Grammars with Context Conditions and Their Applications

ALEXANDER MEDUNA
http://www.fit.vutbr.cz/~meduna
meduna@fit.vutbr.cz

Department of Information Systems
Faculty of Information Technology
Brno University of Technology, Czech Republic, Europe

NOTE: This paper is based on the author's talk at the workshop

Group, Ring, Language and Related Areas in Computer Science

at The Research Institute for Mathematical Sciences (RIMS) organised as an event entitled *Group, Ring, Language and Related Areas in Computer Science* held from 15 February 2023 to 17 February 2023 at the Kyoto University, Japan. The paper is published in the Proceedings of our Symposium in the "RIMS Kokyuroku" series.

Abstract: In the classical formal language theory, we can divide grammatical productions into context-dependent and context-independent productions, and based on this division, we can naturally distinct context-dependent grammars, such as phrase-structure grammars, from context-independent grammars, such as context-free grammars. Making a derivation step according to context-dependent productions depends on rather strict conditions satisfied by the context surrounding the rewritten symbol, while making a step according to context-independent productions does not, so from this point of view, we obviously always prefer using context-independent grammars to the others. Unfortunately, compared to context-dependent grammars, context-independent grammars are significantly less powerful; in fact, most of them are incapable of grasping some aspects of quite common programming languages. On the other hand, most context-dependent grammars are equivalent to Turing machines, and this remarkable power represents their indisputable advantage. These pros and cons inspired the modern language theory to introduce some new grammars that simultaneously satisfy these properties:

(1) They are based on context-independent productions.
(2) Their context conditions are significantly simpler than the strict conditions of classical context-dependent productions.
(3) They are as powerful as classical context-dependent grammars.

In this paper, we overview the most essential types of these grammars, whose alternative context conditions can be classical into these three categories:

(A) Context conditions placed on derivation domains.
(B) Context conditions placed on the use of productions.
(C) Context conditions placed on the neighborhood of the rewritten symbols.

As already pointed out, we want the context conditions to be as small as possible. Therefore, we concentrate the investigation on the reduction of context conditions. Specifically, it reduces the number of some of their components, such as nonterminals or productions. It studies how to achieve this reduction without any decrease in this generative power, which coincides with the power of Turing machines. By achieving this reduction, it makes the partially parallel rewriting more succinct and economical, and this economization is obviously highly appreciated both from a practical and theoretical standpoint.

Regarding each of the grammars discussed, we introduce and study their parallel and sequential versions, which represent two basic approaches to grammatical rewriting in today's formal language theory. That is, during a sequential derivation step, a grammar rewrites a single symbol in the current sentential form, while during a parallel derivation step, a grammar rewrites all symbols. As context-free and E0L grammars represent perhaps the most fundamental sequential and parallel grammars, respectively, we usually base the discussion of sequential and parallel rewriting upon them.

In addition, we illustrate the applications of grammars with context conditions by biologically oriented examples.

This paper is based upon the author's latest book *Modern Language Models and Computation*: *Theory with Applications* (written along with his Ph.D. student Ondřej Soukup), Springer, 2017. *Handbook of Mathematical Models for Languages and Computation* (written along with his Ph.D. students Petr Horáček and Martin Tomko), The Institution of Engineering and Technology.

Keywords: formal grammars; context conditions; basic properties; language families; reduction; simplification; descriptional complexity; applications; perspectives.

1. Introduction

Formal languages play a crucial role in many areas of computer science, ranging from compilers through mathematical linguistics to molecular genetics. In dealing with these languages, we face the problem of choosing appropriate models in the order to capture their structure elegantly and precisely. By analogy with the specification of natural languages, we often base these models on suitable grammars. A grammar generates its language by performing derivation steps that change strings, called sentential forms, to other strings according to its grammatical productions.

During a derivation step, the grammar rewrites a part of its current sentential form with a string according to one of its productions. If in this way it can make a sequence of derivation steps from its start symbol to a sentential form consisting of terminal symbols, that is, the symbols over which the language is defined, the resulting sentential form is called a sentence and belongs to the generated language. The set of all sentences made this way is the language generated by the grammar. In classical formal language theory, we can divide grammatical productions into context-dependent and context-independent productions. Based on this division, we can make a natural distinction between context-dependent grammars, such as phrase-structure grammars, and context-independent grammars, such as context-free grammars.

The derivation step by context-dependent productions depends on rather strict conditions, usually placed on the context surrounding the rewritten symbol, while the derivation step by context-independent productions does not have any restrictions. For this reason, we tend to use context-independent grammars. Unfortunately, compared to context-dependent grammars,

context-independent grammars are far less powerful; in fact, most of these grammars are incapable of grasping some basic aspects of common programming languages. On the other hand, most context-dependent grammars are as powerful as Turing machines, and this remarkable power gives them an indisputable advantage. From a realistic point of view, classical context-independent and context-dependent grammars have some other disadvantages.

Consider, for instance, English. Context-independent grammars are obviously incapable of capturing all the contextual dependencies in this complex language. However, we may find even the classical context-dependent grammars clumsy for this purpose. To illustrate, in an English sentence, the proper form of a verb usually depends on the form of the subject. For instance, we write I do it, not I does it, and it is the subject, I, that implies the proper form of do. Of course, there may occur several words, such as adverbs, between the subject and the verb. We could extend "I do it to" "I often do it", "I very often do it" and infinitely many other sentences in this way.

At this point, however, the classical context-dependent productions whose conditions are placed on the context surrounding the rewritten symbol are hardly of any use. The proper form of the verb follows from a subject that does not surround the verb at all; it can occur many words ahead of the verb. To overcome the difficulties and, at the same time, maintain the advantages described above, modern language theory has introduced some new grammars that simultaneously satisfy these three properties:

- They are based on context-independent productions.
- Their context conditions are significantly more simple and flexible than the strict condition placed on the context surrounding the rewritten symbol in the classical context-dependent grammars.
- They are as powerful as classical context-dependent grammars.

In this paper, we give an overview of the most essential types of these grammars. Their alternative context conditions can be classified into these three categories:

- Context conditions placed on derivation domains.
- Context conditions placed on the use of productions.
- Context conditions placed in the neighborhood of the rewritten symbols.

As already pointed out, we want the context conditions to be as small as possible. For this reason, we pay a lot of attention to the reduction of context conditions in this paper. Specifically, we reduce the number of their components, such as the number of nonterminals or productions.

We study how to achieve this reduction without any decrease of their generative power, which coincides with the power of the Turing machines. By achieving this reduction, we actually make the grammars with context conditions more succinct and economical, and these properties are obviously highly appreciated both from a practical and theoretical standpoint.

Regarding each of the discussed grammars, we introduce and study their parallel and sequential versions, which represent two basic approaches to the grammatical generation of languages in today's formal language theory. To be more specific, during a sequential derivation step, a grammar rewrites a single symbol in the current sentential form, whereas during a parallel derivation step, a grammar rewrites all symbols. As context-free and E0L grammars

represent perhaps the most fundamental sequential and parallel grammars, respectively, we usually base the discussion of sequential and parallel generation of languages on them.

2. Results

Context Conditions Placed on Derivation Domains

In the formal language theory, the relation of a direct derivation is introduced over V*, where V is the total alphabet of a grammar. Algebraically speaking, V* is thus defined over the free monoid whose generators are symbols. We modify this definition by using strings rather than symbols as the generators. More precisely, we introduce this relation over the free monoid generated by a finite set of strings; in symbols, it is defined over W*, where W is a finite language. As a result, this modification represents a very natural context condition: a derivation step is performed on the condition that the rewritten sentential form occurs in W*.

This context condition results in a large increase of generative power of both the sequential and parallel context-independent grammars, represented by context-free grammars and E0L grammars, respectively. In fact, even if W contains strings consisting of no more than two symbols, the resulting power of these grammars is equal to that of Turing machines.

Context Conditions Placed on the Use of Productions

Furthermore, we discuss grammars with context conditions represented by strings associated with productions. We distinguish between two types of these conditions: forbidding conditions and permitting conditions. A production is applicable to a sentential form if each of its permitting conditions occurs in the sentential form, and any of its forbidding conditions does not. We study sequential grammars with context conditions, originally introduced by van der Walt in 1970. In addition, we introduce and discuss parallel versions of these grammars. In both sections, we demonstrate that this concept of context conditions attached to grammatical productions significantly increases the grammatical generative power. Furthermore, in some grammars, we explain how to reduce the number of conditional productions, the length of context conditions, and the number of nonterminals.

Context Conditions Placed on the Neighborhood of Rewritten Symbols

This chapter studies grammars with context conditions placed on the neighborhood of rewritten symbols. First, we investigate grammars with context conditions that strictly require a continuous neighborhood of the rewritten symbols. We discuss both sequential and parallel grammars of this kind. The discussion of sequential grammars naturally leads to the study of classical context-dependent grammars, such as context-sensitive and phrase-structure grammars. Regarding parallel grammars, we base this discussion on EIL grammars. Then, we study scattered context grammars in which rewriting depends on symbols occurring in the sentential form, but these symbols may not form a continuous substring of the sentential form. Rather, these symbols, which are simultaneously rewritten during a single derivation step, may be scattered throughout the sentential form. In all grammars discussed in this chapter, we make their context-dependency uniform, reduced, and easy-to-use in theory and practice.

3. Applications

Although this paper primarily represents a theoretically oriented treatment, most grammars discussed in the previous chapters have realistic applications. Indeed, these grammars are useful to every scientific field that formalizes its results by some strings and studies how these strings are produced from one another under some permitting or, in contrast, forbidding conditions. As numerous areas of science formalize and study their results in this way, any description of applications that cover more than one of these areas would be unbearably sketchy, if not impossible. Therefore, we concentrate our attention on a single application area, microbiology, which appears to be of great interest at present. In this intensively investigated scientific field, we next give three examples with the figures that model the development of plants.

For first example, let us assume that the red alga occurs in some unhealthy conditions in which only some of its parts survive, while the rest dies. This dying process starts from the newly born marginal parts of branches, which are too young and weak to survive, and proceeds toward the older parts, which are strong enough to live under these conditions. To be quite specific, all the red alga parts become gradually dead except for the parts denoted by 2s and 8s. This process can be specified by 0L grammar with forbidding conditions, and Figure 1 shows the dying process corresponding to the next derivation, whose last eight strings correspond to stages (a) through (h) in the figure.

For next example, imagine a situation where the red alga has degenerated. During this degeneration, only the main stem was able to give a birth to new branches while all the other branches lengthened themselves without any branching out. This degeneration could specified by the forbidding 0L grammar, where Figure 2 pictures the degeneration specified by the following derivation, in which the last 10 strings correspond to stages (a) through (j) in the figure.

Lastly, the third example illustrates the idea of plants simulated by parametric 0L grammars with permitting conditions. We demonstrate a plant development with a resource flow controlled by the number of apexes, so it erases the apex and generates two new internodes terminated by apexes. Figure 3 shows 15 developmental stages of a plant simulation based on this model. From the presented example, we see that with permitting conditions, parametric 0L grammars can describe sophisticated models of plants in a very natural way. Particularly, compared to the context-sensitive L grammars, they allow one to refer to modules that are not adjacent to the rewritten module, and this property makes them more adequate, succint, and elegant.

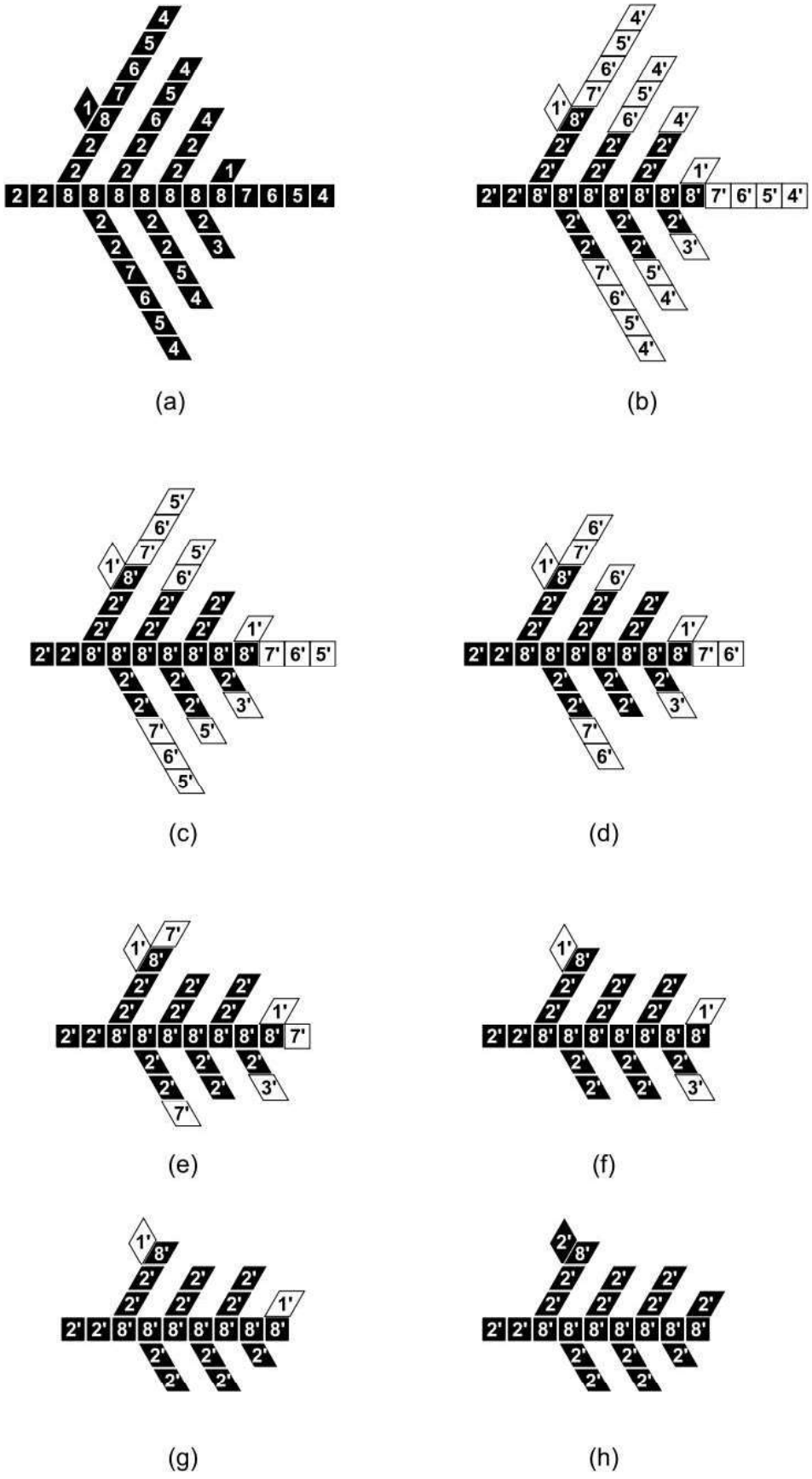

Figure 1: Death of marginal branch parts

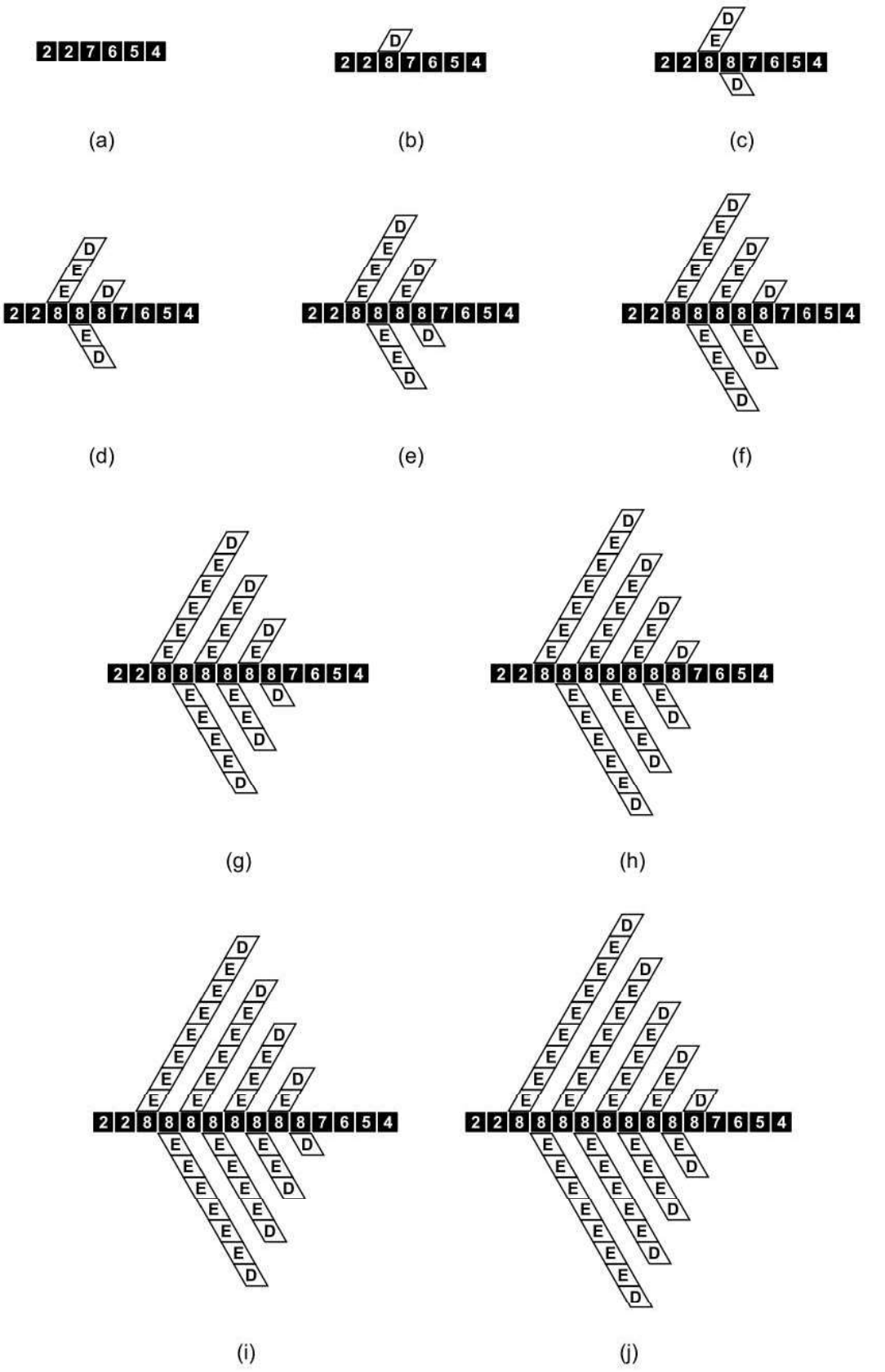

Figure 2: Degeneration.

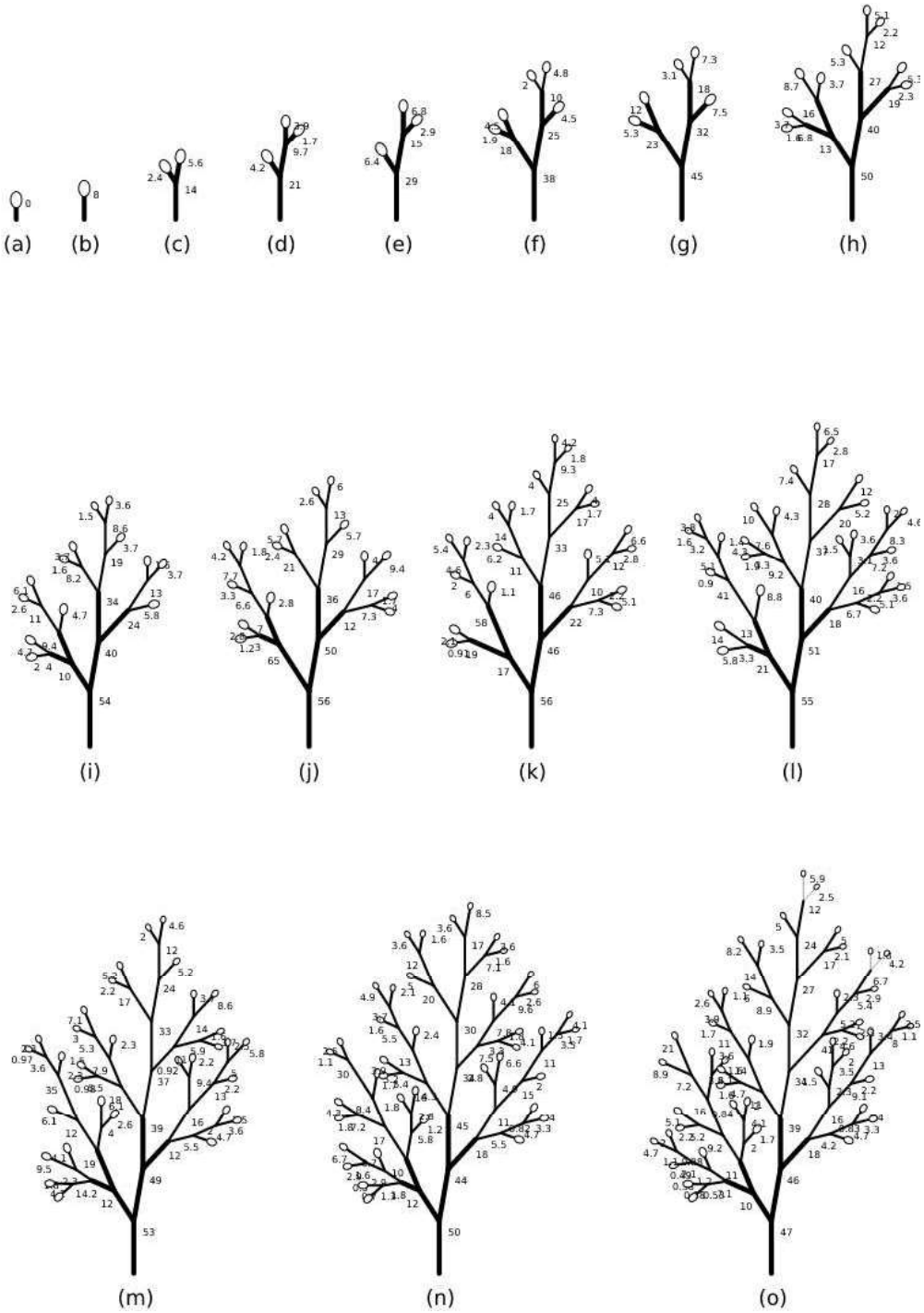

Figure 3: Developmental stages of the plant.

Reference

Meduna, Alexander; Horáček, Petr; *Tomko*, *Martin* (2020): *Handbook of Mathematical Models for Languages and Computation*, The Institution of Engineering and Technology. *ISBN 978-1-78561-660-0.*

About the author:

Author: Prof. Alexander Meduna (born 1957 in Olomouc, Czech Republic) is a theoretical computer scientist and expert on compiler design, formal languages and automata. He is a full professor of Computer Science at the Brno University of Technology. Formerly, he taught theoretical computer science at various European and American universities, including the University of Missouri, where he spent a decade teaching advanced topics of formal language theory. He wrote more than ninety papers related to theoretical computer science. His books include

- *Meduna, Alexander (2000). Automata and Languages: Theory and Applications. Springer Science & Business Media. ISBN 9781852330743.*
- *Meduna, Alexander (2007). Elements of Compiler Design. CRC Press. ISBN 9781420063233.*
- *Meduna, Alexander (2014). Formal Languages and Computation: Models and Their Applications. CRC Press. ISBN 9781466513457.*
- *Meduna, Alexander; Švec, Martin (2005). Grammars with Context Conditions and Their Applications. John Wiley & Sons. ISBN 9780471736554.*
- *Meduna, Alexander; Techet, Jiří (2010). Scattered Context Grammars and Their Applications. WIT Press. ISBN 9781845644260.*
- *Meduna, Alexander; Zemek, Petr (2014). Regulated Grammars and Automata. Springer. ISBN 9781493903696.*
- *Meduna, Alexander; Soukup, Ondřej (2017). Modern Language Models and Computation: Theory with Applications. Springer. ISBN 9783319630991.*
- *Meduna, Alexander; Horáček, Petr*; *Tomko*, *Martin* (2020): *Handbook of Mathematical Models for Languages and Computation*, The Institution of Engineering and Technology. *ISBN 978-1-78561-660-0.*

Acknowledgements

This work was supported by the Research Institute for Mathematical Sciences, an International Joint Usage/Research Center located in Kyoto University. Martin Havel and Prof. Tsunekazu Nishinaka made excellent suggestions concerning the first version of this work.