Chapter 1
Introduction

Abstract This chapter gives an introduction to the present monograph as a whole. It intuitively conceptualizes regulated grammars and automata, which represent the subject of this book, and places this subject into general scientific context. It demonstrates that the subject of this book represents a vivid investigation area of today's computer science. In particular, this investigation is central to the theory of formal languages and automata, and the present chapter gives an informal insight into all the upcoming material, including crucially important concepts and results, in terms of this theory. In addition, the chapter sketches significant applications of regulated grammars and automata in several scientific areas. Some of these application areas, such as compiler writing, belong to computer science while others, such as biology, lie out of it.

Key words: introduction, formulation of the book subject, regulated grammars and automata, their conceptualization and significance, their study in formal language theory, application areas

Formal Languages and Their Regulated Models

Formal languages, such as programming languages, are applied in a great number of scientific disciplines, ranging from biology through linguistics up to informatics (see [1]). As obvious, to use them properly, they have to be precisely specified in the first place. Most often, they are defined by mathematical models with finitely many rules by which the models rewrite sequences of symbols, called strings. Over its history, the theory of formal languages have introduced a great variety of these language-defining models. The present monograph deals with their regulated versions, which are extended by additional mathematical mechanisms by which these models regulate the string-rewriting process.

This book is primarily interested in establishing the *power* of regulated language models. Secondarily, it studies algorithmic *transformations* that modify the models so they satisfy some desired *properties* while the defined languages remain unchanged. It pays a principal attention to two kinds of transformations. First, it studies the *reduction* of these models because their small size makes, in effect, the definition of languages easy-to-follow, economical and succinct. Second, whenever dealing with different types of equally powerful language models, this book is obviously interested in the *conversion* between them so they can be turned to each other at will.

In principle, formal language theory classifies language models into two basic categories—generative and recognition language models. Generative models, better known as *grammars*, define strings of their language so their rewriting process generates them from a special start symbol. On the other hand, recognition models, better known as *automata*, define strings of their language by a rewriting process that starts from these strings and ends in a prescribed set of final strings. Accordingly, this book deals with regulated versions of these two types of language models, hence its title—*regulated grammars and automata*.

Regulated Grammars

Concerning grammars, the classical formal language theory has often classified all grammars into two fundamental categories—context-free grammars and non-context-free grammars. As their name suggests, context-free grammars are based upon context-free rules, by which these grammars rewrite symbols regardless of the context surrounding them. As opposed to them, non-context-free grammars rewrite symbols according to context-dependent rules, whose application usually depends on rather strict conditions placed upon the context surrounding the rewritten symbols, and this way of context-dependent rewriting often makes them clumsy and inapplicable in practice. From this point of view, we obviously always prefer using context-free grammars, but they have their drawbacks, too. Perhaps most importantly, context-free grammars are significantly less powerful than non-context-free grammars. Considering all these pros and cons, it comes as no surprise that modern formal language theory has intensively and systematically struggled to come with new types of grammars that are underlined by context-free rules, but they are more powerful than ordinary context-free grammars. Regulated versions of context-free grammars, briefly referred to as *regulated grammars* in this book, represent perhaps the most successful and significant achievement in this direction. They are based upon context-free grammars extended by additional regulating mechanisms by which they control the way the language generation is performed.

Although regulated grammars are based upon context-free rules just like ordinary context-free grammars, they are significantly stronger than the unregulated versions. In fact, many of them are as powerful as classical non-context-free grammars with context-dependent rules. In addition, regulated grammars control their rewriting

process and, thereby, operate in a more deterministic way than unregulated context-free grammars, which work, in a general case, completely non-deterministically. Considering these indisputable advantages over classical grammars, it comes as no surprise that the theory of formal languages has paid a special attention to regulated grammars, and their investigation has introduced several important types of grammatical regulation and achieved significant results about them, most of which were published in various conference and journal papers. The present book systematically and compactly summarizes the knowledge about grammatical regulation while focusing its attention on the latest trends in the theory of regulated grammars.

However, the knowledge concerning regulated grammars is so huge that we cannot cover it all. Instead, we restrict our main attention to these two types of grammatical regulation—*context-based regulation* and *rule-based regulation*. As their names indicate, the former is based upon various context-related restrictions while the latter is underlain by restrictions placed on the use of rules during the language generation process.

Concerning context-based regulated grammars (see Chapter 4), this monograph primarily deal with two types of these grammars—conditional grammars and scattered context grammars, sketched next.

- In a *context-conditional grammar* G, a set of permitting strings and a set of forbidding strings are assigned to every rule. During a derivation step, a rule like this is applicable provided that all its permitting strings occur in the rewritten sentential form while all the forbidding strings do not occur there.
- A *scattered context grammar* G is based on finite sequences of context-free rules. By rules of this form, G simultaneously rewrites several nonterminals during a single derivation step.

Concerning rule-based regulated grammars (see Chapter 5), this book covers four crucially important types of these grammars—regular-controlled grammars, matrix grammars, programmed grammars, and state grammars. Next, we give their gist. Notice how they differ in their regulating mathematical mechanisms as well as in the way they apply these mechanisms.

- A *regular-controlled grammar* G is extended by a regular control language defined over the set of rules in G. A terminal string x is in the language generated by G if and only if the control set contains a control string according to which G generates x.
- A *matrix grammar* G is extended by a finite language defined over the set of rules in G, and the strings in this language are referred to as matrices. In essence, G makes a derivation so it selects a matrix, and after this selection, it applies all its rules one by one until it reaches the very last rule. Then, it completes its derivation, or it makes another selection of a matrix and continues the derivation in the same way.
- A *programmed grammar* G has a finite set of rules attached to each rule r. If G applies r during a derivation step, then during the next derivation step, it has to apply a rule from its attached set.

- A *state grammar G* is extended by an additional state mechanism that strongly resembles a finite-state control of finite automata. During every derivation step, *G* applies a rule in a left-fashion way, and in addition, it moves from a state to another state, which influences the choice of the rule to be applied in the next step.

Apart from these fundamentals of grammatical regulation, the text discusses several closely related topics, such as special cases of context-based regulation (see Chapter 6), the role of erasing rules in this regulation (see Chapter 7), generation of languages extended by extra symbols (see Chapter 8), grammatical regulation in parallel (see Chapters 10 and 11), algebraically regulated grammars (see Chapters 9 and 12), and sets of mutually communicating regulated grammars (see Chapters 13 and 14).

Regulated Automata

Concerning automata, this book studies regulated versions of finite and pushdown automata, which are central to formal language theory.

This theory conceptualizes the general notion of a *finite automaton M* as a strictly finitary model—that is, all components of *M* are of a fixed size, and none of them can be extended during the course of the rewriting process. More precisely, *M* consists of an input tape, a read head, and a finite state control. The input tape is divided into squares. Each square contains one symbol of an input string *w*. The symbol under the read head is the current input symbol. The finite control is represented by a finite set of states together with a relation, specified as a set of computational rules in this book. On *w*, *M* works by making moves. Each move is made according to a computational rule that describes how the current state is changed and whether the current input symbol is read. If the symbol is read, the read head is shifted one square to the right; otherwise, the read head is kept stationary. The main task of *M* is to decide whether *w* is accepted. For this purpose, *M* has one state defined as the start state and some states designated as final states. If *M* can read *w* by making a sequence of moves from the start state to a final state, *M* accepts *w*; otherwise, *M* rejects *w*. The language of *M* consists of all strings that *M* accepts in this way.

A *pushdown automaton* represents a finite automaton extended by a potentially infinite pushdown list. During a move, according to one of its rules, it reads a symbol, changes the current state, and rewrites the symbol occurring on the pushdown top. If it reads the entire input string, empties the pushdown list and enters a final state, the automaton accepts the input string; the set of all strings accepted in this way is the language that the automaton accepts.

Concerning regulated finite and pushdown automata, this monograph covers the following two fundamental types.

- *Self-regulating automata* select a rule according to which the current move is made based upon the rule applied during the previous move (see Chapter 15). As

obvious, this regulation strongly resembles the regulation of programmed grammars.

- *Language-controlled automata* regulate the application of rules by control languages by analogy with context-free grammars regulated by control languages (see Chapter 16). This book considers finite and pushdown automata regulated by control languages from various language families.

In addition, this monograph covers modified versions of finite and pushdown automata closely related to regulated automata, too. Specifically, it presents *jumping finite automata* that work just like classical finite automata except that they can, after reading a symbol, jump in either direction within their input tapes and continue making moves from there (see Chapter 17). Furthermore, it covers *deep pushdown automata* that can make expansions deeper in the pushdown, not just on the very pushdown top (see Chapter 18).

Applications

Although this book primarily deals with regulated language-defining models from a theoretical viewpoint, it sketches their applications, too. It narrows its attention to the applications of regulated grammars rather than regulated automata, and it restricts their coverage only to three application areas—computational linguistics, molecular biology, and compiler writing. It describes these applications and their perspectives from a general viewpoint (see Chapter 19). In addition, it presents many case studies to show quite specific real-world applications concerning the three scientific fields mentioned above (see Chapter 20).

References

[1] Rozenberg, G., Salomaa, A. (eds.): Handbook of Formal Languages, Volumes 1 through 3. Springer, New York (1997)