$ is a variable with a free occurrence in $\phi$, then $x \in \mathcal{V}$.

(ii) If $x$ is a variable with a bound occurrence in $\phi$, then $x \notin \mathcal{V}$.

(iii) No variable $x$ has bound occurrences in the scopes of two different quantifiers.

We define

$$w \models_\mathcal{I} \phi$$

(read $w$ is a model of $\phi$ under $\mathcal{I}$, or $w$ satisfies $\phi$ with respect to the interpretation $\mathcal{I}$) by induction on the construction of $\phi$:

$$w \models_\mathcal{I} Q_a x$$

if and only if $w$ contains a letter of the form $(a, S)$, where $x \in S$.

$$w \models_\mathcal{I} R^k_1(x_1, \ldots, x_k)$$

if and only if $P(j_1, \ldots, j_k)$, where $P$ is the $k$-ary relation on $\{1, \ldots, |w|\}$ associated to $R^k_1$ by $\mathcal{I}$, and $j_1, \ldots, j_k$ are the positions in $w$ where the variables $x_1, \ldots, x_k$, respectively, occur.

$$w \models_\mathcal{I} (\phi_1 \land \phi_2)$$
II.2. **DEFINITIONS**

If and only if \( w \models_I \phi_1 \) and \( w \models_I \phi_2 \).

\[ w \models_I \lnot \phi \]

if and only if \( w \) is not a model of \( \phi \) with respect to \( I \). Finally, let

\[ w = (a_1, S_1) \cdots (a_r, S_r). \]

Then

\[ w \models \exists x \phi \]

if and only if for some \( i, 1 \leq i \leq r, \)

\[ (a_1, S_1) \cdots (a_i, S_i \cup \{x\}) \cdots (a_r, S_r) \models_I \phi. \]

If \( \phi \) is a sentence; that is, if \( \phi \) has no free variables, then \( \phi \) can be interpreted in a word \( w \in A^* \), since such a word can be viewed as a \( \emptyset \)-structure. In this case we set

\[ L_\phi = \{ w \in A^* : w \models_I \phi \} \]

\( L_\phi \) is the **language defined by \( \phi \)**. More generally, if \( \phi \) is a formula with free variables in \( \mathcal{V} \) then we will denote by \( L_\phi \) the set of \( \mathcal{V} \)-structures that satisfy \( \phi \). This notion depends both on the interpretation \( I \) and on the set \( \mathcal{V} \) of free variables. For example, a formula in which the only variable with a free occurrence is \( x \) can be interpreted in structures over any set of free variables that contains \( x \), and this will lead to different sets \( L_\phi \). It will usually be clear from the context what set of free variables is intended.

Two formulas \( \phi \) and \( \psi \) with free variables in \( \mathcal{V} \) are said to be **equivalent** if \( L_\phi = L_\psi \).

We have said nothing about the universal quantifier \( \forall \). In our formalism

\[ \forall \mathcal{V} \psi \]

is simply an abbreviation of

\[ \lnot \exists \mathcal{V} \lnot \psi. \]

The other boolean operations \((\lor, \rightarrow, \leftrightarrow)\) can be defined in terms of \( \land \) and \( \lnot \) in the usual fashion. It should be noted that our definition of the semantics of formulas is a bit unusual. The interpretation of formulas with free variables in words over an extended alphabet will be
an indispensable tool in the subsequent chapters, but it does present some inconveniences. Consider, for example, the sentence

$$\exists x \exists y (x < y \land \exists x (y < x)).$$

The use of the variable $x$ in two different quantifiers is perfectly reasonable, and it is quite clear which of the two occurrences of $x$ in atomic formulas is associated with which quantifier. However the formalism adopted here does not allow us to interpret this formula in a structure; we do not have the means to define

$$(a, \{ x \})(b, \{ y \})(a, \emptyset) \models \exists x (y < x).$$

There are several ways to get around this problem. We have adopted the expedient, if inelegant, solution of insisting that our definition of satisfaction applies only to formulas in which distinct quantifiers use distinct variables. If we confront a formula, like the one above, in which variables are re-used, we shall rewrite it by introducing new names for the bound variables, and interpret the resulting formula. Thus the sentence introduced above will be replaced by

$$\exists x \exists y (x < y \land \exists z (y < z)).$$

Now we do indeed have

$$(a, \{ x \})(b, \{ y \})(a, \emptyset) \models \exists z (y < z),$$

under the usual interpretation of $\prec$. The language defined by this sentence is the set of all words of length at least 3.

\textit{II.2.a Example.} Consider the unary numerical predicate $\theta(x)$ whose informal interpretation is "$x$ is even". That is, for each $n$ our interpretation assigns to the predicate symbol $\theta$ the subset of $\{1, \ldots, n\}$ consisting of the even elements. Like the numerical relation $\prec$, $\theta$ is actually independent of $n$, and thus defines a (in this case unary) relation on the positive integers. The sentence

$$\exists x \forall y (\neg (x < y) \land \theta(x))$$

defines the set of words of even positive length. This is clear from the informal interpretation. To see how the formal interpretation works, observe that a $\{ x, y \}$-structure will satisfy $\neg (x < y) \land \theta(x)$ if and only
II.2. DEFINITIONS

if $x$ occurs in an even-numbered position and $y$ occurs in the same position or to the left of this position. Thus the $\{x\}$-structures that satisfy

$$\forall y (\neg(x < y) \land \theta(x))$$

are precisely those of even length, with $x$ in the last position.

There is, however, a much simpler sentence that defines this same language. Consider the 0-ary numerical predicate $\eta$ that is true for structures of even length and false for structures of odd length. $\eta$ is a sentence, and is satisfied by $w \in A^*$ if and only if $|w|$ is even and positive.

We now define the interpretation of monadic second-order formulas. Let $\mathcal{V}_1$ be a finite set of first-order variables, and $\mathcal{V}_2$ a finite set of monadic second-order variables. A $(\mathcal{V}_1, \mathcal{V}_2)$-structure over $A$ is a word

$$w = (a_1, S_1, T_1) \cdots (a_n, S_n, T_n) \in (A \times 2^{\mathcal{V}_1} \times 2^{\mathcal{V}_2})^*$$

such that

$$(a_1, S_1) \cdots (a_n, S_n)$$

is a $\mathcal{V}_1$-structure. No restrictions are placed on the occurrences of the second-order variables in the structure. The definition of

$$w \models x \phi$$

is the same as for first-order formulas, with the addition of two new clauses:

$$w \models x X(x),$$

where $X$ is a second-order variable and $x$ is a first-order variable, if and only if $w$ contains a letter $(a_i, S_i, T_i)$, with $x \in S_i$ and $X \in T_i$. If $X$ is a second-order variable, then

$$w \models x \exists X \phi$$

if and only if there is a (possibly empty) set $J$ of positions in $w$ with the following property: The $(\mathcal{V}_1, \mathcal{V}_2)$-structure $w'$ formed by replacing each $(a_i, S_i, T_i)$, with $i \in J$, by $(a_i, S_i, T_i \cup \{X\})$ satisfies $\phi$.

The set $L_\phi$ of $(\mathcal{V}_1, \mathcal{V}_2)$-structures that satisfy a formula $\phi$ is defined as for first-order formulas.
Exercises

1. Describe the languages defined by the formulas

\[ \exists x (\forall z (z \geq x) \land Q_a x) \]

and

\[ \forall x (\forall z (z \geq x) \rightarrow Q_a x) \]

2. Write a sentence that defines the set of words over the alphabet \( \{a, b\} \) that contain an even number of occurrences of the letter \( a \).

3. A \( k \)-ary second-order variable denotes a \( k \)-ary relation on the set of positions in a word. Write second-order sentences, using both 1-ary (i.e., monadic) and 2-ary second-order variables, that define the languages (i) consisting of all words of the form \( a^nb^n \), \( n > 0 \); and (ii) consisting of all words over the alphabet \( \{a, b\} \) in which the number of occurrences of \( a \) is equal to the number of occurrences of \( b \). It will follow from our results in Section III.1 that these languages cannot be defined if one is allowed only monadic second-order quantifiers.

4. (a) Let \( \psi \) be a first-order formula, and suppose that the free variables that appear in \( \psi \) are \( x_1, \ldots, x_k \). Let \( y \) be a variable that does not appear in \( \psi \). We can interpret \( \psi \) in either \( \{x_1, \ldots, x_k\}\)-structures, or in \( \{x_1, \ldots, x_k, y\}\)-structures. Let \( w \) be a \( \{x_1, \ldots, x_k\}\)-structure. Show that \( w \models \phi \) if and only if \( w' \models \phi \), where \( w' \) is any one of the \( \{x_1, \ldots, x_k, y\}\)-structures formed by adjoining \( y \) to one of the positions of \( w \).

(b) Let \( \phi \) and \( \psi \) be first-order formulas, and suppose that the variable \( y \) does not occur in \( \psi \). Show, using the result of the preceding exercise, that \( \exists y (\phi \land \psi) \) is equivalent to \( \exists y \phi \land \psi \), and that \( \exists y (\phi \lor \psi) \) is equivalent to \( \exists y \phi \lor \psi \).

(c) Conclude from this that every first-order formula is equivalent to one which consists of a sequence of universal and existential quantifiers, followed by a quantifier-free formula. (Such a formula is said to be in prefix form. One way to measure the complexity of a formula is to count the number of quantifiers in an equivalent formula in prefix form. Another way is to count the number of maximal blocks of quantifiers, where a block is a sequence that contains only existential quantifiers or only universal quantifiers. A formula in prefix form with \( k \) such blocks...
II.2. DEFINITIONS

is said to be a $\Sigma_k$-formula if the leftmost block contains existential quantifiers, and a $\Pi_k$-formula if the leftmost block contains universal quantifiers.)
Chapter Notes

The part of mathematical logic devoted to the meaning and interpretation of logical formulas is called model theory. As we mentioned above, our treatment of these matters is a bit unorthodox and somewhat over-specialized: Since we are only interested in interpreting formulas in words, and since we do not use second-order variables of arity greater than 1, our present formalism is sufficient. Moreover, it has the advantage of allowing us to treat the structures in which we interpret formulas as being words over an extended finite alphabet. This idea comes from Perrin and Pin [45].

A more general treatment of some of the material from mathematical logic considered here can be found in any of a number of textbooks on logic, for example, Ebbinghaus, Flum, and Thomas [24].
Chapter III

Finite Automata

III.1 Monadic Second-Order Sentences and Regular Languages

The sentences in the examples in Section II.1 all define regular languages. This is no accident. If we restrict the available numerical predicates appropriately, then the language defined by a monadic second-order sentence is a regular language. Moreover, as we will prove in Theorem III.1.1 below, every regular language can be defined in this fashion. We will thus obtain a characterization of the regular languages in terms of logic.

We will consider monadic second-order sentences in which the only numerical predicates are \( R^1_1 \) and \( R^2_2 \), where \( R^1_1(x, y) \) is interpreted as \( x = y \), and \( R^2_2(x, y) \) is interpreted as \( y = x + 1 \). We will write “\( x = y \)” and “\( y = x + 1 \)” explicitly in our formulas, rather than use the symbols \( R^1_1 \) and \( R^2_2 \). We denote by \( SOM[+1] \) the family of all languages over \( A^* \) that are defined by such sentences.

We will also use \( SOM[+1] \) informally to refer to this logical apparatus. Thus we say ‘a formula of \( SOM[+1] \)’ to mean a monadic second-order formula with these particular numerical predicates. It will always be clear from the context whether we are talking about the family of languages in \( A^* \) or the collection of formulas.

III.1.1 Theorem. Let \( L \subseteq A^* \). \( L \in SOM[+1] \) if and only if \( L \) is a regular language.

Proof. First suppose that \( L \) is regular. We may assume that \( L \subseteq A^+ \); otherwise we use the argument below to produce a sentence that defines
the regular language $L \setminus \{1\}$ and take the disjunction of this sentence with $\forall x (x \neq x)$, which defines the set consisting of the empty string alone. Let

$$
\mathcal{M} = (\{q_0, \ldots, q_{k-1}\}, q_0, F, \mathcal{E})
$$

be a nondeterministic finite automaton that recognizes $L$. A word $w \in A^*$ belongs to $L$ if and only if there exist sets

$$X_0, \ldots, X_{k-1} \subseteq \{1, \ldots, |w|\}$$

that satisfy the following conditions:

(i)

$$
\bigcup_{i=0}^{k-1} X_i = \{1, \ldots, |w|\}.
$$

(ii) If $i \neq j$ then $X_i \cap X_j = \emptyset$.

(iii) $1 \in X_0$.

(iv) If $j \in X_i$, $j + 1 \in X_i$ and $a$ is the $i^{\text{th}}$ letter of $w$, then $(q_0, a, q_1) \in \mathcal{E}$.

(v) If $|w| \in X_j$ and $a$ is the last letter of $w$, then there exists $q \in F$ such that $(q_j, a, q) \in \mathcal{E}$.

To see that these properties characterize acceptance of $w$, suppose first that $w \in L$. Choose an accepting path labelled $w$ in the automaton, and define $i \in X_j$ if and only if the automaton, when following this accepting path, is in state $q_j$ after reading the first $i - 1$ letters of $w$. It is trivial to verify that the five conditions above are satisfied. Conversely, if the conditions (i)-(v) are satisfied then for each proper prefix $w'$ of $w$ one can construct, by induction on $|w'|$, a path labelled $w'$ that begins at $q_0$ and ends at $q_j$, where $|w'| + 1 \in X_j$. Condition (v) then gives an accepting path for $w$.

It now suffices to construct a sentence that states these five conditions. The sentence is

$$
\exists X_0 \cdots \exists X_{k-1} (\phi_1 \land \cdots \land \phi_5),
$$

where the formulas $\phi_i$ are:

$$
\phi_1 : \forall x \left[ \bigvee_{i=0}^{k-1} X_i(x) \right].
$$
\[ \phi_2 : \forall x \left( \bigwedge_{0 \leq i < j < k} \neg(X_i(x) \land X_j(x)) \right). \]

\[ \phi_3 : \forall x (\forall y(y \neq x + 1) \rightarrow X_0(x)). \]

\[ \phi_4 : \forall x \left( \forall y(y = x + 1 \rightarrow \bigwedge_{0 \leq i < j < k} \left(X_i(x) \land X_j(y) \rightarrow \bigvee_{S_{i,j}} Q_a x \right) \right), \]
where \( S_{i,j} = \{ a \in A : (q_i, a, q_j) \in E \} \).

\[ \phi_5 : \forall x \left( \forall y(y \neq x + 1) \rightarrow \bigwedge_{i=0}^{k-1} \left(X_i(x) \rightarrow \bigvee_{T_i} Q_a x \right) \right), \]
where \( T_i \) consists of all \( a \in A \) such that for some \( q \in F \), \( (q_i, a, q_j) \in E \).

For the converse, we will prove by induction on the construction of formulas that for any sets \( \mathcal{V}_1 \) and \( \mathcal{V}_2 \) of first- and second-order variables, and every formula \( \phi \) with free first-order variables in \( \mathcal{V}_1 \) and free second-order variables in \( \mathcal{V}_2 \), \( L_\phi \) is a regular language. The theorem is just the case \( \mathcal{V}_1 = \mathcal{V}_2 = \emptyset \) of this claim.

Let \( \mathcal{L} \) denote the set of all \( (\mathcal{V}_1, \mathcal{V}_2) \)-structures. It is easy to check with a finite automaton over the input alphabet \( A \times 2^{\mathcal{V}_1} \times 2^{\mathcal{V}_2} \) that each first-order variable in \( \mathcal{V}_1 \) occurs exactly once in an input string, so \( \mathcal{L} \) is itself a regular language. It is also easy to check with a finite automaton whether a particular first-order variable \( x \) occurs in a letter whose first component is \( a \). The intersection of the set of all such strings with \( \mathcal{L} \) is the set of all structures that satisfy \( Q_a x \). One can also check with a finite automaton whether the first-order variables \( x \) and \( y \) occur in consecutive letters, or in the same letter, and whether any letter has \( x \) in the second component and \( X \) in the third component. Thus the sets of structures that satisfy \( y = x + 1 \), \( x = y \), and \( X(x) \) are regular languages, so the claim is true for atomic formulas.

If the claim is true for formulas \( \phi \) and \( \psi \) then it is true for \( \phi \land \psi \) and \( \neg \phi \). Indeed,

\[ L_{\phi \land \psi} = L_\phi \cap L_\psi \cap \mathcal{L}, \]
and

\[ L_{\neg \phi} = \mathcal{L} \setminus L_\phi, \]
and the conclusion follows from closure properties of regular languages (see Section 1.2). Now suppose \( \phi \) has the form \( \exists x \psi \), and that the claim is true for \( \psi \), so that the set of \( (\mathcal{V}_1 \cup \{ x \}, \mathcal{V}_2) \)-structures that satisfy \( \psi \).
is a regular language. Let $A = (Q, q_0, F, \mathcal{E})$ be a finite automaton that recognizes this regular language. We define a new automaton

$$M = (Q \times \{0, 1\}, (q_0, 0), F \times \{1\}, \mathcal{E}'),$$

where $\mathcal{E}'$ consists of two kinds of edges:

$$((q, u), (a, S, T), (q', u)),$$

where $u \in \{0, 1\}$, $x \notin S$, and $(q, (a, S, T), q') \in \mathcal{E}$, and

$$((q, 0), (a, S \setminus \{x\}, T), (q', 1)),$$

where $x \in S$, and $(q, (a, S, T), q') \in \mathcal{E}$. It is easy to see that $w$ is accepted by $M$ if and only if there is a way to adjoin $x$ to the middle component of a letter of $w$ so as to obtain a word accepted by $A$. Thus $w$ is accepted if and only if $w \models \exists x \psi$.

We use a similar construction for the case where $\phi$ has the form $\exists X \psi$. We replace the automaton $(Q, q_0, F, \mathcal{E})$ by $(Q, q_0, F, \mathcal{E}')$, where

$$\mathcal{E}' = \{(q, (a, S, T \setminus \{X\}), q') : (q, (a, S, T), q') \in \mathcal{E}\}.$$

If the original automaton recognizes $L_\psi$, then this automaton recognizes $L_{\exists X \psi}$.

A monadic second-order formula is said to be existential if it consists of a single block of existential second-order quantifiers, followed by a first-order formula.

**III.1.2 Corollary.** Every sentence of $SOM[+1]$ is equivalent to an existential sentence of $SOM[+1]$.

*Proof. If $\phi$ is a sentence of $SOM[+1]$, then by Theorem III.1.1 $L_\phi$ is regular. The construction in the first part of the theorem now shows that $L_\phi$ is defined by an existential sentence.*

In fact, it is possible to reduce the sentence to one with a single existential quantifier. In Exercise 2 at the end of the chapter we outline a proof of this fact.
III.2. Regular Numerical Predicates

Let us now consider the situation where the alphabet $A$ is reduced to a single letter $a$. We can dispense with the predicate symbol $Q_a$, since it provides no information. A formula $\phi$ of $SOM[+1]$ with free first-order variables in $\{x_1, \ldots, x_k\}$, and no free second-order variables, thus defines a $k$-ary numerical relation. In this section we shall study the numerical relations that can be defined in this fashion. By the results of Section III.1, these are precisely the numerical relation definable by finite automata, so we will call the associated formulas regular numerical predicates.

III.2.a Example. The relation $x < y$ is definable, since it is the set of $\{x, y\}$-structures over $\{a\}$ recognized by the automaton pictured below:

We can also show directly that this relation is definable by writing a defining formula. The formula asserts the existence of a set $X$ of positions such that $x$ is the first element of $X$, $y$ is the last element of $X$, $x \neq y$, every element of $X$ different from $y$ has its successor in $X$, and every element of $X$ different from $x$ has its predecessor in $X$. The reader is encouraged to write this formula explicitly. Of course, the proof of Theorem III.1.1 provides an automatic procedure for constructing such a formula from the automaton pictured above.

III.2.b Example. If $m > 0$ then the relation "$x \equiv 0 \pmod{m}$" is definable, since it too is the set of structures accepted by a finite automaton (pictured below for the case $m = 3$).
In Example II.1.c we saw how to write a defining formula for this relation in the case \( m = 2 \). The same idea can be used to obtain a defining formula for general \( m \).

**III.2.c Example.** Let us prove that the relation \( "x + y = z" \) is not definable. Suppose there is a formula \( \phi(x, y, z) \) that defines it; then the formula \( \psi(x) \) given by

\[
\exists z (\forall y (y \leq z) \land \phi(x, x, z))
\]
defines the numerical relation "\( x \) is half the length". (By the preceding example, the relation \( y \leq z \) used in this formula can be expressed in \( SOM[+1] \).) Now the sentence

\[
\exists x (\psi(x) \land \forall y ((y \leq x \rightarrow Q_{xy}) \land (x < y \rightarrow Q_{sy})))
\]
defines the nonregular language \( \{a^n b^n : n > 0 \} \), contradicting Theorem III.1.1.

The next theorem gives a simple characterization of the numerical relations definable in \( SOM[+1] \).

**III.2.1 Theorem.** A numerical relation is definable in \( SOM[+1] \) if and only if it is definable by a first-order formula in which all the atomic formulas are of the form

\[
x < y,
\]
and

\[
x \equiv 0 \pmod{m},
\]

where \( m > 0 \).

**Proof.** We have seen in Examples III.2.a and III.2.b that the numerical relations in the statement of the theorem are definable, which implies the "if" part of the theorem. We now prove the converse direction. If \( \phi \) is a \( k \)-ary numerical relation definable in \( SOM[+1] \), then by Theorem III.1.1 (or, rather, its proof) the set \( L_\phi \) of \( (\{x_1, \ldots, x_k\}, \emptyset) \)-structures it defines is recognized by a finite automaton. (We will henceforth ignore the third component of the letters of such structures, so that we can view them as \( \{x_1, \ldots, x_k\} \)-structures.) Let us call the order type of a structure of \( w \) the order of the occurrences of
III.2. REGULAR NUMERICAL PREDICATES

the variables \(x_1, \ldots, x_k\) in \(w\). (For example, with \(k = 3\), \((x_1 < x_2 < x_3)\), \((x_1 = x_3 < x_2)\), \((x_1 < x_3 = x_2)\) are all order types. In this case there are 13 different order types.) Let \(L_\tau\) denote the set of structures having order type \(\tau\), and let \(T\) denote the set of all order types on variables \(\{x_1, \ldots, x_k\}\). Easily \(L_\tau\) is regular, and thus \(L_\phi \cap L_\tau\) is regular. It will suffice to obtain a formula \(\psi_\tau\) of the desired sort for \(L_\phi \cap L_\tau\); \(L_\phi\) is then defined by the disjunction of the \(L_\phi \cap L_\tau\) over all \(\tau \in T\). Let us write \(a\) for \((a, \emptyset)\). Each element of \(L_\phi \cap L_\tau\) then has the form

\[w = a^{m_0}(a, S_1)a^{m_1} \cdots (a, S_j)a^{m_j},\]

where the \(S_i\) are nonempty subsets of \(\{x_1, \ldots, x_k\}\) determined by the order type. Observe that \(j \leq k\). Let \(\mathcal{A}\) be an automaton that recognizes \(L_\phi \cap L_\tau\). Let us call a trace of the structure \(w\) any sequence of states

\[\alpha = (p_0, q_0, p_1, q_1, \ldots, p_j, q_j)\]

in \(\mathcal{A}\) such that \(a^{m_i}\) labels a path from \(p_i\) to \(q_i\), and \((a, S_i)\) labels an edge from \(q_{i-1}\) to \(p_i\). For states \(q, q'\) of \(\mathcal{A}\), let \(L_{q,q'}\) denote the set of all words of the form \(a^m\) that label a path from \(q\) to \(q'\). Obviouly, \(L_{q,q'}\) is regular, and thus it is a finite union of sets of the form \(a^r a^s\), for \(r, s \geq 0\). It follows that \(L_\phi \cap L_\tau\) is a finite union of sets of the form

\[L_{p_0,q_0}(a, S_1)L_{p_1,q_1}(a, S_2) \cdots L_{p_j,q_j},\]

where the union ranges over all traces of elements of \(L_\phi \cap L_\tau\). Each such set is itself a finite union of sets of the form

\[L_0(a, S_1)L_1 \cdots L_j,\]

where each \(L_i\) has the form \((a^r a^s)\), for some \(r, s \geq 0\). \(L_\phi\) is thus defined by a disjunction of formulas defining sets of the form displayed above, and each such set is in turn defined by a formula that is a conjunction of clauses expressing the following conditions:

\[\begin{align*}
(i) & \text{ The order type is } \tau; \\
(ii) & x_j = x_i + s + 1, \text{ where } s \geq 0; \\
(iii) & x_j > x_i + s, \text{ where } s \geq 0; \\
(iv) & x_j \equiv x_i + s \pmod{r}, \text{ where } s \geq 0 \text{ and } r > 1.
\end{align*}\]
It remains to show that there are first-order formulas of the desired kind for each of these four conditions. First observe that the equality and successor relations can be expressed in terms of $\prec$. Indeed, $x_i = x_j$ is expressed by
\[ \neg(x_i < x_j) \land \neg(x_j < x_i), \]
and $x_i = x_j + 1$ by
\[ x_j < x_i \land \forall y((x_j < y) \rightarrow ((x_i = y) \lor (x_i < y))). \]

We can then write a formula for (i) expressing the order type; this will be a conjunction of clauses of the form $x_i < x_j$ and $x_i = x_j$. For (ii), if $s = 0$ the condition is expressed by $x_j = x_i + 1$; if $s > 0$, it is expressed by
\[ \exists y_1 \cdots \exists y_s ((y_1 = x_i + 1) \land (x_j = y_s + 1) \land \bigwedge_{m=1}^{s-1} (y_{m+1} = y_m + 1)). \]

Condition (iii) is expressed by a formula that is identical, except $x_j = y_s + 1$ is replaced by $y_s < x_j$. To express condition (iv), first note that if $0 < m < r$ then we can express
\[ x_i \equiv m \pmod{r} \]
by
\[ \exists z((z \equiv 0 \pmod{r}) \land (x_i = z + m)). \]

Now $z \equiv x_i + s \pmod{r}$ is expressed by
\[ \bigvee_{m \in \mathbb{Z}_r} ((x_i \equiv m \pmod{r}) \land (z \equiv m + s \pmod{r})). \]

This completes the proof. \[ \]

III.3 Infinite Words and Decidable Theories

In this section, we describe an analogue of Theorem III.1.1 for infinite sequences of the form
\[ a_1a_2 \cdots, \]
where each $a_i \in A$. This theorem has important consequences in mathematical logic, which we will discuss at the end of the section. We denote the set of all such infinite sequences of elements of $A$ by $A^\omega$. More generally, if $L \subseteq A^+$, then $L^\omega$ denotes the set of all sequences

$$v_1v_2\cdots,$$

where each $v_i \in L$.

It should be clear how to interpret monadic second-order formulas in such infinite strings. A formula beginning with a second-order quantifier

$$\exists X \phi$$

means "there exists a set $X$ of positions such that $\phi(X)$". The set $X$ can be finite or infinite.

We can read elements of $A^\omega$ with ordinary nondeterministic finite automata $(Q, i, F, \mathcal{E})$. An infinite word $\alpha \in A^\omega$ is the label of (possibly infinitely many, because of the nondeterminism) infinite paths

$$(i, a_1, q_1)(q_1, a_2, q_2)\cdots$$

in the automaton. We say that the automaton accepts $\alpha$ if $\alpha$ is the label of a path in which $q_j \in F$ for infinitely many indices $j$. The $\omega$-language recognized by the automaton is the set of all elements of $A^\omega$ accepted by the automaton.

**III.3. Example.** Consider the automaton pictured below.

![Automaton Diagram](image)

The $\omega$-language recognized by this automaton is the set of all strings in $\{a, b\}^\omega$ containing only finitely many occurrences of $b$. We can write $L$ as $\{a, b\}^*a^\omega$, and define it by the sentence

$$\exists x \forall y ((y > x) \rightarrow Q_0x).$$
The complement of $L$ is the set of words containing infinitely many occurrences of $b$. We can write it as $(a^*ba^*)^\omega$. It is recognized by the automaton

![Automaton Diagram]

which is deterministic. There is, however, no deterministic automaton that recognizes $L$. To see this, suppose that such an automaton exists. Then the word $ba^\omega$ must label a sequence of states that includes a final state infinitely often. By the determinism of the automaton, this sequence is unique. There is thus some $k_0 > 0$ such that $ba^{k_0}$ leads from the initial state to a final state. Similarly $ba^{k_0}ba^\omega$ passes a final state infinitely often, so there is some $k_1 > 0$ such that $ba^{k_0}ba^{k_1}$ leads from the initial state to a final state. We continue in this manner and obtain an infinite word

$ba^{k_0}ba^{k_1}\ldots$

that is accepted by the automaton. But this word contains infinitely many $b$'s, a contradiction.

The following theorem is the analogue of Theorem III.1.1 for infinite words.

**III.3.1 Theorem.** Let $L \subseteq A^\omega$. $L$ is recognized by a finite automaton if and only if $L$ is defined by a sentence of SOM[+1].

The proof depends on the following fact, which we will prove later in the section:

**III.3.2 Proposition.** If $L_1, L_2 \subseteq A^\omega$ are recognized by finite automata, then so are $L_1 \cup L_2$, $L_1 \cap L_2$, and $A^\omega \setminus L_1$.

Along the way, we will also show (Proposition III.3.4) that the $\omega$-languages recognized by finite automata are exactly those defined by generalized regular expressions like those that appear in the example above.
III.3. INFINITE WORDS AND DECIDABLE THEORIES

Assuming Proposition III.3.2 we can prove our main theorem:

Proof of Theorem III.3.1. The proof is essentially the same as that of Theorem III.1.1. Given an automaton we construct a sentence as before. However, in place of the condition on the last letter of the word, we must have a clause that states that one of the sets \( X_q \) associated with a final state is infinite. "\( X \) is infinite" is expressed by:

\[
\forall x \exists y ((y > x) \land X(y)).
\]

The converse is proved exactly as in Theorem III.1.1. We use Proposition III.3.2 to handle the boolean connectives.

We now turn to the proof of Proposition III.3.2. This requires a sequence of auxiliary propositions.

III.3.3 Proposition. If \( L_1, L_2 \subseteq A^\omega \) are recognized by finite automata, then so is \( L_1 \cup L_2 \).

Proof. We use a standard construction that works for finite words as well. Let \( \mathcal{M}_j = (Q_j, i_j, F_j, E_j), j = 1, 2 \), be automata recognizing \( L_j \). We can suppose that the sets \( Q_1, Q_2 \) are disjoint. Let us introduce a new state \( i \). Now set

\[
\mathcal{M} = (Q_1 \cup Q_2 \cup \{i\}, i, F_1 \cup F_2, E),
\]

where

\[
E = E_1 \cup E_2 \cup \bigcup_{j=1,2} \{(i, a, q) : (i_j, a, q) \in E_j\}.
\]

Then \( \mathcal{M} \) recognizes \( L_1 \cup L_2 \).

III.3.4 Proposition. \( L \subseteq A^\omega \) is recognized by a finite automaton if and only if \( L \) is a finite union of sets of the form \( JK^\omega \), where \( J \subseteq A^* \), \( K \subseteq A^+ \), are regular languages.

Proof. Let \( L \subseteq A^\omega \) be recognized by a finite automaton \( (Q, i, F, E) \). Then \( \alpha \in A^\omega \) is accepted by the automaton if and only if there is some \( p \in F \) such that infinitely many initial segments of \( \alpha \) label a path from \( i \) to \( p \). For \( q, q' \in Q \), let \( L_{q,q'} \subseteq A^+ \) denote the set of finite words that label a path from \( q \) to \( q' \). Clearly \( L_{q,q'} \) is a regular language. We have

\[
L = \bigcup_{p \in F} L_{i,p} L_{p,p}^\omega.
\]
CHAPTER III. FINITE AUTOMATA

For the converse, we need to prove that if \( J \subseteq A^* , K \subseteq A^+ \), are regular languages, then \( J K^\omega \) is recognized by a finite automaton. (We can then apply Proposition III.3.3 to obtain the desired result.) First, let \((Q, i, F, \mathcal{E})\) be an automaton that recognizes \( K \). We add a new state \( i' \) and new edges

\[(i', a, q)\]

if \((i, a, q) \in \mathcal{E}\), and

\[(q, a, i')\]

if \((q, a, p) \in \mathcal{E}\) for some \( p \in F \). Let \( Q' \) denote the enlarged set of states, and \( \mathcal{E}' \) the enlarged set of edges. Then \((Q', i', \{i'\}, \mathcal{E}')\) recognizes \( K^\omega \). Now let \((P, j, T, \mathcal{D})\) be an automaton that recognizes \( J \). We can suppose \( P \) is disjoint from \( Q' \). Let

\[\mathcal{C} = \{(q, a, i') : \exists t \in T, (q, a, t) \in \mathcal{D}\} \]  

Then \((P \cup Q', j, \{i'\}, \mathcal{C} \cup \mathcal{D} \cup \mathcal{E}')\) recognizes \( J K^\omega \). 

For the next three propositions, we let \( \mathcal{M} = (Q, i, F, \mathcal{E}) \) be a finite automaton. We will now define an equivalence relation \( \equiv \mathcal{M} \) on \( A^+ \). If \( q, q' \in Q \) and \( w \in A^+ \), then we say \( s(q, q', w) \) to mean that there is a path in \( Q \) from \( q \) to \( q' \) labelled \( w \), and \( t(q, q', w) \) if there is such a path that includes a state of \( F \). (In particular, if \( q \in F \) or \( q' \in F \), then \( s(q, q', w) \) implies \( t(q, q', w) \).) We write

\[ w \equiv \mathcal{M} w' \]

if for all \( q, q' \in A \),

\[ s(q, q', w) \leftrightarrow s(q, q', w') \]

and

\[ t(q, q', w) \leftrightarrow t(q, q', w'). \]

III.3.5 Proposition. \( \equiv \mathcal{M} \) is an equivalence relation of finite index, and every equivalence class is a regular language.

Proof. The equivalence class of \( w \) is determined by the sets of pairs \( (q, q') \) such that \( s(q, q', w) \) and \( t(q, q', w) \) both hold. It follows that there are only finitely many equivalence classes (because \( Q \) is finite) and that each equivalence class is a finite boolean combination of sets of the form

\[ \{v : s(q, q', v)\} \]
III.3. INFINITE WORDS AND DECIDABLE THEORIES

and

\{v : t(q, q', v)\}.

It thus suffices to show that each of these sets is a regular language.

We use the notation of the proof of Proposition III.3.4. We have

\{v : s(q, q', v)\} = L_{q,q'}.

If neither q nor q' belongs to F, then

\{v : t(q, q', v)\} = \bigcup_{f \in F} L_{q,f} L_{f,q'}.

If either q \in F or q' \in F, then

\{v : t(q, q', v)\} = \bigcup_{f \in F} L_{q,f} L_{f,q'} \cup L_{q,q'}.

Since sets of the form L_{q,q'} are regular languages, this gives the desired result. \[\square\]

III.3.6 Proposition. Let U, V \subseteq A^+ be \equiv_m\text{-classes. Let } L \subseteq A^\omega \text{ be the set recognized by } M. \text{ If}

UV^\omega \cap L \neq \emptyset

then

UV^\omega \subseteq L.

Proof. Let \alpha \in UV^\omega \cap L. Then

\alpha = uv_1v_2 \cdots

where u \in U, v_j \in V. There is thus a sequence of states i, q_1, q_2, \ldots, such that s(i, q_1, u), s(q_j, q_{j+1}, v_j) for all j \geq 1, and t(q_j, q_{j+1}, v_j) for infinitely j. If \beta \in UV^\omega, then

\beta = u'v_1'v_2' \cdots,

where u' \equiv_m u and v_j' \equiv_m v_j for all j. Thus s(i, q_1, u'), s(q_j, q_{j+1}, v_j') for all j \geq 1, and t(q_j, q_{j+1}, v_j') for infinitely many j. This implies \beta \in L. \[\square\]
III.3.7 Proposition. Let $\alpha \in A^\omega$. Then there exist $\equiv_M$-classes $U$ and $V$ such that $\alpha \in UV^\omega$.

Proof. For each pair $i < j$ of positive integers, let $\alpha_{i,j}$ denote the element of $A^+$ that begins at the $i^{th}$ position of $\alpha$ and ends at the $(j - 1)^{th}$ position. We color the set $\{i,j\}$ with the equivalence class of $\alpha_{i,j}$. This defines a finite coloring of the infinite set $\{(m,n) : 1 \leq m < n\}$. By Ramsey's theorem, there exists an infinite set $I = \{i_1 < i_2 < \cdots\}$ such that all two-element subsets of $I$ have the same color. Let the equivalence class $V$ be this color, and let $U$ be the equivalence class of $\alpha_{i_1,i_1}$. Then

$$\alpha = \alpha_{i_1,i_1} \alpha_{i_1,i_2} \cdots \in UV^\omega.$$  

(In the case where $i_1 = 1$ we take $U = V$.)

Proof of Proposition III.3.2. Closure under union was proved in Proposition III.3.3. Once we have proved closure under complement, closure under intersection will follow. Let $M$ be a finite automaton that recognizes $L \subseteq A^\omega$. We claim that $A^\omega \setminus L$ is the union of the sets $JK^\omega$, where $J$ and $K$ are $\equiv_M$-classes and $JK^\omega \cap L = \emptyset$. Obviously this union is contained in $A^\omega \setminus L$. If $\alpha \in A^\omega \setminus L$, then by Proposition III.3.7, there exist equivalence classes $J, K$ such that $\alpha \in JK^\omega$. By III.3.6, $JK^\omega \subseteq L$. Thus $L$ is equal to the union, as claimed. By Proposition III.3.4 and Proposition III.3.5, this union is recognized by a finite automaton.

In the case where $A$ is reduced to a single letter, a formula with $k$ free first-order variables and no free second-order variables defines, as in Section III.2, a $k$-ary relation on $Z^+$. If the formula is a sentence, it states a property of $Z^+$, which is either true or false. We can thus view monadic second-order logic with the successor relation and without the predicates $Q_n$ as a language in which to express properties of ordinary arithmetic. Following the traditions in the literature, we call this language $S1S$.

It is an important question in mathematical logic to know whether a logical theory is decidable; that is, whether there is an algorithm to determine if a given sentence is true or false. For example, if we consider first-order sentences over the positive integers with addition and multiplication as numerical predicates (Peano arithmetic), the resulting theory is undecidable, because one can code undecidable questions about Turing machines by sentences of Peano arithmetic. The theory $S1S$ allows stronger quantification than Peano arithmetic, but is weaker
III.3. INFINITE WORDS AND DECIDABLE THEORIES

in terms of numerical predicates. We can use the automaton-theoretic considerations of this section to prove:

III.3.8 Theorem. $S_1S$ is decidable.

Proof. Consider a sentence of $S_1S$. We can effectively construct a non-deterministic automaton that recognizes the subset of $\{a\}^\omega$ defined by this sentence. Effectiveness of the automaton constructions was given explicitly in the proofs, except for the complementation operation. In this case observe that if we are given an automaton $\mathcal{M}$ recognizing $L \subseteq A^\omega$, we can effectively construct automata recognizing all the $\equiv_{\mathcal{M}}$ classes, and thus construct automata recognizing each of the sets $JK^\omega$, where $J$ and $K$ are $\equiv_{\mathcal{M}}$-classes. We need to determine for each of these sets whether it is contained in $L$ or in the complement of $L$. To do this we choose words $u \in J$ and a word $v \in K$ and test whether there is a state $q$ such that $s(i, q, uv^m)$ and $t(q, q, v^m)$ for some $m, n \leq |Q|$.

The automaton we construct either accepts the string $a^\omega$ (if the sentence is true) or does not (if the sentence is false). The matter is settled by determining whether there is a final state $q$ and paths in the automaton from the initial state to $q$, and from $q$ to itself. This problem is decidable, since we only need to inspect paths of length less than or equal to $|Q|$.

Exercises

1. Let $\phi$ be a monadic second-order sentence containing $n$ symbols. Give an upper bound on the number of states in a deterministic automaton that recognizes $L_\phi$.

2. Prove that a language is regular if and only if it is defined by a sentence of $SOM[+1]$ of the form $\exists X \phi$, where $\phi$ has only first-order quantifiers.

Hint. We proved in Theorem III.1.1 that a language $L$ recognized by a finite automaton $\mathcal{M}$ is in $SOM[+1]$. This was done by showing that $w \in L$ if and only if there is a sequence of states

$q_1 \cdots q_{|w|}$
that satisfies certain consistency conditions with respect to \( w \) and \( \mathcal{M} \).
The existence of the sequence is then expressed by an existential sentence of \( SOM[+1] \). Suppose that \( \mathcal{M} \) has \( k \) states, which we order somehow, and that the \( j^{th} \) state is encoded by the sequence of \( k + 3 \) bits
\[
0^{j-1}10^{k-j}011.
\]
Then \( w \in L \) if and only if there is a sequence of bits
\[
c_{i_1} \cdots c_{i_r} 0^m,
\]
where \( r = \lfloor |w|/(k + 3) \rfloor \), and \( m = |w| - r(k - 3) \), that satisfies some consistency requirements with respect to \( \mathcal{M} \) and \( w \). Formulate these conditions precisely, and show that the existence of such a bit sequence can be expressed by a sentence of \( SOM[+1] \) with a single existential second-order quantifier.

3. We can change the interpretation of monadic second-order formulas in infinite words by interpreting the second-order variables as ranging over finite sets only. The resulting logic is called weak monadic second-order logic. In the case where the input alphabet is restricted to a single letter, we obtain weak second-order arithmetic. Use Theorem III.3.8 to show that weak second-order arithmetic is decidable.

4. Compute the equivalence classes \( \equiv_{\mathcal{M}} \) for the automaton \( \mathcal{M} \) pictured in Figure III.3, and construct a deterministic finite automaton that recognizes each of these classes. Write the complement of the subset of \( A^\omega \) recognized by \( \mathcal{M} \) as a union of sets \( J K^\omega \), where \( J \) and \( K \) are equivalence classes.

5. We can encode a positive integer \( n \) as a nonempty finite subset of \( \mathbb{Z}^+ \) by using the binary expansion of \( n \) : If
\[
n = 2^{i_1} + \cdots + 2^{i_r},
\]
where \( i_1 > i_2 > \cdots > i_r \geq 0 \), then the encoding of \( n \) is the set \( \{i_1, \ldots, i_r\} \). Thus the relation \( x + y = z \) on positive integers corresponds to a relation \( r(X, Y, Z) \) on finite subsets of positive integers.

(a) Express this relation by a formula of \( S1S \).

(b) Presburger arithmetic is the first-order theory of the positive integers in which sentences contain a symbol for 1 and atomic formulas of the form \( x = y \) and \( x + y = z \). Use part (a) of this exercise and Theorem III.3.8 to show that Presburger arithmetic is decidable.
III.3. INFINITE WORDS AND DECIDABLE THEORIES

6. A positive real can be encoded as a pair consisting of a finite subset of \( \mathbb{Z}^+ \) for the integer part, and an infinite subset \( \mathbb{Z}^+ \) for the fractional part. This is carried out in a fashion analogous to the preceding exercise. Use this and Theorem III.3.8 to show that the first-order theory of the positive reals with addition is decidable.

Chapter Notes

Theorems III.1.1, III.3.1, and III.3.8 are due to Büchi [15, 16]. McNaughton [40] proved the equivalence between recognition of an infinite sequence by a nondeterministic automaton, as defined here, and a special kind of deterministic automaton.

Perhaps the farthest-reaching decidability result exploiting finite automata is that of Rabin [48] concerning automata on infinite trees. A thorough account of these results is given in Thomas [65].

Theorem III.2.1 in the form given here is from Straubing [56], but the underlying idea can be found in Büchi [15]. Similar results appear in Péladau [44].

Exercise 2 is from Thomas [63].

If we allow second-order variables of higher arity, that is, if we allow quantification over binary, ternary, etc., relations on \( \{1, \ldots, n\} \) then the existential second-order sentences define precisely the languages in the computational complexity class \( NP \) (nondeterministic polynomial time) and the second-order sentences define the languages in the polynomial time hierarchy (Fagin [25], Lynch [38], Stockmeyer [54]). We will return to such contacts with computational complexity theory later in the book, when we discuss circuits.
Chapter IV

Model-Theoretic Games

IV.1 The Ehrenfeucht-Fraïssé Game

We will begin to answer the question: What languages can be defined with first-order sentences? The answer, of course, depends on what numerical predicates we are allowed to use. Throughout this section we will assume that we are working with first-order formulas with some fixed finite set of numerical predicates and a fixed interpretation \( I \). In the subsequent sections of this chapter we will make specific choices for these numerical predicates and prove some limitations on the power of first-order sentences. For example, we will show that the numerical predicate \( x < y \) cannot be defined by a first-order formula in which the only numerical predicates are of the form \( x = y \) and \( y = x + 1 \), and that there are regular languages that cannot be defined by first-order sentences in which \( x < y \) is the only numerical predicate allowed.

The question of first-order definability will be investigated using an interesting method from model theory: Ehrenfeucht-Fraïssé games.

We begin by defining the quantifier complexity of a formula \( \phi \), denoted \( c(\phi) \): If \( \phi \) is an atomic formula, then its quantifier complexity is 0. The complexity of nonatomic formulas is defined inductively, according to the following rules:

\[
\begin{align*}
c(\neg \phi) &= c(\phi), \\
c(\phi \land \psi) &= \max(c(\phi), c(\psi)), \\
c(\exists x \phi) &= c(\phi) + 1.
\end{align*}
\]

We will adopt a kind of normal form for formulas, which we define by induction on the quantifier complexity. A quantifier-free formula
(a formula of quantifier complexity 0) can be written in disjunctive normal form; that is, as a disjunction of conjunctions of atomic formulas and negated atomic formulas. Because disjunction and conjunction are idempotent, we can eliminate any repetition of a conjunct, and any repetition of a disjunct. This will constitute our normal form for quantifier-free formulas. A formula of quantifier complexity $c + 1$ can be written as a disjunction of conjunctions of formulas, each of which is of one of the forms $\exists x \phi$, $\neg \exists x \phi$, or $\phi$, where $c(\phi) \leq c$. We can write each of the formulas $\phi$ in normal form, and eliminate repeated conjuncts and disjuncts. This constitutes our normal form for formulas of complexity $c + 1$.

**IV.1.1 Proposition.** Let $\mathcal{V}$ be a finite set of first-order variables, and let $c \geq 0$. There are only finitely many normal-form formulas of complexity $c$ in which every variable belongs to $\mathcal{V}$.

*Proof.* We prove this by induction on $c$. There are $m$ atomic formulas, where $m$ depends on the number of numerical predicate symbols of each arity and on the cardinality of $\mathcal{V}$. There are thus at most $2^{2^m}$ disjunctive normal forms of atomic formulas and negated atomic formulas. Now suppose there are $p$ formulas of complexity less than $c$. There are then $2p$ formulas of the form $\exists x \phi$ or $\neg \exists x \phi$ with $c(\phi) < c$, and thus $2^{2p}$ conjunctions of these. We may add to this conjunction one of the $p$ formulas of complexity less than $c$, giving at most $q = (p + 1)2^{2p}$ possible disjuncts, and thus at most $2^q$ formulas in normal form. $
$

A formula is *logically equivalent* to its normal form, in the sense that the two formulas will be satisfied by exactly the same structures, no matter how we interpret the atomic formulas. Let $w_1, w_2$ be $\mathcal{V}$-structures and let $r \geq 0$. We write

$$w_1 \sim_r w_2$$

if and only if $w_1$ and $w_2$ satisfy precisely the same formulas of complexity no more than $r$. As an immediate consequence of Proposition IV.1.1, we have

**IV.1.2 Corollary.** $\sim_r$ is an equivalence relation of finite index. $
$

We now define the $r$-round game in $(w_1, w_2)$. The game is played with two $\{y_1, \ldots, y_p\}$-structures $w_1$ and $w_2$. There are two players, I and
IV. 1 THE EHRENFEUCHT-FRAISSE GAME

II. Player I will try to show that the two structures are different, while Player II will try to show that they are the same. Each player has \( r \) pebbles, labelled \( z_1, \ldots, z_r \). At the \( i^{th} \) round, Player I places his pebble labelled \( z_i \) on one of the letters of one of the two structures. Player II must place her pebble labelled \( z_i \) on a letter of the other structure. Once a pebble is played it may not be moved. The game ends after \( r \) rounds, at which point both players will have used all their pebbles.

Who won? At the end of the game we have two \( \{y_1, \ldots, y_p, z_1, \ldots, z_r\} \)-structures \( w'_1 \) and \( w'_2 \). II wins if for every atomic formula \( \alpha \), \( w'_i \models \alpha \) if and only if \( w'_j \models \alpha \). That is, Player II wins if and only if \( w'_1 \sim_0 w'_2 \). Otherwise Player I wins.

IV.1.a Example. Let \( \mathcal{V} = \emptyset \), and consider the game with \( w_1 = ab \) and \( w_2 = baa \) at the outset. Suppose that the only numerical relation we use is equality. Player I can show that the two words are different with respect to this relation by showing that \( w_2 \) contains two distinct positions with the same letter. Note, however, that he cannot do this in only one round, since Player II always has a satisfactory reply to the first play. However, Player I can win in two rounds: He places the pebble \( z_1 \) on the first \( a \) of baa. Player II is obliged to reply on the first letter of \( ab \). Player I now plays \( z_2 \) on the last letter of \( baa \). If Player II responds by placing her remaining pebble on \( a \), the resulting structure will satisfy \( z_1 = z_2 \). If she plays on \( b \), then it will fail to satisfy \( Q_a z_2 \).

IV.1.b Example. Let us now work with formulas in which the only numerical predicates are of the form \( x < y \). We consider two words \( w_1, w_2 \in \{a, b\}^* \), such that the last letter of \( w_1 \) is \( a \) and the last letter of \( w_2 \) is \( b \). Player I can win the two-round game by playing \( z_1 \) on the last letter of \( w_1 \). If \( w_2 \) does not contain the letter \( a \), then Player II has no satisfactory response, and Player I will win regardless of what happens in the next round. Otherwise Player II must place her pebble \( z_1 \) on an \( a \) in \( w_2 \). Player I now places his pebble \( z_2 \) in \( w_2 \), somewhere to the right of Player II's \( z_1 \). The resulting structure satisfies the atomic formula \( z_1 < z_2 \). There is no place for Player II to play the pebble \( z_2 \) in \( w_1 \) to obtain a structure satisfying this formula. Thus Player I wins.

We note that for a given choice of structures \( w_1, w_2 \) and a given number of rounds, one of the players has a winning strategy. To see
this, consider the tree of all possible sequences of plays. The nodes of this tree are all possible pairs of structures obtainable by some sequence of plays, with the pair \((w_1, w_2)\) at the root, and configurations of finished games at the leaves. The root corresponds to the first move by Player I, the children of the root to the first play by Player II. We now define recursively a labelling of the nodes of the tree, beginning at the leaves. Since the game cannot end in a tie, we can label each leaf I or II, depending upon whether the leaf corresponds to a winning configuration for Player I or for Player II. An interior node corresponding to a play by Player I is labelled I if and only if some child is labelled I, otherwise the node is labelled II; similarly an interior node corresponding to a play by Player II is labelled II if and only if some child is labelled II, otherwise the node is labelled I. Easily, the label at the root determines which of the players has a winning strategy.

In the next theorem we show that the existence of a winning strategy for Player II characterizes first-order equivalence of structures.

**IV.1.3 Theorem.** Let \(w_1, w_2\) be \(\mathcal{V}\)-structures and let \(r \geq 0\). Then \(w_1 \sim_r w_2\) if and only if Player II possesses a winning strategy in the \(r\)-round game in \((w_1, w_2)\).

**Proof.** We first assume \(w_1 \sim_r w_2\) and prove by induction on \(r\) that Player II has a winning strategy. If \(r = 0\) then Player II has won, since by assumption \(w_1\) and \(w_2\) satisfy all the same atomic formulas. Now assume that \(r > 0\) and that this direction of the theorem is true for \(r - 1\). Suppose further that the conclusion for \(r\) is false—that is, that Player II does not have a winning strategy. By the remarks preceding the theorem, Player I has a winning strategy. Let Player I make the first move of his strategy (let us say that it is in \(w_1\)). There results a structure \(w'_1\) such that for any response of Player II in \(w_2\), giving a structure \(w'_2\), Player I has a winning strategy in the \((r - 1)\)-round game in \((w'_1, w'_2)\). It follows from the inductive hypothesis that \(w'_1\) and \(w'_2\) are not \(\sim_{(r-1)}\)-equivalent. Let \(\psi\) be the conjunction of all normal form formulas of complexity less than \(r\) satisfied by \(w'_1\). Then \(w'_2 \not\models \psi\). Since this holds for any structure \(w'_2\) obtained from \(w_2\) by placing the pebble \(z_1\), we have

\[
\begin{align*}
    w_2 \not\models \exists z_1 \psi, \\
    w_1 \models \exists z_1 \psi,
\end{align*}
\]

a contradiction. Thus Player II must have a winning strategy.
The converse direction is also proved by induction on $r$. If $r = 0$ and Player II has a winning strategy then $w_1$ and $w_2$ satisfy the same atomic formulas, and thus the same formulas of complexity 0. Suppose now that $r > 0$ and that this direction of the theorem is true for $r - 1$. Suppose Player II has a winning strategy in the $r$-round game in $(w_1, w_2)$. If $w_1$ and $w_2$ are not $\sim_r$-equivalent, then there exists a formula $\psi$ of complexity $r$ such that

$$w_1 \models \psi$$

and

$$w_2 \not\models \psi.$$  

We can assume that $\psi$ has the form $\exists z_1 \phi$, where $c(\phi) = r - 1$. Since $w_1 \models \psi$, Player I can place his pebble $z_1$ at a position in $w_1$ such that the resulting structure $w'_1$ satisfies $\phi$. Player II follows her winning strategy and replies in $w_2$ to obtain a structure $w'_2$. By assumption $w'_2$ does not satisfy $\phi$. But now Player II has a winning strategy in the $(r - 1)$-round game in $(w'_1, w'_2)$. Thus, by the inductive hypothesis, $w'_1$ and $w'_2$ must both satisfy or both fail to satisfy $\phi$, a contradiction.

**IV.1.c Example.** Let us return to examples IV.1.a and IV.1.b above. In the first case, the result of the game, along with Theorem IV.1.3, tells us that $ab$ and $baa$ satisfy all the same formulas of complexity 1, but not all the same formulas of complexity 2. A formula on which they fail to agree says, informally, that the word contains two distinct positions with the letter $a$. Such a formula is

$$\exists x \exists y (\neg (x = y) \land Q_a x \land Q_a y),$$

which has complexity 2. We also conclude, not surprisingly, that the same property cannot be expressed by a sentence of quantifier complexity 1. In the second example we can conclude that there is a sentence of quantifier complexity no more than 2 that is satisfied by exactly one of the two words. Such a sentence says, informally, that the last letter is $b$, and is given by

$$\exists x (\forall y \neg (x < y) \land Q_b x).$$

**IV.1.d Example.** Theorem IV.1.3 is, of course, entirely general, and we
can use it to compare any two structures in which we interpret formulas of first-order logic with a finite number of relation symbols. Let us thus consider the logic of first-order formulas with binary relation symbols $<$ and $=$, and interpretations in sets $S$ with a binary relation $<_S$. (We will always interpret $=$ by equality.) Two pairs $(S, <_S)$ and $(T, <_T)$ are regarded as equivalent if they satisfy all the same first-order sentences. Observe that transitivity, reflexivity, antisymmetry, presence of a least or greatest element, totality (meaning that every pair of elements is $<$-comparable) and denseness (between any two distinct elements there is a third element), are all easily expressed by first-order sentences. Alternatively, we can prove that these properties are definable using games—for example, Player I can always distinguish between a total and a nontotal partial order in two rounds, and between a dense and a nondense total order in three rounds.

We can conclude from this that $(Q, <)$ and $(R, <)$ (respectively the rational and the real numbers with the usual ordering) satisfy exactly the same sentences. Whatever the number of rounds in the game, Player II wins by placing her pebble so as to keep the same ordering among the $z_i$ in the two structures. This can always be done because of the denseness of the orders. Indeed, any two dense total orders without a least or a greatest element satisfy the same first-order sentences.

On the other hand $R$ and $Q$ can be distinguished by a monadic second-order sentence that expresses the least upper bound property. These observations thus constitute a proof that the least upper bound property cannot be expressed by a first-order sentence.

### IV.2 Application to $FO[<]$

We denote by $FO[<]$ the family of languages defined by first-order sentences in which the only numerical predicates are of the form $x < y$.

### IV.2.1 Theorem. The set of words of even length is not in $FO[<]$.

**Proof.** Suppose that there is a sentence $\phi$ that defines the set of words of even length. Let $r = c(\phi)$. Thus for $a \in A$, $a^k \models \phi$ if and only if $k$ is even. We shall show that for all $r > 0$,

$$a^{2^r} \sim_r a^{2^{r-1}},$$
and thus if one of these words satisfies \( \phi \), then the other does, which gives a contradiction.

By Theorem IV.1.3, it suffices to show that if \( k \geq 2^r - 1 \), then Player II has a winning strategy for the \( r \)-round game in \((a^k, a^{k+1})\). The proof is by induction on \( r \). If \( r = 1 \), then Player II wins the game with any response to Player I's move. Suppose now that \( r > 1 \) and that the claim is true for \( r - 1 \). Let \( k \geq 2^r - 1 \). Player I puts his pebble \( z_1 \) in one of the two words, giving a structure

\[
(a, \emptyset)^s(a, \{z_1\})(a, \emptyset)^t.
\]

Either \( s \leq (k - 1)/2 \), or \( t \leq (k - 1)/2 \). Let us suppose \( s \leq (k - 1)/2 \); the reasoning is the same in the other case. Player II places her pebble \( z_1 \) on the \((s + 1)\)th letter of the other word, giving a structure

\[
(a, \emptyset)^s(a, \{z_1\})(a, \emptyset)^{t'},
\]

where \( t' = t + 1 \) or \( t' = t - 1 \). Since

\[
2^r - 1 \leq k = \min (t, t') + s + 1 \leq \min (t, t') + (k - 1)/2 + 1,
\]

we obtain

\[
\min (t, t') \geq (k - 1)/2 \geq 2^{r-1} - 1.
\]

Thus, by the inductive hypothesis, Player II has a winning strategy for the \((r - 1)\)-round game with initial structures \((a^i, a^{t'})\). We use this to describe the rest of Player II's strategy for the game in \((a^k, a^{k+1})\): Whenever Player I plays on the \(i\)th letter of a structure, with \( i \leq s + 1 \), Player II plays on the \(i\)th letter of the other structure. However, if Player I plays in the right-hand portion of one structure, so that \( i > s + 1 \), then Player II plays in the right-hand portion of the other structure, following her strategy for the \((r - 1)\)-round game in \((a^i, a^{t'})\). We claim that Player II has won. At the end of the \( r \)-round game, we have two structures \( w'_1, w'_2 \). If \( i \leq s + 1 \) then \( z_j \) appears in the \(i\)th position of \( w'_1 \) if and only if \( z_j \) appears in the \(i\)th position of \( w'_2 \). Let \( u'_1 \) and \( u'_2 \) be the structures obtained by erasing the first \( s + 1 \) letters. Then in \( u'_1, u'_2 \) we find the structures that result after \( r' \leq r - 1 \) rounds of play of Player II's winning strategy in the \((r - 1)\)-round game in \((a^i, a^{t'})\). In particular, \( u'_1 \) and \( u'_2 \) satisfy the same atomic formulas. Obviously \( w'_1 \) and \( w'_2 \) both satisfy \( Q_{a,z_j} \) for all \( j \). Suppose \( w'_1 \models z_i < z_j \). If \( z_i, z_j \) both occur in the first \( s + 1 \) letters of \( w'_1 \), then they occur in the corresponding positions of \( w'_2 \), and thus \( w'_2 \models z_i < z_j \). If they both occur in the last
$|w'_1| - s - 1$ positions, then $v'_1 \models z_i < z_j$, whence $v'_2 \models z_i < z_j$, and thus $w'_2 \models z_i < z_j$. Finally, if $z_i$ appears among the first $s + 1$ letters of $w'_1$ and $z_j$ appears after the first $s + 1$ letters, then the same is true of $w'_2$, and thus $w'_2 \models z_i < z_j$. The identical argument shows that $w'_1$ satisfies all the atomic formulas satisfied by $w'_2$. This completes the proof.  

**IV.2.2 Corollary.** $FO[<]$ is strictly contained in $SOM[+1]$

*Proof.* The set of words of even length is a regular language, so by Theorem III.1.1 it is in $SOM[+1]$.

We shall have more to say about $FO[<]$ in the next chapter.

**IV.3 Application to $FO[+1]$**

We denote by $FO[+1]$ the class of languages defined by first-order sentences in which the numerical predicates are of the form $x = y$ and $y = x + 1$. Throughout the section the equivalence relations $\sim_r$ and the rules of the Ehrenfeucht-Fraissé games are to be understood as defined with respect to this class of formulas. We will give a characterization of $FO[+1]$.

Let $v \in A^*$, $w \in A^+$. $Fact(w,v)$ denotes the number of times $v$ occurs as a factor of $w$. For example,

$Fact(abbababba, bb) = 3$.

Let $k, r > 0$. We define an equivalence relation $\approx^k_r$ on $A^*$ as follows. If $|w_1| < k$, then $w_1 \approx^k_r w_2$ if and only if $w_1 = w_2$. Otherwise, $w_1 \approx^k_r w_2$ if and only if the following three conditions are satisfied:

(i) $w_1$ and $w_2$ have the same prefix of length $k - 1$.

(ii) $w_1$ and $w_2$ have the same suffix of length $k - 1$.

(iii) For all $v \in A^+$ with $|v| \leq k$, either

$Fact(w_1, v) = Fact(w_2, v) < r$,

or both $Fact(w_1, v) \geq r$ and $Fact(w_2, v) \geq r$.  

IV.3. APPLICATION TO FO[+1]

It is obvious that $\approx_k$ is an equivalence relation. Furthermore, the equivalence class of a word $w$ of length at least $k$ is determined by the prefix of $w$ of length $k - 1$, the suffix of $w$ of length $k - 1$, and the map

$$v \mapsto \max(\text{Fact}(w, v), r)$$

restricted to $\{v : |v| \leq k\}$. Thus this equivalence relation has finite index. A language $L \subseteq A^*$ is said to be locally threshold testable if it is the union of $\approx_k$-classes for some $k, r > 0$. (Let us explain this curious terminology: The case $r = 1$ was studied first, and these languages were called "locally testable", because one could determine whether a word belonged to the language by scanning the word with a window of width $k$, and checking which strings appeared in the window. In the general case we count the number of occurrences of each string we see in the window up to a certain threshold $r").

IV.3.1 Lemma. Let $L \subseteq A^*$. If $L$ is locally threshold testable then $L \in \text{FO[+1]}$.

Proof. It suffices to prove that each $\approx_k$-class is in $\text{FO[+1]}$. Let $v = a_1 \cdots a_{k-1}$. We can express "$v$ is a prefix of $w$" by

$$\exists x_1 \cdots \exists x_{k-1}(\bigwedge_{i=1}^{k-1} (x_{i+1} = x_i + 1) \land \bigwedge_{i=1}^{k} Q_{a_i} \land \text{first}(x_1)),$$

where $\text{first}(x)$ is an abbreviation for

$$\neg\exists y (x = y + 1).$$

We similarly express "$v$ is a suffix of $w$" by a sentence using the analogous numerical predicate $\text{last}$. If $s > 0$ and $v = a_1 \cdots a_l$, where $l \leq k$, then we express \'\text{Fact}(w, v) \geq s\' by

$$\exists x_{1,1} \cdots \exists x_{1,l} \cdots \exists x_{s,l}(\bigwedge_{i=1}^{s} \bigwedge_{j=1}^{l-1} (x_{i,j+1} = x_{i,j} + 1) \land \bigwedge_{1 \leq i < j \leq s} (x_{i,1} \neq x_{j,1}) \land \bigwedge_{i=1}^{s} \bigwedge_{j=1}^{l} Q_{a_j} x_{i,j}).$$
We can express "\( \text{Fact}(w, v) = 0 \)" by "\( \neg(\text{Fact}(w, v) \geq 1) \)" , and for \( s > 0 \), "\( \text{Fact}(w, v) = s \)" by

\[
(\text{Fact}(w, v) \geq s) \land \neg(\text{Fact}(w, v) \geq s + 1).
\]

It follows immediately that each \( \approx^i \)-class is defined by a sentence of \( \text{FO}[+1] \).

If \( w \in A^+ \) and \( 1 \leq i < j \leq |w| + 1 \), we denote by \( w[i, j] \) the factor of \( w \) beginning at the \( i \)th position and ending at the \( (j - 1) \)th position. The set \( \{i, \ldots, j\} \) is said to be the underlying set of the factor. Two factors are said to be disjoint if their underlying sets are disjoint; that is, if there is a gap between the two factors. If two factors of \( w \) are not disjoint then there is a factor \( v \) of \( w \) whose underlying set is the union of the underlying sets of the two factors—\( v \) is said to be the union of the two factors. More generally, we define the union of an arbitrary set \( S \) of factors of \( w \) to be the set \( T \) of pairwise disjoint factors such that the union of the underlying sets of \( S \) is the union of the underlying sets of \( T \).

**IV.3.2 Lemma.** Let \( w_1, w_2 \in A^+ \), \( r \geq 0 \). Let \( R = 3^r \). If \( w_1 \approx^R_{3R} w_2 \), then \( w_1 \sim_r w_2 \).

**Proof.** We shall show that if \( w_1 \approx^R_{3R} w_2 \), then Player II has a winning strategy in the \( r \)-round game in \( (w_1, w_2) \). By Theorem IV.1.3, this implies the desired result. Suppose \( i \) rounds have been played, so that pebbles labelled \( z_1, \ldots, z_i \) have been placed in each word. For each position \( m \) on which a pebble has been placed in \( w_j \) (\( j = 1, 2 \)), consider the factor

\[
w_j[m - 3^{r-i} + 1, m + 3^{r-i}] .
\]

(If \( m - 3^{r-i} + 1 < 1 \), or \( m + 3^{r-i} > |w_j| + 1 \), then we consider instead the prefix \( w_j[1, m + 3^{r-i}] \) or the suffix \( w_j[m - 3^{r-i} + 1, |w_j| + 1] \).) The union of these factors is a set of pairwise disjoint factors

\[
\{u_{i,j,1}, \ldots, u_{i,j,k_{i,j}}\}.
\]

We will describe a strategy for Player II such that for each \( 1 \leq p \leq i \), the factor \( u_{i,1,p} \) in which \( z_p \) is placed is equal to the factor \( u_{i,2,p} \) in which \( z_p \) is placed, and that the pebble is placed on the same position in the two factors. The case \( i = r \) gives the desired result, since if