BestMasters

RESEARCH

Laura Wirth

Weighted Automata, Formal Power Series and Weighted Logic

Springer Spektrum

BestMasters

Mit **"BestMasters"** zeichnet Springer die besten Masterarbeiten aus, die an renommierten Hochschulen in Deutschland, Österreich und der Schweiz entstanden sind. Die mit Höchstnote ausgezeichneten Arbeiten wurden durch Gutachterinnen und Gutachter zur Veröffentlichung empfohlen und behandeln aktuelle Themen aus unterschiedlichen Fachgebieten der Naturwissenschaften, Psychologie, Technik und Wirtschaftswissenschaften. Die Reihe wendet sich an Personen aus Praxis und Wissenschaft gleichermaßen und soll insbesondere auch dem wissenschaftlichen Nachwuchs Orientierung geben.

Springer awards **"BestMasters"** to the best master's theses which have been completed at renowned Universities in Germany, Austria, and Switzerland. The studies received highest marks and were recommended for publication by supervisors. They address current issues from various fields of research in natural sciences, psychology, technology, and economics. The series addresses practitioners as well as scientists and, in particular, offers guidance for early stage researchers.

Laura Wirth

Weighted Automata, Formal Power Series and Weighted Logic

Springer Spektrum

Laura Wirth
University of Konstanz
Konstanz, Germany

ISSN 2625-3577 ISSN 2625-3615 (electronic)
BestMasters
ISBN 978-3-658-39322-9 ISBN 978-3-658-39323-6 (eBook)
https://doi.org/10.1007/978-3-658-39323-6

Responsible Editor: Marija Kojic
This Springer Spektrum imprint is published by the registered company Springer Fachmedien Wiesbaden GmbH, part of Springer Nature.
The registered company address is: Abraham-Lincoln-Str. 46, 65189 Wiesbaden, Germany

To my mom, for always being by my side,
and in loving memory of Opa Helmut, the biggest fan of my studies.

Acknowledgements

I would like to express my deepest gratitude to my supervisor Prof. Dr. Salma Kuhlmann for her patient guidance throughout my studies. Special thanks shall also go to my second supervisor Prof. Dr. Sven Kosub, whom I could always ask to clarify computer-scientific aspects of any kind. I had the advantage of being able to ask both of them for advice and answers to resolve difficulties at all times. Their remarks and comments have been a valuable help and powerful encouragement.

Furthermore, I would like to thank Prof. Dr. Manfred Droste for our exchange regarding Section 2.5. I could ask him to obtain references, and he gave me a hint that finally resulted in a proof of Theorem 2.5.19, which generalizes the classical result of Büchi, Elgot and Trakhtenbrot.

I owe a particular debt to Dr. Lothar Sebastian Krapp, who calmly supported me in overcoming all problems that arose during this work. His deep mathematical insight has given rise to many invaluable remarks and comments, which have essentially improved the presentation in many respects. In particular, I wish to thank him for taking the trouble to attentively proofread several iterations of this work. Beyond that, I am lucky enough to have him as a friend.

Most warmly, I thank my family – my sister Anna-Lena, my parents Bernd and Manuela, as well as Oma Anne and Oma Ria – for their less mathematical but more loving and emotional support, while I was studying and writing obscure theses that none of them is likely to ever read.

Finally, I wish to thank my fellow students Carl Eggen, Moritz Link, Patrick Michalski, including Philipp Huber, with whom I could share my enthusiasm for the subject of this work.

Abstract

A basic concept from Theoretical Computer Science for the specification of formal languages are finite automata. By equipping the states and transitions of these finite automata with weights, one obtains the quantitative model of weighted automata. The included weights may model e.g. the amount of resources needed for the execution of a transition, the involved costs, or the reliability of its successful execution. To obtain a uniform model, the underlying weight structure is usually modeled by an abstract semiring. The behavior of a weighted automaton is then represented by a formal power series. A formal power series is defined as a map assigning to each word over a given alphabet an element of the semiring, i.e. some weight associated with the respective word.

In this work, we put emphasis on the expressive power of weighted automata. More precisely, the main objective is to represent the behaviors of weighted automata by expressively equivalent formalisms. These formalisms include rational operations on formal power series, linear representations by means of matrices, and weighted monadic second-order logic.

To this end, we first exhibit the classical language-theoretic results of Kleene, Büchi, Elgot and Trakhtenbrot, which concentrate on the expressive power of finite automata. We further derive a generalized version of the Büchi–Elgot–Trakhtenbrot Theorem addressing formulas, which may have free variables, whereas the original statement concerns only sentences. Then we use the language-theoretic approaches and methods as starting point for our investigations with regard to formal power series. We establish Schützenberger's extension of Kleene's Theorem, referred to as Kleene–Schützenberger Theorem. Moreover, we introduce a weighted version of monadic second-order logic, which is due to Droste and Gastin, and analyze its expressive power. By means of this weighted logic, we derive an extension of the Büchi–Elgot–Trakhtenbrot Theorem. Thus, we point out relations among the different specification approaches for formal power series. Further, we relate the notions and results concerning formal power series to their respective counterparts in Language Theory.

Overall, our investigations shed light on the interplay between languages, classical as well as weighted automata, formal power series and monadic second-order logic. Hence, the topic of this work lies at the interface between Theoretical Computer Science, Algebra and Logic or, more generally, Model Theory.

Contents

1. Introduction

A fundamental concern of Computer Science is the study and processing of information and data. Any information requires to be specified, and depending on the application context, a suitable representation or specification is chosen in order to interpret the information in a targeted manner. More precisely, the processing of data, and in particular the amount of resources required for this, depends on the chosen method or formal model for the representation of the conveyed information. Especially when working with *infinite* objects or structures, it is essential to have a *finite* specification of the conveyed information. In this work, we focus on the description-oriented research in Theoretical Computer Science, rather than on algorithm-oriented results. However, the two research areas really are highly, even inseparably, connected.

The information conveying structures that are considered in this work are formal languages and formal power series. Formal languages are sets of words over a given alphabet, and the formal power series, which concern us, are maps associating elements of an abstract semiring to words over a given alphabet. Thus, languages model qualitative information concerning the membership of words, whereas formal power series provide quantitative data about words. In fact, formal power series can be regarded as quantitative extensions of languages.

In Theoretical Computer Science, the basic tool for the specification of languages are finite automata. Historically, finite automata originate in the mid-1950s in the work of Kleene [18]. In his fundamental result, which is commonly referred to as Kleene's Theorem, Kleene characterized the languages that are recognizable by finite automata as rational languages[1]. Mainly motivated by decidability questions, the expressive equivalence of finite automata and monadic second-order logic was derived independently by Büchi [4], Elgot [15] and Trakhtenbrot[2] [43] in the early 1960s. Their equivalence result, referred to as Büchi–Elgot–Trakhtenbrot Theorem, establishes a very early connection between the theory of finite automata and Mathematical Logic. The two approaches often complement each other in a synergetic way, and their relation is highly relevant for multiple application domains, e.g. in verification and knowledge representation, for the design of combinatorial and sequential circuits, as well as in natural language processing. In Theorem 2.5.19, we further present an extension of the Büchi–Elgot–Trakhtenbrot Theorem to monadic second-order formulas, which may have free variables, whereas the original result addresses only sentences. The proof of

[1] A language is called rational if it can be constructed from finitely many finite languages by applying the rational operations union, concatenation and Kleene star.

[2] Due to different transliteration from the Cyrillic, various spellings of this name are common in Latin script.

L. Wirth, *Weighted Automata, Formal Power Series and Weighted Logic*,
BestMasters, https://doi.org/10.1007/978-3-658-39323-6_1

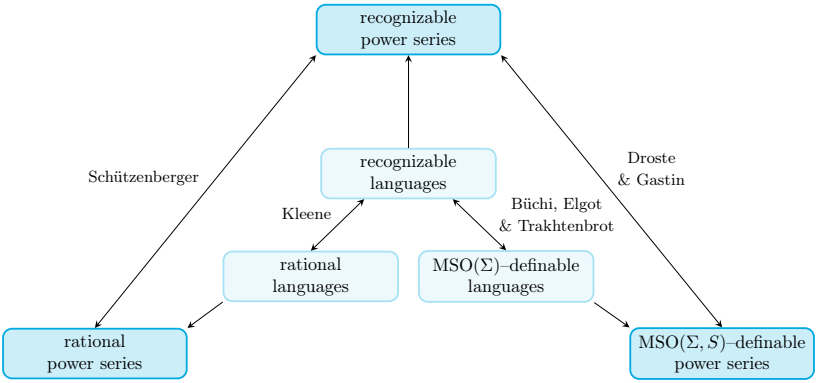
Theorem 2.5.19 is a new contribution of this work[3]. Overall, we are provided with three expressively equivalent tools for the specification of languages: finite automata, rational operations, and monadic second-order logic.

However, the main focus of this work concentrates on the study of formal power series. On the one hand, formal power series support the modeling of quantitative phenomena, e.g. the vagueness or uncertainty of a statement, length of time periods, or resource consumption, whereas languages are not suited to account such subtleties. On the other hand, formal power series constitute a powerful tool in relation to languages and automata, since they, in a sense, lead to the *arithmetization* of the theory. In addition, formal power series are of interest in various branches of Mathematics, particularly in Algebra. We therefore consider extensions of the above-mentioned language-theoretic specification tools to the realm of formal power series. By equipping the transitions and states of classical finite automata with weights, we arrive at the concept of weighted automata. The behavior of a weighted automaton is represented by a formal power series. Weighted automata provide a quantitative model in the sense that they take into account weights associated to computations. Therefore, weighted automata are employed for the description of quantitative properties in various areas such as image compression, probabilistic systems and speech-to-text processing. The notion of a weighted automaton was first introduced by Schützenberger [37] in 1961. Furthermore, Schützenberger was the first to investigate rational formal power series in the context of languages and automata. In particular, he proved an extension of Kleene's Theorem, referred to as Kleene–Schützenberger Theorem. The results of Kleene, Büchi, Elgot, Trakhtenbrot and Schützenberger have inspired a wealth of extensions as well as further research, and also led to recent practical applications, e.g. in verification of finite-state programs, in digital image compression and in speech-to-text processing (cf. Droste and Gastin [6, page 69]). In 2005, Droste and Gastin [5], [6], [10] established a quantitative extension of the Büchi–Elgot–Trakhtenbrot Theorem by introducing a weighted version of monadic second-order logic. More precisely, they could prove that, under certain assumptions, particular restrictions of their weighted logic are expressively equivalent to weighted automata. From a theoretical point of view, the logical description by means of classical as well as weighted monadic second-order formulas is particularly interesting. The formalism of monadic second-order logic can be applied for the abstract investigation of languages and formal power series, respectively, in order to derive theoretical results. On the other hand, formulas can be used to formalize qualitative as well as quantitative requirements or properties of systems.

As the central concepts under consideration are weighted automata, formal power series and weighted monadic second-order logic, the topic of this work is situated at the interface between Theoretical Computer Science, Algebra and Logic or, more generally, Model Theory. The main objective of this work is to "build bridges", i.e. to establish connections between the different notions and formalisms. On the one hand, we draw connections between the classical results from Language Theory and

[3] A self-contained proof of this generalized version of the Büchi–Elgot–Trakhtenbrot Theorem has not been available in the literature. The proof presented in this work is based on a suggestion of Manfred Droste, for which we are very grateful.

the ones concerning formal power series. On the other hand, we study the expressive power of the different approaches for the specification of languages and formal power series, respectively, and relate them to each other. More precisely, we derive that all approaches are expressively equivalent. Graphically, the main results in this work can be summarized in the following diagram:



Next, we give a short overview by outlining the contents of this work. More detailed introductions are given at the beginning of the respective chapters and sections.

In Chapter 2, we start by gathering some general preliminaries on words and languages, which are the underlying structures for the entirety of this work. Furthermore, we consider the three above-mentioned approaches for the specification of languages: finite automata, rational operations and monadic second-order logic. We assume the reader to be familiar with the basics of Mathematical Logic, and particularly of Model Theory. However, Appendix A.1 provides a detailed introduction to the fundamental model-theoretic notions. Our treatment of monadic second-order logic in the context of words and languages is complemented by Appendix A.2. We refer to the corresponding section of Appendix A whenever it is convenient. In Theorem 2.3.6 we present the classical result of Kleene [18] showing that the sets of recognizable and rational languages are identical. Theorem 2.5.1 establishes the expressive equivalence of automata and monadic second-order logic, which is due to Büchi [4], Elgot [15] and Trakhtenbrot [43]. Moreover, we derive an extension of the Büchi–Elgot–Trakhtenbrot Theorem to formulas in Theorem 2.5.19. The corresponding proof presented at the end of Chapter 2 is a new contribution of this work[4].

The language-theoretic notions, methods and results are then used as starting point for our investigations in the subsequent chapters. At each point, we attempt to relate the respective extensions to their language-theoretic counterpart.

[4] A self-contained proof of this generalized version of the Büchi–Elgot–Trakhtenbrot Theorem has not been available in the literature. The proof presented in this work is based on a suggestion of Manfred Droste, for which we are very grateful.

Our study of weighted automata starts in Chapter 3. More precisely, we first introduce the fundamental theory of semirings and formal power series. We present several particular semirings that will occur throughout the whole work and serve as examples of possible weight structures. We then formally introduce the notion of a weighted automaton and its behavior. As we will demonstrate, these quantitative notions can be regarded as natural generalizations of finite automata and recognizable languages, respectively. However, we also shed light on some differences. Finally, we represent weighted automata in terms of matrices. These linear representations provide a further, more algebraic approach for the specification of formal power series. We prove that this approach even has the same expressive power as weighted automata.

Chapter 4 is devoted to the investigation of rational formal power series. To this end, we first define operations on formal power series having language-theoretic operations as natural counterparts. We further show that these operations yield a natural generalization of rational languages in the realm of formal power series. In particular, we derive an extension of Kleene's Theorem, which is due to Schützenberger [37] (see Theorem 4.3.1). More precisely, we first proceed by structural induction to show that every rational power series can be represented as the behavior of a weighted automaton. We then exploit interconnections between rational power series and linear systems of equations to prove, conversely, that the behavior of any weighted automaton is a rational power series.

The main subject of Chapter 5 is the logical examination of weighted automata and their behaviors. We introduce a weighted version of monadic second-order logic, which is due to Droste and Gastin [5]. By employing the formalism of weighted monadic second-order formulas, we obtain another method for the specification or representation of formal power series. However, we will demonstrate by the help of examples that the semantics of weighted formulas are in general more expressive than the behavior of weighted automata. Therefore, we restrict the weighted logic and introduce several fragments of it. By outlining the main steps of Droste and Gastin [6], we then derive that for commutative semirings the behaviors of weighted automata are precisely the power series definable by restricted weighted sentences (see Theorem 5.2.1). Ultimately, we introduce the notion of locally finite semirings, and we show that for this large class of semirings also unrestricted sentences define recognizable power series. Hence, locally finite commutative semirings admit an "unrestricted" generalization of the Büchi–Elgot–Trakhtenbrot Theorem (see Theorem 5.2.20).

Finally, in Chapter 6 we collect the main results and list the correspondences we establish throughout this work. Moreover, we indicate several research directions that tie into the theory presented in this work. These guide towards further work.

This work is intended to composed in a largely self-contained manner. Thus, we try to give a complete treatment of the included concepts as far as they are used in this work. Apart from the basics of Model Theory, which are presented in Appendix A, no particular previous knowledge from graduate level Mathematics and Computer Science is assumed. Whenever further notions or results are needed, they will be fully described or at least referenced. Furthermore, in order to illustrate the expressive power and

diversity of languages, formal power series, and their respective representation tools, for each topic we present various examples. We hope that these fulfill the purpose of pointing out possible application scenarios of the theory.

For the rest of this work, we let $\mathbb{N} = \{0, 1, 2, 3, \dots\}$ be the set of natural numbers and we denote by $\mathbb{N}^+ := \mathbb{N} \setminus \{0\}$ the set of positive natural numbers.

2. Languages, Automata and Monadic Second-Order Logic

Although the main subject of this work is to examine formal power series, weighted automata and weighted monadic second-order logic, we start by considering the classical notions and results in the context of languages, finite automata and monadic second-order logic. These classical formalisms are the starting point of those in the weighted setting that will be considered in the subsequent chapters. Throughout this chapter, we further establish a clear overview over the classical results from Theoretical Computer Science, whose extensions and generalizations we will derive in the subsequent chapters. Thus, this chapter forms the basis of the whole work, as it provides all necessary foundations and preliminaries. More detailed introductions are given at the beginning of each section.

In Section 2.1, we introduce the most fundamental notions from Formal Language Theory. Section 2.2 is devoted to the study of finite automata and recognizable languages. In particular, we present a list of several decidability properties of finite automata and recognizable languages, respectively. In Section 2.3, we treat rational languages, prove several closure properties of recognizable languages and state Kleene's Theorem. In order to state and prove the Büchi–Elgot–Trakhtenbrot Theorem in Section 2.5, we study monadic second-order logic and its connection to words and languages in Section 2.4. We conclude this chapter with a Büchi–Elgot–Trakhtenbrot type result for formulas and not just for sentences. As a complete proof for this result has not been available in the literature, the proof presented at the end of Section 2.5 is a new contribution of this work.

2.1. Words and Formal Languages

> *"Formal language theory is—together with automata theory (which is really inseparable from language theory)—the oldest branch of theoretical computer science. In some sense, the role of language and automata theory in computer science is analogous to that of philosophy in general science: it constitutes the stem from which the individual branches of knowledge emerge."*
>
> Salomaa [36, page 105]

This section gathers the fundamental structures and preliminary notions in the context of Formal Language Theory that will be used throughout this work. One could say that words and languages are the playthings in this chapter, as the subsequent sections

L. Wirth, *Weighted Automata, Formal Power Series and Weighted Logic*,
BestMasters, https://doi.org/10.1007/978-3-658-39323-6_2

are devoted to the introduction of modeling approaches to represent them. However, the classical notions and their properties, which are introduced in this section, also play a central role in the weighted setting, with which the subsequent chapters are concerned.

We set up notation and terminology mainly following Sakarovitch [33, Chapter 0] and Droste [11, § 1].

2.1.1 Definition. An **alphabet** is a non-empty finite set Σ. The elements of an alphabet are called **letters** or **symbols**.

2.1.2 Example. The Latin alphabet $\Sigma = \{a, A, b, B, c, C, \ldots, z, Z\}$ is probably the first alphabet one thinks of. However, elements of an alphabet may also be

- numbers, e.g. $\Sigma = \{0, 1\}$,
- words, e.g. $\Sigma = \{\texttt{bye}, \texttt{tschuess}, \texttt{ciao}, \texttt{farvel}\}$,
- or various symbols, e.g. $\Sigma = \{\mathrm{A}, \mathrm{B}, 1, 2, 3, 4, +, \times, \text{☺}\}$.

In the upcoming examples, we usually work with (subsets of) the alphabet

$$\Sigma = \{a, b, c, d\}.$$

2.1.3 Definition. Let Σ be an alphabet.

a) A (finite) **word** over Σ is a finite sequence of the form

$$w = a_1 \ldots a_n$$

with $n \in \mathbb{N}$ and $a_1, \ldots, a_n \in \Sigma$. The word obtained for $n = 0$ is referred to as the **empty word** and is denoted by ε. Words are also called **strings**.

b) Let $w = a_1 \ldots a_n$ be a word over Σ. Then n is called the **length** of w and is denoted by $|w|$. In particular, we have $|\varepsilon| = 0$.

c) Given two words $w = a_1 \ldots a_n$ and $v = b_1 \ldots b_m$ over Σ, their **concatenation** is defined to be the word

$$w \cdot v := a_1 \ldots a_n b_1 \ldots b_m.$$

Thus, we obtain the concatenation of w and v by writing the word v immediately after the word w. Instead of $w \cdot v$ we usually write wv.

d) Given a word w over Σ, we define **powers** of w inductively by

$$w^0 := \varepsilon,$$
$$w^{n+1} := w^n \cdot w \quad (\text{for } n \in \mathbb{N}).$$

2.1.4 Remark.

a) Formally, a word over Σ is a tuple $w = (a_1, \ldots, a_n)$ with $n \in \mathbb{N}$ and $a_1, \ldots, a_n \in \Sigma$. In particular, two words $(a_1, \ldots, a_n)$ and $(b_1, \ldots, b_m)$ over Σ are equal if they have the same length $n = m$ and fulfill $a_i = b_i$ for any $i \in \{1, \ldots, n\}$. Thus, the equality of words depends on the underlying alphabet.

b) For convenience, we simply denote the tuple $(a_1, \dots, a_n)$ by the sequence $a_1 \dots a_n$. However, this notation does not work in general. For instance, if we consider the alphabet $\Sigma = \{a, aa\}$, then it is not clear whether the sequence $w = aaa$ stands for the word (aa, a), (a, aa), or (a, a, a). Therefore, we assume that every alphabet Σ in this work allows an unambiguous assignment of sequences to the letters in Σ they consist of. This assumption guarantees that every word in Σ^* has a unique written form as concatenation of letters in Σ.

2.1.5 Example. Consider the alphabet $\Sigma = \{\texttt{hello}, \texttt{my}, \texttt{friend}, \texttt{␣}, \texttt{!}, \texttt{?}\}$. Then

$$u = \texttt{hello?hello?}$$
$$v = \texttt{?!my␣}$$
$$w = \texttt{hello␣my␣friend!}$$

are words over Σ. This example shows that, over an appropriate alphabet, also sentences can be interpreted as words. We note further that the word u has length 4 with respect to the alphabet Σ, whereas it has length 12 with respect to the alphabet $\Sigma' = \{\texttt{h}, \texttt{e}, \texttt{l}, \texttt{o}, ?\}$.

2.1.6 Definition. Let Σ be an alphabet. We set

$$\Sigma^* := \{a_1 \dots a_n \mid n \in \mathbb{N}, a_1, \dots, a_n \in \Sigma\},$$

i.e. Σ^* is the set containing all (finite) words over Σ. Moreover, we let

$$\Sigma^+ := \Sigma^* \setminus \{\varepsilon\}$$

be the set of all **non-empty** words over Σ. Given $n \in \mathbb{N}$, we denote by Σ^n the set of all words of length n over Σ.

We treat Σ as a subset of Σ^*, i.e. we do not distinguish between letters of Σ and words of length 1 over Σ. In particular, we have

$$\Sigma^* = \bigcup_{n \in \mathbb{N}} \Sigma^n \text{ and } \Sigma^+ = \bigcup_{n \in \mathbb{N}^+} \Sigma^n,$$

with

$$\Sigma^0 = \{\varepsilon\},\ \Sigma^1 = \Sigma,\ \Sigma^2 = \{ab \mid a, b \in \Sigma\},\ \text{etc.}$$

We note further that the concatenation of words provides a binary operation $\cdot$ on Σ^*.

2.1.7 Example.

a) For the singleton alphabet $\Sigma = \{a\}$ we obtain

$$\Sigma^* = \{\varepsilon, a, aa, aaa, \dots\} = \{a^n \mid n \in \mathbb{N}\},$$
$$\Sigma^+ = \{a^n \mid n \in \mathbb{N}^+\},$$
$$\Sigma^n = \{a^n\} \quad (\text{for } n \in \mathbb{N}).$$

b) For the alphabet $\Sigma = \{a, b\}$ we obtain

$$\Sigma^* = \{\varepsilon, a, b, aa, ab, ba, bb, aaa, aab, aba, abb, baa, bab, bba, bbb, \dots\}.$$

To further study Σ^* and its properties, we recall some algebraic notions in the following.

2.1.8 Definition.

a) A **monoid** is a triple $(M, \cdot, 1)$ consisting of a non-empty set M, an associative binary operation $\cdot\colon M \times M \to M,\ (m_1, m_2) \mapsto m_1 \cdot m_2 := \cdot(m_1, m_2)$ and a neutral element $1 \in M$ with $m \cdot 1 = 1 \cdot m = m$ for any $m \in M$. For convenience, the tuple $(M, \cdot, 1)$ is simply denoted by M if there is no confusion likely to arise. Further, we often omit the symbol $\cdot$ of the binary operation and write $m_1 m_2$ instead of $m_1 \cdot m_2$ for elements $m_1, m_2 \in M$.

b) A monoid M is called **commutative** if $m_1 \cdot m_2 = m_2 \cdot m_1$ holds for any $m_1, m_2 \in M$.

It is well-known that the neutral element of a monoid M is unique. Indeed, if we are given neutral elements 1 and $1'$ of M, then we obtain $1 = 1 \cdot 1' = 1'$.

2.1.9 Example. A typical example of a monoid are the natural numbers $(\mathbb{N}, +, 0)$ where $+$ denotes the usual addition on $\mathbb{N}$, which is even a commutative operation.

2.1.10 Definition. Let $(M, \cdot, 1)$ and $(M', \circ, 1')$ be monoids.

a) A **monoid homomorphism** from M into M' is a map $h\colon M \to M'$ fulfilling
   - $h(1) = 1'$ and
   - $h(m_1 \cdot m_2) = h(m_1) \circ h(m_2)$ for any $m_1, m_2 \in M$.

b) A bijective monoid homomorphism is called a **monoid isomorphism**.

c) The monoids M and M' are called **isomorphic** if there exists a monoid isomorphism from M into M'.

2.1.11 Example. Let Σ be an alphabet.

a) The most important type of a monoid in this work is $(\Sigma^*, \cdot, \varepsilon)$, called the **free monoid generated by the alphabet** Σ. Indeed, the concatenation of words over Σ is an associative operation whose neutral element is given by the empty word. One can even show that $(\Sigma^*, \cdot, \varepsilon)$ is the smallest monoid that contains Σ and is closed under concatenation (cf. Eilenberg [14, page 5]).

b) The monoid Σ^* is commutative if and only if the underlying alphabet Σ contains just a single letter. In fact, if Σ contains two distinct letters $a \neq b$, then the inequality $a \cdot b \neq b \cdot a$ implies that Σ^* is not commutative. For the converse, we assume that $\Sigma = \{a\}$ is a singleton. The free monoid Σ^* is then isomorphic to the commutative monoid $(\mathbb{N}, +, 0)$ via the monoid isomorphism $\mathbb{N} \to \Sigma^*,\ n \mapsto a^n$, whose inverse is given by the length function $\Sigma^* \to \mathbb{N}, a^n \mapsto |a^n| = n$. Therefore, $(\mathbb{N}, +, 0)$ is often said to be a *free* monoid as well.

The reason for calling monoids generated by some alphabet *free* monoids is the following *universal property* (cf. Sakarovitch [33, page 24]), which we will apply various times throughout this work.

2.1.12 Lemma. *Let Σ be an alphabet and $(M, \circ, 1)$ a monoid. Then every map $h\colon \Sigma \to M$ can be uniquely extended to a monoid homomorphism $\hat{h}$ from the free monoid $(\Sigma^*, \cdot, \varepsilon)$ into the monoid $(M, \circ, 1)$. In particular, every monoid homomorphism h from the free monoid Σ^* into a monoid M is uniquely determined by its restriction $h\big|_\Sigma$ to the underlying alphabet Σ.*

Proof. We put $\hat{h}(\varepsilon) = 1$ and $\hat{h}(a_1 \dots a_n) = h(a_1) \circ \dots \circ h(a_n)$ for any $n \in \mathbb{N}^+$ and any $a_1, \dots, a_n \in \Sigma$. Clearly, the map $\hat{h}$ extends h and is a monoid homomorphism by definition. To prove the uniqueness of $\hat{h}$, let $\tilde{h}$ be another monoid homomorphism from the free monoid $(\Sigma^*, \cdot, \varepsilon)$ into the monoid $(M, \circ, 1)$ extending h. Then for each word $w = a_1 \dots a_n \in \Sigma^*$ we obtain

$$\begin{aligned} \tilde{h}(w) &= \tilde{h}(a_1 \dots a_n) \\ &= \tilde{h}(a_1) \circ \dots \circ \tilde{h}(a_n) \\ &= h(a_1) \circ \dots \circ h(a_n) \\ &= \hat{h}(w). \end{aligned}$$

□

We denote the above described (unique) extension $\hat{h}$ of a map h again by h if no confusion is likely to arise.

2.1.13 Example. Consider the constant map $h\colon \Sigma \to \mathbb{N},\ a \mapsto 1$. It is easy to see that its extension $\hat{h}$ is precisely the length function $|\cdot|\colon \Sigma^* \to \mathbb{N},\ w \mapsto |w|$. In other words, the length function is the unique monoid homomorphism from $(\Sigma^*, \cdot, \varepsilon)$ into $(\mathbb{N}, +, 0)$ that maps each letter of Σ to 1. In particular, we have

$$|w \cdot v| = |w| + |v| \text{ and } |w^n| = n \cdot |w|$$

for each $w, v \in \Sigma^*$ and $n \in \mathbb{N}$.

Usually, the monoid homomorphisms we deal with are of the form $h\colon \Sigma^* \to \Gamma^*$, where both Σ and Γ are alphabets. Such maps allow us to change the underlying alphabet, as they assign words over Γ to words over Σ. Based on Droste and Kuich [9, page 16 f.], we now introduce some properties of such monoid homomorphisms.

2.1.14 Definition. Let Σ, Γ be alphabets and $h\colon \Sigma^* \to \Gamma^*$ a monoid homomorphism. We call h

- **non-extending** if $|h(w)| \leq |w|$ for any $w \in \Sigma^*$.
- **non-deleting** if $|h(w)| \geq |w|$ for any $w \in \Sigma^*$.
- **length-preserving** if $|h(w)| = |w|$ for any $w \in \Sigma^*$.