Rheinisch-Westfälische Technische Hochschule Aachen
Lehrstuhl für Informatik VII

November 29, 2010

Classifying Regular Languages via Cascade Products of Automata

Diploma Thesis

Marcus Gelderie

Notes on the Revised Version

The revised version of this diploma thesis contains fixes of some minor errors in notation. It also contains additional references, which were not included in the original version. In particular, the author would like to point out that the results of Section 4.1.2 have been obtained by Brzozowski and Fich in [BF80], a paper the author only learned about after completing his thesis. The proof from [BF80] is slightly different from the one presented in this thesis.

In a similar matter, the results from Section 4.1.1 are similar to a result from Straubing [Str80][1], in which $\mathcal{J}$-trivial monoids are characterized in terms of upper triangular matrices over the commutative semiring $\mathbb{B} = \{0, 1\}$. The result from [Str80] is purely algebraic, whereas our proof in Section 4.1.1 is purely automaton theoretic. A similar investigation is found in [ST88], where partially ordered monoids are considered. In this paper the result from [Str80] is picked up to also obtain a decomposition of $\mathcal{J}$-trivial monoids in terms of restricted semidirect products. This decomposition is very close to the one we give. Its proof is again purely algebraic. This reference was also brought to the author's attention only recently.

Last updated on February 28, 2011

[1] This publication as well as [ST88] was pointed out to the author during the review process of a paper about the present thesis.

Abstract

A significant result in the structure theory of regular languages is the Krohn-Rhodes Theorem, which states that any finite automaton can be decomposed into simple "prime factors". This theorem can be stated in different versions using different products on different structures, namely the cascade product of automata, the wreath product of transformation semigroups and the block product of semigroups. We explore the connection between these three products and use these results to state how the three versions mentioned relate.

We then use the Krohn-Rhodes Theorem to characterize families of regular languages in terms of the decompositions of the corresponding minimal automata. We study the case of piecewise testable, $\mathcal{R}$-trivial and commutative languages. We introduce the concept of a *biased reset* and a *locally i-triggered cascade product* in order to characterize piecewise testable languages. Dropping the requirement of a locally i-triggered product, we then show that a language is $\mathcal{R}$-trivial iff its minimal automaton is covered by a cascade product of biased resets. In order to characterize commutative languages, we introduce the notion of a *one letter automaton (OLA)* and a *one letter cascade product*, in which acceptance of a word is determined solely by the number of occurrences of a single alphabet letter. We show that a language is commutative iff its minimal automaton is covered by a direct product of a one letter cascade products of biased resets and one letter simple cyclic grouplike automata, i.e. grouplike automata, the transition monoid of which is a simple cyclic group.

Finally we introduce the *scope* of resets within a cascade product in order to further refine our analysis of the Krohn-Rhodes decomposition. The scope measures a notion of locality in the product. As initial results we show that the scope of cascade products recognizing $\mathcal{R}$-trivial languages is bounded by a constant and that for a certain family of languages $(L_n)_{n\geq 1}$, such that L_n is of dot-depth n, there exists a product of scope precisely n, which recognizes L_n.

Erklärung

Hiermit versichere ich, dass ich diese Diplomarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Die Stellen meiner Arbeit, die dem Wortlaut oder dem Sinn nach anderen Werken entnommen sind, habe ich in jedem Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht.

Aachen, den 29. November 2010

Contents

1 Introduction **1**
- 1.1 A Product Survey . . . 2
- 1.2 Classifying Regular Languages . . . 4
  - 1.2.1 $\mathcal{R}$-trivial and Piecewise Testable Languages . . . 4
  - 1.2.2 Commutativity . . . 5
  - 1.2.3 Introducing the Scope as a Measure of Locality . . . 6

2 Technical Preliminaries **7**
- 2.1 Sets, Relations and Functions . . . 7
- 2.2 Semigroups and Groups . . . 9
- 2.3 Words, Languages and Automata . . . 14
- 2.4 Products . . . 21

3 A Survey of Cascade, Wreath and Block Products **27**
- 3.1 Cascade and Wreath Products . . . 27
- 3.2 Wreath and Block Products . . . 32
- 3.3 Krohn-Rhodes Theory . . . 33

4 Cascade Products and their Languages **38**
- 4.1 Biased Resets . . . 38
  - 4.1.1 Piecewise Testable Languages . . . 38
  - 4.1.2 $\mathcal{R}$-trivial Languages . . . 43
- 4.2 Commutative Languages . . . 45
- 4.3 The Scope of Cascade Products . . . 53
  - 4.3.1 Language Classes with Constant Scope . . . 54
  - 4.3.2 Scope and Dot-Depth . . . 57
  - 4.3.3 A Tradeoff between Length and Scope . . . 59

5 Conclusion and Future Work **61**
- 5.1 Summary . . . 61
- 5.2 Future Work . . . 62

Bibliography **63**

1 Introduction

In the theory of automata there are prominent results on decomposing automata into "prime" factors. One notable example is the celebrated Krohn-Rhodes Theorem [KR65, Gin68], which states that every automaton can be covered (i.e. "simulated" in a certain sense) by a cascade product of such prime factors. There are two basic types of prime automata used in this decomposition. Firstly, one uses permutation-free two-state automata (called *2-state resets*). Secondly, one uses simple permutation automata (also called *simple grouplike automata*), i.e. automata, the transition monoid of which is a simple group.

This decomposition of automata using the cascade product can also be seen from a semigroup theoretic perspective. In this situation transformation semigroups are decomposed into small *aperiodics* –that is semigroups, which do not contain non-trivial groups– and simple groups using the wreath product. Similarly, the block product can be used to decompose semigroups directly, without resorting to transformation semigroups [Str94, Str89, Pin86, Pin97].

These decompositions of semigroups have been studied extensively over the past decades, for instance in [ST02, Str89, Wei89]. The fundamental Eilenberg Theorem [Eil74b] relates varieties of semigroups with varieties of regular languages, thereby classifying regular languages in terms of the algebraic nature of their syntactic semigroups and monoids. Since then, several classifications of regular languages in terms of algebraic properties of their syntactic semigroups have been found [Sim75, BS71, STT88]. These results have been the subject of several monographs, such as [Eil74a, Eil74b] or [Pin86] as well as survey articles [Wei89, Pin95, Pin97]. However, the decomposition of automata has not been subject to an equally extensive investigation.

Nevertheless, the Krohn-Rhodes Theorem is a strong result in the structure theory of regular languages. Cohen and Brzozowski established a connection to the dot-depth of regular languages in their famous paper [CB71], giving an upper bound for the dot-depth of a language based on the number of factors of a cascade product recognizing it. We will return to this result later in this thesis. The decomposition has also been used to give an alternative proof of the famous Schützenberger Theorem [Mey69, Sch65]. This work suggests, that further work on using the Krohn-Rhodes decomposition to classify regular languages promises results. It is the goal of this thesis to explore the possibilities of obtaining such classification results.

We first give an overview of the cascade, wreath and block products and show how they relate. This comparison is then used to investigate the relationship of the three corresponding versions of the Krohn-Rhodes theorem. We then embark on classifying families of regular languages in terms of their Krohn-Rhodes decomposition. To this end, we consider the cases of piecewise testable languages, $\mathcal{R}$-trivial languages and commutative languages. We provide a characterization for each of these families in

terms of the Krohn-Rhodes decomposition of the minimal automata of languages in those families.

To this end we introduce new concepts to the Krohn-Rhodes decomposition, such as *biased* resets, which are a special case of the usual two-state resets. We also introduce the idea of a *locally i-triggered* cascade product, which restricts the way automata in a cascade react to the states of the preceding automata.

Finally, we introduce the *scope* of a cascade product, which describes a notion of locality in the product. We investigate how products of different scope relate and analyze the expressiveness of such products as well as the *length* (i.e. the number of factors) of such products.

1.1 A Product Survey

The Krohn-Rhodes Theorem can be stated in terms of several kinds of products (on different structures). The three forms of the Krohn-Rhodes Theorem considered in this thesis are those stated in terms of the cascade product of semiautomata, the wreath product of transformation semigroups and the block product of semigroups.

Since all three of these are versions of essentially the same theorem, it is not surprising that the various notions of a product are intimately related. We explore how to formalize this intuitive notion of "related" and use these ideas to shed light on the relationship of the three versions of the Krohn-Rhodes Theorem.

Cascade and Wreath Products

While in a direct product of two automata $\mathfrak{A} \times \mathfrak{B}$ both automata act independently on any given input, the situation in a cascade product is somewhat more entangled. Here the second automaton, $\mathfrak{B}$, changes its state not only depending on its own current state and the input. Instead it also takes the state of $\mathfrak{A}$ into account. A natural question that arises is: Given the transition monoids $M(\mathfrak{A})$ and $M(\mathfrak{B})$, how can we describe the transition monoid $M(\mathfrak{A} \circ \mathfrak{B})$ of the cascade product of $\mathfrak{A}$ and $\mathfrak{B}$?

We show that this monoid can be embedded into the monoid described by the wreath product $(Q^B, M(\mathfrak{B})) \wr (Q^A, M(\mathfrak{A}))$. This relationship can be stated as follows: The transformation monoid $(Q_{A \circ B}, M(\mathfrak{A} \circ \mathfrak{B}))$ *divides* the wreath product $(Q^B, M(\mathfrak{B})) \wr (Q^A, M(\mathfrak{A}))$. We can lift this statement to an arbitrary number of factors by a simple induction.

A converse statement holds as well. Consider a wreath product $(X, S) \wr (Y, T)$ and a transformation monoid (Z, U) dividing this product. Then for every automaton $\mathfrak{A}$ with corresponding transformation monoid (Z, U) we can find automata $\mathfrak{B}$ and $\mathfrak{C}$, corresponding to (Y, T) and (X, S) respectively, such that $\mathfrak{A}$ is covered by the cascade product $\mathfrak{B} \circ \mathfrak{C}$. Again this statement can be extended to an arbitrary number of factors using an induction argument.

Wreath and Block Products

Given two semigroups S and Y we can define generic transformation semigroups (S^1, S) and (Y^1, Y), where S^1 is S adjoined with an identity if necessary. If we consider the wreath product $(Y^1 \times S^1, M) := (Y^1, Y) \wr (S^1, S)$ of these transformation semigroups, what can be said about M?

In [Str94] it is shown how to obtain the Krohn-Rhodes Theorem for semigroups (using the block product) from a version for transformation semigroups (using the wreath product). However, the version for transformation semigroups used there is slightly weaker than the version we consider here. Nevertheless, from the exposition in [Str94] we can extract a proof for the following statement: The monoid M obtained from the wreath product of two transformation semigroups of the kind used above can be embedded into the block product $Y \boxdot S$. This statement can also be lifted to iterated wreath products with an arbitrary number of factors.

A Comparison of the Concepts of Decomposition

Using the above results, we can investigate the three mentioned versions of the Krohn-Rhodes Theorem. We start out with the Krohn-Rhodes Theorem for semiautomata as stated in [Gin68]. On the basis of this theorem we show how to obtain the Krohn-Rhodes Theorem for transformation semigroups with little effort.

The transition from the Krohn-Rhodes Theorem for transformation semigroups using wreath products to the version for semigroups using block products has been illustrated in [Str94]. In fact, the statement from [Str94] is an even stronger one, since the version of the theorem for transformation semigroups is a weaker version of the theorem we consider here. Nevertheless, we state this transition here for the sake of completeness, bearing in mind that it can be done in a more general context.

Finally, we also show how to obtain the Krohn-Rhodes Theorem for semiautomata from the Krohn-Rhodes Theorem for transformation semigroups. This result is a bit more surprising, since, as was illustrated above, the wreath product is "more general" in a certain sense (recall that we can embed the transition monoid of a cascade into the monoid described by the corresponding wreath product). Nevertheless, with some technical effort, we can return to the seemingly less general case.

One might wonder whether one can find a similar transition from the theorem using block products to the theorem using wreath products. We illustrate why the approach used in the wreath vs. cascade product exposition seems unsuitable for the transition from block products to wreath products. The main reason is that the block products seems more expressive than the wreath product or the cascade product. This is suggested by the fact that using the block product, we can decompose monoids using simple groups and $U_1 = (\{0, 1\}, \cdot)$ factors, whereas for wreath products using only factors of the form (U_1, U_1), as well as simple groups, is a proper restriction (see for instance [Str89]).

1.2 Classifying Regular Languages

In order to classify regular languages, we consider the Krohn-Rhodes decomposition of the minimal deterministic automaton of a language. From this decomposition we establish necessary and sufficient conditions to ensure that all languages recognized by such a decomposition are in a certain family of regular languages. In this thesis we consider the cases of piecewise testable languages, $\mathcal{R}$-trivial languages and commutative languages.

1.2.1 $\mathcal{R}$-trivial and Piecewise Testable Languages

Next to the well known Schützenberger Theorem there are other important algebraic characterizations of regular languages. Simon's Theorem is one such example [Sim75]. It states that a language L is piecewise testable iff the syntactic monoid $M(L)$ is $\mathcal{J}$-trivial. Another prominent example is the theorem of Brzozowski and Simon [BS71] characterizing locally testable languages in terms of their syntactic monoids.

Since all $\mathcal{J}$-trivial languages are in particular $\mathcal{R}$- and $\mathcal{L}$-trivial, it is not surprising that one can find similarly related automaton models for these language families. In [Pin86] it is shown that a language L is $\mathcal{R}$-trivial iff the minimal automaton $\mathfrak{A}_L$ is *extensive*. One calls an automaton $\mathfrak{A}$ with state set Q and input alphabet Σ *extensive*, if Q is equipped with a total order $\leq$, compatible with the transitions of $\mathfrak{A}_L$ in the following sense: $q \leq q\overline{\sigma}$ where for all $\sigma \in \Sigma$ we have that $\overline{\sigma}$ is the mapping on Q induced by σ. In [Sim75] it is shown that a language is piecewise testable iff $\mathfrak{A}_L$ can be covered by a direct product of *chain resets*, which are a special kind of extensive automata. More precisely, in a chain reset we require that for any state q_i and any $\sigma \in \Sigma$ $q_i\overline{\sigma} = q_i$ or $q_i\overline{\sigma} = q_{i+1}$, where $q_1, \ldots, q_n$ is an enumeration of Q with respect to $\leq$, i.e. $q_i \leq q_{i+1}$.

$\mathcal{R}$-trivial languages

We add to these characterizations by introducing a characterization in form of Krohn-Rhodes decompositions of the minimal automata $\mathfrak{A}_L$. More precisely, we show that a language L is $\mathcal{R}$-trivial iff $\mathfrak{A}_L$ is covered by an iterated cascade product of *biased resets*, that is, 2-state reset automata, which allow only one change of state during a run. This result is an automaton theoretic equivalent of the well known semigroup theoretical fact, that a monoid is $\mathcal{R}$-trivial iff it divides an iterated wreath products of U_1 factors (see, for instance, [Str89]). In fact, we can deduce this algebraic characterization from our purely automaton theoretic result.

Piecewise Testable Languages

We also give a characterization of piecewise testable languages in terms of special cascade products of biased resets. In order to obtain a characterization, we restrict the way the automata in the cascade may react to the states of the automata preceding them. Assuming that the state set of every biased resets is $\mathbb{B} = \{0, 1\}$, we require that every input read by the i-th automaton induces the identity, if the $(i-1)$-th automaton

is in state 0. Furthermore, if the $(i-1)$-th automaton is in state 1, then the state of all other preceding automata is irrelevant to the transition behavior of the i-th automaton.

This leads us to the notion of a *locally* 1*-triggered* cascade product, which is introduced in Section 4.1.1. Locally 1-triggered cascade products ensure a certain locality in the cascade product. We show that this restriction is sufficient to ensure that a language recognized by such a cascade product is indeed piecewise testable. It is easy to see, that these locally i-triggered cascade products of biased resets are in fact chain resets. Thus, the Krohn-Rhodes characterization we obtain is a refinement of Simon's characterization.

1.2.2 Commutativity

A language L is commutative iff for every $w \in L$ all permutations of w are also in L. From this definition it is easy to deduce that a language L is commutative iff $M(L)$ is commutative. An automaton model also exists: We call an automaton $\mathfrak{A}$ *commutative* if for any two states p, q and any word w mapping p to q every permutation of w also maps p to q. It is again easy to deduce that L is commutative iff $\mathfrak{A}_L$ is commutative.

It seems intuitive that membership of a word w in a commutative language L is determined only by the number of occurrences of the alphabet letters $\sigma_1, \ldots, \sigma_n$ in w. A restriction refers to a single letter σ only. We get languages, which are solely determined by the number of occurrences of σ. We will call such languages *1-semilinear* (an explanation of the terminology, as well as a precise definition is given in Section 4.2).

We show that every commutative language is a Boolean combination of 1-semilinear languages. This result is a formal statement of the intuitive notion of "number of occurrences" mentioned above. The decomposition theorem is itself rather algebraic. It entails, however, a nice decomposition of commutative languages in terms of cascade products of semiautomata.

We call an automaton a *one letter automaton (OLA)*, if there exists exactly one input, which does not induce the identity mapping on its state set. A cascade product is called a *one letter cascade product*, if the automaton defined by it is an OLA. Clearly such automata are commutative and recognize 1-semilinear languages. We show that a language is commutative iff its minimal automaton is covered by a direct product of a one letter cascade product of biased resets and a direct product of *cyclic* OLAs, the number of states of each of which is a prime power. The proof of this statement uses the well known decomposition of cyclic groups of order m into a direct product of cyclic groups of order $m_i = p_i^{k_i}$, where $m = \prod_{i=1}^{r} p_i^{k_i}$ is the prime decomposition of m.

These cyclic grouplike automata can in turn be decomposed into simpler factors using the cascade product and factors, which are cyclic of prime order (i.e. the number of states of which is a prime). In particular, these grouplike automata are simple, i.e. the groups associated with them are simple. Thus, we have arrived at a true Krohn-Rhodes decomposition, using as factors only simple groups and 2-state resets.

1.2.3 Introducing the Scope as a Measure of Locality

When classifying $\mathcal{R}$-trivial and commutative languages, we considered the *nature of the factors* occurring in a Krohn-Rhodes decomposition of the minimal automata recognizing those languages. This is one possible way to distinguish between cascade products and to determine certain properties of the languages such a product will accept.

However, when classifying piecewise testable languages, we made an additional assumption on the *nature of the product* itself. More precisely, we required the factors in the product to only react to the reset immediately preceding it and even required, that it will react to this reset in a very special way.

This is very restrictive. We can loosen our assumptions slightly by first of all dropping every requirement regarding what mappings are invoked if the preceding automaton is in state x. Secondly, instead of requiring that automaton i is only sensitive to the automaton $(i-1)$, we can instead require that automaton i is sensitive only to the first k automata immediately preceding it. This leads us to the *scope* of a cascade product.

The scope describes a notion of locality. If an automaton is "towards the end" of a cascade product, it will not be sensitive to changes that happen to automata, which are close to the beginning. This effectively means, that the amount of information used by any factor of a product is bounded by a constant, regardless of the length of the product.

We show that every piecewise testable language is recognized by a scope 1 cascade product of 2-state resets. Furthermore, we show that every $\mathcal{R}$-trivial language is recognized by a scope 2 cascade product of 2-state resets. However, the locality obtained comes at a price. We risk an exponential blowup in the length in the cascade.

To illustrate this, we give a language L, which can be recognized by a scope n cascade product with n factors, but the canonical scope 1 cascade product recognizing L has $(n+1)!$ factors. We do not know, whether this blowup can be avoided. This is a problem open to further research.

Finally, we give an example of a family of languages L_n of dot-depth n, where the shortest product recognizing these languages generically has scope n. This result suggests a certain connection between dot-depth and scope. However, this connection cannot be tight, as there exist scope 1 products, which recognize languages of dot-depth 2.

2 Technical Preliminaries

2.1 Sets, Relations and Functions

Sets

A set is a collection of objects. If a set A is a subset of B we write $A \subseteq B$. If we wish to indicate that A is a proper subset of B we write $A \subsetneq B$. The *union* of two sets A and B is denoted $A \cup B$. The *intersection* of A and B is written $A \cap B$. The *relative complement* of B in A is written $A \setminus B$. If $B \subseteq A$, then then $A \setminus B$ is also called the *complement of B in A*. If A is clear from context, we write $\overline{B} := A \setminus B$. The empty set is denoted $\varnothing$. If A is a set then $\wp(A) = \{B | B \subseteq A\}$ denotes the *powerset* of A. If A is a set, then $|A|$ denotes the *cardinality* of A. A *partition* of A is a set $B \subseteq \wp(A)$, such that $A = \bigcup_{A_0 \in B} A_0$, $A_0 \neq \varnothing$ and $A_0 \cap A_1 = \varnothing$ for all $A_0, A_1 \in B$, $A_0 \neq A_1$.

We will use the following notation for common sets, which are used throughout the text:

- $\mathbb{N} = \{1, 2, 3, 4, \ldots\}$ for the set of *natural numbers*
- $\mathbb{N}_0 = \{0, 1, 2, 3, 4, \ldots\}$ for the set of *non-negative integers*
- $\mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$ for the set of *integers*
- $\mathbb{B} = \{0, 1\}$ for the *binary digits*

If A and B are sets, then $A \times B = \{(a, b) | a \in A,\ b \in B\}$ denotes the *Cartesian product* of A and B. Similarly let $A_1, \ldots, A_n$ be sets for $n \in \mathbb{N}$. Then the set

$$\bigtimes_{i=1}^{n} A_i := A_1 \times \cdots \times A_n := \{(a_1, \ldots, a_n) | a_i \in A_i,\ 1 \leq i \leq n\}$$

is called the *n-fold Cartesian product*. If $A_1 = A_2 = \cdots = A_n = A$, we also write A^n for $\bigtimes_{i=1}^{n} A_i$. The elements of A^n are called *n-tuples* or *sequences*. If A is any set, we define the set of all finite, nonempty sequences with entries in A:

$$A^+ := \bigcup_{n \in \mathbb{N}} A^n$$

In addition we set $A^0 := \bigtimes_{i=1}^{0} A_i := \{\varepsilon\} := \{()\}$, where ε is the *empty sequence*. Using this we define

$$A^* := \bigcup_{n \in \mathbb{N}_0} A^n$$

to be the set of all finite sequences.

Relations

A *relation* between A and B is a set $R \subseteq A \times B$. If $(a,b) \in R$ we say a *is in relation to* b. For any relation R we define the *inverse of* R, $R^{-1} = \{(b,a)|(a,b) \in R\} \subseteq B \times A$. The *image* of R, is the set $\mathrm{im}(R) = \{b \in B|\exists a \in A : (a,b) \in R\}$. The *domain* of R is the set $\mathrm{dom}(R) = \{a \in A|\exists b \in B : (a,b) \in R\}$. If $A_0 \subseteq A$, then $(A_0)R = \{b \in B|(a,b) \in R \text{ for some } a \in A_0\}$ denotes the *image of* A_0 *under* R. Similarly, if $B_0 \subseteq B$ the set $(B_0)R^{-1} = \{a \in A|(a,b) \in R \text{ for some } b \in B_0\}$ is called the *preimage of* B_0 *under* R. In other words: the image of A under R is the preimage of A under R^{-1} and vice versa.

Let $R \subseteq A \times A$ for some set A. Such a relation R is called *relation over* A. We say R is *transitive*, if for every $a,b,c \in A$, such that $(a,b) \in R$ and $(b,c) \in R$, we also have $(a,c) \in R$. R is called *reflexive*, if $(a,a) \in R$ for every $a \in A$. Conversely, R is called *irreflexive* if for all $a \in A$ we have $(a,a) \notin R$. We call R *symmetric* if for every $(a,b) \in R$ we also have $(b,a) \in R$. It is called *antisymmetric* if $(a,b) \in R$ and $(b,a) \in R$ implies $a = b$.

We are now ready to define a few very important special cases of relation over a set A. Let $R \subseteq A \times A$.

- R is a *preorder* if R is transitive and reflexive.
- R is an *equivalence relation*, if R is reflexive, transitive and symmetric (i.e. a symmetric preorder).
- R is a *partial order*, if R is reflexive, transitive and antisymmetric (i.e. an antisymmetric preorder).
- R is a *total order*, if R is a partial order, such that for all $a,b \in A$ we have either $(a,b) \in R$ or $(b,a) \in R$.

Note that any preorder R on A can be turned into an equivalence relation $\hat{R}$ by setting $\hat{R} = \{(a_1,a_2)|(a_1,a_2) \in R \text{ and } (a_2,a_1) \in R\}$. We say $\hat{R}$ is the equivalence relation *induced* by R.

Let R be an equivalence relation on A. Denote by $[a]_R = \{b \in A|(a,b) \in R\}$ the class of all elements $b \in A$, which are in relation to a. $[a]_R$ is called the *equivalence class* of a under R. If R is clear from context, we drop the subscript and simply write $[a]$. The set $A/R = \{[a]_R|a \in A\}$ forms a partition of A, i.e. $[a] \cap [b] = \varnothing$ if $a \notin [b]$. The number of equivalence classes is called the *index of* R *in* A and is denoted by $[A : R] = |\{[a]_R|a \in A\}|$.

Functions

A *function* (or *mapping*) f from A into B is a relation $f \subseteq A \times B$, such that $\mathrm{dom}(f) = A$ and $|(\{a\})f| = 1$ for all $a \in A$. We write $f : A \to B$. If $a \in A$ we write $(a)f$ for the unique element of $(\{a\})f \subseteq B$, called the *image of* a *under* f. Notice that we write function arguments to the left of the function symbol. If A and B are sets, the set of all functions from A into B is denoted B^A. As a convention, we will sometimes drop the brackets, if the function symbol and the operand are clear from context, i.e. xf instead of $(x)f$.

Some important functions, which we will use frequently, are given below:

- For any set A, we define the *identity on* A, $\mathrm{id}_A : A \to A$, by $(a)\mathrm{id}_A = a$.
- If $u, v \in A^*$ are finite sequences, $u = (a_1, \ldots, a_n)$, $v = (b_1, \ldots, b_m)$, then $u \cdot v := (a_1, \ldots, a_n, b_1, \ldots, b_m)$ is again a finite sequence in A^*. Hence $\cdot : A^* \times A^* \to A^*$ is a mapping. Note that for all $u \in A^*$ we have $u \cdot \varepsilon = \varepsilon \cdot u = u$. We refer to $\cdot$ as the *concatenation (of sequences).*
- Let $A = \bigtimes_{i=1}^n A_i$ for sets $A_1, \ldots, A_n$. We define the *i-th projection map* $\mathrm{pr}_i : A \to A_i$ by $\big((a_1, \ldots, a_n)\big)\mathrm{pr}_i = a_i$ for all $i = 1, \ldots, n$.
- If $f : A \to B$ and $A_0 \subseteq A$, then $f|_{A_0} = f \cap (A_0 \times B)$ is a function, called the *restriction of f to A_0*.
- If $f : A \to B$ and $g : B \to C$ are functions, then so is $f \circ g : A \to C$, defined by $(a)\big(f \circ g\big) = \big((a)f\big)g$, the *composition of f and g*.
- If A is a set and $\cdot : A \times A \to A$ is a function, we call $\cdot$ an *operation (on A).* In this situation we write $\cdot$ in infix notation, i.e. $a \cdot b := (a, b)\cdot$. As a convention, we will often drop $\cdot$ and simply write ab instead of $a \cdot b$. For subsets $B, C \subseteq A$, we write $B \cdot C := \{b \cdot c \,|\, b \in B, c \in C\}$. If B or C is a singleton, containing $b \in B$ or $c \in C$, we write $b \cdot C$ or $B \cdot c$.

We say a mapping $f : A \to B$ is *surjective*, if $(A)f = B$, that is, if the image of A is B. In this situation we sometimes say f is a mapping *onto* B. We call f *injective*, if $|(\{b\})f^{-1}| \leq 1$ for all $b \in B$. f is called *bijective*, if it is both injective and surjective. If $f : A \to B$ is bijective, then f^{-1} is a bijective function from B into A, such that $f \circ f^{-1} = \mathrm{id}_A$ and $f^{-1} \circ f = \mathrm{id}_B$. In this situation f^{-1} is called the *inverse of f*. If A and B are sets, such that there exists a bijective function $f : A \to B$, then $|A| = |B|$.

2.2 Semigroups and Groups

Semigroups and Monoids

Let S be a set and let $\cdot$ be an operation on S. An operation $\cdot$ is *associative*, if $(a \cdot b) \cdot c = a \cdot (b \cdot c)$ for all $a, b, c \in S$. When dealing with associative operations on a set S, we will often use the following abbreviational notation for every $n \in \mathbb{N}$:

$$a^n := \underbrace{a \cdots a}_{n\text{-times}}$$

Similarly, we define for $A \subseteq S$ and $n \in \mathbb{N}$:

$$A^n := \underbrace{A \cdots A}_{n\text{-times}} = \{a_1 \cdots a_n | a_1, \ldots, a_n \in A\}$$

If $\cdot$ is an associative operation on a set S, we call the pair $(S, \cdot)$ a *semigroup*. Usually we identify $(S, \cdot)$ with S, calling S a semigroup. An operation $\cdot$ is called *commutative*,

if $a \cdot b = b \cdot a$ for all $a, b \in S$. If S is a semigroup with a commutative operation, then S is called a *commutative semigroup*. If S is a semigroup and $H \subseteq S$ is closed under $\cdot$ (i.e. $h_1 \cdot h_2 \in H$ for all $h_1, h_2 \in H$), we call H a *subsemigroup* of S. If $S_0 \subseteq S$ is a subset of S, then

$$\langle S_0 \rangle := \bigcap_{\substack{H \in \mathcal{S} \\ S_0 \subseteq H}} H$$

where $\mathcal{S}$ is the set of all subsemigroups of S, is a subsemigroup of S. It is the smallest subsemigroup of S containing S_0. We say S_0 *generates* $\langle S_0 \rangle$. It can be shown, that $\langle S_0 \rangle$ is precisely the set $\bigcup_{n \in \mathbb{N}} S_0^n$. If $S_0 = \{s_1, \ldots, s_n\}$ is finite we often write $\langle s_1, \ldots, s_n \rangle$ instead of $\langle S_0 \rangle$. A semigroup S is called *monogenic*, if there exists an element $s \in S$, such that $S = \langle s \rangle$.

Let M be a semigroup. An element $1 \in M$, such that $1 \cdot m = m \cdot 1 = m$ for all $m \in M$ is called a *neutral element* of M. In every semigroup there can exist at most one neutral element. If M possesses a neutral element, M is called a *monoid*. If M is a monoid and $m \in M$, we call m *invertible*, if there exists $n \in M$, such that $m \cdot n = n \cdot m = 1$. For any such m the element n satisfying this property is unique. It is usually denoted m^{-1}. A *submonoid* of M is a subsemigroup $H \subseteq M$ such that $1 \in H$. If S is any semigroup, then the set $S^I := S \cup \{I\}$ becomes a monoid by setting $I \cdot s = s \cdot I = s$ for all $s \in S$. We define:

$$S^1 := \begin{cases} S & \text{if } S \text{ is a monoid} \\ S^I & \text{otherwise} \end{cases}$$

If M is a monoid and $M_0 \subseteq M$ is a subset, then $\langle M_0 \rangle$ is not necessarily a monoid (it will be if $1 \in M_0$). If we wish the subsemigroup of M generated by M_0 to be a submonoid of M, we will adjoin the neutral element of $1 \in M$ to $\langle M \rangle$. We set $\langle M_0 \rangle_M := \langle M_0 \rangle \cup \{1\}$.

Let S be a semigroup. An element $s \in S$, such that $s' \cdot s = s$ for all $s' \in S$ is called *right zero*. Similarly, an element $s \in S$, such that $s \cdot s' = s$ for all $s' \in S$ is called *left zero*. An element s, which is both left and right zero is simply called a *zero (of S)*. Any semigroup can have at most one zero, which (if it exists) is denoted 0. A semigroup S is a *left zero semigroup*, if all its elements are left zero. If S is a monoid, then S is called *left zero monoid*, if all elements $s \neq 1$ are left zero. Right zero semigroups and monoids are defined in the same way. Note that a right zero semigroup cannot be a monoid and that a right zero monoid is *not* a right zero semigroup.

We have already seen examples of semigroups and monoids. For any set A, the pair $(A^+, \cdot)$, where $\cdot$ refers to the concatenation of sequences, is a semigroup. The set A^* even forms a monoid, since ε is a neutral element with respect to the concatenation of sequences. Let $1 < n \in \mathbb{N}$. We define an operation $\cdot$ on $U_n = \{1, a_1, \ldots, a_n\}$ by setting $a_i \cdot a_j = a_j$ and $1 \cdot a_i = a_i \cdot 1 = a_i$ for all $i, j \in \{1, \ldots, n\}$. Then U_n is a right zero monoid for all $n \in \mathbb{N}$. We will be especially interested in the cases U_1 and U_2. Notice that U_1 possesses a zero, namely a_1. Furthermore, for all sets X the set X^X of all mappings from X into X is a monoid.

Let $(S, \cdot)$ and $(R, \diamond)$ be semigroups and $\varphi : S \to R$ be a mapping. If φ satisfies $(s_1 \cdot s_2)\varphi = (s_1)\varphi \diamond (s_2)\varphi$ for all $s_1, s_2 \in S$, we call φ a *semigroup homomorphism*, or simply *homomorphism*. The image $\mathrm{im}\,(\varphi)$ of a homomorphism φ is a semigroup. If $\varphi : S \to R$ is injective, then φ is called an *embedding*, denoted $\varphi : S \hookrightarrow R$. We say S

can be *embedded* into R. If φ is bijective the inverse φ^{-1} is again a homomorphism. In this situation we call φ an *isomorphism*. If there exists an isomorphism between two semigroups S and R, we write $S \cong R$ and say S and R are *isomorphic*. For example $A^* \cong (A^+)^1$.

If M and N are monoids with neutral elements 1_M and 1_N respectively and $\varphi : M \to N$ is a semigroup homomorphism, such that $(1_M)\varphi = 1_N$, we call φ a *monoid homomorphism*. Embeddings and isomorphisms are defined as before.

Groups

A monoid G where every $g \in G$ is invertible is called a *group*. $|G|$ is called the *order* of G. A *subgroup* $H \leq G$ of a group G is a subset $H \subseteq G$, such that H is again a group. This is the case if and only if $1 \in H$, for every $a, b \in H$ $a \cdot b \in H$ and $a^{-1} \in H$. If H is a subgroup of G, G finite, then $|H|$ divides $|G|$. For every group G, $\{1\}$ and G are subgroups of G, the *trivial* subgroups of G. Consequently, $H \leq G$ is called *nontrivial*, if $\{1\} \neq H \neq G$. A monoid homomorphism (isomorphism) between two groups G and H is called a *group homomorphism (isomorphism)*.

Let $n \in \mathbb{N}$. A very generic example of a group is the *symmetric group on n points*, denoted S_n of all bijective functions from $\{1, \ldots, n\}$ into $\{1, \ldots, n\}$ together with the composition of functions. The elements of S_n are also called *permutations*. A monogenic group G is called *cyclic*. Subgroups of cyclic groups are again cyclic. If $g \in G$, then $\langle g \rangle$ is a cyclic subgroup of G. The order of $\langle g \rangle$ is called the *order of g*. If G is a commutative group, G is called *abelian*.

A subgroup of $N \leq G$ is called *normal subgroup*, if $gN = Ng$ for all $g \in G$. We write $N \trianglelefteq G$ to indicate that N is a normal subgroup of G. For every group G, the trivial subgroups $\{1\}$ and G are normal subgroups of G. A group $\{1\} \neq G$ is called *simple*, if $\{1\}$ and G are the only normal subgroups of G. In an abelian group G, every subgroup is normal. We state Cauchy's Theorem:

Theorem 2.2.1 (Cauchy, see [KS04])**.** *Let G be a finite group and $p \in \mathbb{N}$ be prime, such that p divides $|G|$. Then there exists an element $g \in G$ of order p. In particular, G has a subgroup of order p.*

We deduce that a finite abelian group is simple iff it has prime order.

If S is a semigroup and $G \neq \{1\}$ is a subsemigroup of S, which is a nontrivial group, then we say S *contains groups*. S is *group free* or *aperiodic* if S does not contain groups, i.e. if every subsemigroup of S, which is a group, is trivial.

Congruences, Quotients and Division

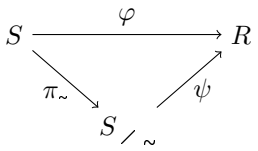
Let S be a semigroup and let $\sim \subseteq S \times S$ be an equivalence relation on S. If for every pair $s, s' \in S$ with $s \sim s'$ and every $x, y \in S$ we have $xsy \sim xs'y$ we call $\sim$ a *congruence*. Every homomorphism $\varphi : S \to R$ induces a congruence $\sim_\varphi$ on S, defined by

$$s \sim_\varphi s' \quad \Leftrightarrow \quad \varphi(s) = \varphi(s')$$

If $\sim$ is a congruence on S, the set $S/\sim$ forms a semigroup. The operation is defined by $[s] \cdot [s'] := [s \cdot s']$. Since $\sim$ is a congruence, this operation is well defined. A semigroup R, such that $R \cong S/\sim$ for some congruence $\sim$, is called *quotient of S (with respect to $\sim$)*. If S is a monoid, then $S/\sim$ is a monoid with neutral element $[1]$. The mapping $\pi_\sim : S \to S/\sim$ defined by $s \mapsto [s]_\sim$ is a surjective homomorphism of semigroups (or monoids, where this applies).

We state the following important theorem:

Theorem 2.2.2 (Homomorphism Theorem (for Semigroups), see [How95])**.** *Let S and R be semigroups and let $\varphi : S \to R$ and let $\sim \subseteq \sim_\varphi$. Then there exists a unique homomorphism $\psi : S/\sim \to R$, such that $\varphi = \pi_\sim \circ \psi$ i.e. the following diagram commutes:*



If $\sim = \sim_\varphi$ then ψ is injective. In particular, $S/\sim_\varphi \cong \operatorname{im}(\varphi)$.
If S and R are monoids and φ is a monoid homomorphism, then so is ψ.

An immediate consequence of Theorem 2.2.2 is that if a semigroup R is the image of a semigroup S under some homomorphism φ, then R is a quotient of S.

If a semigroup R is a quotient of a subsemigroup $S_0 \subseteq S$, we say R *divides* S and write $R \prec S$. By Theorem 2.2.2 we conclude that $R \prec S$ if and only if R is the image of some subsemigroup $S_0 \subseteq S$ under some (surjective) homomorphism $\varphi : S_0 \to R$. Notice that division of semigroups is a transitive relation on the class of all semigroups.

Green's Relations

Let S be a semigroup and $a \in S$. The *left ideal of S generated by a* is the set S^1a. Similarly the *right ideal generated by a* is the set aS^1. The *(two-sided) ideal of S generated by a* is the set $S^1aS^1 = \{sas' | s, s' \in S^1\}$. Any left ideal I in S satisfies $sI \subseteq I$ for all $s \in S$. Similarly $Is \subseteq I$ for all $s \in S$ if I is a right ideal and finally $sI \subseteq I \supseteq Is$ if I is an ideal.

Ideals give us a way of defining preorders on S. We define:

- $a \leq_{\mathcal{L}} b$ iff $S^1a \subseteq S^1b$
- $a \leq_{\mathcal{R}} b$ iff $aS^1 \subseteq bS^1$
- $a \leq_{\mathcal{J}} b$ iff $S^1aS^1 \subseteq S^1bS^1$

These orders are called the $\mathcal{L}$*-ordering*, $\mathcal{R}$*-ordering* and $\mathcal{J}$*-ordering* respectively. We denote the equivalence relations induced by these preorders $\mathcal{L}$, $\mathcal{R}$ and $\mathcal{J}$ in that order. If $(a, b) \in \mathcal{L}$ we often write $a\mathcal{L}b$ for short (similarly for $\mathcal{R}$ and $\mathcal{J}$). The equivalence classes with respect to $\mathcal{L}$, $\mathcal{R}$ and $\mathcal{J}$ are called $\mathcal{L}$-, $\mathcal{R}$- and $\mathcal{J}$-classes.

A semigroup S is called *$\mathcal{L}$-trivial* (*$\mathcal{R}$-trivial*, *$\mathcal{J}$-trivial*) if all its $\mathcal{L}$-classes ($\mathcal{R}$-classes, $\mathcal{J}$-classes) are singletons. Since $a\mathcal{L}b$ implies $a\mathcal{J}b$ and similarly $a\mathcal{R}b$ implies $a\mathcal{J}b$, we get that every $\mathcal{J}$-trivial semigroup is in particular $\mathcal{L}$- and $\mathcal{R}$-trivial.

Suppose S is a right zero semigroup (in particular S is not a monoid). Then $S^1s = \{s\}$ for all $s \in S$. Hence S is $\mathcal{L}$-trivial. If M is a right zero monoid, then either $Mm = \{m\}$ for $Mm = M$ for all $m \in M$. The second case arises precisely if $m = 1$. Hence M is an $\mathcal{L}$-trivial semigroup. Similarly left zero semigroups and monoids are $\mathcal{R}$-trivial. Hence U_n is $\mathcal{L}$-trivial for all $n \in \mathbb{N}$. Since U_1 is both a left and a right zero monoid, it is both $\mathcal{L}$- and $\mathcal{R}$-trivial. Notice that U_1 is also $\mathcal{J}$-trivial.

The equivalence relations $\mathcal{L}$, $\mathcal{R}$ and $\mathcal{J}$ are three of the five *Green's relations*. The remaining two relations are $\mathcal{H} := \mathcal{L} \wedge \mathcal{R}$ and $\mathcal{D} := \mathcal{L} \vee \mathcal{R}$, where $\mathcal{L} \wedge \mathcal{R} = \{(a,b) | a\mathcal{L}b \text{ and } a\mathcal{R}b\}$ and $\mathcal{L} \vee \mathcal{J}$ is the smallest equivalence relation containing both $\mathcal{L}$ and $\mathcal{R}$. We only state the definition of $\mathcal{H}$ and $\mathcal{D}$ for the sake of completeness. Only $\mathcal{L}$, $\mathcal{R}$ and $\mathcal{J}$ will be of interest in this text. For more information the reader is referred to [How95, Pin86, Pin97].

Transformation Semigroups

Let X be a set and $S \subseteq X^X$ be a set of mappings from X into X, such that S is closed under composition. Then $(S, \circ)$ is a semigroup. We will usually write $\cdot$ instead of $\circ$. The pair (X, S) is called a *transformation semigroup* or *TS* for short. If S contains the identity function on X, then S is obviously a monoid. In this situation we also call (X, S) a *transformation monoid*, or *TM*. A TS (X, S) is called *monogenic* if there exists an element $x_0 \in X$, such that $X = x_0S$. The element x_0 is called a *generating element* of (X, S).

Notice that if S is a semigroup, then (S^1, S) can be thought of as a TS: We identify with $s \in S$ the mapping $f_s : S^1 \to S^1$, $(x)f_s = xs$. Notice that (S, S) need not be a TS, if S is not a monoid. If S is, for instance, a left zero semigroup, then $xs = xs' = x$ for all $x, s, s' \in S$. Hence $f_s = f_{s'} = \mathrm{id}_S$ and we cannot think of S as a *set* of mappings.

If (X, S) is a TS, then we denote by $(X, S)^1$ the TM obtained by adding id_X to the set S of mappings, if necessary. We denote by $\overline{(X, S)}$ the TS obtained by adding all constant functions $g_c : X \to X$, $x \mapsto c$ for all $c \in X$ to S.

Definition 2.2.1. Let (X, S) and (Y, T) be two transformation semigroups. We say that (X, S) *divides* (Y, T) and write $(X, S) \prec (Y, T)$ if there exists a surjective mapping $f : Y_0 \to X$ from a subset $Y_0 \subseteq Y$ onto X and a mapping $\psi : S \to T$, such that $(yf)s = \big(y(s\psi)\big)f$ for all $y \in Y_0$ and $s \in S$.

The notion of division of transformation semigroups is related to the notion of division of semigroups:

Lemma 2.2.3 (see [Str94])**.** *Let (X, S) and (Y, T) be transformation semigroups such that $(X, S) \prec (Y, T)$. Then $S \prec T$.*

Semigroup Actions

Let A be a set and S be a semigroup. A mapping $. : A \times S \to A$, which satisfies $a.(s \cdot s') = (a.s).s'$ for all $s, s' \in S$, is called a *(right) semigroup action (of S on A)* or simply *action (of S on A)*. We then say S *acts* on A. Left semigroup actions are defined analogously. Of course, every right action can be transformed into a left action and vice versa. If $S = S^1$ is a monoid and if $a.1 = a$ for all $a \in A$, we call . a *(right) monoid action (of S on A)*. Left monoid actions are defined in the same way.

A semigroup action of S on A is *faithful* if for every $s, s' \in S$, there exists $a \in A$, such that $a.s \neq a.s'$. If (X, S) is a TS, then S acts on X in a very natural way: $x.s := (x)s$ for all $x \in X$ and all $s \in S$. S even acts faithfully on X, because we have defined S to be a set of mappings. Conversely, every pair (X, S) of a semigroup S acting faithfully on a set X can be thought of as a TS (X, S).

2.3 Words, Languages and Automata

Words and Languages

An *alphabet* is a finite set Σ of *letters*. Recall the definition of the set Σ^* of all finite sequences with entries in Σ. If Σ is an alphabet, we will write the elements of Σ^* as *strings* of elements of Σ, i.e. $(\sigma_1, \ldots, \sigma_n)$ is written as $\sigma_1 \cdots \sigma_n$. We will then refer to the elements of Σ^* as *words (over Σ)*. The empty sequence ε is called the *empty word*. We call Σ^* the set of all *finite words (over Σ)*. Similarly the set Σ^+ is referred to as the set of all finite, nonempty words over Σ. A *language* is a subset $L \subseteq \Sigma^*$, i.e. a set of words. For two words $u, v \in \Sigma^*$ the *concatenation* $u \cdot v$ of u and v defined to be the concatenation of sequences.

Given a word $w = \sigma_1 \cdots \sigma_n \in \Sigma^*$ for some $n \in \mathbb{N}_0$ (note that if $n = 0$ then $w = \varepsilon$), we set $|w| := n$. This defines a mapping $|\cdot| : \Sigma^* \to \mathbb{N}_0$. We call $|w|$ the *length* of w. Given a letter $\sigma \in \Sigma$, we define $|\cdot|_\sigma : \Sigma^* \to \mathbb{N}_0$ by $|\sigma_1 \cdots \sigma_n|_\sigma = |\{i | \sigma_i = \sigma\}|$. $|w|_\sigma$ denotes the number of *occurrences* of σ in w. If $w = \sigma_1 \ldots \sigma_n$, then every word $\varepsilon, \sigma_1, \sigma_1\sigma_2, \ldots, w$ is called a *prefix* of w. If v is a prefix of w we write $v \sqsubseteq w$. Similarly every word $\varepsilon, \sigma_n, \sigma_{n-1}\sigma_n, \ldots, w$ is a *suffix* of w. We write $u \sqsupseteq w$ if u is a suffix of w.

Automata and Regular Languages

A *deterministic finite automaton (DFA)* is a tuple $\mathfrak{A} = (Q, \Sigma, q_0, \delta, F)$ with a finite set Q of *states*, an *input alphabet* Σ, an *initial state* $q_0 \in Q$, a mapping $\delta : \Sigma \to Q^Q$ and a set $F \subseteq Q$ of *final states*. We usually write $\overline{\sigma}$ (or $\overline{\sigma}^A$, if we wish to stress the automaton in question is $\mathfrak{A}$) for the image $(\sigma)\delta$ of σ under δ. Let $w = \sigma_1 \cdots \sigma_n \in \Sigma^*$. A *run of $\mathfrak{A}$ on w* is a mapping $\rho : \{0, \ldots, n\} \to Q$, such that $(0)\rho = q_0$ and for all $i = 1, \ldots, n$ we have $(i)\rho = \big((i-1)\rho\big)\overline{\sigma_i}$. Notice that for every word $w \in \Sigma^*$ and every DFA $\mathfrak{A}$ the run ρ is unique. We often write ρ_i for $(i)\rho$, where $i = 0, \ldots, n$. Notice $\rho_n = (q_0)\overline{\sigma_1} \circ \overline{\sigma_2} \circ \cdots \circ \overline{\sigma_n} =: (q_0)\overline{w}$. A run ρ is *accepting*, if $\rho_n \in F$. If the run of $\mathfrak{A}$ on w is accepting, the automaton $\mathfrak{A}$ *accepts* w. The language $L(\mathfrak{A})$ of all words accepted

by $\mathfrak{A}$ is called the *language accepted by* $\mathfrak{A}$. A language $L \subseteq \Sigma^*$ is called *regular* if there exists a DFA $\mathfrak{A}$, such that $L = L(\mathfrak{A})$.

Proposition 2.3.1 (see [HMU01])**.** *Regular languages are closed under union, complement and therefore under intersection.*

If a set S of sets satisfies the closure properties stated in Proposition 2.3.1 and if there exist two sets $I, 0 \in S$, such that $I \cap A = A$ and $0 \cup A = A$ for all $A \in S$, we call S a *Boolean algebra*. In this terminology, the regular languages from a Boolean algebra.

A *semiautomaton* is a tuple $\mathfrak{A} = (Q, \Sigma, \delta)$, where Q, Σ and δ are defined as before. We can extend any semiautomaton to a DFA by specifying an initial state $q_0 \in Q$ and a set $F \subseteq Q$ of final states. In this case we write $\mathfrak{A}_{q_0,F} := (\mathfrak{A}, q_0, F)$. Since a semiautomaton $\mathfrak{A}$ neither has initial nor final states, we do not have a concept of the language of a semiautomaton. Instead we can think of a semiautomaton as the basic building block for several DFAs, all of which recognize a well defined language. Given a semiautomaton $\mathfrak{A} = (Q, \Sigma, \delta)$, we will therefore speak of the *family of languages* $\mathcal{L}(\mathfrak{A})$ *of* $\mathfrak{A}$, where $\mathcal{L}(\mathfrak{A}) = \{L \subseteq \Sigma^* | L = L(\mathfrak{A}_{q_0,F}),\ q_0 \in Q,\ F \subseteq Q\}$. Every $L \in \mathcal{L}(\mathfrak{A})$ is said to be *recognized by* $\mathfrak{A}$. Notice, that for every $\mathfrak{A}$ we have $\varnothing, \Sigma^* \in \mathcal{L}(\mathfrak{A})$.

Let $\mathfrak{A} = (Q, \Sigma, \delta)$ be a semiautomaton. Suppose $Q_0 \subseteq Q$ and $\Sigma_0 \subseteq \Sigma$. If $q\overline{\sigma} \in Q_0$ for all $q \in Q_0$ and all $\sigma \in \Sigma_0$, then $\mathfrak{A}_0 = (Q_0, \Sigma_0, \delta|_{\Sigma_0})$ is a *subsemiautomaton* of $\mathfrak{A}$. Evidently, for every $L \in \mathcal{L}(\mathfrak{A}_0)$ there exists $L' \in \mathcal{L}(\mathfrak{A})$, such that $L \subseteq L'$. If $\Sigma_0 = \Sigma$, then we even get $\mathcal{L}(\mathfrak{A}_0) \subseteq \mathcal{L}(\mathfrak{A})$.

Let $\mathfrak{A} = (Q^A, \Sigma, q_0^A, \delta^A, F^A)$ and $\mathfrak{B} = (Q^B, \Sigma, q_0^B, \delta^B, F^B)$ be two DFAs. A mapping $\varphi : Q^A \to Q^B$, which satisfies $(q_0^A)\varphi = q_0^B$, $(F^A)\varphi \subseteq F^B$ and $(q\overline{\sigma}^A)\varphi = (q\varphi)\overline{\sigma}^B$ for all $q \in Q^A$ is called a *DFA-homomorphism*. We often write $\varphi : \mathfrak{A} \to \mathfrak{B}$ to indicate that φ is a homomorphism from $\mathfrak{A}$ into $\mathfrak{B}$. In this situation we have $L(\mathfrak{A}) \subseteq L(\mathfrak{B})$. $\varphi : \mathfrak{A} \to \mathfrak{B}$ is called a *DFA-isomorphism*, if φ is bijective and $(F^A)\varphi = F^B$. If a DFA-isomorphism between $\mathfrak{A}$ and $\mathfrak{B}$ exists, we write $\mathfrak{A} \cong \mathfrak{B}$.

If $\mathfrak{A} = (Q^A, \Sigma, \delta^A)$ and $\mathfrak{B} = (Q^B, \Sigma, \delta^B)$ are semiautomata, then a mapping $\varphi : Q^A \to Q^B$ satisfying $(q\overline{\sigma}^A)\varphi = (q\varphi)\overline{\sigma}^B$ is called a *homomorphism of semiautomata*. The image $\mathrm{im}(\varphi)$ of a homomorphism $\varphi : \mathfrak{A} \to \mathfrak{B}$ of semiautomata is a subsemiautomaton of $\mathfrak{B}$. If φ is bijective, then φ is called an *isomorphism of semiautomata*. We again write $\mathfrak{A} \cong \mathfrak{B}$ if an isomorphism of semiautomata exists between $\mathfrak{A}$ and $\mathfrak{B}$.

Regular Expressions

Suppose Σ is an alphabet. The *regular expressions (over* Σ*)* are defined inductively as follows: $\varnothing$ is a regular expression and for every $\sigma \in \Sigma$, σ is a regular expression. $\varnothing$ and σ are *atomic regular expressions*. If r is a regular expression, then so is r^*. If r_1, r_2 are regular expressions, then $r_1 \cdot r_2$ and $r_1 + r_2$ are regular expressions. For notational convenience, we introduce another operator $^+$ on regular expressions, which we define by $r^+ := r \cdot r^*$.

Given any regular expression r the *language* $L(r)$ *defined by* r is defined inductively over the construction of r: $L(\varnothing) = \varnothing$. For $\sigma \in \Sigma$ we have $L(\sigma) = \{\sigma\}$. If r is an

expression defining $L = L(r)$, then

$$L(r^*) := L^* := \bigcup_{n \in \mathbb{N}_0} L^n$$

with the convention that $L^0 = \{\varepsilon\}$. $^* : \wp(\Sigma^*) \to \wp(\Sigma^*)$ is called the *Kleene star*. If r_1 and r_2 are regular expressions defining languages L_1 and L_2 respectively, then $L(r_1 + r_2) = L_1 \cup L_2$ and $L(r_1 \cdot r_2) = L_1 \cdot L_2$. We often identify the language $L(r)$ with r.

Proposition 2.3.2 (Kleene, see [HMU01])**.** *Let Σ be an alphabet and $L \subseteq \Sigma^*$. The following are equivalent:*

(a) L is regular.

(b) There exists a DFA $\mathfrak{A}$, such that $L = L(\mathfrak{A})$.

(c) There exists a regular expression r, such that $L = L(r)$.

Let r be a regular expression. We introduce a new operator $\sim$ on regular expressions. We define $L(\sim r) = \Sigma^* \setminus L(r)$. From Proposition 2.3.1 and 2.3.2 it follows that introducing $\sim$ to the set of operations defining regular expressions adds nothing to their expressiveness. Expressions composed using this extended set of operators are called *generalized regular expressions*.

A regular language L is *star-free*, if it can be defined by a generalized regular expression without using the Kleene star *, i.e. if it is constructed using the operators $\sim, \cdot, +$ as well as the usual atomic regular expressions. For instance, $\Sigma^* = \sim \varnothing$ is star-free. By definition, the star-free languages form a Boolean algebra. We denote the set of all star-free languages by $\mathcal{SF}$.

Congruences, Quotients and Coverings

Let $\mathfrak{A} = (Q, \Sigma, \delta)$ be a semiautomaton and let $\approx \subseteq Q \times Q$ be an equivalence relation. If $q\overline{\sigma} \approx q'\overline{\sigma}$ for all $q \approx q'$ and all $\sigma \in \Sigma$, we call $\approx$ a *congruence (on $\mathfrak{A}$)*. $\approx$ is a congruence on the DFA $\mathfrak{B} = (\mathfrak{A}, q_0, F)$ if it is a congruence on the semiautomaton $\mathfrak{A}$. Given a semiautomaton $\mathfrak{A} = (Q, \Sigma, \delta)$ and a congruence $\approx$ on $\mathfrak{A}$, we can define the *quotient automaton (of $\mathfrak{A}$ with respect to $\approx$)*, denoted $\mathfrak{A}/\approx$, by

$$\mathfrak{A}_{/\approx} = \left(Q_{/\approx}, \Sigma, \hat{\delta}\right)$$

where $\hat{\delta} : \Sigma \to Q/\approx$, $\sigma \mapsto \overline{\sigma}^{\approx}$, $[q]\overline{\sigma}^{\approx} = [q\overline{\sigma}]$. $\overline{\sigma}^{\approx}$ is well defined, because $\approx$ is a congruence. In the case of a DFA $\mathfrak{B} = (\mathfrak{A}, q_0, F)$ define $\mathfrak{B}/\approx = (\mathfrak{A}/\approx, [q_0], F/\approx)$, where $F/\approx = \{[q] | q \in F\}$. In analogy to the case of semiautomata, $\mathfrak{B}/\approx$ is also called the *quotient automaton (of $\mathfrak{B}$ with respect to $\approx$)*.

Regardless of whether $\mathfrak{A}$ is a semiautomaton or a DFA, we call $\mathfrak{A}/\approx$ a *quotient (of $\mathfrak{A}$)*. The map $\pi_\approx : \mathfrak{A} \to \mathfrak{A}/\approx$, $q \mapsto [q]_\approx$ is a surjective homomorphism of semiautomata (or a DFA-homomorphism, where this applies).

Every DFA-homomorphism (and consequently every homomorphism of semiautomata) induces a congruence. Let $\varphi : \mathfrak{A} \to \mathfrak{B}$ be a DFA homomorphism. Then $\approx_\varphi$ defined by $p \approx_\varphi q$ iff $(p)\varphi = (q)\varphi$ is a congruence. A statement analogous to Theorem 2.2.2 holds (the proof is straightforward and analogous to the proof of Theorem 2.2.2):

Theorem 2.3.3 (Homomorphism Theorem (for Automata)). *Let $\mathfrak{A}$ and $\mathfrak{B}$ be semiautomata and $\varphi : \mathfrak{A} \to \mathfrak{B}$ a homomorphism. Suppose $\approx$ is a congruence on $\mathfrak{A}$, such that $\approx \subseteq \approx_\varphi$. Then there exists a unique homomorphism $\psi : \mathfrak{A}/\approx \to \mathfrak{B}$, such that $\varphi = \pi_\approx \circ \psi$. If $\approx = \approx_\varphi$, then ψ is injective. In particular, $\mathfrak{A}/\approx_\varphi \cong \operatorname{im}(\varphi)$.*

If $\mathfrak{A}$ and $\mathfrak{B}$ are semiautomata, then we say $\mathfrak{B}$ *covers* $\mathfrak{A}$ and write $\mathfrak{A} \leq \mathfrak{B}$, if there exists a surjective homomorphism of semiautomata $\varphi : \mathfrak{B}_0 \to \mathfrak{A}$ for some subsemiautomaton $\mathfrak{B}_0$ of $\mathfrak{B}$. By Theorem 2.3.3 we see that $\mathfrak{A} \leq \mathfrak{B}$ iff $\mathfrak{A}$ is isomorphic to a quotient of a subsemiautomaton of $\mathfrak{B}$. Covering of semiautomata is a transitive relation on the class of all semiautomata.

It should be mentioned that a more general definition of covering than given above exists (we will be using this more general form only once, in Section 3.1). In [Gin68] this more general form is used. Whenever we cite [Gin68] we will be referring to the notion of covering just introduced, bearing in mind that some cases a little adjustment may be needed to adapt the results to our more specific setting.

Languages and Semigroups

For any language $L \subseteq \Sigma^*$ we can define an equivalence relation $\sim_L \subseteq \Sigma^* \times \Sigma^*$ by $w \sim_L w'$ iff for all $u, v \in \Sigma^*$ we have

$$uwv \in L \Leftrightarrow uw'v \in L$$

Notice that this implies either $[w] \subseteq F$ or $[w] \cap F = \varnothing$ for all $w \in \Sigma^*$. $\sim_L$ is a congruence (of semigroups) with respect to the concatenation of words. The monoid $M(L) := \Sigma^*/\sim_L$ is called the *syntactic monoid of* L.

Let M be a monoid and $L \subseteq \Sigma^*$. Then L is *recognized* by M, if there exists a monoid homomorphism $\varphi : \Sigma^* \to M$ and a subset $P \subseteq M$, such that $L = (P)\varphi^{-1}$. If $L \subseteq \Sigma^*$, then $M(L)$ recognizes L. Choose $P = \{[w]_{\sim_L} | w \in L\}$ and choose $\varphi = \pi_{\sim_L}$ (recall that $(w)\pi_{\sim_L} = [w]_{\sim_L}$). Then $L = (P)\pi_{\sim_L}^{-1}$.

Let $\mathfrak{A} = (Q, \Sigma, \delta)$ be a semiautomaton. Consider the set $A = \{\overline{\sigma} | \sigma \in \Sigma\} \subseteq Q^Q$. The submonoid of Q^Q generated by A is called the *transition monoid of* $\mathfrak{A}$ and is denoted $M(\mathfrak{A}) := \langle A \rangle_M$. The transition monoid $M(\mathfrak{A})$ recognizes every language $L \subseteq \Sigma^*$ recognized by $\mathfrak{A}$. Notice that $M(\mathfrak{A})$ is always finite.

We have seen that every semiautomaton $\mathfrak{A}$ (and therefore every DFA $(\mathfrak{A}, q_0, F)$) uniquely determines a monoid $M(\mathfrak{A})$. The converse is also true: Given a finite monoid M we obtain a semiautomaton $\mathfrak{A}_M = (M, M, \delta_M)$, where $\overline{m} := (m)\delta_M$ is defined by $m'\overline{m} = m' \cdot m$ for all $m, m' \in M$. We have $M(\mathfrak{A}_M) \cong M$. Given a mapping $\omega : \Sigma \to M$, we obtain an automaton $\mathfrak{A}_M^\omega = (M, \Sigma, \delta_M^\omega)$, where $\delta_M^\omega = \omega \circ \delta_M$. $\mathfrak{A}_M^\omega$ will recognize every language over Σ^*, which M recognizes.

There is a connection between coverings of automata and division of monoids:

Proposition 2.3.4 (see [Gin68]). *Let $\mathfrak{A}$ and $\mathfrak{B}$ be semiautomata, such that $\mathfrak{A} \leq \mathfrak{B}$. Then $M(\mathfrak{A}) \prec M(\mathfrak{B})$.*

The syntactic monoid satisfies the following property:

Proposition 2.3.5 (see [Str94]). *Let N be a monoid. N recognizes L iff $M(L) \prec N$.*

In particular, for every regular language L the syntactic monoid $M(L)$ is finite, since, for an arbitrary automaton $\mathfrak{A}$ with $L(\mathfrak{A}) = L$, $M(\mathfrak{A})$ is finite and recognizes L. Another immediate consequence of this is that $M(L)$ has the minimal number of elements among all monoids, which recognize L and is furthermore unique up to isomorphism. The syntactic monoid $M(L)$ is used in several algebraic characterizations of properties of languages. Perhaps the most prominent example of this is the following theorem:

Theorem 2.3.6 (Schützenberger, [Sch65]). *A language L is star-free iff $M(L)$ is group free.*

The transition monoid also leads to a very natural correspondence between semiautomata and transformation semigroups. More formally if $\mathfrak{A} = (Q, \Sigma, \delta)$ is a semiautomaton, then $(Q, M(\mathfrak{A}))$ is a TM. We call $(Q, M(\mathfrak{A}))$ the *transformation monoid equivalent to* $\mathfrak{A}$. In this situation we also say $\mathfrak{A}$ *and* $(Q, M(\mathfrak{A}))$ *correspond* and write $\mathfrak{A} \sim_{TS} (Q, M(\mathfrak{A}))$. Notice that in general several semiautomata can correspond to a single TS.

Recall that a TS (X, S) is called monogenic, if there exists an element $x_0 \in X$, such that $X = x_0 S$. We say a semiautomaton $\mathfrak{A} = (Q, \Sigma, \delta)$ is *monogenic* if there exists $q_0 \in Q$, such that for every state $q \in Q$ there exists a word $x_q \in \Sigma^*$ satisfying $q_0 \overline{x_q}^A = q$. We evidently have:

Remark 2.3.1. Let $\mathfrak{A}$ be a semiautomaton with state set Q. Then $\mathfrak{A}$ is monogenic if and only if $(Q, M(\mathfrak{A}))$ is.

Syntactic Monoids

Given a monoid M and a subset $P \subseteq M$, we define $\sim_P$ by $m \sim_P n$ iff for all $u, v \in M$ we have $umv \in P \Leftrightarrow unv \in P$. The relation $\sim_P$ is a congruence on M. The quotient $M/\sim_P$ is called *syntactic monoid of P and M*. A monoid M is called *syntactic* if there exists a subset $P \subseteq M$, such that $M \cong M/\sim_P$, i.e. $\sim_P = \mathrm{id}$. We have:

Proposition 2.3.7 (see [Eil74b]). *A monoid M is syntactic iff there exists and alphabet Σ and a language $L \subseteq \Sigma^*$, such that $M \cong M(L)$.*

Let $\varnothing \neq \mathbf{V}$ be a set of finite monoids closed under finite direct products and division, i.e. if $M, N \in \mathbf{V}$, then $M \times N \in \mathbf{V}$ and if $M \in \mathbf{V}$ and $N \prec M$ then $N \in \mathbf{V}$. A set $\mathbf{V}$ fulfilling these conditions is called a *pseudovariety*, a *variety of finite monoids* or often just a *variety*. Examples[1] of varieties are $\mathbf{M}$, the variety of all finite monoids, $\mathbf{Com}$, the variety of commutative monoids and $\mathbf{R}$, the variety of $\mathcal{R}$-trivial monoids. Another important example is the variety $\mathbf{A}$ of aperiodic monoids. Notice that, since $\mathbf{R}$ is a variety, all quotients of $\mathcal{R}$-trivial semigroups are in turn $\mathcal{R}$-trivial.

[1] For a proof that these sets are indeed varieties see, for instance, [Pin86].

Proposition 2.3.8 (see [Eil74b]). *If* $\mathbf{V}$ *is a variety, such that* $M \in \mathbf{V}$*, then there exists a syntactic monoid* $N \in \mathbf{V}$*, such that* $M \prec N$*.*

In other words, if M is a commutative ($\mathcal{R}$-trivial) monoid, then there exists a commutative ($\mathcal{R}$-trivial) syntactic monoid N, such that $M \prec N$.

Minimization

Define $\sim_L^R \subseteq \Sigma^* \times \Sigma^*$ by $u \sim_L^R v$, iff for all $w \in \Sigma^*$ we have

$$uw \in L \Leftrightarrow vw \in L$$

Clearly $\sim_L^R$ is a right congruence over Σ^*, i.e. for every $u \sim_L^R v$ and every $x \in \Sigma^*$ we have $ux \sim_L^R vx$. We define $\mathfrak{A}_L = (\Sigma^*/\sim_L^R, \Sigma, [\varepsilon], \delta, F)$ by $[w]\overline{\sigma} := [w \cdot \sigma]$ for all $\sigma \in \Sigma$ and $F = \{[w] | w \in L\}$. $\mathfrak{A}_L$ recognizes L. It is called the *canonical DFA for* L and has the minimal number of states among all automata recognizing L. We have $M(L) \cong M(\mathfrak{A}_L)$ (see [Str94]).

Let $\mathfrak{A} = (Q, \Sigma, q_0, \delta, F)$ be a DFA, such that all states $q \in Q$ are reachable from q_0, and let $L = L(\mathfrak{A})$. Define $\approx_L \subseteq Q \times Q$ by $p \approx_L q$ iff

$$p\overline{w} \in F \Leftrightarrow q\overline{w} \in F$$

for all $w \in \Sigma^*$. $\approx_L$ is a congruence of automata. The quotient $\mathfrak{A}_{red} := \mathfrak{A}/\approx_L$ recognizes L. We will often identify the DFA $\mathfrak{A}_{red}$ with its semiautomaton. Notice that $\mathfrak{A}_{red}$ is a homomorphic image of $\mathfrak{A}$ under the projection homomorphism $\pi_{\approx_L}$.

Proposition 2.3.9 ([HMU01]). *If* $\mathfrak{A}$ *is a DFA recognizing* $L \subseteq \Sigma^*$*, then* $\mathfrak{A}_{red} \cong \mathfrak{A}_L$*. In particular* $\mathfrak{A}_{red}$ *has the minimal number of states among all automata which recognize* L*.*

It follows that if $\mathfrak{A}$ and $\mathfrak{B}$ are two automata recognizing L, which have the minimal number of states, then $\mathfrak{A} \cong \mathfrak{B}$.

We will need the following characterization:

Proposition 2.3.10. *Let* $L \subseteq \Sigma^*$ *and let* $\mathfrak{A}$ *be a semiautomaton with input alphabet* Σ*. Then* $L \in \mathcal{L}(\mathfrak{A})$ *iff* $\mathfrak{A}_L \leq \mathfrak{A}$*.*

Proof. Suppose $L \in \mathcal{L}(\mathfrak{A})$. We consider $\mathfrak{A}$ as a DFA (having chosen a suitable initial state and set of final states). Then $\mathfrak{A}_L \cong \mathfrak{A}_{red}$, which is a homomorphic image of $\mathfrak{A}$. The DFA-homomorphism is in particular a homomorphism of semiautomata and the claim follows.

Conversely, if $\mathfrak{A}_L \leq \mathfrak{A}$, we can find a subsemiautomaton $\mathfrak{A}_0$ of $\mathfrak{A}$ and a surjective homomorphism $\varphi : \mathfrak{A}_0 \to \mathfrak{A}_L$. Since $\mathfrak{A}$ has input alphabet Σ we get that the states of $\mathfrak{A}_0$ form a subset of Q^A closed under the mappings $\overline{\sigma}$ for all $\sigma \in \Sigma$. Hence, every language over Σ recognized by $\mathfrak{A}_0$ will also be recognized by $\mathfrak{A}$.

Now suppose $L = L(\mathfrak{A}_L, q_0, F)$. Pick $p_0 \in (q_0)\varphi^{-1}$ arbitrary and set $F' = (F)\varphi^{-1}$. Let $w \in \Sigma^*$. Then φ becomes a DFA-homomorphism from $(\mathfrak{A}_0, p_0, F')$ onto $(\mathfrak{A}_L, q_0, F)$. Therefore the inclusion $L(\mathfrak{A}_0, p_0, F') \subseteq L$ is clear. For the converse, let $w \in L$. Then $(p_0\overline{w}^{A_0})\varphi = q_0\overline{w}^{A_L}$ since φ is a homomorphism. Now $q_0\overline{w}^{A_L} \in F$ and therefore $p_0\overline{w}^{A_0} \in F'$. Hence $L \subseteq L(\mathfrak{A}_0, p_0, F')$ and the claim follows. □

Permutation and Reset Automata

Suppose $\mathfrak{A} = (Q, \Sigma, \delta)$ is a semiautomaton and $\sigma \in \Sigma$. If $\overline{\sigma} \equiv q \in Q$ is a constant mapping with $\mathrm{im}(\overline{\sigma}) = \{q\}$, then we call σ a *reset input* of $\mathfrak{A}$. The state q is called the *target* of σ. If every input $\sigma \in \Sigma$ induces either the identity mapping $\mathrm{id}_Q = \overline{\sigma}$ or is a reset input, we call $\mathfrak{A}$ a *reset automaton*. If $\mathfrak{R} = (Q, \Sigma, \delta)$ is a reset automaton with $|Q| = n$, we call $\mathfrak{R}$ an n-state reset automaton, or n-RA for short. We will be particularly interested in the case, where $n = 2$, i.e. $\mathfrak{R}$ is a 2-RA. In this case, it is no restriction to assume that $Q = \mathbb{B}$. In Section 2.4 we will see why 2-RAs are sufficient in the study of reset automata. For this reason, we will from that point on usually refer to 2-RAs simply as *resets* or *reset automata* (see Remark 2.4.1), bearing in mind, that this name is also used in the more general case of n-RAs. Notice that the transition monoid of every reset automaton is group free. In fact, if $\mathfrak{A}$ is an n-RA, then $M(\mathfrak{A})$ can be embedded into U_n.

A letter $\sigma \in \Sigma$, such that $\overline{\sigma}$ is bijective, is called *permutation input*. If σ is a permutation input, then the inverse mapping $\overline{\sigma}^{-1}$ is of the form $\overline{\sigma}^k$ for some $k \in \mathbb{N}$. In particular $\overline{\sigma}$ possesses an inverse in $M(\mathfrak{A})$. An automaton, for which every input is a permutation input, is called a *permutation automaton*. Notice that the transition monoid $M(\mathfrak{A})$ of a permutation automaton is a group. The group $M(\mathfrak{A})$ is then called the *group associated with* $\mathfrak{A}$ or the group *corresponding to* $\mathfrak{A}$. We call a permutation automaton simple (cyclic) if the corresponding group is simple (cyclic). Permutation automata of the form $\mathfrak{G} = (G, \Sigma, \delta^G)$ with $M(\mathfrak{G}) = G$ for some group G are called *grouplike automata*. In other words, in a grouplike automaton we can identify the states with elements of the group $M(\mathfrak{G})$.

The Straubing Hierarchy and the Brzozowski Hierarchy

The *Straubing Hierarchy (over Σ)* is defined inductively as follows. Level 0, denoted $\mathcal{V}_0$, consists of the languages Σ^* and $\varnothing$. Given level $n \in \mathbb{N}_0$, we define $\mathcal{V}_{n+1}$ to be the Boolean closure of all languages of the form $L_1\sigma_1L_2\cdots L_{n-1}\sigma_{m-1}L_m$, where $L_i \in \mathcal{V}_n$ for $i = 1, \ldots, m$ and $\sigma_i \in \Sigma$ for $i = 1, \ldots, m-1$. Notice that in particular $\mathcal{V}_n \subseteq \mathcal{V}_{n+1}$ for all $n \in \mathbb{N}_0$. The *level* of a star-free language L is the minimal number $n \in \mathbb{N}_0$, such that $L \in \mathcal{V}_n$. To see that the level of every star-free language is well-defined, we state:

Proposition 2.3.11 (see [Pin86]). *$\mathcal{SF} = \bigcup_{n\in\mathbb{N}_0} \mathcal{V}_n$ and for every $n \in \mathbb{N}_0$ we have $\mathcal{V}_n \subsetneq \mathcal{V}_{n+1}$.*

A language $L \in \mathcal{V}_1$ is called *piecewise testable*. From the definition of the hierarchy we immediately get:

Proposition 2.3.12. *The piecewise testable languages are precisely the Boolean closure of all languages of the form $\Sigma^*\sigma_1\Sigma^*\cdots\Sigma^*\sigma_n\Sigma^*$.*

Piecewise testable languages have an interesting algebraic characterization:

Theorem 2.3.13 (Simon, [Sim75]). *A language L is piecewise testable iff $M(L)$ is $\mathcal{J}$-trivial.*

A similar hierarchy is the so called *Brzozowski Hierarchy (over Σ)*, which is defined as follows: $\mathcal{B}_0$ is the class of all finite and co-finite languages (i.e. languages, the complement of which is a finite set). Then for $n \in \mathbb{N}_0$ we define $\mathcal{B}_{n+1}$ as the Boolean closure of all languages of the form $L_1\sigma_1 L_2\sigma_2 L_3 \cdots L_{m-1}\sigma_{m-1}L_m$ with $L_i \in \mathcal{B}_n$ for all $i = 1, \ldots, m$ and $\sigma_i \in \Sigma$ for all $i = 1, \ldots, m-1$. Again we note that $\mathcal{B}_n \subseteq \mathcal{B}_{n+1}$. Also, we again have:

Proposition 2.3.14 (see [Pin86])**.** *$\mathcal{SF} = \bigcup_{n\in\mathbb{N}_0} \mathcal{B}_n$ and for every $n \in \mathbb{N}_0$ we have $\mathcal{B}_n \subsetneq \mathcal{B}_{n+1}$.*

Given a star-free language L, we call the least $n \in \mathbb{N}_0$, such that $L \in \mathcal{B}_n$, the *dot-depth* of L. It is apparent from the definition that $\mathcal{V}_n \subseteq \mathcal{B}_n$ for all $n \in \mathbb{N}_0$.

Commutative languages

Let $w = \sigma_1 \ldots, \sigma_n \in \Sigma^*$. We define $\mathrm{Perm}(w) = \{\sigma_{(1)\pi} \cdots \sigma_{(n)\pi} | \pi \in S_n\}$. A language $L \subseteq \Sigma^*$ is called *commutative*, if it is closed under permutations, i.e. if for every $w \in L$ we have $\mathrm{Perm}(w) \subseteq L$. Again, there exists an algebraic characterization:

Proposition 2.3.15. *L is commutative iff $M(L)$ is commutative.*

Proof. If $M(L)$ is commutative then clearly L is commutative. Hence, let L be commutative. Then for all $w \in \Sigma^*$ we have $\mathrm{Perm}(w) \subseteq [w]_{\sim_L}$ (if $w' \in \mathrm{Perm}(w)$ then $xuv \in L$ iff $uw'v \in L$ since the right side is a permutation of the left). Therefore the congruence $\sim$ induced by Perm (i.e. $u \sim v$ iff v is a permutation of u) is finer than $\sim_L$, i.e. $\sim \subseteq \sim_L$. By Theorem 2.2.2 there exists a surjective homomorphism from $\Sigma^*/\sim$ onto $M(L)$. Since $\Sigma^*/\sim$ is clearly commutative, $M(L)$ must be commutative as well. □

Notice that this statement holds even if L is not regular.

If L_1, L_2 are regular commutative languages, then clearly $\overline{L_1}$, $L_1 \cup L_2$ and therefore $L_1 \cap L_2$ are regular commutative languages. Furthermore Σ^* and $\varnothing$ are commutative languages. Thus the commutative languages form a Boolean algebra.

2.4 Products

Direct Products of Automata

Given two semiautomata $\mathfrak{A} = (Q^A, \Sigma, \delta)$ and $\mathfrak{B} = (Q^B, \Sigma, \delta^B)$ over the same input alphabet Σ, we define the *direct product* $\mathfrak{A} \times \mathfrak{B}$ of $\mathfrak{A}$ and $\mathfrak{B}$ by

$$\mathfrak{A} \times \mathfrak{B} = (Q^A \times Q^B, \Sigma, \delta^{A\times B})$$

where $(q^A, q^B)\overline{\sigma}^{A\times B} = (q^A\overline{\sigma}^A, q^B\overline{\sigma}^B)$ for all $\sigma \in \Sigma$. Evidently every language in $\mathcal{L}(\mathfrak{A} \times \mathfrak{B})$ is a Boolean combination of languages in $\mathcal{L}(\mathfrak{A})$ and $\mathcal{L}(\mathfrak{B})$.

Direct products give us a means of simplifying n-RAs (a slightly weaker[2] form of this lemma is also stated in [Gin68]):

[2] The lemma in [Gin68] does not make a statement about the number n of 2-RAs needed.

Lemma 2.4.1. *Let $\mathfrak{A} = (Q, \Sigma, \delta)$ be an n-RA. Then there exist n 2-RAs $\mathfrak{R}_1 \ldots, \mathfrak{R}_n$ with the same input alphabet Σ, such that*

$$\mathfrak{A} \leq \mathfrak{R}_1 \times \cdots \times \mathfrak{R}_n$$

Proof. Let $Q = \{q_1, \ldots, q_n\}$. Define for all $q \in Q$ the set $\Sigma_q = \{\sigma \in \Sigma | \overline{\sigma}^A \equiv q\}$ of all inputs with target q. Let $\Sigma_{\mathrm{id}} = \{\sigma \in \Sigma | \overline{\sigma}^A = \mathrm{id}\}$. Then $\Sigma_{\mathrm{id}}, \Sigma_{q_1}, \ldots, \Sigma_{q_n}$ forms a partition of Σ. We define $\mathfrak{R}_i = (\mathbb{B}, \Sigma, \delta^{R_i})$ by

$$q\overline{\sigma}^{R_i} = \begin{cases} 1 & \sigma \in \Sigma_{q_i} \\ 0 & \sigma \notin (\Sigma_{\mathrm{id}} \cup \Sigma_{q_i}) \\ q & \text{otherwise} \end{cases}$$

for all $q \in \mathbb{B}$. Denote $\mathfrak{B} = \mathfrak{R}_1 \times \cdots \times \mathfrak{R}_n$. Let $Q_0 = \{(x_1, \ldots, x_n) \in \mathbb{B}^n | \sum_{i=1}^n x_i = 1\}$ be the set of all states of $\mathfrak{B}$ for which exactly one automaton is in state 1. Then $\mathfrak{B}_0 = (Q_0, \Sigma, \delta^B)$ forms a subsemiautomaton of $\mathfrak{B}$. Define $\varphi : \mathfrak{B}_0 \to \mathfrak{A}$ by $(x_1, \ldots, x_n)\varphi = q_i$ iff $x_i = 1$. This mapping is well defined and is furthermore a homomorphism of semiautomata. Clearly φ is also surjective. The claim follows. □

Remark 2.4.1. This result enables us to always reduce the case of an n-RA to that of several 2-RAs. For this reason we will, unless otherwise stated, henceforth mean a 2-RA, whenever we say *reset* or *reset automaton*.

Suppose $\mathfrak{A}$,$\mathfrak{B}$ and $\mathfrak{C}$ are semiautomata, such that $\mathfrak{A} \leq \mathfrak{C}$. Pick a subsemiautomaton $\mathfrak{C}_0$ of $\mathfrak{C}$ and a surjective homomorphism $\varphi : \mathfrak{C}_0 \to \mathfrak{A}$. Then $\mathfrak{C}_0 \times \mathfrak{B}$ is a subsemiautomaton of $\mathfrak{C} \times \mathfrak{B}$ and $\psi : \mathfrak{C}_0 \times \mathfrak{B} \to \mathfrak{A} \times \mathfrak{B}$, $(q_1, q_2) \mapsto ((q_1)\varphi, q_2)$ is a surjective homomorphism. Thus $\mathfrak{A} \times \mathfrak{B} \leq \mathfrak{C} \times \mathfrak{B}$. Since $\mathfrak{A} \times \mathfrak{B} \cong \mathfrak{B} \times \mathfrak{A}$ an analogous statement holds if $\mathfrak{C} \geq \mathfrak{B}$. Furthermore, since $\leq$ is a transitive relation, we get the following lemma:

Lemma 2.4.2. *If $\mathfrak{A}, \mathfrak{B}, \mathfrak{C}$ and $\mathfrak{D}$ are semiautomata, such that $\mathfrak{A} \leq \mathfrak{C}$ and $\mathfrak{B} \leq \mathfrak{D}$, then:*

$$\mathfrak{A} \times \mathfrak{B} \leq \mathfrak{C} \times \mathfrak{D}$$

Cascade Products of Automata

Let $\mathfrak{A} = (Q^A, \Sigma^A, \delta^A)$ and $\mathfrak{B} = (Q^B, \Sigma^B, \delta^A)$ be semiautomata. Suppose $\Sigma^A \times Q^A \subseteq \Sigma^B$. We can define transitions $\overline{\sigma}^{A \circ B}$ on $Q^A \times Q^B$ for each $\sigma \in \Sigma^A$ by

$$(q^A, q^B)\overline{\sigma}^{A \circ B} = \left(q^A\overline{\sigma}^A, q^B\overline{(\sigma, q^A)}^B\right) \tag{2.4.1}$$

Equation (2.4.1) gives rise to the following definition:

Definition 2.4.1 (Cascade Product)**.** Let $\mathfrak{A} = (Q^A, \Sigma^A, \delta^A)$ and $\mathfrak{B} = (Q^B, \Sigma^B, \delta^A)$ be semiautomata such that $\Sigma^A \times Q^A \subseteq Q^B$. Then the *cascade product of $\mathfrak{A}$ and $\mathfrak{B}$* is given by $\mathfrak{A} \circ \mathfrak{B} = (Q^A \times Q^B, \Sigma^A, \delta^{A \circ B})$, where the mappings $\overline{\sigma}^{A \circ B}$ are defined as in (2.4.1).

The following theorem states a few important properties of the cascade product:

Theorem 2.4.3 (see [Gin68]). *Let $\mathfrak{A}$, $\mathfrak{B}$ and $\mathfrak{C}$ be semiautomata, such that $\mathfrak{A} \leq \mathfrak{B}$. Then*

(a) $\mathfrak{C} \circ \mathfrak{A} \leq \mathfrak{C} \circ \mathfrak{B}$.

(b) $(\mathfrak{A} \circ \mathfrak{B}) \circ \mathfrak{C} \cong \mathfrak{A} \circ (\mathfrak{B} \circ \mathfrak{C})$, *i.e. the cascade product is associative.*

Direct Products of Monoids

Let M and N be monoids. Then $M \times N$ is a monoid with multiplication $(m,n) \cdot (m',n') = (mm',nn')$. The neutral element is $(1_M, 1_N)$. All languages recognized by $M \times N$ are Boolean combination of languages recognized by M and N. If $\mathfrak{A}$ and $\mathfrak{B}$ are semiautomata, then $M(\mathfrak{A} \times \mathfrak{B})$ can be embedded into $M(\mathfrak{A}) \times M(\mathfrak{B})$. In general, $M(\mathfrak{A} \times \mathfrak{B})$ and $M(\mathfrak{A}) \times M(\mathfrak{B})$ are not isomorphic.

We have a statement similar to Lemma 2.4.1:

Lemma 2.4.4. *Let $n \in \mathbb{N}$. Then $U_n \prec \underbrace{U_2 \times \cdots \times U_2}_{n\text{-times}}$.*

Proof. Consider $\mathfrak{A} = (U_n \setminus \{1\}, U_n, \delta)$, with $u\overline{v} = u \cdot v$ for $u \in U_n \setminus \{1\}$ and $v \in U_2$. Then $\mathfrak{A}$ is an n-RA and by Lemma 2.4.1 we have $\mathfrak{A} \leq \mathfrak{R}_1 \times \cdots \times \mathfrak{R}_n$ for suitable resets $\mathfrak{R}_i$, $i = 1, \ldots, n$. Now $M(\mathfrak{R}_1 \times \cdots \times \mathfrak{R}_n) \prec M(\mathfrak{R}_1) \times \cdots \times M(\mathfrak{R}_n)$ and we have already seen that $M(\mathfrak{R}_i) \prec U_2$ for $i = 1, \ldots, n$. By Proposition 2.3.4 the claim follows. □

The Wreath Product of Transformation Semigroups

Let (X_1, S_1) and (X_2, S_2) be two TSs. We set $S := S_1^{X_2} \times S_2$, where $S_1^{X_2}$ is the set of all mappings from X_2 into S_1. The set $S_1^{X_2}$ is a semigroup with pointwise multiplication. More formally, if $F_1, F_2 \in S_1^{X_2}$ we define $F_1 \cdot F_2$ by $x \mapsto (x)F_1 \cdot (x)F_2$ where the operation on the right-hand side is the operation of S_1. It follows that, if S_1 is a monoid, then so is $S_1^{X_2}$. We now define a left action of S_2 on $S_1^{X_2}$. To this end let $s.F$ be defined by $x \mapsto (xs)F$ for all $F \in S_1^{X_2}$, $s \in S_2$ and $x \in X_2$. It is easy to see that, if S_2 is a monoid, this is a monoid action. Notice that the action is distributive with respect to the multiplication in $S_1^{X_2}$:

$$s.(F \cdot G) = (s.F) \cdot (s.G) \tag{2.4.2}$$

for all $s \in S_2$ and all $F, G \in S_1^{X_2}$. We are now ready to define a multiplication on $S_1^{X_2} \times S_2$. We set

$$(F_1, s_1) \cdot (F_2, s_2) = (F_1 \cdot (s_1.F_2), s_1 s_2) \tag{2.4.3}$$

Note that it is not necessary to bracket the expression $s_1.F_2$ in the above equation. The brackets were used for readability but shall often be omitted in what follows. Similarly, we will often write $s_1 F_1$ instead of $s_1.F_1$ so as not to be required to make excessive use of various operation symbols. We verify that the multiplication from

(2.4.3) is associative:

$$\begin{aligned}\big((F,f)\cdot(G,g)\big)\cdot(H,h) &= (F\cdot fG, fg)\cdot(H,h)\\ &= (F\cdot fG\cdot fgH, fgh)\\ &= \big(F\cdot f(G\cdot gH), fgh\big) && \text{by (2.4.2)}\\ &= (F,f)\cdot(G\cdot gH, gh)\\ &= (F,f)\cdot\big((G,g)\cdot(H,h)\big)\end{aligned}$$

Hence, with (2.4.3), the set $S_1^{X_2}\times S_2$ becomes a semigroup. If S_1 and S_2 are monoids, then so is $S_1^{X_2}\times S_2$. In this case the identity is $(F^1, 1_{S_2})$, where $F^1 \equiv 1_{S_1}$ is the function with constant value 1_{S_1}. We can define a right action of $S_1^{X_2}\times S_2$ on $X_1\times X_2$:

$$(x_1,x_2).(F,s) = (x_1(x_2)F, x_2 s)$$

Then we have

$$\begin{aligned}(x_1,x_2).\big((F_1,s_1)(F_2,s_2)\big) &= (x_1,x_2).(F_1\cdot s_1.F_2, s_1s_2)\\ &= (x_1(x_2)F_1\cdot(x_2s_1)F_2, x_2s_1s_2)\\ &= \big((x_1,x_2).(F_1,s_1)\big).(F_2,s_2)\end{aligned}$$

Hence the mapping defined above is really a semigroup action of $S_1^{X_2}\times S_2$ on $X_1\times X_2$. Obviously this action is also a monoid action, provided $S_1^{X_2}\times S_2$ is a monoid. We want to show, that $(X_1\times X_2, S_1^{X_2}\times S_2)$ is a TS. To this end we need to prove that the action described above is faithful. Let (F_1,s_1) and (F_2,s_2) be such that $(x_1,x_2).(F_1,s_1) = (x_1,x_2).(F_2,s_2)$ for all $(x_1,x_2)\in X_1\times X_2$. Then obviously $s_1 = s_2$ since (X_2,S_2) is a TS. Furthermore $(x_2)F_1 = (x_2)F_2$ for all $x_2\in X_2$. Hence $(F_1,s_1) = (F_2,s_2)$.

Definition 2.4.2 (Wreath-Product)**.** Let (X_1,S_1) and (X_2,S_2) be transformation semigroups. Then $(X_1\times X_2, S_1^{X_2}\times S_2)$ together with the operation from (2.4.3) is a transformation semigroup, called the *wreath product* $(X_1,S_1)\wr(X_2,S_2)$ of (X_1,S_1) and (X_2,S_2).

The wreath product is associative:

Lemma 2.4.5 (see [Str94])**.** *If (X,S), (Y,T) and (Z,U) are TSs, then*

$$\big((X,S)\wr(Y,T)\big)\wr(Z,U) \cong (X,S)\wr\big((Y,T)\wr(Z,U)\big)$$

The following lemma states that the wreath product satisfies a certain monotonicity:

Lemma 2.4.6 (see [Str94])**.** *Let $(P_1,S_1)\prec(P_2,S_2)$ and $(Q_1,T_1)\prec(Q_2,T_2)$ be four TSs. Then*

$$(P_1,S_1)\wr(Q_1,T_1)\prec(P_2,S_2)\wr(Q_2,T_2)$$

The Block Product of Semigroups

Let M and N be semigroups. Let U denote the semigroup $M^{N\times N}$ of all mappings from $N \times N$ into M with pointwise multiplication. We define a left action of N on U by $(n_1, n_2)(n.F) = (n_1 n, n_2)F$ for all $F \in U$ and all $n \in N$. We likewise define a right action by $(n_1, n_2)(F.n) = (n_1, nn_2)F$ for all $F \in U$ and all $n \in N$. Both actions are monoid actions in the case where N is a monoid. We now observe that both actions satisfy the following property, which we only state for the left action:

$$n.(F \cdot G) = n.F \cdot n.G \tag{2.4.4}$$

Notice that the actions defined are similar to the action used in the preceding paragraph on wreath products. Also, a similar version of (2.4.4) was already stated in (2.4.2). Furthermore we have that

$$(n_1.F).n_2 = n_1.(F.n_2) \tag{2.4.5}$$

for all $n_1, n_2 \in N$ and all $F \in U$. A left and a right action satisfying (2.4.5) are said to be *compatible*. Since the above actions are compatible, we can drop the brackets and write $n_1.F.n_2$ without ambiguity. For readability we will often drop the dot in the action of N on U and write nF instead of $n.F$ and Fn instead of $F.n$.

We are now ready to define an operation on $U \times N$:

$$(F, n) \cdot (G, m) = ((Fm) \cdot (nG), nm) \tag{2.4.6}$$

for all $F, G \in U$ and all $m, n \in N$. In order to enhance readability, we use the convention that the action of N on U precedes the operation on N in the order of operation, i.e. $Fn \cdot mG$ should be interpreted as $(Fn) \cdot (nG)$ as opposed to $F(nm)G$ (which would not even have clear semantics), $(Fnm)G$ or $F(nmG)$. Because of (2.4.4) we have

$$\begin{aligned}
\big((F, n) \cdot (G, m)\big) \cdot (H, o) &= (Fm \cdot nG, nm) \cdot (H, o) \\
&= \big((Fm \cdot nG)o) \cdot (nmH), nmo\big) \\
&= (Fmo \cdot nGo \cdot nmH, nmo) \\
&= \big(Fmo \cdot (n(Go \cdot mH)), nmo\big) \\
&= (F, n) \cdot (Go \cdot mH, mo) \\
&= (F, n) \cdot \big((G, m) \cdot (H, o)\big)
\end{aligned}$$

Hence the multiplication is associative and $U \times N$ becomes a semigroup with the multiplication given by (2.4.6). Obviously $U \times N$ is a monoid, if U and N are monoids.

Definition 2.4.3 (Block Product)**.** Suppose M and N are semigroups. Then the semigroup $M^{N\times N} \times N$ with the multiplication given by (2.4.6) is called the *block product of M and N*, denoted $M \wr N$.

The block product is not associative in general (not even up to isomorphism) as a simple counting argument shows. Suppose M and N are finite semigroups with $|M| > 1$ and $|N| = 2$. Then

$$|(M \wr N) \wr N| = (|M|^4 \cdot 2)^4 \cdot 2 = |M|^{16} \cdot 32$$

but

$$|M \square \underbrace{(N \square N)}_{2^4 \cdot 2 = 32 \text{ elements}} | = |M|^{32^2} \cdot 32$$

We will need the following lemma:

Lemma 2.4.7. *Let S, T and U be semigroups. If S can be embedded into T, then $S \square U$ can be embedded into $T \square U$.*

Proof. Let $\varphi : S \to T$ be an injective homomorphism. We then define $\psi : S \square U \to T \square U$ by $(F, u)\psi = ((F \circ \varphi), u)$. We have

$$\begin{aligned}\big((F,u)(G,v)\big)\psi &= (Fv \cdot uG, uv)\psi \\ &= \big((Fv \cdot uG) \circ \varphi, uv\big) \\ &= \big((Fv) \circ \varphi \cdot (uG) \circ \varphi, uv\big) \\ &= \big((F \circ \varphi)v \cdot u(G \circ \varphi), uv\big) \\ &= (F \circ \varphi, u) \cdot (G \circ \varphi, v) = (F,u)\psi \cdot (G,v)\psi\end{aligned}$$

Thus, ψ is a homomorphism. To see that ψ is injective, suppose $(F, u)\psi = (G, v)\psi$. Then $(F \circ \varphi, u) = (G \circ \varphi, v)$. Obviously $u = v$ and because φ is injective we have that $(x)F \circ \varphi = (x)G \circ \varphi$ for all $x \in U \times U$ iff $(x)F = (x)G$ for all $x \in U \times U$. □

Notice that if φ is a monoid homomorphism, then so is ψ.

3 A Survey of Cascade, Wreath and Block Products

In this chapter we investigate the relationship between the various products introduced in Section 2.4. We then apply the results of this work to the famous Krohn-Rhodes Theorem in order to show how the different versions of this theorem are related.

3.1 Cascade and Wreath Products

Consider a semiautomaton $\mathfrak{A} \circ \mathfrak{B}$. How is the transition monoid $M(\mathfrak{A} \circ \mathfrak{B})$ of this automaton related to the monoids $M(\mathfrak{A})$ and $M(\mathfrak{B})$? The following theorem addresses this question:

Theorem 3.1.1. *Let $\mathfrak{A} = (Q^A, \Sigma, \delta^A)$ and let $\mathfrak{B} = (Q^B, \Sigma', \delta^B)$ where $\Sigma' \supseteq \Sigma \times Q^A$. Furthermore let $(Q^A, M(\mathfrak{A}))$ and $(Q^B, M(\mathfrak{B}))$ be the transformation monoids corresponding to $\mathfrak{A}$ and $\mathfrak{B}$ respectively. Suppose $(Q^B, M(\mathfrak{B})) \wr (Q^A, M(\mathfrak{A})) = (Q^B \times Q^A, M)$. Then $M(\mathfrak{A} \circ \mathfrak{B})$ can be embedded into M.*

Proof. Let $x = x_1 \cdots x_n \in \Sigma^*$ and $(q_A, q_B) \in Q^A \times Q^B$. Starting from $\overline{x}^{A \circ B}$ we wish to isolate the mapping acting on q_B. However, in general this mapping will depend on q_A. We apply $\overline{x}^{A \circ B}$ to (q_A, q_B) to obtain a clear description of how $\overline{x}^{A \circ B}$ acts on q_B.

$$
\begin{aligned}
(q_A, q_B)\overline{x}^{A \circ B} &= (q_A, q_B)\overline{x}_1^{A \circ B} \cdots \overline{x}_n^{A \circ B} \\
&= (q_A \overline{x}_1^A, q_B \overline{(q_A, x_1)}^B)\overline{x}_2^{A \circ B} \cdots \overline{x}_n^{A \circ B} \\
&\vdots \\
&= (q_A \overline{x}^A, q_B \underbrace{\overline{(q_A, x_1)}^B \cdots \overline{(q_A \overline{x_1 \cdots x_{n-1}}^A, x_n)}^B}_{= (\overline{x}^{A \circ B}, q_A)\pi}) \qquad (*)
\end{aligned}
$$

As indicated above, we wish to denote the mapping acting on q_B by $(\overline{x}^{A \circ B}, q_A)\pi$. However, the description of this mapping in $(*)$ strongly depends on $x \in \Sigma^*$. Hence the problem of well definedness arises. To see that $(\overline{x}^{A \circ B}, q_A)\pi$ is well defined, suppose $y \in \Sigma^*$ induces the same mapping $\overline{y}^{A \circ B} = \overline{x}^{A \circ B}$. Then in particular $(q_A, q_B)\overline{y}^{A \circ B} = (q_A, q_B)\overline{x}^{A \circ B}$ and hence $(\overline{y}^{A \circ B}, q_A)\pi = (\overline{x}^{A \circ B}, q_A)\pi$.

We are now ready to define a homomorphism $\psi : M(\mathfrak{A} \circ \mathfrak{B}) \to M$ by $(\overline{x}^{A \circ B})\psi = (F_{\overline{x}^{A \circ B}}, \overline{x}^A)$, where $(q)F_{\overline{x}^{A \circ B}} = (\overline{x}^{A \circ B}, q)\pi \in M(\mathfrak{B})$ for all $q \in Q^A$. Then

$$\begin{aligned}(\overline{x}^{A \circ B} \cdot \overline{y}^{A \circ B})\psi &= (\overline{xy}^{A \circ B})\psi = (F_{\overline{xy}^{A \circ B}}, \overline{xy}^A) \\ &= (F_{\overline{x}^{A \circ B}} \cdot \overline{x}^A . F_{\overline{y}^{A \circ B}}, \overline{x}^A \overline{y}^A) \qquad \text{by } (*) \\ &= (F_{\overline{x}^{A \circ B}}, \overline{x}^A)(F_{\overline{y}^{A \circ B}}, \overline{y}^A) \\ &= (\overline{x}^{A \circ B})\psi \cdot (\overline{y}^{A \circ B})\psi\end{aligned}$$

We evidently have that $(q_A, q_B)\overline{x}^{A \circ B} = (q_1, q_2)$ iff $(q_B, q_A)\big((\overline{x}^{A \circ B})\psi\big) = (q_2, q_1)$ for all $q^A \in Q^A$ and all $q^B \in Q^B$. We see that both $\overline{x}^{A \circ B}$ and $(\overline{x}^{A \circ B})\psi$ essentially have the same effect. As both $M(\mathfrak{A} \circ \mathfrak{B})$ and M act faithfully on the corresponding sets, we conclude that ψ is injective. □

Suppose the cascade product $\mathfrak{A} \circ \mathfrak{B}$ of $\mathfrak{A}$ and $\mathfrak{B}$ is direct, i.e. $\overline{(\sigma, q)}^B = \overline{(\sigma, q')}^B$ for all $q, q' \in Q^A$ and all $\sigma \in \Sigma$. In that case $M(\mathfrak{A} \circ \mathfrak{B}) \hookrightarrow M(\mathfrak{A}) \times M(\mathfrak{B})$. In particular we have $|M(\mathfrak{A} \circ \mathfrak{B})| \leq |M(\mathfrak{A})| \cdot |M(\mathfrak{B})|$. However, if $|M(\mathfrak{A})|, |M(\mathfrak{B})| \geq 2$ we have $|M| = |M(\mathfrak{B})|^{|M(\mathfrak{A})|} \cdot |M(\mathfrak{A})| > |M(\mathfrak{A})| \cdot |M(\mathfrak{B})|$. Hence we obtain the following remark:

Remark 3.1.1. In general M and $M(\mathfrak{A} \circ \mathfrak{B})$ are not isomorphic.

We can restate Theorem 3.1.1 in a slightly different way:

Corollary 3.1.2. *Suppose $\mathfrak{A}$ and $\mathfrak{B}$ are as defined above. Then*

$$(Q^A \times Q^B, M(\mathfrak{A} \circ \mathfrak{B})) \prec (Q^B, M(\mathfrak{B})) \wr (Q^A, M(\mathfrak{A}))$$

Proof. By Theorem 3.1.1 $M(\mathfrak{A} \circ \mathfrak{B})$ can be embedded into the semigroup of the TS on the right-hand side. We pick the homomorphism ψ from the proof of Theorem 3.1.1. It now suffices to find a surjective map $f : Q^B \times Q^A \to Q^A \times Q^B$ and to prove that f and ψ commute in the sense of Definition 2.2.1.

A natural choice for f is to use the map $(q^B, q^A) \mapsto (q^A, q^B)$. We then have

$$\begin{aligned}((q^B, q^A)f)\overline{x}^{A \circ B} &= (q^A, q^B)\overline{x}^{A \circ B} \\ &= ((q^B, q^A)(\overline{x}^{A \circ B}\psi))f \qquad \text{see proof of Theorem 3.1.1}\end{aligned}$$

for all $q^A \in Q^A$ and all $q^B \in Q^B$ as required. □

We can extend this result to products with an arbitrary (but finite) number of factors. Suppose $\mathfrak{A}_1, \ldots, \mathfrak{A}_n$ are semiautomata, $\mathfrak{A}_i = (Q_i, \Sigma_i, \delta^i)$, and let $\mathfrak{B} = \mathfrak{A}_1 \circ \cdots \circ \mathfrak{A}_n$ denote their cascade product (we assume that the alphabets Σ_i of the $\mathfrak{A}_i$ are of the appropriate form). We claim that

$$(Q_1 \times \cdots \times Q_n, M(\mathfrak{B})) \prec (Q_n, M(\mathfrak{A}_n)) \wr \cdots \wr (Q_1, M(\mathfrak{A}_1)) \tag{3.1.1}$$

To prove this, we proceed by induction on n. For $n = 2$ this is precisely the statement of the previous corollary. Now suppose $n > 2$. Let

$$\mathfrak{C} = \mathfrak{A}_1 \circ \cdots \circ \mathfrak{A}_{n-1}$$

Then $\mathfrak{B} = \mathfrak{C} \circ \mathfrak{A}_n$ and by the previous corollary:

$$(Q_1 \times \cdots \times Q_n, M(\mathfrak{B})) \prec (Q_n, M(\mathfrak{A}_n)) \wr (Q_1 \times \cdots \times Q_{n-1}, M(\mathfrak{C}))$$

By the induction hypothesis and by Lemma 2.4.6, we obtain:

$$(Q_n, M(\mathfrak{A}_n)) \wr (Q_1 \times \cdots \times Q_{n-1}, M(\mathfrak{C})) \prec (Q_n, M(\mathfrak{A}_n)) \wr \cdots \wr (Q_1, M(\mathfrak{A}_1))$$

and (3.1.1) holds for n. Hence:

Corollary 3.1.3. *Let $\mathfrak{A}_1, \ldots, \mathfrak{A}_n$, $\mathfrak{A}_i = (Q_i, \Sigma_i, \delta^i)$, $n \geq 2$, be semiautomata and let $\mathfrak{B} = \mathfrak{A}_1 \circ \cdots \circ \mathfrak{A}_n$ denote their cascade product. Then*

$$(Q_1 \times \cdots \times Q_n, M(\mathfrak{B})) \prec (Q_n, M(\mathfrak{A}_n)) \wr \cdots \wr (Q_1, M(\mathfrak{A}_1))$$

We have seen that a cascade product of semiautomata yields a wreath product of transformation semigroups, such that the factors of both products correspond. We now want to investigate the opposite direction. Given a wreath product of transformation semigroups, can we find a cascade product of semiautomata, such that the factors of both products correspond? The answer is yes, but the proof is a bit more involved.

We first need a broader notion of coverings of automata than we have used before. We will only be using this more general form of covering in the context of investigating the question asked in the previous paragraph. To emphasize this, we use a different notation:

Definition 3.1.1. Let $\mathfrak{A} = (Q^A, \Sigma^A, \delta^A)$ and $\mathfrak{B} = (Q^B, \Sigma^B, \delta^B)$. If there exists a subset $Q_0^B \subseteq Q^B$ of states, as well as mappings $\xi : \Sigma^A \to \Sigma^B$ and $\eta : Q_0^B \to Q^A$, where η is surjective, such that

$$(q\eta)\overline{\sigma}^A = \left(q\overline{(\sigma\xi)}^B\right)\eta$$

for all $q \in Q_0^B$, we write $\mathfrak{A} \leq_G \mathfrak{B}$.

The following lemma states the connection between this general notion of covering and the definition used so far:

Lemma 3.1.4 (see [Gin68])**.** *If $\xi = \mathrm{id}$, that is (in particular) $\Sigma^A \subseteq \Sigma^B$, then $\mathfrak{A} \leq_G \mathfrak{B}$ implies $\mathfrak{A} \leq \mathfrak{B}$.*

The key observation in the proof of this lemma is that, in this situation, the mapping η is in fact a homomorphism of semiautomata.

The previous lemma can be used to find a way of reducing $\leq_G$ to $\leq$ at the expense of changing the alphabet of $\mathfrak{B}$. The basic idea is to consider $\mathrm{im}\,(\xi) \subseteq \Sigma^B$ and duplicate each letter $\sigma \in \mathrm{im}\,(\xi)$ sufficiently many times and then rename the duplicates so that $\sigma_A\xi = \sigma_A$ for all $\sigma_A \in \Sigma^A$.

Lemma 3.1.5. *Let* $\mathfrak{A} = (Q^A, \Sigma^A, \delta^A)$, $\mathfrak{B} = (Q^B, \Sigma^B, \delta^B)$ *and* $\mathfrak{C}$ *be semiautomata, such that* $\mathfrak{A} \leq_G \mathfrak{B} \circ \mathfrak{C}$. *Then there exist semiautomata* $\hat{\mathfrak{B}} = (Q^B, \Sigma^{\hat{B}}, \delta^{\hat{B}})$, $\hat{\mathfrak{C}} = (Q^C, \Sigma^{\hat{C}}, \delta^{\hat{C}})$, *such that* $M(\mathfrak{B}) = M(\hat{\mathfrak{B}})$, $M(\mathfrak{C}) = M(\hat{\mathfrak{C}})$ *and* $\mathfrak{A} \leq \hat{\mathfrak{B}} \circ \hat{\mathfrak{C}}$.

Proof. Pick $\xi : \Sigma^A \to \Sigma^B$ and η according to Definition 3.1.1. Let $\Gamma := \operatorname{im}(\xi)$. We now set $\Sigma^{\hat{B}} = (\Sigma^B \setminus \Gamma) \cup \Sigma^A$ and define:

$$\overline{\sigma}^{\hat{B}} = \begin{cases} \overline{\sigma}^B & \sigma \in \Sigma^B \setminus \Gamma \\ \overline{\sigma\xi}^B & \sigma \in \Sigma^A \end{cases}$$

for all $\sigma \in \Sigma^{\hat{B}}$.

The adjustments made to $\mathfrak{C}$ are analogous. Clearly the transition monoids of the two automata remain unchanged by these modifications. By construction, we now have $\mathfrak{A} \leq_G \hat{\mathfrak{B}} \circ \hat{\mathfrak{C}}$ with η unchanged and $\xi' = \mathrm{id}_{\Sigma^A}$. By Lemma 3.1.4 the claim follows. □

We can now proceed to investigate the transition from wreath products of transformation monoids to cascade products of automata:

Lemma 3.1.6. *Let* (Q, S), (X, T) *and* (Y, U) *be transformation monoids such that*

$$(Q, S) \prec (X, T) \wr (Y, U) =: (X \times Y, V)$$

Then for each automaton $\mathfrak{A} \sim_{TS} (Q, S)$ *there exist automata* $\mathfrak{B} \sim_{TS} (Y, U)$ *and* $\mathfrak{C} \sim_{TS} (X, T)$ *such that* $\mathfrak{A} \leq \mathfrak{B} \circ \mathfrak{C}$.

Proof. Choose $Z \subseteq X \times Y$ and $f : Z \to Q$ surjective, as well as $\psi : S \to V$ such that the conditions from Definition 2.2.1 are satisfied. Let $\mathfrak{A} = (Q, \Sigma^A, \delta^A)$ and let $U = \{u_1, \ldots, u_n\}$.

We partition $(S)\psi$ into sets $[(S)\psi]_i$ (some of which may be empty) for $i \in \{1, \ldots, n\}$ where:

$$[(S)\psi]_i = \{(F, u) \in (S)\psi | u = u_i\} \qquad n_i := |[(S)\psi]_i|$$

Now define $[(S)\psi]_U = \{u \in U | \exists (F, u) \in (S)\psi\}$ as the set of all $u \in U$ which occur in the image of ψ. Set $U_\perp := U \setminus [(S)\psi]_U$.

Define $\Sigma_i^B = \{\sigma_{i,1}, \ldots, \sigma_{i,n_i}\}$. Then we set $\Sigma^B = \Sigma_1^B \cup \Sigma_2^B \cup \cdots \cup \Sigma_n^B \cup U_\perp$ where we require that $\Sigma_i^B \cap \Sigma_j^B = \varnothing$ for $i \neq j$ and $\Sigma_i^B \cap U_\perp = \varnothing$ for all $i \in \{1, \ldots, n\}$. We are now ready to define $\mathfrak{B}$. The mappings $\overline{\sigma}^B$ for $\sigma \in \Sigma^B$ are defined as follows:

$$y\overline{\sigma}^B = \begin{cases} yu_i & \sigma \in \Sigma_i^B \\ yu & \sigma = u \in U_\perp \end{cases}$$

We set $\mathfrak{B} = (Y, \Sigma^B, \delta^B)$, which obviously corresponds to (Y, U).

Now let $[(S)\psi]_i = \{(F_{i,1}, u_i), \ldots, (F_{i,n_i}, u_i)\}$ and let

$$[(S)\psi]_{T^Y} = \{F_{i,j} | 1 \leq i \leq n \wedge 1 \leq j \leq n_i\}$$

Set

$$\Sigma_\perp^C = \{t \in T | \forall y \in Y, \forall F \in [(S)\psi]_{T^Y} : (y)F \neq t\}$$

Then define $\Sigma^C = (\Sigma^B \times Y) \dot\cup \Sigma_\perp^C$. For each $(\sigma, y) \in \Sigma^B \times Y$ set

$$x\overline{(\sigma,y)}^C = \begin{cases} x((y)F_{i,j}) & \sigma = \sigma_{i,j} \\ x & \sigma \in U_\perp \end{cases}$$

If $\sigma \in \Sigma_\perp^C$ set $x\overline{\sigma}^C = x\sigma$. We define $\mathfrak{C} = (X, \Sigma^C, \delta^C)$. By construction we have $\mathfrak{C} \sim_{TS} (X, T)$.

We now show that $\mathfrak{A} \leq_G \mathfrak{B} \circ \mathfrak{C}$. Define $Z^{-1} = \{(y,x) | (x,y) \in Z\} \subseteq Y \times X$. Let $\varphi : Z^{-1} \to Q$ be defined by $(y,x) \mapsto (x,y)f$. Notice that by the definition of $\mathfrak{B}$ and $\mathfrak{C}$ for every $(F, f) \in (S)\psi \subseteq V$ there exists a unique $\sigma_{i,j}$ such that $\overline{\sigma}_{i,j}^{A \circ B}$ acts on (y,x) in the same way (F,f) acts on (x,y) for all $x \in X$ and all $y \in Y$. We now define a mapping $g : \Sigma^A \to \Sigma^B$, which maps every $\sigma \in \Sigma^A$ to the unique $\sigma_{i,j} \in \Sigma^B$ determined by $(\overline{\sigma}^A)\psi = (F_{i,j}, u_i)$ for suitable i, j.

We have

$$\begin{aligned}
((y,x)\varphi)\overline{\sigma}^A &= (x,y)f\overline{\sigma}^A \\
&= \big((x,y)(\overline{\sigma}^A\psi)\big)f \\
&= \big((x,y)(F_{i,j},u_i)\big)\varphi && \text{for suitable } i,j \\
&= \big((y,x)(\overline{\sigma}_{i,j}^{B \circ C})\big)\varphi && \text{by construction} \\
&= \big((y,x)\overline{(\sigma g)}^{B \circ C}\big)\varphi
\end{aligned}$$

Now by Lemma 3.1.5 the claim follows. □

We can again extend this result to products with arbitrarily, but finitely many factors:

Corollary 3.1.7. *Let $(Q,S), (X_1,S_1), \ldots, (X_n,S_n)$ be transformation monoids, such that*

$$(Q,S) \prec (X_1,S_1) \wr \cdots \wr (X_n,S_n)$$

Then for every automaton $\mathfrak{A} \sim_{TS} (Q,S)$ there exist automata $\mathfrak{F}_1, \ldots, \mathfrak{F}_n$, where $\mathfrak{F}_i \sim_{TS} (X_i,S_i)$, such that

$$\mathfrak{A} \leq \mathfrak{F}_n \circ \cdots \circ \mathfrak{F}_1$$

Proof. We proceed by induction on n. We have already seen this statement to be true for $n = 2$. Now suppose $n > 2$. Let

$$(Y,T) := (X_1,S_1) \wr \cdots \wr (X_{n-1},S_{n-1})$$

We have $(Q,S) \prec (Y,T) \wr (X_n,S_n)$ and hence there exist automata $\mathfrak{B}$ and $\mathfrak{C}$ such that $\mathfrak{A} \leq \mathfrak{B} \circ \mathfrak{C}$ and $\mathfrak{B} \sim_{TS} (X_n,S_n)$ as well as $\mathfrak{C} \sim_{TS} (Y,T)$. By the induction hypothesis we can find automata $\mathfrak{F}_1, \ldots, \mathfrak{F}_{n-1}$ such that

$$\mathfrak{C} \leq \mathfrak{F}_{n-1} \circ \cdots \circ \mathfrak{F}_1$$

and $\mathfrak{F}_i \sim_{TS} (X_i,S_i)$. Now, by Theorem 2.4.3 we have:

$$\mathfrak{A} \leq \mathfrak{B} \circ \mathfrak{C} \leq \mathfrak{B} \circ (\mathfrak{F}_{n-1} \circ \cdots \circ \mathfrak{F}_1)$$

□

3.2 Wreath and Block Products

The following Theorem is taken from a proof in [Str94] (proof of Lemma A.4.6):

Theorem 3.2.1. *Let (X,S) and (Y,T) be transformation semigroups and let $(X \times Y,U) = (X,S) \wr (Y,T)$ denote their wreath product. If (Y,T) is monogenic, then every subsemigroup of U is isomorphic to a subsemigroup of $S \square T$. In particular, U can be embedded into $S \square T$.*

Proof. Let M be a subsemigroup of U and let y_0 be a generating element of (Y,T). We define a mapping $\psi : M \to S \square T$ by $(F,t)\psi = (\tilde{F},t)$, where $(t_1,t_2)\tilde{F} = (y_0t_1)F$. Then ψ is a homomorphism:

$$\begin{aligned}\big((F_1,t_1)(F_2,t_2)\big)\psi &= (F_1 \cdot t_1.F_2, t_1t_2)\psi \\ &= (\widetilde{F_1 \cdot t_1.F_2}, t_1t_2) \\ &= (\tilde{F}_1.t_2 \cdot t_1.\tilde{F}_2, t_1t_2) \qquad (*) \\ &= (\tilde{F}_1,t_1) \cdot (\tilde{F}_2,t_2) \\ &= (F_1,t_1)\psi \cdot (F_2,t_2)\psi\end{aligned}$$

In $(*)$ the equality $\tilde{F}_1.t_2 = \tilde{F}_1$ was tacitly used (recall that $\tilde{F}_1$ is constant in the second argument). To see that ψ is injective it is sufficient to observe that $\tilde{F}_1 = \tilde{F}_2$ implies $F_1 = F_2$. Indeed if $\tilde{F}_1 = \tilde{F}_2$ then $(y_0t)F_1 = (y_0t)F_2$ for all $t \in T$. Since (Y,T) is monogenic with generating element y_0 we have $F_1 = F_2$. □

It is again easy to deduce that U and $S \square T$ are not isomorphic in general. To get an intuitive idea of this, we simply observe that for the injective homomorphism ψ in the previous proof it was sufficient to use mappings $\tilde{F}$ constant in the second argument. However, we can formalize this intuition by again using a counting argument: If $|T| \geq |Y| \geq 2$, then $|U| = |S|^{|Y|} \cdot |T| < |S|^{|T|^2} \cdot |T| = |S \square T|$.

Remark 3.2.1. In general U and $S \square T$ are not isomorphic.

We will use the convention that iterated block products are evaluated from left to right:

$$S_1 \square S_2 \square \cdots \square S_n = (\cdots(S_1 \square S_2) \square S_3 \cdots) \square S_n \qquad (3.2.1)$$

It should be mentioned at this point that left-to-right bracketing is inherently weaker than right-to-left bracketing. Nevertheless, this convention will be sufficient for our purposes. More rigour on the "strength" of these two bracketing conventions will follow in Section 3.3.

Let $(X_1,S_1),\ldots,(X_n,S_n)$, $n \geq 2$, be monogenic[1] transformation semigroups and let

$$(X_1,S_1) \wr \cdots \wr (X_n,S_n) =: (X,U)$$

[1] It is actually sufficient to only require that (X_i,S_i) is monogenic for $i > 1$.

denote their wreath product. Using Theorem 3.2.1 we can show by induction on n that every subsemigroup of U is isomorphic to a subsemigroup of

$$S_1 \,\square\, S_2 \,\square\, \cdots \,\square\, S_n$$

If $n = 2$ this is exactly the statement of Theorem 3.2.1. Now suppose $n > 2$. By the induction hypothesis every subsemigroup of T given by

$$(X_1^1, S_1) \wr \cdots \wr (X_{n-1}^1, S_{n-1}) =: (Y, T)$$

is isomorphic to a subsemigroup of $S_1 \,\square\, S_2 \,\square\, \cdots \,\square\, S_{n-1} =: M$. In particular T itself is isomorphic to a subsemigroup of M. By Lemma 2.4.7 it follows that $T \,\square\, S_n$ is isomorphic to a subsemigroup of $M \,\square\, S_n$.

Now $(X, U) = (Y, T) \wr (X_n, S_n)$ and by Theorem 3.2.1 we see that every subsemigroup of U is isomorphic to a subsemigroup of $T \,\square\, S_n$. In particular, every subsemigroup of U is isomorphic to a subsemigroup of $M \,\square\, S_n$ and the claim follows for n. We have shown:

Corollary 3.2.2. *Let $(X_1, S_1), \ldots, (X_n, S_n)$, $n \geq 2$, be monogenic transformation semigroups and let*

$$(X_1, S_1) \wr \cdots \wr (X_n, S_n) = (X, U)$$

denote their wreath product. Then every subsemigroup of U is isomorphic to a subsemigroup of

$$S_1 \,\square\, \cdots \,\square\, S_n$$

To complete our investigation of cascade, wreath and block products, we state a consequence of Theorem 3.1.1 and Theorem 3.2.1 about the relationship between the cascade product of automata and the block product of monoids:

Theorem 3.2.3. *Let $\mathfrak{A}$ and $\mathfrak{B}$ be semiautomata. If $\mathfrak{A}$ is monogenic, $M(\mathfrak{A} \circ \mathfrak{B})$ can be embedded into $M(\mathfrak{B}) \,\square\, M(\mathfrak{A})$.*

3.3 Krohn-Rhodes Theory

In this section we will state all three versions of the Krohn-Rhodes theorem –for semiautomata using the cascade product, for transformation semigroups using the wreath product and for monoids using the block product. We will show, using the results of the previous sections, that if one proves the version stated in terms of semiautomata, one can deduce the version for wreath products and even a version for block products, which, however, is slightly weaker than the the version of the theorem usually stated in terms of the block product. Similarly, we will show how one can prove the Krohn-Rhodes Theorem for semiautomata on the basis of the Krohn-Rhodes Theorem for transformation semigroups.

Theorem 3.3.1 (Krohn-Rhodes for Semiautomata)**.** *Let $\mathfrak{A}$ be a semiautomaton. Then there exist semiautomata $\mathfrak{F}_1, \ldots, \mathfrak{F}_n$, such that $\mathfrak{A} \leq \mathfrak{F}_1 \circ \mathfrak{F}_2 \circ \cdots \circ \mathfrak{F}_n$, where each $\mathfrak{F}_i$ is either a reset or a grouplike automaton. Furthermore, if $\mathfrak{F}_i$ is a grouplike automaton, then the group G_i associated with $\mathfrak{F}_i$ is simple and divides $M(\mathfrak{A})$.*

A direct proof of this theorem is given in [Gin68].

Suppose a semiautomaton $\mathfrak{A}$ is a reset. Then $M(\mathfrak{A})$ is a right zero monoid. If $\mathfrak{A}$ has two reset inputs $\sigma_1, \sigma_2 \in \Sigma$, such that $\overline{\sigma}_1 \neq \overline{\sigma}_2$ then $M(\mathfrak{A}) = \{1, \overline{\sigma}_1, \overline{\sigma}_2\}$. Otherwise all reset inputs of $\mathfrak{A}$ act alike and we have that $M(\mathfrak{A}) = \{1, \overline{\sigma}\}$ for some $\sigma \in \Sigma$ or $M(\mathfrak{A}) = \{1\}$ if $\mathfrak{A}$ has no reset inputs. In any event, $M(\mathfrak{A})$ can obviously be embedded into the monoid U_2 (see Section 2.2 to recall the definition of U_2).

Given an arbitrary finite semigroup S we consider the TM (S^I, S^I).

Lemma 3.3.2. *Any nontrivial group G dividing S^I already divides S.*

Proof. Pick a group G which divides S^I but not S, say $G = \mathrm{im}(\varphi)$ for some group $U \subseteq S^I$ and some surjective monoid homomorphism $\varphi : U \to G$. Then $I \in U$ is the identity of U, since, as a group, U contains precisely one idempotent. However for all $s, s' \in S = S^I \setminus \{I\}$ we have $ss' \in S$. Therefore U is trivial and so is G. □

From (S^I, S^I) we can construct a semiautomaton $\mathfrak{A} = (S^I, S^I, \delta^S)$. Observe that (S^I, S^I) is the transformation monoid equivalent to $\mathfrak{A}$. Theorem 3.3.1 now yields a decomposition:

$$\mathfrak{A} \leq \underbrace{\mathfrak{F}_1 \circ \cdots \circ \mathfrak{F}_n}_{=:\mathfrak{B}}$$

where each $\mathfrak{F}_i$ is either a reset or a grouplike automaton, associated with a simple group G_i dividing $M(\mathfrak{A}) = S^I$. However, since simple groups are nontrivial by definition, we have that G_i already divides S.

Let $\mathfrak{F}_i = (Q_i, \Sigma_i, \delta^i)$ and let $\mathfrak{B} = (Q^B, \Sigma, \delta)$. By Corollary 3.1.3 we can see that

$$(Q^B, M(\mathfrak{B})) \prec (Q_n, M(\mathfrak{F}_n)) \wr (Q_{n-1}, M(\mathfrak{F}_{n-1})) \wr \cdots \wr (Q_1, M(\mathfrak{F}_1)) \tag{3.3.1}$$

Observe that every monoid $M(\mathfrak{F}_i)$ is either isomorphic to a simple group dividing S or divides U_2 as was outlined above.

Lemma 3.3.3. *Let $\mathfrak{F}$ be a reset. The equivalent transformation monoid $(Q^F, M(\mathfrak{F}))$ divides (U_2, U_2).*

Proof. Indeed, if $Q^F = \{q_a, q_b\}$ we can define a surjective mapping $f : U \to Q^F$ by $af = q_a$ and $bf = q_b$ where $U = \{a, b\} \subset U_2$. We then define $\psi : M(\mathfrak{F}) \to U_2$ by $1\psi = 1$ and

$$\overline{x}^F \mapsto \begin{cases} a & \text{if } q\overline{x}^F = q_a \text{ for all } q \in Q^F \\ b & \text{otherwise} \end{cases}$$

for all $\overline{x}^F \neq 1$. Then we have $(uf)\overline{x}^F = (u(\overline{x}^F\psi))f$ for all $u \in U$ and all $\overline{x}^F \in M(\mathfrak{F})$. □

Notice that for each grouplike automaton $\mathfrak{F}$ the equivalent transformation monoid is already of the form $(M(\mathfrak{F}), M(\mathfrak{F}))$, since the states of $\mathfrak{F}$ are essentially the elements of the associated group $G_{\mathfrak{F}}$.

We need the following corollary of Lemma 2.4.6:

Corollary 3.3.4. *Let $(X_1, S_1), \ldots, (X_1, S_1)$ be transformation semigroups, such that $(X_j, S_j) \prec (Y, T)$ for some transformation semigroup (Y, T). Then:*

$$\begin{aligned}&(X_1, S_1) \wr \cdots \wr (X_{j-1}, S_{j-1}) \wr (X_j, S_j) \wr (X_{j+1}, S_{j+1}) \wr \cdots (X_n, S_n)\\ \prec\ &(X_1, S_1) \wr \cdots \wr (X_{j-1}, S_{j-1}) \wr (Y, T) \wr (X_{j+1}, S_{j+1}) \wr \cdots (X_n, S_n)\end{aligned}$$

Using this we can now substitute the factors $(Q_i, M(\mathfrak{F}_i))$ in (3.3.1) appropriately by $(M(\mathfrak{F}_i), M(\mathfrak{F}_i))$ or (U_2, U_2). Since $\mathfrak{A} \leq \mathfrak{B}$ we get $(S^I, S^I) \prec (Q^B, M(\mathfrak{B}))$. Now obviously $(S^1, S) \prec (S^I, S^I)$. Altogether we obtain the second version of the Krohn-Rhodes theorem:

Theorem 3.3.5 (Krohn-Rhodes for Transformation Semigroups)**.** *Let S be any finite semigroup. Then there exist semigroups $X_1, \ldots, X_n$ such that*

$$(S^1, S) \prec (X_1^1, X_1) \wr \cdots \wr (X_n^1, X_n)$$

Furthermore, for all $i \in \{1, \ldots, n\}$ we have $X_i = U_2$ or $X_i = G_i$ for some simple group G_i dividing S.

We now proceed to prove the third version of the Krohn-Rhodes theorem, a version using block products. Given a semigroup S we can find semigroups $X_1, \ldots, X_n$ such that each X_i is either a simple group dividing S or $X_i = U_2$ and

$$(S^1, S) \prec (X_1^1, X_1) \wr \cdots \wr (X_n^1, X_n) =: (X, U)$$

We need to show that (X_i^1, X_i) is monogenic for every $i \in \{1, \ldots, n\}$. If X_i is a group, then this is immediately clear. If $X_i = U_2$, we see that 1 is a generating element of the transformation monoid. Corollary 3.2.2 yields that every subsemigroup of U is isomorphic to a subsemigroup of

$$X_1 \boxdot X_2 \boxdot \cdots \boxdot X_n =: X$$

In particular $U \prec X$ and because of Lemma 2.2.3 $S \prec U$. We immediately get a first, weak version of the Krohn-Rhodes Theorem for semigroups:

Theorem 3.3.6 (Krohn-Rhodes for Semigroups, Weak Version)**.** *Let S be any finite semigroup. Then there exist semigroups $X_1, \ldots, X_n$, where either $X_i = U_2$ or $X_i = G_i$ for some simple group G_i dividing S, such that:*

$$S \prec X_1 \boxdot \cdots \boxdot X_n$$

The reason we called the last theorem a "weak" version of the Krohn-Rhodes theorem is the nature of the semigroups used in the decomposition. Recall that one of the building blocks of the decomposition in Theorem 3.3.5 was the monoid U_2. However, the block product is more powerful, as the following lemma shows.

Lemma 3.3.7. *$U_2 \hookrightarrow U_1 \boxdot U_1$. In particular $U_2 \prec U_1 \boxdot U_1$.*

Proof. Recall that $U_2 = \{1, a, b\}$. We define $F_a : U_1 \times U_1 \to U_1$ by

$$(u_1, u_2)F_a = \begin{cases} 1 & \text{if } u_2 = 0 \\ 0 & \text{otherwise} \end{cases}$$

Note that F_a is constant in the first argument. Let $F_b \equiv 1$ be the function with constant value 1. Then the neutral element of $U_1 \,\square\, U_1$ is $(F_b, 1)$. We now define a mapping $\psi : U_2 \to U_1 \,\square\, U_1$ by $1\psi = (F_b, 1)$, $b\psi = (F_b, 0)$ and $a\psi = (F_a, 0)$. Then we have for all $u_1, u_2 \in U_2$ such that $u_2 \neq 1$:

$$\begin{aligned}(u_1 \cdot u_2)\psi = u_2\psi &= (F_{u_2}, 0) \\ &= (\underbrace{F_{u_1}.0}_{\equiv 1} \cdot \underbrace{0.F_{u_2}}_{=F_{u_2}}, 0) \\ &= (F_{u_1}, 0) \cdot (F_{u_2}, 0) = u_1\psi \cdot u_2\psi\end{aligned}$$

Hence suppose $u_2 = 1$. If $u_1 = u_2 = 1$ there is nothing to show. Therefore suppose $u_1 \neq 1$. Then

$$\begin{aligned}(u_1 \cdot u_2)\psi = u_1\psi &= (F_{u_1}, 0) \\ &= (\underbrace{F_{u_1}.1}_{=F_{u_1}} \cdot \underbrace{0.F_b}_{\equiv 1}, 0) \\ &= (F_{u_1}, 0) \cdot (F_b, 1) = u_1\psi \cdot u_2\psi\end{aligned}$$

We see that ψ is a homomorphism. Since ψ is injective by definition, the claim follows. □

Recall that Corollary 3.3.4 allowed us to substitute single factors in a decomposition of transformation semigroups. Since the wreath product is associative we could even have substituted by products of transformation semigroups. However, the block product is not associative in general. We are not able to simply substitute single factors in a decomposition by block products of simpler factors, since that would violate the implicit bracketing (recall our convention stated in equation (3.2.1)). In particular, we cannot replace occurrences of U_2 by $U_1 \,\square\, U_1$ without proving that this can be done in a manner consistent with (3.2.1).

In fact this is altogether impossible, if we wish to stick to the bracketing convention from (3.2.1). This is shown in [ST02] by Straubing and Thérien. More precisely, the statement from [ST02] is:

Theorem 3.3.8 (Straubing, Thérien, [ST02])**.** *Every monoid $M = U_1 \,\square\, \cdots \,\square\, U_1$ (with the bracketing convention from (3.2.1)) is in the variety* **DA**.

As we do not want to go into detail about **DA**, it suffices to say that it is a proper subvariety of **A**, the variety of all aperiodic monoids.

We can introduce a new bracketing convention, which will be right-to-left (the notation is adopted from [Str94]):

$$\square(S_1, \ldots, S_n) := S_1 \,\square\, (S_2 \,\square\, (\cdots(S_{n-1} \,\square\, S_n)\cdots))$$

Then we can in fact state:

Theorem 3.3.9 (Krohn-Rhodes for Semigroups, see [Str94])**.** *Let S be any finite semigroup. Then there exist semigroups $X_1, \ldots, X_n$, where either $X_i = U_1$ or $X_i = G_i$ for some simple group G_i dividing S, such that:*

$$S \prec \square(X_1, \ldots, X_n)$$

It is interesting to note at this point that, although the left-to-right bracketing of block products is intrinsically weaker than the right-to-left bracketing, a Krohn-Rhodes decomposition theorem is still possible, if we allow the more complex U_2 factors. We can conclude:

Proposition 3.3.10. *The expressiveness of iterated block products of simple groups and U_2 factors is independent of left to right or right to left bracketing.*

We now proceed to show that Theorem 3.3.5 implies Theorem 3.3.1. We start out with an automaton $\mathfrak{A} = (Q, \Sigma, \delta)$. Let $M := M(\mathfrak{A})$ denote the transition monoid of $\mathfrak{A}$. We consider the equivalent TM (Q, M). We evidently have $(Q, M) \prec \overline{(Q, M)}$. Now let $\overline{(Q, M)} = (Q, \overline{M})$.

Lemma 3.3.11. $(Q, \overline{M}) \prec (\overline{M}, \overline{M})$

Proof. Let $M_0 \subseteq \overline{M}$ be the subset of all constant mappings in $\overline{M}$. Denote by $g_q \in M_0$ the mapping with constant value q. Then define $f : M_0 \to Q$, $(g_q)f = q$. Clearly f is surjective. Define $\psi = \mathrm{id}$. Then $(g_q f)m = (q)m = (g_q m)f$ for all $q \in Q$. □

Notice that every nontrivial group dividing $\overline{M}$ already divides M.

By Theorem 3.3.5 we can cover $(\overline{M}, \overline{M})$ by a product $(X_1^1, X_1) \wr \cdots \wr (X_n^1, X_n)$. Now by Corollary 3.1.7 we can find automata $\mathfrak{F}_1, \ldots, \mathfrak{F}_n$, $\mathfrak{F}_i \sim_{TS} (X_i^1, X_i)$ for all $i \in \{1, \ldots, n\}$, such that

$$\mathfrak{A} \leq \mathfrak{F}_n \circ \cdots \circ \mathfrak{F}_1$$

Recall that all X_i are either a simple group dividing $\overline{M}$ (and therefore M) or are of the form $U_2 = \{1, a, b\}$. Obviously, if X_i is a group then $\mathfrak{F}_i$ is a grouplike automaton. However, if $X_i = U_2$, we do not immediately get a 2-RA. In that case $\mathfrak{F}_i$ will be a 3-RA, since $|U_2| = 3$. Nevertheless, since any n-RA automaton can be covered by a direct product of 2-RAs automata by Lemma 2.4.1, we obtain Theorem 3.3.1.

4 Cascade Products and their Languages

4.1 Biased Resets

Let $\mathfrak{R} = (\mathbb{B}, \Sigma, \delta)$ be a reset. Denote by Σ_0 the set of all inputs σ, such that $\overline{\sigma} \equiv 0$, i.e. $\overline{\sigma}$ is the constant 0 mapping. Similarly let Σ_1 denote the set of all inputs, such that $\overline{\sigma} \equiv 1$. If $\Sigma_i = \varnothing$ we say $\mathfrak{R}$ is $(1-i)$-*biased* (see Figure 4.1)[1].

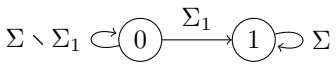


Figure 4.1: A 1-biased reset.

In the following two sections we will show how biased resets can be used to characterize piecewise testable languages and $\mathcal{R}$-trivial languages (i.e. languages L for which $M(L)$ is $\mathcal{R}$-trivial). We turn to piecewise testable languages first.

4.1.1 Piecewise Testable Languages

We first show that biased resets are sufficient to recognize all piecewise testable languages:

Lemma 4.1.1. *Let $L = \Sigma^*\sigma_1\Sigma^*\cdots\Sigma^*\sigma_m\Sigma^*$ for some alphabet Σ and for $\sigma_1, \ldots, \sigma_m \in \Sigma$. Then L is recognized by a cascade product of m 1-biased resets $\mathfrak{R}_1, \ldots, \mathfrak{R}_m$.*

Proof. The proof is by induction on m. If $m = 1$ then $L = \Sigma^*\sigma_1\Sigma^*$ and we have that $\mathfrak{R} = (\mathbb{B}, \Sigma, \delta)$ defined by

$$\overline{\sigma} = \begin{cases} 1 & \sigma = \sigma_1 \\ \mathrm{id} & \text{otherwise} \end{cases}$$

recognizes L. Indeed, let $\mathfrak{B} = (\mathfrak{R}, 0, \{1\})$. Then $w = a_1\cdots a_n \in L(\mathfrak{B})$ iff the run ρ of $\mathfrak{B}$ on w is accepting iff there exists an index $i > 0$ such that $\rho_i = 1$ iff $a_i = \sigma_1$ iff $w \in L$.

Now let $m > 1$ and suppose the statement has already been shown for $m - 1$. Let $L = \Sigma^*\sigma_1\Sigma^*\sigma_2\Sigma^*\cdots\Sigma^*\sigma_m\Sigma^*$. Then by the induction hypothesis we can find 1-biased

[1] These resets were also considered by, e.g., Brzozowski and Fich [BF80]. There they were called *half resets*.

resets $\mathfrak{R}_1, \ldots, \mathfrak{R}_{m-1}$, such that $L' = \Sigma^*\sigma_1\Sigma^*\sigma_2\Sigma^*\cdots\Sigma^*\sigma_{m-1}\Sigma^*$ is recognized by

$$\mathfrak{A} = (Q, \Sigma, \delta) = \mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_{m-1}$$

say by $L' = L(\mathfrak{A}, q_0, F)$. Now let $\mathfrak{R}_m = (\mathbb{B}, \Sigma', \delta^{R_m})$ with $\Sigma' \supseteq \Sigma \times Q$. be defined by

$$\overline{(\sigma, q)}^{R_m} = \begin{cases} 1 & \sigma = \sigma_m \wedge q \in F \\ \mathrm{id} & \text{otherwise} \end{cases}$$

Let $\mathfrak{B} = \mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_m$ and set $F' = \{(q, i) \in Q \times \mathbb{B} | \, i = 1\}$ as well as $q_0' = (q_0, 0)$. Define $\mathfrak{C} = (\mathfrak{B}, q_0', F')$. We claim that $L = L(\mathfrak{C})$. Indeed, let $w = a_1 \cdots a_n \in L(\mathfrak{C})$. Let ρ be the accepting run of $\mathfrak{C}$ on w. Then $\rho_n = (q, 1)$ for some $q \in Q$. By the definition of q_0' and $\mathfrak{R}$ we see that this implies that there exists an index $i < n$ such that $\rho_i = (p, 0)$ for some $p \in F$. Because $(\mathfrak{A}, q_0, F)$ an automaton recognizing L' we see that then $\rho_j = (q_j, x_j)$ with $q_j \in F$ for every $j > i$ (since $u \in L'$ implies $u \cdot v \in L'$ for all $v \in \Sigma^*$). Hence we see that $\mathfrak{C}$ accepts w iff $a_1 \cdots a_i \in L'$ and $a_{i+1} \cdots a_n \in \Sigma^*\sigma_m\Sigma^*$ for some i. Thus $L(\mathfrak{C}) = L' \cdot \Sigma^*\sigma_m\Sigma^* = L$. □

As an immediate corollary we obtain:

Corollary 4.1.2. *Every piecewise testable language is recognized by a direct product of cascade products of 1-biased resets.*

Since direct products are special cases of cascade products we conclude:

Corollary 4.1.3. *Every piecewise testable language is recognized by a cascade product of 1-biased resets.*

Hence biased resets are expressive enough to recognize all piecewise testable languages. However, they are not restrictive enough to ensure that all languages recognized are in turn piecewise testable. Let $\Sigma = \{a, b\}$. If we define $\mathfrak{R}_1 = (\mathbb{B}, \Sigma, \delta^{R_1})$ by

$$\overline{\sigma}^{R_1}(q) = \begin{cases} 1 & \sigma = a \\ q & \text{otherwise} \end{cases}$$

and $\mathfrak{R}_2 = (\mathbb{B}, \Sigma \times \mathbb{B}, \delta^{R_2})$ by

$$\overline{(\sigma, x)}^{R_2}(q) = \begin{cases} 1 & \sigma = b \wedge x = 0 \\ q & \text{otherwise} \end{cases}$$

then $\mathfrak{R}_1$ and $\mathfrak{R}_2$ are 1-biased resets. Nevertheless, $L(\mathfrak{R}_1 \circ \mathfrak{R}_2, (0,0), \{(0,1)\}) = \overline{\Sigma^*a\Sigma^*}b\Sigma^*$ is not piecewise testable (see [CB71]).

We will now investigate what restrictions are necessary to ensure that every language recognized by a cascade product of biased resets is piecewise testable. If we look at the example of the previous paragraph, we notice that our problem is the alternation in the "orientation" of the two resets: One checked for the occurrence of a pattern and the other one checked for absence of a pattern. We therefore have to ensure, that this kind of alternation is avoided. First we need to understand better when these effects occur. To this end we will consider the languages recognized by a cascade product of an arbitrary automaton $\mathfrak{A}$ and a biased reset $\mathfrak{R}$:

Lemma 4.1.4. *Let $\mathfrak{A} = (Q, \Sigma, \delta^A)$ be a semiautomaton and let $\mathfrak{R} = (\mathbb{B}, \Sigma \times Q, \delta^R)$ be a 1-biased reset. Then every language recognized by $\mathfrak{A} \circ \mathfrak{R}$ is a finite union of languages of one of the following three forms:*

(1)

$$L \in \mathcal{L}(\mathfrak{A})$$

(2)

$$L = \bigcup_{\substack{(\sigma,q)\in\Sigma\times Q \\ \overline{(\sigma,q)}^R \equiv 1}} L(\mathfrak{A}_{p,q}) \cdot \sigma \cdot L(\mathfrak{A}_{q\overline{\sigma}^A,r})$$

(3)

$$L = \left(\overline{\bigcup_{\substack{(\sigma,q)\in\Sigma\times Q \\ \overline{(\sigma,q)}^R \equiv 1}} L(\mathfrak{A}_{p,q}) \cdot \sigma \cdot \Sigma^*} \right) \cap L(\mathfrak{A}_{p,r})$$

for some $p, r \in Q$.

Proof. Consider the automaton $\mathfrak{B} = (\mathfrak{A} \circ \mathfrak{R}, q_0, F)$, where $q_0 \in Q \times \mathbb{B}$ and $F \subseteq Q \times \mathbb{B}$. Let $q_0 = (p, i_0)$ for some $p \in Q$ and $i_0 \in \mathbb{B}$. Let furthermore $F = \{(r_1, i_1), \ldots, (r_n, i_n)\}$. We evidently have $L(\mathfrak{B}) = \bigcup_{j=1}^n L(\mathfrak{B}_{q_0,(r_j,i_j)})$. It therefore suffices to show that the languages $L(\mathfrak{B}_{q_0,(r_j,i_j)})$ are of the desired form.

If $i_0 = 1$ then we may assume that $i_1, \ldots, i_n = 1$, since if $i_j = 0$ we have $L(\mathfrak{B}_{q_0,(r_j,i_j)}) = \varnothing$ in that case. However, since $\mathfrak{R}$ is 1-biased, the run of $\mathfrak{B}$ is accepting iff the run of $\mathfrak{A}_{p,r_i}$ is. Hence we see that in this case the language is already recognizable by $\mathfrak{A}$.

Now assume that $i_0 = 0$. We consider an accepting state $(r, 1)$. Let $L = L(\mathfrak{B}_{q_0,(r,1)})$ and let $w = \sigma_1 \cdots \sigma_m \in \Sigma^*$. Consider the run ρ of $\mathfrak{A}_{p,r}$ on w. We have $w \in L$ iff $w \in L(\mathfrak{A}_{p,r})$ and there exists $k \in \{1, \ldots, m\}$ such that $\overline{(\sigma_k, \rho_{k-1})}^R \equiv 1$. Hence L is of form (2) above.

Suppose we have an accepting state $(r, 0)$. Let w and ρ be defined as before. Then $w \in L$ iff $w \in L(\mathfrak{A}_{p,r})$ and there exists no $k \in \{1, \ldots, m\}$, such that $\overline{(\sigma_k, \rho_{k-1})}^R \equiv 1$. This is evidently case (3). □

Suppose now that $\mathfrak{R}$ is a 1-biased reset. Then $M(\mathfrak{R}) = U_1$, which is $\mathcal{J}$-trivial. Hence $\mathfrak{R}$ recognizes only piecewise testable languages by Simon's Theorem (Theorem 2.3.13). Indeed:

Lemma 4.1.5. *Let $\mathfrak{R} = (\mathbb{B}, \Sigma, \delta)$ be a 1-biased reset. Then*

$$\mathcal{L}(\mathfrak{R}) = \{\varnothing, \Sigma^*, \bigcup_{\sigma\in\Sigma_1} \Sigma^*\sigma\Sigma^*, \bigcap_{\sigma\in\Sigma_1} \overline{\Sigma^*\sigma\Sigma^*}\}$$

Proof. We consider $\mathfrak{R}_{p,q}$. Let $L = L(\mathfrak{R}_{p,q})$. If $p = q = 1$, then evidently $L = \Sigma^*$. Furthermore, if $p = 1$ and $q = 0$, then $L = \varnothing$. Hence we suppose $p = 0$.

If $q = 0$, then we have $w = \sigma_1 \cdots \sigma_n \in L$ iff there exists no index i, such that $\sigma_i \in \Sigma_1$. Hence $L = \bigcap_{\sigma\in\Sigma_1} \overline{\Sigma^*\sigma\Sigma^*}$. Conversely, if $q = 1$, then $w \in L$ iff there exists i, such that $\sigma_i \in \Sigma_1$. Therefore $L = \bigcup_{\sigma\in\Sigma_1} \Sigma^*\sigma\Sigma^*$. □

From the previous proof we see that $L(\mathfrak{R}, q_0, F)$ is a finite union languages of the form $\Sigma^*\sigma\Sigma^*$ iff $q_0 = 0$ and $F = \{1\}$. We proceed to characterize piecewise testable languages:

Definition 4.1.1. Let $\mathfrak{R}_1, \ldots, \mathfrak{R}_n$ be 1-biased resets, each with state set $\mathbb{B}$. We call the cascade product $\mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_n$ *strictly locally 1-triggered*, if we have

(a) for all $(x_1, \ldots, x_{i-1}) \in \mathbb{B}^{i-1}$, $i = 2, \ldots, n$:

$$\overline{(\sigma, (x_1, \ldots, x_{i-1}))}^{R_i} = \mathrm{id} \qquad \text{if } x_{i-1} \neq 1$$

(b) and for all $(x_1, \ldots, x_{i-2}, 1), (y_1, \ldots, y_{i-2}, 1) \in \mathbb{B}^{i-1}$, $i = 2, \ldots, n$:

$$\overline{(\sigma, (x_1, \ldots, x_{i-2}, 1))}^{R_i} = \overline{(\sigma, (y_1, \ldots, y_{i-2}, 1))}^{R_i}$$

for all $\sigma \in \Sigma$. It is said to be *locally 1-triggered*, if it is a direct product of strictly locally 1-triggered cascade products of 1-biased resets.

Inspecting the proof of Lemma 4.1.1 we see that the cascade product recognizing $\Sigma^*\sigma_1\Sigma^*\cdots\Sigma^*\sigma_n\Sigma^*$ may be constructed such that it is strictly locally 1-triggered. Furthermore every direct product of locally 1-triggered cascade products of resets is again a locally 1-triggered cascade product of resets. Thus every piecewise testable language is recognized by a locally 1-triggered cascade product of 1-biased resets.

We now turn to the converse:

Lemma 4.1.6. *If $\mathfrak{A} := \mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_n$ is a strictly locally 1-triggered cascade product of 1-biased resets $\mathfrak{R}_i = (\mathbb{B}, \Sigma^i, \delta^{R_i})$, $i = 1 \ldots, n$, then every language recognized by $\mathfrak{A}$ with accepting states $\{(x_1, \ldots, x_n) | x_n = 1\} \subseteq \mathbb{B}^n$ is a finite union of languages of the form $\Sigma^*\sigma_1\Sigma^*\cdots\Sigma^*\sigma_m\Sigma^*$, where $\Sigma = \Sigma^1$.*

Proof. We prove the claim by induction over the length n (i.e. the number of factors) of the cascade product. If $n = 1$ then by Lemma 4.1.5 and the corresponding proof there is nothing to show. Hence we may suppose that $n > 1$ and that the claim has already been shown for all $m < n$. Denote $\mathfrak{B} = \mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_{n-1}$. By the induction hypothesis all languages from $\mathcal{L}(\mathfrak{B})$, which are accepted with a set $F' = \{(x_1, \ldots, x_{n-1}) | x_{n-1} = 1\}$, are of the desired form. Let $L = L(\mathfrak{A}, q_0, F)$, where $F = \{(x_1, \ldots, x_n) | x_n = 1\}$ and q_0 is arbitrary.

If $q_0 = (q_{0,1}, \ldots, q_{0,n})$ contains a non-zero entry, say $q_{0,i}$, then the state of the resets $\mathfrak{R}_1, \ldots \mathfrak{R}_i$ will have no effect on the behavior of the resets $\mathfrak{R}_{i+1}, \ldots, \mathfrak{R}_n$, since the $\mathfrak{R}_i$ are all 1-biased and the product is strictly locally 1-triggered. Because of the definition of F they will therefore have no effect on the acceptance behavior of the automaton. Thus we may assume $q_0 = (0, \ldots, 0)$.

Then by Lemma 4.1.4 and its proof we have that L is a finite union of languages $L(\mathfrak{B}, (0, \ldots, 0), F') \cdot \sigma \cdot L(\mathfrak{B}, (q_1, \ldots, q_{n-1}, 1), F')$. However, $L(\mathfrak{B}, (q_1, \ldots, q_{n-1}, 1), F') = \Sigma^*$ and $L(\mathfrak{B}, (0, \ldots, 0), F')$ is a finite union of languages of the appropriate form. Then, by the distributivity of the concatenation with respect to the union of languages, the claim follows. □

We can now prove the main theorem of this section[2]:

Theorem 4.1.7. *L is piecewise testable iff L can be recognized by a locally 1-triggered cascade product of 1-biased resets.*

Proof. We have already seen that every piecewise testable language can be recognized by a locally 1-triggered cascade product of 1-biased resets. Hence let $\mathfrak{A}$ be a locally 1-triggered cascade product of 1-biased resets. Then $\mathfrak{A}$ is a direct product of strictly locally 1-triggered cascade products of 1-biased resets. Therefore all languages from $\mathcal{L}(\mathfrak{A})$ are Boolean combinations of the languages recognized by strictly locally 1-triggered products. Since the piecewise testable languages form a Boolean algebra, it is sufficient to show that all strictly locally 1-triggered cascade products of 1-biased resets recognize only piecewise testable languages.

Hence assume w.l.o.g. that $\mathfrak{A} := \mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_n$ is strictly locally 1-triggered. Let $q_0 \in \mathbb{B}^n$ and let $F = \{(f_{1,1}, \ldots, f_{1,n}), \ldots, (f_{r,1}, \ldots, f_{r,n})\}$. Since

$$L(\mathfrak{A}, q_0, F) = \bigcup_{i=1}^{r} L(\mathfrak{A}, q_0, \{(f_{i,1}, \ldots, f_{i,n})\})$$

we may assume that $r = 1$ and $F = \{(f_1, \ldots, f_n)\}$. Suppose $q_0 = (q_{0,1}, \ldots, q_{0,n})$ and let $q_{0,i} = 1$. Then, because $\mathfrak{A}$ is strictly locally 1-triggered, $L = L(\mathfrak{A}, q_0, F)$ is the intersection of the languages

$$J = L(\mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_i, (q_{0,1}, \ldots, q_{0,i}), \{(f_1, \ldots, f_i)\})$$
$$K = L(\tilde{\mathfrak{R}}_{i+1} \circ \mathfrak{R}_{i+2} \circ \cdots \circ \mathfrak{R}_n, (q_{0,i+1}, \ldots, q_{0,n}), \{(f_{i+1}, \ldots, f_n)\})$$

where $\tilde{\mathfrak{R}}_{i+1}$ is obtained from $\mathfrak{R}_{i+1}$ by treating all inputs $\sigma \in \Sigma$ as $(\sigma, (0, \ldots, 0, 1))$. Since both resulting products are again strictly locally 1-triggered, we may assume that $q_0 = (0, \ldots, 0)$.

Now assume that $f_n = 1$. Then, since $q_0 = (0, \ldots 0)$ and since $\mathfrak{A}$ is strictly locally 1-triggered and since all resets are 1-biased, we have $f_1 = \ldots = f_n = 1$ or the language $L = L(\mathfrak{A}, q_0, \{(f_1, \ldots, f_n)\})$ is empty. Hence $L(\mathfrak{A}, q_0, \{(x_1, \ldots, x_n) | x_n = 1\}) = L$, since only $(1, \ldots, 1)$ is reachable from q_0. By Lemma 4.1.6 we see that L is a finite union of languages of the form $\Sigma^* \sigma_1 \Sigma^* \cdots \Sigma^* \sigma_r \Sigma^*$.

If $f_n = 0$ we pick $i \in \{1, \ldots, n-1\}$ maximal (if it exists) with $f_i = 1$. If no such i exists, then clearly $L = \overline{L(\mathfrak{R}_1, 0, \{1\})}$, which is piecewise testable. Hence we assume such an index i exists. We have:

$$L = L(\mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_i \circ \mathfrak{R}_{i+1} \circ \cdots \circ \mathfrak{R}_n, q_0, \{(f_1, \ldots, f_n)\})$$

[2] Straubing obtained a very similar result in terms of semigroups (see [Str80]). He uses matrices over the commutative semiring $\mathbb{B}$. A related investigation is found in [ST88], where partially ordered semigroups are considered. In the second paper the results are combined to obtain a decomposition of $\mathcal{J}$-trivial monoids in terms of restricted semidirect products. This decomposition is indeed very close the one we give. Our proof is purely automaton theoretic, whereas the proof from [ST88] is purely algebraic. Both references were not mentioned in the non-revised version, since the corresponding work was brought to the author's attention only after submitting his thesis (see also page iii).

Since $\mathfrak{A}$ is strictly 1-triggered we see that $f_1 = \cdots = f_i = 1$, since otherwise $(f_1, \ldots, f_n)$ is again unreachable from q_0. Furthermore we must have $f_{i+2} = \cdots = f_n = 0$ for the same reason. This however means, that $\mathfrak{R}_{i+2}, \ldots, \mathfrak{R}_n$ are irrelevant to the acceptance behavior of $(\mathfrak{A}, q_0, (f_1, \ldots, f_n))$. Thus we may assume that $i = n - 1$.

We may use the results from the case $f_n = 1$ to see that

$$K := L(\mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_i, (0, \ldots, 0), \{(1, \ldots, 1)\})$$

is a finite union of languages of the form $\Sigma^*\sigma_1\Sigma^*\cdots\Sigma^*\sigma_r\Sigma^*$. Now by Lemma 4.1.4 we get that L is a finite union of languages of the form $\overline{K\sigma\Sigma^*} \cap K$. Since the piecewise testable languages are closed under the Boolean operations the claim follows. □

4.1.2 $\mathcal{R}$-trivial Languages

The main question we investigate in this section is: What class of languages is recognized by biased resets? We will show that this turns out to be the class of $\mathcal{R}$-trivial languages, i.e. languages L such that $M(L)$ is $\mathcal{R}$-trivial[3]. We first need a suitable automaton model (adopted from [Pin86]):

Definition 4.1.2. Let $\mathfrak{A} = (Q, \Sigma, \delta)$ be a semiautomaton. $\mathfrak{A}$ is called *extensive*, if there exists a total ordering $\leq$ on Q, which is compatible with the transition structure of $\mathfrak{A}$ in the following way: $q \leq q\overline{\sigma}$ for all $q \in Q$, $\sigma \in \Sigma$.

In [Pin86] it is shown that a language L is $\mathcal{R}$-trivial iff $\mathfrak{A}_L$ is extensive. We will add another equivalence (in terms of the cascade product) to this characterization. As a first step, we state the following lemma:

Lemma 4.1.8. *Let M be an $\mathcal{R}$-trivial monoid. Then the monoid S defined by $(M \times U_1, S) := (M, M) \wr (U_1, U_1)$ is $\mathcal{R}$-trivial.*

Proof. Consider $s_1, s_2 \in S$, such that $s_1 \mathcal{R} s_2$. Let $s_1 = (F_1, u_1)$ and $s_2 = (F_2, u_2)$. By assumption there exist (G_1, t_1) and (G_2, t_2), such that

$$(F_2, u_2) = (F_1, u_1) \cdot (G_1, t_1) = (F_1 \cdot u_1.G_1, u_1 t_1) \qquad (*)$$
$$(F_1, u_1) = (F_2, u_2) \cdot (G_2, t_2) = (F_2 \cdot u_2.G_2, u_2 t_2) \qquad (**)$$

Since U_1 is $\mathcal{R}$-trivial it is evident that $u_1 = u_2$. Pick an arbitrary $u \in U_1$. Consider the operation on $(1, u) \in M \times U_1$:

$$\begin{aligned}
& (1,u).(F_1,u_1) = (1,u).(F_2 \cdot u_2.G_2, u_2t_2) \\
\Leftrightarrow\ & (1,u).(F_1,u_1) = (1,u).(F_2 \cdot u_1.G_2, u_1) \\
\Leftrightarrow\ & (1 \cdot (uF_1), uu_1)) = (1 \cdot (uF_2) \cdot (uu_1G_2), uu_1) \\
\Leftrightarrow\ & ((uF_1), uu_1)) = ((uF_2) \cdot (uu_1G_2), uu_1)
\end{aligned}$$

[3] This result was first obtained by Brzozowski and Fich in [BF80]. In the non-revised version of this thesis this fact was not mentioned since the author was not aware of it (see also page iii).

We see that there exists $r = uu_1G_2 \in M$, such that $uF_1 = (uF_2) \cdot r$. Using $(*)$ instead of $(**)$ we can show completely analogously that there exists $r' \in M$, such that $uF_2 = (uF_1) \cdot r'$. Hence $(uF_1)\mathcal{R}(uF_2)$ and by the $\mathcal{R}$-triviality of M $uF_1 = uF_2$. Since u was arbitrary we have $F_1 = F_2$ and S is $\mathcal{R}$-trivial. □

We say M is an *iterated wreath product of U_1 factors*, if M is the monoid of the TM $(U_1, U_1) \wr \cdots \wr (U_1, U_1)$. We deduce:

Corollary 4.1.9. *Every iterated wreath product of U_1 factors is $\mathcal{R}$-trivial.*

We are now ready to state the following theorem:

Theorem 4.1.10. *Let $L \subseteq \Sigma^*$ be regular. The following are equivalent:*

(a) $M(L)$ is $\mathcal{R}$-trivial.

(b) $\mathfrak{A}_L$ is extensive.

(c) L is recognizable by a cascade product of biased resets.[4]

Proof. (a) ⇒ (b): This is shown in [Pin86].
(b) ⇒ (c): We show $\mathfrak{A}_L \leq \mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_m$, for biased resets $\mathfrak{R}_i$. The claim then follows from Proposition 2.3.10.

Let $\mathfrak{A}_L = (Q, \Sigma, \delta)$ and let $n = |Q|$. Order the elements of Q according to $\leq$, say $q_1, \ldots, q_n$. Denote by Σ_i^j the inputs, which map to state q_i when the automaton is in state q_j for $1 \leq j \leq i \leq n$. Set $\Sigma_1^0 := \Sigma$. For a bit vector $x = (x_1, \ldots, x_k) \in \mathbb{B}^k$ we define $\max(x) := \max\{i | x_i = 1\}$ to be the index of the maximal component of x, which is 1, if at least one such component exists. If $x_1 = x_2 = \cdots = x_k = 0$, we set $\max(x) = 0$. We define biased resets $\mathfrak{R}_i = (\mathbb{B}, \Sigma \times \mathbb{B}^{i-1}, \delta^{R_i})$ for $i = 1, \ldots, n$ in the following way:

$$\overline{(\sigma, x)}^{R_i} = \begin{cases} 1 & \max(x) = j \text{ and } \sigma \in \Sigma_i^j \\ \mathrm{id} & \text{otherwise} \end{cases}$$

Notice that in the special case of $i = 1$, we will always be in the first case.

Now set

$$\mathfrak{B} := \mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_n$$

Denote the mappings of $\mathfrak{B}$ by $\{\overline{\sigma}^B | \sigma \in \Sigma\}$. Let $P_0 := \mathbb{B}^n \setminus \{(0, \ldots, 0)\}$. Clearly the states of P_0 are closed under the transitions of $\mathfrak{B}$. Hence (P_0, Σ, δ^B) is a subsemiautomaton of $\mathfrak{B}$. Define $\varphi : P_0 \to Q$ by $x\varphi = q_{\max(x)}$. For all $\sigma \in \Sigma$ we have for x with $\max(x) = j$

$$\begin{aligned} (x\overline{\sigma}^B)\varphi &= y\varphi && \text{for some } y \text{ with } \max(y) = i \text{ iff } \sigma \in \Sigma_i^j \\ &= q_i \\ &= q_j\overline{\sigma} \\ &= (x\varphi)\overline{\sigma} \end{aligned}$$

[4] Statement (c) is originally due to Brzozowski and Fich (see [BF80]).

Hence φ is a homomorphism of semiautomata. Since φ is obviously surjective we see that $\mathfrak{A}_L \leq \mathfrak{B}$.
(c) $\Rightarrow$ (a): Pick a cascade product of biased resets $\mathfrak{B}$, such that $\mathfrak{B}$ recognizes L. Then by Proposition 2.3.10 we have $\mathfrak{A}_L \leq \mathfrak{B}$ and by Corollary 3.1.3 we see that

$$(Q^B, M(\mathfrak{B})) \prec (\mathbb{B}, U_1) \wr \cdots \wr (\mathbb{B}, U_1)$$

Now by Proposition 2.3.4 and Lemmas 2.2.3 and 4.1.8 the claim follows (every monoid, which divides an $\mathcal{R}$-trivial monoid, is itself $\mathcal{R}$-trivial, since $\mathbf{R}$ is a variety; see Section 2.3). □

Since the set $\mathbf{R}$ of all $\mathcal{R}$-trivial monoids forms a variety, we obtain the following Corollary from the preceding Theorem (also stated in, e.g. [Str89]):

Corollary 4.1.11. *A semigroup is $\mathcal{R}$-trivial iff it divides an iterated wreath product of U_1 factors.*

Proof. Use the preceding theorem as well as Propositions 2.3.7 and 2.3.8 and Corollary 4.1.9. □

4.2 Commutative Languages

In this section we consider commutative languages and characterize them in terms of their cascade decompositions.

Recall that a language $L \subseteq \Sigma^*$ is commutative iff $\mathrm{Perm}(w) \subseteq L$ for all $w \in L$. Define $\sim \subseteq \Sigma^* \times \Sigma^*$ by $u \sim v$ iff $\mathrm{Perm}(u) = \mathrm{Perm}(v)$. Clearly $\sim$ is a congruence with infinite index. Furthermore if L is commutative, $\sim$ is finer than $\sim_L$, i.e. $\sim \subseteq \sim_L$. There exists a natural bijective mapping $f : \Sigma^*/\sim \to \mathbb{N}_0^n$, $[w] \mapsto (|w|_{\sigma_1}, \ldots, |w|_{\sigma_n})$. Because of the definition of $\sim$ this map is well defined. Notice that $\mathbb{N}_0^n$ is a monoid with componentwise addition: $(a_1 \ldots, a_n) + (b_1, \ldots, b_b) = (a_1 + b_1, \ldots, a_n + b_n)$. In this situation, f is even a monoid homomorphism and thus $\mathbb{N}_0^n \cong \Sigma^*/\sim$.

Since $\sim$ is finer than $\sim_L$, every $\sim_L$-class will be the union of several $\sim$ classes. The map $\psi : \mathbb{N}_0^n \to M(L)$, which maps every tuple to the unique $\sim_L$ class, in which the corresponding $\sim$ class is contained, is therefore well defined and surjective.

Remark 4.2.1. ψ is a homomorphism of monoids.

Proof. Let $(a_1, \ldots, a_n), (b_1, \ldots, b_n) \in \mathbb{N}_0^n$. Then $w_a = \sigma_1^{a_1} \cdots \sigma_n^{a_n}$ is a representative of the $\sim$-class corresponding to $(a_1, \ldots, a_n)$. Let w_b be defined in the same way. Then $(a_1, \ldots, a_n)\psi = [w_a]_{\sim_L}$ and:

$$\begin{aligned}((a_1, \ldots, a_n) + (b_1, \ldots, b_n))\psi &= (a_1 + b_1, \ldots, a_n + b_n)\psi \\ &= [w_a \cdot w_b]_{\sim_L} \\ &= [w_a]_{\sim_L} \cdot [w_b]_{\sim_L} \\ &= (a_1, \ldots, a_n)\psi \cdot (b_1, \ldots, b_n)\psi\end{aligned}$$

□

By Theorem 2.2.2 we have $\mathbb{N}_0^n/\sim_\psi \cong M(L)$. We will therefore fail to distinguish between $M(L)$ and $\mathbb{N}_0^n/\sim_\psi$.

Notice that $\sim_\psi$ is essentially $\sim_L$ adapted to the monoid $\mathbb{N}_0^n$. Define $\hat{L} \subseteq \mathbb{N}_0^n$ by $a \in \hat{L}$ iff $(a)f^{-1} \subseteq L$. Then $a \sim_\psi b$ iff for all $x \in \mathbb{N}_0^n$: $a + x \in \hat{L} \Leftrightarrow b + x \in \hat{L}$. We will therefore also refer to $\sim_\psi$ as $\sim_L$. It will be clear from context, whether we refer to the congruence of words or the congruence of tuples.

Now let L be a regular and commutative and let $\delta_i = (\delta_{i,1}, \ldots, \delta_{i,n})$ be defined by[5]

$$\delta_{i,j} = \begin{cases} 1 & j = i \\ 0 & j \neq i \end{cases}$$

For $i = 1, \ldots, n$ define $\sim_i$ by

$$(a_1, \ldots, a_n) \sim_i (b_1, \ldots, b_n) \quad \text{iff} \quad \delta_i \cdot a_i \sim_L \delta_i \cdot b_i$$

Then $\sim_i$ is a congruence on $\mathbb{N}_0^n$ for $i = 1, \ldots, n$: Let $k = (k_1, \ldots, k_n) \in \mathbb{N}_0^n$ and $(a_1, \ldots, a_n) \sim_i (b_1, \ldots, b_n)$. Then $\delta_i \cdot (a_i + k_i) \sim_L \delta_i \cdot (b_i + k_i)$.

Lemma 4.2.1. $M(L) \prec \mathbb{N}_0^n/\sim_1 \times \cdots \times \mathbb{N}_0^n/\sim_n$

Proof. For $i = 1, \ldots, n$ define $\eta_i : \mathbb{N}_0^n/\sim_i \to M(L)$ by $[(a_1, \ldots, a_n)]_{\sim_i} \mapsto [\delta_i \cdot a_i]_{\sim_L}$. η_i is well defined and is a homomorphism of monoids. Now define $\eta : \mathbb{N}_0^n/\sim_1 \times \cdots \times \mathbb{N}_0^n/\sim_n \to M(L)$ by $([c_1]_{\sim_1}, \ldots, [c_n]_{\sim_n}) \mapsto ([c_1])\eta_1 + \cdots + ([c_n])\eta_n$. Since $M(L)$ is commutative, η is a homomorphism. Clearly η is surjective and thus the claim follows. □

Notice that $[\mathbb{N}_0^n : \sim_i] \leq [\mathbb{N}_0^n : \sim_L] < \infty$. Hence every monoid in the direct product above is finite. Observe that $\mu_i : \mathbb{N}_0 \to \mathbb{N}_0^n/\sim_i$, $n \mapsto [\delta_i \cdot n]_{\sim_i}$ is a surjective monoid homomorphism for all $i = 1, \ldots, n$. Thus $\mathbb{N}_0^n/\sim_i$ is isomorphic to $\mathbb{N}_0/\approx_i$ for a suitable $\approx_i$ of finite index.

We will proceed to characterize commutative languages. To this end we need the following definition:

Definition 4.2.1. Let Σ be an alphabet and $L \subseteq \Sigma^*$ be a regular language. We say L is *1-semilinear*, if there exists $\sigma \in \Sigma$ and $N \subseteq \mathbb{N}_0$ such that for all $w \in \Sigma^*$:

$$w \in L \quad \Leftrightarrow \quad |w|_\sigma \in N$$

We also say L is *1-semilinear with respect to* σ.

Clearly every 1-semilinear language is commutative.

Denote by $\Psi : \Sigma^* \to \mathbb{N}_0^n$ the mapping $w \mapsto ([w]_\sim)f = (|w|_{\sigma_1}, \ldots, |w|_{\sigma_n})$, i.e. Ψ maps every word w to the tuple encoding the number of occurrences of each letter in w. Obviously Ψ is a homomorphism. In literature Ψ is usually called *Parikh mapping*. L is commutative iff $(L)\Psi \circ \Psi^{-1} = L$.

Before proceeding to characterize commutative languages, we want to illustrate, why we called the languages just defined '1-semilinear'. Recall that a set $M \subseteq \mathbb{N}_0^n$ is called

[5] In other words: $\delta_{i,j}$ is the *Kronecker-Delta*

linear, if there exist $\alpha_0, \ldots, \alpha_m \in \mathbb{N}_0^n$, such that $M = \{\alpha_0 + r_1\alpha_1 + \cdots + r_m\alpha_m | r_1, \ldots, r_n \in \mathbb{N}_0\}$. M is called *semilinear* if it is a finite union of linear sets. By Parikh's Theorem, the set $(L)\Psi$ is semilinear for all context free (and therefore all regular) languages L.

Let L be 1-semilinear with respect to σ and set $M = (L)\Psi$. We have $(\sigma)\Psi = \delta_i$ for some $i \in \{1, \ldots, n\}$. Then clearly M is closed under addition with vectors $\alpha = \sum_{j=1, j\neq i}^{n} r_j\delta_j$ for $r_j \in \mathbb{N}_0$, i.e. for any $\beta \in M$ and any such α we have $\alpha + \beta \in M$. Also, for any $\beta \notin M$ we have $\alpha + \beta \notin M$. Membership in M is determined by only one dimension. Hence the term 1-semilinear.

Turning back to commutative languages in general, we prove the following lemma:

Lemma 4.2.2. *A regular language L is commutative iff L is a finite Boolean combination 1-semilinear languages.*

Proof. If L is a Boolean combination of 1-semilinear languages, then L is clearly commutative, since commutative languages are closed under the Boolean operations.

Conversely, let L be commutative. Consider a decomposition of $M(L)$ as in Lemma 4.2.1:

$$M(L) \prec \mathbb{N}_0^n / \sim_1 \times \cdots \times \mathbb{N}_0^n / \sim_n =: N$$

Define $\varphi : \Sigma^* \to N$ by

$$w \mapsto \left([(w)\Psi]_{\sim_1}, \ldots, [(w)\Psi]_{\sim_n}\right)$$

Then φ is a homomorphism. Consider the homomorphism $\eta : N \to M(L)$ from the proof of Lemma 4.2.1. Denote $P = (F)\eta^{-1}$, where $F \subseteq M(L)$ is the set of all $\sim_L$-classes contained in $\hat{L}$. We claim that $L = (P)\varphi^{-1}$. To see this let $w \in \Sigma^*$. We have $(w)\varphi = ([w\Psi]_{\sim_1}, \ldots, [w\Psi]_{\sim_n}) =: x$. Now $(x)\eta = ([w\Psi]_{\sim_1})\eta_1 + \cdots + ([w\Psi]_{\sim_n})\eta_n$. Let $w\Psi = (a_1, \ldots, a_n)$. Then $([w\Psi]_{\sim_i})\eta_i = [\delta_i \cdot a_i]_{\sim_L}$. Hence

$$(x)\eta = [\delta_1 \cdot a_1]_{\sim_L} + \cdots + [\delta_n \cdot a_n]_{\sim_L} = [(a_1, \ldots, a_n)]_{\sim_L}$$

If $w \in L$, then $[(a_1, \ldots, a_n)]_{\sim_L} \subseteq \hat{L}$ and therefore $x \in P$. Conversely, if $w \notin L$, then $[(a_1, \ldots, a_n)]_{\sim_L} \cap \hat{L} = \varnothing$. Hence $x \notin P$.

Now P is finite and we may write $P = \{p_1, \ldots, p_m\}$, where $p_i = (p_{i,1}, \ldots, p_{i,n}) \in N$. Evidently $L = \bigcup_{i=1}^{m} L_i$, where $L_i = (\{p_i\})\varphi^{-1}$. Furthermore $L_i = \bigcap_{j=1}^{n} L_{ij}$, where

$$L_{ij} = (\mathbb{N}_0^n/\sim_1 \times\cdots\times \mathbb{N}_0^n/\sim_{j-1} \times\{p_{i,j}\} \times \mathbb{N}_0^n/\sim_{j+1} \times\cdots\times \mathbb{N}_0^n/\sim_n)\varphi^{-1}$$

which is a 1-semilinear language by the definition of φ. □

Before stating the main theorem of this section, we need to define a suitable automaton model for commutative languages:

Definition 4.2.2. Let $\mathfrak{A} = (Q, \Sigma, \delta)$ be a semiautomaton. We say $\mathfrak{A}$ is *commutative*, if for every pair $p, q \in Q$ of states and every $w \in \Sigma^*$ we have

$$p\overline{w} = q \quad \Rightarrow \quad p\overline{v} = q \qquad \forall\, v \in \mathrm{Perm}(w)$$

An automaton is called *one letter automaton (OLA) (with respect to σ)*, if there exists σ, such that $\overline{\sigma} \neq \mathrm{id}$ and for all $\sigma \neq \sigma' \in \Sigma$ we have $\overline{\sigma'} = \mathrm{id}$. A cascade product is called a *one letter cascade product* if the automaton defined by it is an OLA.

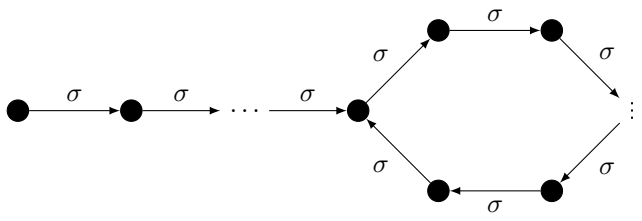


Figure 4.2: Basic scheme of a minimal OLA. The loops on each state (for letters $\neq \sigma$) have been omitted.

Remark 4.2.2. Every OLA is commutative.

If $\mathfrak{A}$ is an OLA, then $\mathfrak{A}$ recognizes only 1-semilinear languages (with respect to the uniquely determined letter σ for which $\overline{\sigma} \neq \mathrm{id}$). Conversely, if L is a 1-semilinear language, then $\mathfrak{A}_L = (Q, \Sigma, q_0, \delta, F)$ is an OLA: Suppose L is a 1-semilinear language with respect to $\sigma \in \Sigma$. Then for every $q \in Q$ and $p = q\overline{\sigma'}$ for $\sigma' \neq \sigma$ we have:

$$p\overline{w} \in F \quad \Leftrightarrow q\overline{w} \in F \qquad \forall\, w \in \Sigma^*$$

By the minimality of $\mathfrak{A}_L$ we have $p = q$ and thus $\overline{\sigma'} = \mathrm{id}$. This leads to the following remark:

Remark 4.2.3. A regular language L is a 1-semilinear language iff $\mathfrak{A}_L$ is an OLA.

In particular, 1-semilinear languages are closed under complement.

Lemma 4.2.3. *Let L be a 1-semilinear language and let $\mathfrak{A}_L = (Q, \Sigma, \delta)$ be the minimal semiautomaton recognizing L. Then Q is the union of two disjoint sets Q_1, Q_2, where Q_1 may be empty, such that:*

(a) For every $p, q \in Q_1$, $p \neq q$, we have either $p = q\overline{\sigma}^r$ for some $r \in \mathbb{N}_0$ or $q = p\overline{\sigma}^r$ for some $r \in \mathbb{N}_0$ (i.e. in Q_1 there are no loops).

(b) For every $p, q \in Q_2$ there exist $r_1, r_2 \in \mathbb{N}_0$ such that $q = p\overline{\sigma}^{r_1}$ and $p = q\overline{\sigma}^{r_2}$.

In other words, $\mathfrak{A}_L$ has the form depicted in Figure 4.2.

Proof. Let $q_0 \in Q$ and $F \subseteq Q$, such that $L = L(\mathfrak{A}_L, q_0, F)$. Since Q is finite there must be a repetition in the sequence $q_0\overline{\sigma}^1, q_0\overline{\sigma}^2, q_0\overline{\sigma}^3, \ldots$. Since $\mathfrak{A}_L$ is an OLA and since $\mathfrak{A}_L$ has the minimal number of states among all automata accepting L, all the states from Q occur in this sequence. The claim follows. □

We call Q_2 the *cyclic part of* $\mathfrak{A}_L$.

We now turn to cascade decompositions of minimal OLAs. Suppose L is a 1-semilinear language and $\mathfrak{A}_L$ is the canonical semiautomaton recognizing L. Let $Q = Q_1 \dot\cup Q_2$ according to the previous lemma, $Q_1 = \{q_0, \ldots, q_{n-1}\}$, $q_{i+1} = q_i\overline{\sigma}^{A_L}$ for $i = 0, \ldots, n-2$. We have:

Lemma 4.2.4. *$\mathfrak{A}_L$ can be covered by a product*

$$\mathfrak{B} := \mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_n \times \mathfrak{G}$$

where $\mathfrak{R}_i$ is a biased one letter reset for $i = 1, \ldots, n$ and $\mathfrak{G} = (Q_2, \Sigma, \delta)$ is a cyclic permutation automaton with:

(i) $|Q_2| = |M(\mathfrak{G})|$

(ii) $\mathfrak{G}$ *is an OLA with respect to* σ.

(iii) All states from Q_2 occur in the sequence $q\overline{\sigma}^G, q\overline{\sigma^2}^G, q\overline{\sigma^3}^G, \ldots$ for arbitrary $q \in Q_2$.

Proof. Define $\mathfrak{G} = (Q_2, \Sigma, \delta)$ by

$$\overline{\sigma}^G = \overline{\sigma}^{A_L}|_{Q_2}$$

By the definition of Q_2 in the previous lemma, we have $|Q_2| = |M(\mathfrak{G})|$. Furthermore, since $\mathfrak{A}_L$ is an OLA, so is $\mathfrak{G}$. The third point follows by the construction of Q_2.

Define $\mathfrak{R}_i = (\mathbb{B}, \Sigma \times \mathbb{B}^{i-1}, \delta_i)$ by

$$q\overline{(\sigma', (x_1, \ldots, x_{i-1}))}^{R_i} = \begin{cases} 1 & x_1 = x_2 = \cdots x_{i-1} = 1 \wedge \sigma' = \sigma \\ q & \text{otherwise} \end{cases}$$

for all $q \in \mathbb{B}$ with the convention that $\Sigma \times \mathbb{B}^0 = \Sigma$.

Recall that $n = |Q_1|$. Let $p_0 \in Q_2$ be such that $p_0\overline{\sigma^n} = q_0\overline{\sigma^n}$. Notice that $q_0\overline{\sigma^n}$ is the first state from Q_2 reached in the sequence $q_0\overline{\sigma}, q_0\overline{\sigma \cdot \sigma}, \ldots$. Define the set $A \subseteq \mathbb{B}^n$ by $(x_1, \ldots, x_n) \in A$ iff there exists $i \in \{1, \ldots, n\}$ such that $x_1 = \cdots = x_{i-1} = 1$ and $x_j = 0$ for all $j \geq i$. Notice that $(1, \ldots, 1) \notin A$. A is the set of all bit vectors, in which all positions with entry 1 are left of all the positions with entry 0 and which posses at least one zero entry. For $x = (x_1, \ldots, x_n) \in A$ define $\max(x) := \sum_{i=1}^n x_i$. Notice $\max(x) = 0$ iff $x = (0, \ldots, 0)$.

Define $P_0 \subseteq \mathbb{B}^n \times Q_2$ by $(x_1, \ldots, x_n, q) \in P_0$ iff $x = (x_1, \ldots, x_n) \in A$ and either $q = p_0\overline{\sigma^i}$ and $\max(x) = i < n$ or $\max(x) = n$ and $q \in Q_2$ arbitrary. Then $\mathfrak{B}_0 = (P_0, \Sigma, \delta^B|_{P_0})$ is a subsemiautomaton of $\mathfrak{B}$ by construction. Recall $Q_1 = \{q_0, \ldots, q_{n-1}\}$. We define $\varphi : \mathfrak{B}_0 \to \mathfrak{A}_L$ by

$$(x, q)\varphi = \begin{cases} q_j & \max(x) = j < n \\ q & \max(x) = n \end{cases}$$

Then $((x_1, \ldots, x_n, q)\overline{\gamma}^B)\varphi = (x_1, \ldots, x_n, q)\varphi\overline{\gamma}^{A_L}$ for all $\gamma \in \Sigma$ by the choice of p_0. Clearly φ is surjective and hence the claim follows. □

The following remark is obtained directly from the proof:

Remark 4.2.4. In the decomposition above, $\mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_n$ is an OLA and hence commutative.

We now want to decompose $\mathfrak{G} = (Q_2, \Sigma, \delta^G)$ from the previous lemma into direct and cascade products of simple grouplike automata. We recall that every cyclic group of order m is isomorphic to $\mathbb{Z}/m\mathbb{Z} = \{[k]_m | k \in \mathbb{Z}\}$, where $k_1 \equiv_m k_2$ iff $k_1 \mod m = k_2 \mod m$ for all $k_1, k_2 \in \mathbb{Z}$. We will identify $M(\mathfrak{G})$ with $\mathbb{Z}/m\mathbb{Z} =: H$. In subsequent paragraphs, when dealing with $\equiv_m$ classes, we will often drop the subscript m.

Let $|Q_2| = m$. As before we will denote by σ the unique letter from Σ, which does not induce the identity mapping on Q_2. Define $\mathfrak{H} = (H, \Sigma, \delta^H)$ by

$$[k]\overline{\gamma} = \begin{cases} [k+1] & \gamma = \sigma \in \Sigma \\ [k] & \text{otherwise} \end{cases}$$

A first step towards decomposing $\mathfrak{G}$ is the following lemma:

Lemma 4.2.5. $\mathfrak{G} \cong \mathfrak{H}$. *In particular* $\mathfrak{G} \leq \mathfrak{H}$.

Proof. Pick $q_0 \in Q_2$ arbitrary and define $\varphi : H \to Q_2$ by $([k])\varphi = q_0\overline{\sigma}^{k_0}$, where $0 \leq k_0 \in [k]$ is the smallest, non-negative representative[6] of $[k]$. Then, by the construction of $\mathfrak{G}$, φ is surjective and is furthermore a homomorphism of semiautomata:

$$\begin{aligned} ([k]\overline{\sigma})\varphi &= ([k+1])\varphi \\ &= q_0\overline{\sigma}^{k+1} \\ &= q_0\overline{\sigma}^{k}\overline{\sigma} \\ &= ([k])\varphi\overline{\sigma} \end{aligned}$$

For $\sigma' \in \Sigma$, $\sigma' \neq \sigma$, there is nothing to show. Since Q_2 and H are finite sets of the same cardinality, we deduce that φ is bijective. The claim follows. □

Notice that this essentially means, that we can think of $\mathfrak{G}$ as a grouplike automaton, i.e. we can identify the states with elements of the group $M(\mathfrak{G})$ (recall the definition of a grouplike automaton from Section 2.3).

Now we will decompose $\mathfrak{H}$. Let $m = \prod_{i=1}^{r} p_i^{k_i}$, where $p_1, \ldots p_r \in \mathbb{P} = \{2, 3, 5, 7, \ldots\}$ are pairwise distinct primes, and define $m_i := p_i^{k_i}$. Let $\mathfrak{C}_i = (\mathbb{Z}/m_i\mathbb{Z}, \Sigma, \delta^{C_i})$, where

$$[k]_{m_i}\overline{\alpha}^{C_i} = \begin{cases} [k+1]_{m_i} & \alpha = \sigma \in \Sigma \\ [k] & \text{otherwise} \end{cases}$$

Notice that each $\mathfrak{C}_i$ is an OLA with respect to σ.

The next lemma is essentially an automaton theoretic equivalent of the well known fact that $\mathbb{Z}/m\mathbb{Z} \cong \bigtimes_{i=1}^{r} \mathbb{Z}/m_i\mathbb{Z}$.

Lemma 4.2.6. $\mathfrak{H} \cong \mathfrak{C}_1 \times \cdots \times \mathfrak{C}_r$

[6] Of course all non-negative representatives will induce the same mapping, since $|Q_2| = m$. However, it is convenient to use the smallest among these representatives in order to avoid questions about well-definedness.

Proof. Denote $\mathfrak{B} = \mathfrak{C}_1 \times \cdots \times \mathfrak{C}_r$. Define $\varphi : \mathfrak{H} \to \mathfrak{B}$ by

$$([k]_m)\varphi = ([k]_{m_1}, \ldots, [k]_{m_r})$$

As a consequence of the well known Chinese-Remainder Theorem, φ is well defined and bijective. It is therefore sufficient to show that φ is indeed a homomorphism of semiautomata. To this end let $\alpha \in \Sigma$. We want to show

$$([k]_m \overline{\alpha}^H)\varphi = ([k]_m)\varphi \overline{\alpha}^B$$

If $\alpha \in \Sigma \setminus \{\sigma\}$ there is nothing to show. Hence let $\alpha = \sigma$. Then:

$$\begin{aligned}
([k]_m \overline{\sigma}^H)\varphi &= ([k]_m + [1]_m)\varphi \\
&= ([k+1]_m)\varphi \\
&= ([k+1]_{m_1}, \ldots, [k+1]_{m_r}) \\
&= ([k]_{m_1}, \ldots, [k]_{m_r}) + ([1]_{m_1}, \ldots, [1]_{m_r}) \\
&= ([k]_{m_1}, \ldots, [k]_{m_r})\overline{\sigma}^B
\end{aligned}$$

□

Hence every minimal OLA $\mathfrak{A}_L$ can be covered by a direct product of:

1. A one letter cascade product of biased resets
2. A direct product of cyclic grouplike OLA, the size of which (i.e. the number of states) is a prime power. Furthermore, the primes occurring in this direct product are pairwise distinct. The product over all prime powers is precisely the prime decomposition of the size of the cyclic part of $\mathfrak{A}_L$.

By [Gin68] each grouplike automaton $\mathfrak{C}_i$ in the direct product above can be covered by a cascade product of simple grouplike automata, the groups of which divide $M(\mathfrak{C}_i)$. Recall $|M(\mathfrak{C}_i)| = p^k$ for some prime p and some $k \in \mathbb{N}$. If G is a group and U is a group dividing G, then U must be the quotient of a subgroup V of G. It is well known that $|V|$ divides $|G|$ and that furthermore $|U|$ divides $|V|$. Thus $|U|$ divides $|G|$. In the situation above we get (with $G = M(\mathfrak{G})$) that all simple groups dividing $M(\mathfrak{C}_i)$ must be of order p^t for some $t \leq k$. Since $M(\mathfrak{C}_i)$ is cyclic and thus abelian and since every abelian group is simple iff it is of prime order, we get that all simple groups dividing $M(\mathfrak{C}_i)$ are of prime order.

This analysis, however, is insufficient for our purposes, since we want to ensure that all languages accepted by the cascade are commutative. We therefore have to show that the decomposition of $\mathfrak{C}_i$ into a cascade of cyclic automata of prime order can be done in such a way, that this cascade will only accept commutative languages.

Lemma 4.2.7. *Let $\mathfrak{C} = (\mathbb{Z}/p^k\mathbb{Z}, \Sigma, \delta^C)$ be cyclic and an OLA with respect to $\sigma \in \Sigma$. Then:*

$$\mathfrak{C} \leq (\mathbb{Z}/p\mathbb{Z}, \Sigma, \delta^{C_1}) \circ (\mathbb{Z}/p\mathbb{Z}, \Sigma \times \mathbb{Z}/p\mathbb{Z}, \delta^{C_2}) \circ \cdots \circ (\mathbb{Z}/p\mathbb{Z}, \Sigma \times (\mathbb{Z}/p\mathbb{Z})^{k-1}, \delta^{C_k})$$

where the cascade is an OLA with respect to σ.

Proof. The proof is by induction on k. If $k = 1$ then $\mathfrak{C}$ is already of the desired form.

Let $k > 1$. Define $\mathfrak{D} = (\mathbb{Z}/p\mathbb{Z}, \Sigma, \delta^D)$ by $[r]_p\overline{\gamma}^D = [r+1]_p$ if $\gamma = \sigma$ and $[r]_p$ otherwise. Now define $\mathfrak{H} = (\mathbb{Z}/p^{k-1}\mathbb{Z}, \Sigma \times \mathbb{Z}/p\mathbb{Z}, \delta^H)$ by $[r]_{p^{k-1}}\overline{\gamma}^H = [r+1]_{p^{k-1}}$ if $\gamma = (\sigma, [p-1]_p)$ and $[r]_{p^{k-1}}$ otherwise. Then $\mathfrak{H}$ is an OLA. This means we can apply the induction hypothesis to $\mathfrak{H}$. Therefore, by Theorem 2.4.3, we are done if we have shown that $\mathfrak{C} \leq \mathfrak{D} \circ \mathfrak{H} =: \mathfrak{A}$.

Observe that $|\mathbb{Z}/p\mathbb{Z} \times \mathbb{Z}/p^{k-1}\mathbb{Z}| = p^k$. Hence we will not define a subsemiautomaton of $\mathfrak{A}$ but rather use the full set of states for the covering. Define $\varphi : \mathfrak{A} \to \mathfrak{C}$ by $([r]_p, [s]_{p^{k-1}}) \mapsto [(r \mod p) + p \cdot s]_{p^k}$. We have to show that this mapping is well defined. To this end, let $r' \equiv_p r$ and $s' \equiv_{p^{k-1}} s$. Then w.l.o.g. $r' = r + p \cdot a$ and $s' = s + p^{k-1} \cdot b$ for some $a, b \in \mathbb{Z}$. We have:

$$\begin{aligned}[(r' \mod p) + p \cdot s']_{p^k} &= [r_0 + p \cdot (s + p^{k-1} \cdot b)]_{p^k} \\ &= [(r \mod p) + p \cdot s + p^k \cdot b]_{p^k} \\ &= [(r \mod p) + p \cdot s]_{p^k}\end{aligned}$$

We show that φ is a homomorphism, i.e.

$$(([r]_p, [s]_{p^{k-1}})\overline{\gamma}^A)\varphi = (([r]_p, [s]_{p^{k-1}})\varphi)\overline{\gamma}^C$$

If $\gamma \neq \sigma$ there is nothing to show. Furthermore, if $[r]_p \neq [p-1]_p$ and $\gamma = \sigma$, then

$$\begin{aligned}(([r]_p, [s]_{p^{k-1}})\overline{\sigma}^A)\varphi &= ([r+1]_p, [s]_{p^{k-1}})\varphi \\ &= [(r+1 \mod p) + s \cdot p]_{p^k} \\ &= [(r \mod p) + 1 + s \cdot p]_{p^k} \\ &= [(r \mod p) + s \cdot p]_{p^k}\overline{\sigma}^C \\ &= (([r]_p, [s]_{p^{k-1}})\varphi)\overline{\sigma}^C\end{aligned}$$

If $[r]_p = [p-1]_p$ and $\gamma = \sigma$, then:

$$\begin{aligned}(([p-1]_p, [s]_{p^{k-1}})\overline{\sigma}^A)\varphi &= ([0]_p, [s+1]_{p^{k-1}})\varphi \\ &= [0 + (s+1) \cdot p]_{p^k} \\ &= [0 + s \cdot p + p]_{p^k} \\ &= [p - 1 + s \cdot p + 1]_{p^k} \\ &= [p - 1 + s \cdot p]_{p^k}\overline{\sigma}^C \\ &= (([p-1]_p, [s]_{p^{k-1}})\varphi)\overline{\sigma}^C\end{aligned}$$

Clearly φ is surjective and the claim follows. □

We can now state the main theorem of this section:

Theorem 4.2.8. *Let $L \subseteq \Sigma^*$ be regular. The following are equivalent:*

(a) L is commutative.

(b) $M(L)$ is commutative.

(c) $\mathfrak{A}_L$ is commutative.

(d) L is a finite Boolean combination of 1-semilinear languages.

(e) L is recognized by a direct product of OLAs.

(f) L is recognized by a direct product of

(i) one letter cascade products of biased resets

(ii) one letter cascade products of cyclic, grouplike automata of prime order.

Proof. (a) $\Rightarrow$ (b): Proposition 2.3.15

(b) $\Rightarrow$ (c): $M(L) \cong M(\mathfrak{A}_L)$, hence for every $w \in \Sigma^*$ and every $w' \in \mathrm{Perm}(w)$ we have $\overline{w} = \overline{w'}$ in $\mathfrak{A}_L$.

(c) $\Rightarrow$ (a): Let $L = L(\mathfrak{A}_L, q_0, F)$ and $w \in L$. Then $q_0\overline{w} \in F$ and hence $q_0\overline{w'} \in F$ for all $w' \in \mathrm{Perm}(w)$.

(a) $\Leftrightarrow$ (d): Lemma 4.2.2

(d) $\Rightarrow$ (e): Let $L = \bigcup_{i=1}^{k} \bigcap_{j=1}^{m_1} L_{ij}$ for 1-semilinear languages L_{ij}. Then $\mathfrak{A}_{L_{ij}}$ is an OLA for every pair $i = 1, \ldots, k$, $j = 1 \ldots, m_i$. L is recognized by the product $\bigtimes_{i=1}^{k} \bigtimes_{j=1}^{m_i} \mathfrak{A}_{L_{ij}}$.

(e) $\Rightarrow$ (f): Use Lemmas 2.4.2, 4.2.4, 4.2.5, 4.2.6 and Proposition 2.3.10 to obtain the decomposition. By Remark 4.2.4 and by Lemma 4.2.7 the automata defined by the cascade products are OLAs.

(f) $\Rightarrow$ (a): Since all cascade products are OLAs they accept only commutative languages. Hence every language recognized by the direct product of these elements will only recognize commutative languages. □

We obtain the following well-known algebraic characterization of commutative monoids:

Corollary 4.2.9. *A monoid M is commutative iff M divides a direct product of products of the form $U_1^n \times C$, where U_1^n is an iterated wreath product and C is a cyclic group.*

Proof. Use the previous Theorem and Propositions 2.3.7 and 2.3.8. □

4.3 The Scope of Cascade Products

So far we have looked at the types of factors occurring in a cascade product, as well as at the number of factors. We will now investigate another possible measure for the "complexity" of a cascade product, which we define as follows: Given a cascade product $\mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_n$ of n factors over the alphabet Σ with state space $Q_1 \times \cdots \times Q_n$, we define the *scope of the factor* $\mathfrak{R}_i$, to be the maximal number k, such that for arbitrary $(p_1, \ldots, p_k) \in Q_{i-k} \times \cdots \times Q_{i-1}$ all inputs of the form $(\sigma, (q_1, \ldots, q_{i-k-1}, p_1, \ldots, p_k))$ and $(\sigma, (q_1', \ldots, q_{i-k-1}', p_1, \ldots, p_k))$ induce the same mapping in $\mathfrak{R}_i$ for all choices of $(q_1, \ldots q_{i-k-1}), (q_1', \ldots, q_{i-k-1}') \in Q_1 \times \cdots \times Q_{i-k-1}$ and all $\sigma \in \Sigma$. The *scope of the product* $\mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_n$ is then the maximum over the scopes of all factors.

In other words: the scope is the maximal number of automata immediately preceding $\mathfrak{R}_i$, the state of which has an influence on the transition behavior of $\mathfrak{R}_i$. For instance, if we have a direct product between $\mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_{i-1}$ and $\mathfrak{R}_i \circ \cdots \circ \mathfrak{R}_n$, then the scope of $\mathfrak{R}_i$ is 0. The scope of $\mathfrak{R}_i$ is trivially bounded by $i-1$. We now want to investigate, what restriction the scope of cascade products imposes on the class of languages recognized by these products. It is evident that without any restriction on the number of states of the factors, the scope will be no restriction at all. For any given language L we can then simply pick $\mathfrak{A}_L$ and thus have a trivial scope 0 product, which recognizes L. For this reason, we will henceforth consider the scope only in the context of cascade products of resets.

4.3.1 Language Classes with Constant Scope

In this paragraph we will investigate which language classes can be recognized by cascade products of resets with constant scope. More precisely, given a class $\mathcal{C}$ of regular languages (i.e. piecewise, testable languages, $\mathcal{R}$-trivial languages etc.), does there exists a number $k \in \mathbb{N}_0$, such that all languages $L \in \mathcal{C}$ can be recognized by a cascade product of scope at most k? If such a k exists, we say *$\mathcal{C}$ has constant scope k* or just *$\mathcal{C}$ has constant scope*.

We have, in fact, already established one result on languages, which are recognizable by a scope 1 cascade product of resets. Careful inspection of the proof of Lemma 4.1.1 yields:

Proposition 4.3.1. *Every piecewise testable language can be recognized by a cascade product of 1-biased resets with scope 1.*

Notice, that this is no equivalence: The language $\overline{\Sigma^* a \Sigma^*} b \Sigma^*$, where $\Sigma = \{a, b, c\}$ is of dot-depth 2 [CB71], but can be recognized by a cascade product of two resets, which trivially has scope 1.

What other language families can be recognized by cascade products with constant scope? As we will prove shortly, all $\mathcal{R}$-trivial languages can be recognized by a scope 2 product of resets. We first state a result from Eilenberg:

Theorem 4.3.2 (Eilenberg, see [Pin86])**.** *Every $\mathcal{R}$-trivial language $L \subseteq \Sigma^*$ is the disjoint union of languages $\Sigma_0^* \sigma_1 \Sigma_1^* \sigma_2 \cdots \sigma_n \Sigma_n^*$, where $\sigma_1, \ldots, \sigma_n \in \Sigma$ and $\Sigma_i \subseteq \Sigma \setminus \{\sigma_{i+1}\}$ for $i = 0, \ldots, n-1$ and $\Sigma_n \subseteq \Sigma$.*

We call languages of this form *left deterministic products*. It is now sufficient to show that left deterministic products can be recognized by a scope 2 cascade product of resets.

Lemma 4.3.3. *Let $L = \Sigma_0^* \sigma_1 \Sigma_1^* \sigma_2 \cdots \sigma_n \Sigma_n^*$ be a left deterministic product. Then L can be recognized by a scope 2 cascade product of 1-biased resets.*

Before proving this Lemma, we consider an automaton, which recognizes left deterministic products. For the case $n = 3$ we consider Figure 4.3. We first illustrate the

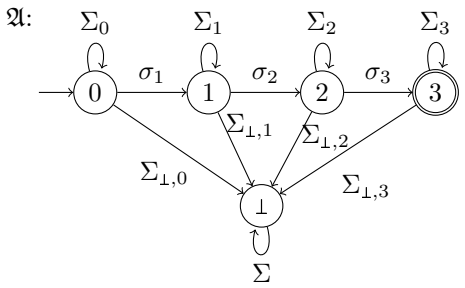


Figure 4.3: The automaton given above recognizes $\Sigma_0^*\sigma_1\Sigma_1^*\sigma_2\cdots\sigma_3\Sigma_3^*$. The set $\Sigma_{\perp,i}$ is short-hand for $\Sigma \setminus (\Sigma_i \cup \{\sigma_{i+1}\})$.

idea of the proof for the automaton given in Figure 4.3. Consider the following cascade product:

$$\mathfrak{R}^{\Sigma_{\perp,0}} \circ \mathfrak{R}^{(0,\sigma_1)} \circ \mathfrak{R}^{(1,\Sigma_{\perp,1})} \circ \mathfrak{R}^{(1,0,\sigma_2)} \circ \mathfrak{R}^{(1,\Sigma_{\perp,2})} \circ \mathfrak{R}^{(1,0,\sigma_3)} \circ \mathfrak{R}^{(1,\Sigma_{\perp,3})}$$

The superscripts denote the inputs, which induce the constant 1 mapping in the automaton. For instance, all inputs from $\Sigma_{\perp,0}$ induce the constant 1 mapping in the first automaton. The fourth automaton will move the state 1 iff the second automaton is in state 1 already, the third automaton is in state 0 and the input σ_2 is read. We claim that this automaton covers the automaton $\mathfrak{A}$ from Figure 4.3.

Since a formal proof will be given below for the general construction, we will only outline how the identification of states works. Consider the state $(0, 0, 0, \ldots, 0)$. This state will be identified with 0. If we read an input from $\Sigma_{\perp,0}$ when in state 0 then $\mathfrak{A}$ moves to $\perp$. The cascade above will move to state $(1, 0, \ldots, 0)$ upon reading such an input, which we identify with $\perp$. It is easy to verify, that this is a sink state for this cascade. If we read σ_1, while in 0 the automaton $\mathfrak{A}$ will move to 1. The cascade will move to $(0, 1, 0, \ldots, 0)$, which we will, consequently, associate with 1. It is easy to verify that the mapping of states thus defined is indeed a homomorphism of a subsemiautomaton of the cascade onto $\mathfrak{A}$.

Proof of Lemma 4.3.3. We prove a slightly stronger statement by induction on n: Any left-deterministic language $L = \Sigma_0^*\sigma_1\Sigma_1^*\sigma_2\cdots\sigma_n\Sigma_n^*$ can be recognized by a scope 2 cascade product of $2n + 1$ biased resets, such that a word $w \in \Sigma$ is accepted iff the last reset of the cascade is in state 0 and the second to last reset is in state 1.

If $n = 1$ we have $L = \Sigma_0^*\sigma_1\Sigma_1^*$. Define $\Sigma_{\perp,0} = \Sigma \setminus (\Sigma_0 \cup \{\sigma_1\})$ and $\Sigma_{\perp,1} = \Sigma \setminus \Sigma_1$. We the define $\mathfrak{R}_1 = (\mathbb{B}, \Sigma, \delta^{R_1})$ by

$$\overline{\sigma}^{R_1} \equiv \begin{cases} 1 & \sigma \in \Sigma_{\perp,0} \\ \mathrm{id} & \text{otherwise} \end{cases}$$

We then define $\mathfrak{R}_2 = (\mathbb{B}, \Sigma \times \mathbb{B}, \delta^{R_2})$ by

$$\overline{(\sigma, x)}^{R_2} \equiv \begin{cases} 1 & \sigma = \sigma_1 \wedge x = 0 \\ \mathrm{id} & \text{otherwise} \end{cases}$$

Finally we define $\mathfrak{R}_3 = (\mathbb{B}, \Sigma \times \mathbb{B}^2, \delta^{R_3})$ in the following way:

$$\overline{(\sigma, (x_1, x_2))}^{R_3} \equiv \begin{cases} 1 & \sigma \in \Sigma_{\perp,1} \wedge x_2 = 1 \\ \mathrm{id} & \text{otherwise} \end{cases}$$

Set $\mathfrak{A} = \mathfrak{R}_1 \circ \mathfrak{R}_2 \circ \mathfrak{R}_3$. We claim that this cascade recognizes L. To this end we choose $q_0 = (0,0,0)$ and set $F = \{(0,1,0),(1,1,0)\}$. This set of final states fulfills the requirements of the claim, we wish to prove. We pick $w \in L$. Then we can factorize $w = u\sigma_1 v$ with $u \in \Sigma_0^*$ and $v \in \Sigma_1^*$. Clearly this word is accepted by $(\mathfrak{A}, q_0, F)$. Conversely, if $w \in \Sigma^*$ is accepted by the automaton, then there must exist a prefix $u \sqsubseteq w$, such that $u \in \Sigma_0^*$ and $w = u\sigma_1 v$ for some suffix v of w. This is because upon reading the first letter from $\Sigma_{\perp,0} = \Sigma \setminus (\Sigma_0 \cup \{\sigma_1\})$ the automaton $\mathfrak{R}_1$ will move to state 1, thereby preventing $\mathfrak{R}_2$ from reaching 1. By the definition of $\mathfrak{R}_3$ and the set F of accepting states, it follows that v must be in Σ_1^*.

We now assume that $n > 1$. Write $L = \Sigma_0^*\sigma_1\Sigma_1^*\sigma_2 \cdots \sigma_n\Sigma_n^*$. We consider the language $L' = \Sigma_0^*\sigma_1\Sigma_1^*\sigma_2 \cdots \sigma_{n-1}(\Sigma_{n-1} \cup \{\sigma_n\})^*$. By the induction assumption we can choose a scope 2 cascade product $\mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_{2n-1}$, which recognizes L' with accepting states $F' = \{x \in \mathbb{B}^{2n-1} | x_{2n-2} = 1 \wedge x_{2n-1} = 0\}$ and initial state $q_0' \in \mathbb{B}^{2n-1}$.

We now define $\mathfrak{R}_{2n} = (\mathbb{B}, \Sigma \times \mathbb{B}^{2n-1}, \delta^{R_{2n}})$ and $\mathfrak{R}_{2n+1} = (\mathbb{B}, \Sigma \times \mathbb{B}^{2n}, \delta^{R_{2n+1}})$ as follows:

$$\overline{(\sigma, (x_1, \ldots, x_{2n-1}))}^{R_{2n}} \equiv \begin{cases} 1 & \sigma = \sigma_n \wedge x_{2n-1} = 0 \wedge x_{2n-2} = 1 \\ \mathrm{id} & \text{otherwise} \end{cases}$$

and

$$\overline{(\sigma, (x_1, \ldots, x_{2n}))}^{R_{2n+1}} \equiv \begin{cases} 1 & \sigma \notin \Sigma_n \wedge x_{2n} = 1 \\ \mathrm{id} & \text{otherwise} \end{cases}$$

We claim that the cascade product $\mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_{2n+1}$ recognizes L, with initial state $q_0 = (q_0', 0, 0)$ and final states $F = \{x \in \mathbb{B}^{2n+1} | x_{2n} = 1 \wedge x_{2n+1} = 0\}$. To this end consider a word w accepted by this automaton. By construction we may write $w = u\sigma_n v$, where $u \in L'$. Now by the definition of F and $\mathfrak{R}_{2n+1}$ we have $v \in \Sigma_n^*$. Conversely, every word $w \in L$ has a prefix $u \in L'$, such that $w = u\sigma_n v$ for some $v \in \Sigma_n^*$. It is straightforward to verify that this word is accepted by the automaton and the claim follows. □

In summary we have shown:

Theorem 4.3.4. *Every $\mathcal{R}$-trivial language is recognized by a scope 2 cascade product of biased resets.*

4.3.2 Scope and Dot-Depth

We have already seen in the beginning of the previous section, that there are languages, which are not of dot-depth 1, but can be recognized by a scope 1 cascade product. Hence there cannot be a tight correspondence between the dot-depth of a language and the minimal scope among all cascade products recognizing that language.

However, for some languages there seems to be a connection. Let $\Sigma = \{a, b\}$ and define

$$L_n = \{w \in \Sigma^* | \forall x \sqsubseteq w : 0 \leq |x|_a - |x|_b \leq n \wedge |w|_a = |w|_b\} \tag{4.3.1}$$

We have $L_n \in \mathcal{B}_n \setminus \mathcal{B}_{n-1}$ for all $n \geq 1$ (see [Tho87]). We will show that there is a very natural way to recognize L_n with a cascade product of resets, which has both the minimal number of factors among all cascade products of resets recognizing L_n and also has scope n for all $n \in \mathbb{N}$.

Lemma 4.3.5. *Let $\mathfrak{A} = \mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_n$ be a cascade product of resets and $\sigma \in \Sigma$. Let $q \in \mathbb{B}$. Then we have $q\overline{\sigma^n} = q\overline{\sigma^m}$ for all $m \geq n$.*

Proof. It is sufficient to prove $q\overline{\sigma^n} = q\overline{\sigma^{n+1}}$. The more general case $m = n + k$ then follows by a trivial induction on k.

We prove the claim by induction on n. For $n = 1$ there is nothing to show, since $\overline{\sigma}$ is either constant, or the identity.

Suppose now that $n > 1$. Write $\mathfrak{B} = \mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_{n-1}$, i.e. $\mathfrak{A} = \mathfrak{B} \circ \mathfrak{R}_n$. Let $p = (p_B, p_n) = q\overline{\sigma^{n-1}}$ for $p_B \in \mathbb{B}^{n-1}$ and $p_n \in \mathbb{B}$. We have:

$$\begin{aligned}
q\overline{\sigma^{n+1}} = p\overline{\sigma^2} &= (p_B, p_n)\overline{\sigma}\,\overline{\sigma} \\
&= \big(p_B\overline{\sigma}, p_n\overline{(\sigma, p_B)}\big)\overline{\sigma} \\
&= \big(p_B, p_n\overline{(\sigma, p_B)}\big)\overline{\sigma} && \text{by the induction hypothesis} \\
&= \big(p_B\overline{\sigma}, p_n\overline{(\sigma, p_B)}\;\overline{(\sigma, p_B)}\big) \\
&= \big(p_B\overline{\sigma}, p_n\overline{(\sigma, p_B)}\big) && \text{since } \mathfrak{R}_n \text{ is a reset automaton} \\
&= p\overline{\sigma} = q\overline{\sigma^n}
\end{aligned}$$

The claim follows. □

As a corollary we obtain:

Corollary 4.3.6. *Let $\mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_m$ be a cascade product of resets recognizing L_n. Then $m \geq n + 1$.*

Proof. If $m \leq n$, then for any initial state q_0 we have $q_0\overline{a^n} = q_0\overline{a^{n+1}}$. This is a contradiction to $a^n \in L_n$ but $a^{n+1} \notin L_n$. □

Next we give a cascade product for L_n, which has minimal length and scope precisely

n. To this end, let $\mathfrak{R}_1, \ldots, \mathfrak{R}_n$ be defined in the following way:

$$\mathfrak{R}_i = (\mathbb{B}, \Sigma \times \mathbb{B}^{i-1}, \delta^{R_i})$$

$$\overline{(\sigma, (x_1, \ldots, x_{i-1}))}^{R_i} = \begin{cases} 1 & x_1 = \cdots = x_{i-1} = 1 \wedge \sigma = a \\ 0 & x_1 = \cdots = x_{i-1} = 0 \wedge \sigma = b \\ \mathrm{id} & \text{otherwise} \end{cases}$$

with the convention $\Sigma \times B^0 = \Sigma$. Furthermore, define

$$\mathfrak{R}_\perp = (\mathbb{B}, \Sigma \times \mathbb{B}^n, \delta^{R_\perp})$$

$$\overline{(\sigma, (x_1, \ldots, x_n))}^{R_\perp} = \begin{cases} 1 & x_1 = \cdots = x_n = 0 \wedge \sigma = b \\ 1 & x_1 = \cdots = x_n = 1 \wedge \sigma = a \\ \mathrm{id} & \text{otherwise} \end{cases}$$

Define $\mathfrak{A}_0 = \mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_n$. Let $q_0 = (0, \ldots, 0)$. For $w \in \Sigma^*$ let

$$\Delta_n(w) = \begin{cases} |w|_a - |w|_b & 0 \leq |w|_a - |w|_b \leq n \\ \perp & \text{otherwise} \end{cases}$$

Observe that $w \notin L_n$ iff there exists a prefix $u \sqsubseteq w$ with $\Delta_n(u) = \perp$ or if $\Delta_n(w) > 0$.

Lemma 4.3.7. *Let $w \in L_n$, $u \sqsubseteq w$ be a prefix of w and let $(x_1, \ldots, x_n) = q_0 \overline{u}^{A_0}$. Then*

$$\Delta_n(u) = \sum_{i=1}^{n} x_i$$

Proof. We prove this by induction on $m = |u|$. If $m = 0$ then there is nothing to show. Hence suppose that $m > 0$, say $u = u' \cdot \sigma$ for some $\sigma \in \Sigma$. By the induction hypothesis we have $\Delta_n(u') = \sum_{i=1}^{n} x_i$, where $(x_1, \ldots, x_n) = q_0 \overline{u'}^{A_0}$. Now suppose $\Delta_n(u') = n$. Then because $w \in L_n$ we see that $\sigma = b$ and

$$q_0 \overline{u}^{A_0} = (0, x_2, \ldots, x_n) = (0, 1, \ldots, 1)$$

Conversely, if $\Delta_m(u') = 0$ then because of $w \in L_n$ we have $\sigma = a$ and hence

$$q_0 \overline{u}^{A_0} = (1, x_2, \ldots, x_n) = (1, 0, \ldots, 0)$$

Otherwise $0 < \Delta_n(u') < n$. Let $i_0 \in \{1, \ldots, n\}$ be minimal, such that $x_{i_0} = 0$ and $i_1 \in \{1, \ldots, n\}$ minimal such that $x_{i_1} = 1$ (observe that both indices must exist). If $\sigma = a$ then $\mathfrak{R}_{i_0}$ will receive input $(a, (1, \ldots, 1))$ and will therefore change its state to 1. All other resets maintain their state and hence $\sum_{i=1}^{n} y_i = \Delta_n(u)$ where $(y_1, \ldots, y_n) = x\overline{\sigma}^{A_0} = q_0 \overline{u}^{A_0}$.

If $\sigma = b$ then $\mathfrak{R}_{i_1}$ will receive the input $(b, (0, \ldots, 0))$ and will therefore change its state to 0. All other reset again maintain their state and hence we again have $\sum_{i=1}^{n} y_i = \Delta_n(u)$ where $(y_1, \ldots, y_n) = x\overline{\sigma}^{A_0} = q_0 \overline{u}^{A_0}$. □

Now we are ready to define the cascade product recognizing L_n:

Lemma 4.3.8. *L_n is recognized by the following cascade product*

$$\mathfrak{A} = \mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_n \circ \mathfrak{R}_\perp$$

with initial state $q_0 = (0, \ldots, 0)$ and final states $F = \{q_0\}$.

Proof. Let $w \in L_n$. Then for every prefix $u \sqsubseteq w$ of w we have $0 \leq \Delta_n(u) \leq n$. Furthermore $\Delta_n(w) = 0$. Hence $\mathfrak{R}_\perp$ never receives an input of the form $(a, (1, \ldots, 1))$ or $(b, (0, \ldots, 0))$. Also, $q_0\overline{w} = q_0 = (0, \ldots, 0)$. Hence $(\mathfrak{A}, q_0, \{q_0\})$ accepts w.

If $w \notin L_n$ then there exists $u \sqsubseteq w$, such that $\Delta_n(u) = \perp$ or $\Delta_n(w) > 0$. In the first case pick u minimal with $\Delta_n(u) = \perp$. Then u is nonempty and we can write $u = u' \cdot \sigma$. We have either $\Delta_n(u') = n$ and $\sigma = a$ or $\Delta_n(u') = 0$ and $\sigma = b$. In both cases $\mathfrak{R}_\perp$ will change its state to 1. Since $\mathfrak{R}_n$ is biased, q_0 is not reachable from $q_0\overline{u}$ and therefore $(\mathfrak{A}, q_0, \{q_0\})$ rejects w. In the second case, i.e. $\Delta_n(w) = k > 0$, we see by the previous lemma (w can be completed to $w' = wb^k \in L_n$) that the automaton ends in a state different from q_0 and hence the automaton also rejects w. □

Remark 4.3.1. The product constructed has scope n.

We have seen that we can define a cascade product of minimal length for L_n in a very natural way, such that this cascade product has scope n.

It should be mentioned that there exists a connection between the length of a cascade product of resets, which recognizes L, and the dot-depth of L. This connection was found by Cohen and Brzozowski in their very first paper on the dot-depth of star-free languages:

Theorem 4.3.9 (Cohen, Brzozowski, [CB71])**.** *Let L be star-free and $\mathfrak{R}_1 \circ \cdots \circ \mathfrak{R}_n$ a cascade product of resets, which recognizes L. Then L has dot-depth at most $n + 1$.*

The preceding observations show, that the family $(L_n)_{n\in\mathbb{N}}$ (defined by (4.3.1)) matches this bound up to a constant term. Unfortunately this is not always the case:

Proposition 4.3.10. *For every $n \in \mathbb{N}$ there exists $L_n \in \mathcal{V}_1$, such that every cascade product of resets has at least n factors.*

Proof. We set $L_n = (\Sigma^*\sigma\Sigma^*)^n$ for some $\sigma \in \Sigma$. In analogy to Corollary 4.3.6 one concludes from Lemma 4.3.5, that every cascade product of resets recognizing L_n has at least n factors. L_n is obviously piecewise testable and hence in $\mathcal{V}_1$. □

4.3.3 A Tradeoff between Length and Scope

In trying to find cascade products with minimal scope for a given language L, we may have to increase the number of factors. This is illustrated by the following example:

Example 4.3.1. We already know that all piecewise testable languages can be recognized by a scope 1 cascade product of resets. Those products will generally not be minimal. For instance, let $\Sigma = \{\sigma_1, \ldots, \sigma_n, b\}$ and consider the commutative language

$$K = \{w \in \Sigma^* \mid |w|_{\sigma_1} > 0, \ldots, |w|_{\sigma_n} > 0\}$$

Then define $L = K \cdot b\Sigma^*$. An easy way to recognize L is to define 1-biased resets $\mathfrak{R}_i = (\mathbb{B}, \Sigma, \delta^{R_i})$ by $q\overline{\sigma}^{R_i} = 1$ iff $\sigma = \sigma_i$. Then $\mathfrak{R}_1 \times \cdots \times \mathfrak{R}_n$ recognizes K. Define a 1-biased reset $\mathfrak{R} = (\mathbb{B}, \Sigma \times \mathbb{B}^n, \delta^R)$ by $q\overline{(x,\sigma)}^R = 1$ iff $x = (1, \ldots, 1)$ and $\sigma = b$. Obviously $\mathfrak{R}_1 \times \cdots \times \mathfrak{R}_n \circ \mathfrak{R}$ recognizes L and has scope n.

If we now want to recognize L with a scope 1 product, we can proceed as follows: For every permutation $\pi \in S_n$ we set $L_\pi := \Sigma^*\sigma_{(1)\pi}\Sigma^* \cdots \Sigma^*\sigma_{(n)\pi}\Sigma^* b\Sigma^* \subseteq L$. Furthermore, for every $w \in L$, there exists $\pi \in S_n$, such that $w \in L_\pi$. Then $L = \bigcup_{\pi \in S_n} L_\pi$. We know how to find scope 1 products $\mathfrak{A}_\pi$ recognizing L_π for every $\pi \in S_n$. Hence L is recognized by $\mathfrak{B} := \bigtimes_{\pi \in S_n} \mathfrak{A}_\pi$. For every $\pi \in S_n$ we can construct $\mathfrak{A}_\pi$ in such a way, that it has exactly $n+1$ factors. Hence $\mathfrak{B}$ has $n! \cdot (n+1) = (n+1)!$ factors.

We conjecture that any scope 1 product of resets, which recognizes a language of the form $\Sigma^*\sigma_1\Sigma^* \cdots \Sigma^*\sigma_n\Sigma^*$ has at least n factors. However, we cannot prove this statement yet. If one accepts this conjecture as being true, one can at least conclude that if the language L from the previous example is recognized by a scope 1 cascade product of resets, then this product cannot be constructed in the same way as was done in the example. In other words, taking the union over all permutations will not result in a shorter scope 1 product.

The statement of the conjecture cannot be weakened, as the next example shows:

Example 4.3.2. Let $\Sigma = \{a, b\}$ and let $L = \Sigma^* a\Sigma^* b\Sigma^* a\Sigma^* b\Sigma^*$. We will define a cascade product of resets with three factors, which recognizes L:

$$\mathfrak{A} := \mathfrak{R}_b^a \circ \mathfrak{R}^{(b,1)} \circ \mathfrak{R}^{(1,1,b)} \tag{4.3.2}$$

The superscripts in equation (4.3.2) are to be understood as the (unique) input inducing a mapping with constant value 1. Likewise the subscripts denote the unique input, which induces a constant 0 mapping. All the inputs, which appear neither as sub- nor superscript induce the identity mapping.

We claim that the product defined in (4.3.2) recognizes L, namely

$$L = L(\mathfrak{A}, (0,0,0), \{(x_1, \ldots, x_3) \in \mathbb{B}^4 | x_3 = 1\})$$

To this end, we consider a word $w \in L$. We may write $w = u_1 \cdot a \cdot u_2 \cdot b \cdot u_3 \cdot a \cdot u_4 \cdot b \cdot u_5$ with $u_1 \in b^*$, $u_2 \in a^*$, $u_3 \in b^*$, $u_4 \in a^*$ and $u_5 \in \Sigma^*$. Consider the run ρ of $\mathfrak{A}$ on w. Then $\rho_{|u_1|+i} = (1,0,0)$ for all $i \in \{1, \ldots, 1+|u_2|\}$. Thus $\rho_{|u_1|+|u_2|+1+i} = (0,1,0)$ for all $i \in \{1, \ldots, 1+|u_3|\}$. From this we get $\rho_{r+i} = (1,1,0)$, where $r = |u_1| + |u_2| + |u_3| + 2$ and $i \in \{1, \ldots, 1+|u_4|\}$. If $r' = r+|u_4|+1$ then finally $\rho_{r'+i} = (*,1,1)$ for all $i \in \{1, \ldots, 1+|u_5|\}$, where $*$ denotes an arbitrary entry.

Conversely, if $w \in \Sigma^*$ is recognized locally by $\mathfrak{A}$ (with initial state $(0,0,0)$ and final bit 1), then there exists a shortest prefix $u \sqsubseteq w$, such that u is accepted by the automaton. Then the last letter of u must be a b. Write $u = v \cdot b$ and observe that the run ρ of $\mathfrak{A}$ on v must end in a state $(1,1,0)$. By (4.3.2) this can only happen if the state $(0,1,0)$ was seen at some time earlier in the run.

5 Conclusion and Future Work

5.1 Summary

In this thesis we considered the Krohn-Rhodes Theorem in different formulations and its applications in formal language theory. In Chapter 3 we first gave an overview of the Krohn-Rhodes Theorem, one stated in terms of the cascade product on semiautomata, one in terms of the wreath product on transformation semigroups and one in terms of the block product on semigroups. In a comparison of these three theorems, it was shown that the cascade product version and the wreath product version can be derived directly from one another. It was also illustrated that a version using the block product can be derived from the wreath product version of the theorem. Finally we mentioned that there exists a stronger statement in terms of the block product than the one we derived. In doing so, we observed that even with the inherently weaker left-to-right bracketing convention used, we can still obtain a decomposition result (Proposition 3.3.10).

In Chapter 4 we then proceeded to classify classes of regular languages in terms of the decomposition obtained from the Krohn-Rhodes Theorem. To this end we introduced the concepts of biased resets, locally i-triggered cascade products and one-letter-automata. We showed how to characterize:

- $\mathcal{R}$-trivial languages using biased resets (Theorem 4.1.10)
- piecewise testable languages using locally i-triggered cascade products (Theorem 4.1.7)
- commutative languages using one letter automata (Theorem 4.2.8)

Finally we introduced the notion of scope of resets within a cascade product as an attempt to further refine the cascade product classification. We investigated which classes of regular languages can be recognized with cascade products of constant scope. Initial results show that all piecewise testable languages can be recognized by a scope 1 cascade product (Proposition 4.3.1) and that all $\mathcal{R}$-trivial languages can be recognized by a scope 2 cascade product (Theorem 4.3.4). Subsequently, we showed that for a certain family of languages $(L_n)_{n\in\mathbb{N}}$ separating the levels of the dot-depth hierarchy we have a natural correspondence between the dot-depth and the scope of a minimal cascade product recognizing this language. Finally we illustrated how decreasing the scope of a cascade product may lead to a blowup in the number of factors.

5.2 Future Work

The major focus of this thesis was, with the exception of commutative languages, on star-free languages. An interesting question would be the characterization of other, more general classes of regular languages, such as languages definable in the logic $\mathrm{FO} + \mathsf{MOD}$, i.e. first order logic with modular counting.

Another interesting point of future research is a closer study of the scope of cascade products. We have not yet established how expressive scope k products are. Denote by $\mathcal{SC}_k$ the class of languages, which are recognizable by a cascade product of scope k. Do we have $\mathcal{SC}_k \subsetneq \mathcal{SC}_{k+1}$ for all $k \in \mathbb{N}$?

Finally we would like to mention the possibility of lifting the theory to the class of ω-regular languages. Since a cascade product covering a semiautomaton is independent of the acceptance condition, we can construct a cascade product covering the transition graph of any deterministic Muller automaton. Using the homomorphism from this covering, we can identify accepting runs in both automata, thereby specifying an acceptance condition on the cascade product. The question arises, which types of ω-regular languages we can characterize in this way.

Bibliography

[BF80] J. A. Brzozowski and Faith E. Fich. Languages of J-trivial monoids. *Journal of Computer and System Sciences*, 20(1):32 – 49, 1980. iii, 38, 43, 44

[BS71] Janusz A. Brzozowski and Imre Simon. Characterizations of locally testable events. *Foundations of Computer Science, Annual IEEE Symposium on*, 0:166–176, 1971. 1, 4

[CB71] Rina S. Cohen and Janusz A. Brzozowski. Dot-depth of star-free events. *Journal of Computer and System Sciences*, 5(1):1 – 16, 1971. 1, 39, 54, 59

[Eil74a] Samuel Eilenberg. *Automata, Languages, and Machines: Volume A*. Pure and Applied Mathematics. Elsevier, Burlington, MA, 1974. 1

[Eil74b] Samuel Eilenberg. *Automata, Languages, and Machines: Volume B*. Pure and Applied Mathematics. Elsevier, Burlington, MA, 1974. 1, 18, 19

[Gin68] Abraham Ginzburg. *Algebraic theory of automata*. Academic Press, New York, 1968. 1, 3, 17, 18, 21, 23, 29, 34, 51

[HMU01] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison Wesley, 2001. 15, 16, 19

[How95] John M. Howie. *Fundamentals of Semigroup Theory, Volume 12 of London Mathematical Society Monographs. New Series*. The Clarendon Press Oxford University Press, New York, 1995. 12, 13

[KR65] Kenneth Krohn and John Rhodes. Algebraic theory of machines. I. Prime decomposition theorem for finite semigroups and machines. *Trans. Amer. Math. Soc*, 116:450–464, 1965. 1

[KS04] Hans Kurzweil and Bernd Stellmacher. *The Theory of Finite Groups: An Introduction*. Springer Verlag, 2004. 11

[Mey69] Albert R. Meyer. A note on star-free events. *J. ACM*, 16(2):220–225, 1969. 1

[Pin86] Jean-Eric Pin. *Varieties Of Formal Languages*. Plenum Publishing Co., 1986. 1, 4, 13, 18, 20, 21, 43, 44, 54

[Pin95] Jean-Eric Pin. Finite semigroups and recognizable languages: An introduction. In John Fountain, editor, *Semigroups, Formal Languages and Groups*. 1995. 1

[Pin97] Jean-Eric Pin. Syntactic semigroups. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages, Vol. 1: word, language, grammar*, pages 679–746. Springer-Verlag New York, Inc., New York, NY, USA, 1997. 1, 13

[Sch65] Marcel-Paul Schützenberger. On finite monoids having only trivial subgroups. *Information and Control*, 8(2):190 – 194, 1965. 1, 18

[Sim75] Imre Simon. Piecewise testable events. In *Proceedings of the 2nd GI Conference on Automata Theory and Formal Languages*, pages 214–222, London, UK, 1975. Springer-Verlag. 1, 4, 20

[ST88] Howard Straubing and Denis Thérien. Partially ordered finite monoids and a theorem of I. Simon. *J. Algebra*, 119(2):393–399, 1988. iii, 42

[ST02] Howard Straubing and Denis Thérien. Weakly iterated block products of finite monoids. In *LATIN '02: Proceedings of the 5th Latin American Symposium on Theoretical Informatics*, pages 91–104, London, UK, 2002. Springer-Verlag. 1, 36

[Str80] Howard Straubing. On finite J-trivial monoids. *Semigroup Forum*, 19:107–110, 1980. iii, 42

[Str89] Howard Straubing. The wreath product and its applications. In *Proceedings of the LITP Spring School on Theoretical Computer Science*, pages 15–24, London, UK, 1989. Springer-Verlag. 1, 3, 4, 45

[Str94] Howard Straubing. *Finite Automata, Formal Logic, and Circuit Complexity.* Birkhauser Verlag, Basel, Switzerland, Switzerland, 1994. 1, 3, 13, 18, 19, 24, 32, 36, 37

[STT88] Howard Straubing, Denis Thérien, and Wolfgang Thomas. Regular languages defined with generalized quantifiers. In Timo Lepistö and Arto Salomaa, editors, *Automata, Languages and Programming*, volume 317 of *Lecture Notes in Computer Science*, pages 561–575. Springer Berlin / Heidelberg, 1988. 1

[Tho87] Wolfgang Thomas. A concatenation game and the dot-depth hierarchy. *Computation theory and logic*, pages 415–426, 1987. 57

[Wei89] Pascal Weil. Concatenation product: a survey. In J. Pin, editor, *Formal Properties of Finite Automata and Applications*, volume 386 of *Lecture Notes in Computer Science*, pages 120–137. Springer Berlin / Heidelberg, 1989. 1