RWTH Aachen University, Germany
Chair of Computer Science 7 - Logic and Theory of Discrete Systems
Prof. Dr. Dr.h.c. Wolfgang Thomas

Bachelor Thesis

# Finite Automata over Infinite Alphabets

Sarah Winter

May 2011

Hiermit versichere ich, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

Aachen, den 20. Mai 2011

## Abstract

We present an automaton model that recognizes relations of finite or infinite words over infinite alphabets. We show that the relations recognized by our automaton model are monadic second-order definable and show a general equivalence between monadic second-order definability and recognizability in case of infinite alphabets.

From this result we obtain a new application to monadic chain logic. We provide an extension to chain logic over infinitely branching trees by additionally allowing monadic second-order definable connections between siblings.

## Kurzfassung

Wir präsentieren ein Automatenmodel welches Relationen von endlichen oder unendlichen Wörten über unendlichen Alphabeten erkennt. Wir zeigen, dass die von unserem Automatenmodel erkannten Relation definierbar sind in monadischer Logik zweiter Stufe, sowie die generelle Äquivalenz zwischen MSO Definierbarkeit und Erkennbarkeit im Fall von unendlichen Alphabeten.

Unsere Ergebnisse ermöglichen uns eine neue Anwendung der Kettenlogik. Wir erweitern diese über unendlich verzweigte Bäume durch MSO-definierbare Verbindungen von Geschwisterknoten.

# Contents

# Chapter 1

# Introduction

An equivalence between languages recognized by finite automata and monadic second-order logic over finite words was found in the early 1960s. Büchi [Büc60], Elgot [Elg61], and Trakhtenbrot [Tra62] proved that monadic second-order logic definable languages coincide with regular languages. Büchi [Büc62] showed that an equivalence exists between finite automata and monadic second-order logic over infinite words and Rabin [Rab69] showed such an equivalence between finite automata and monadic second-order logic over infinite trees. This is the essential background result for this work. Rabin's tree theorem states that the monadic second-order theory of the infinite binary tree is decidable.

In applications more restricted logics are of interest. Over trees, path logic and chain logic are fragments of monadic second-order logic in which set quantifications are restricted to paths and chains, i.e. subsets of paths, respectively. These logics were first studied in [Tho87]. The restriction allows to extend the expressibility beyond successors. Adding the binary equal level predicate allows to connect vertices of the same tree level. It is well known that extending monadic second-order logic over the infinite binary tree by the equal level predicate is undecidable, see [Tho90], whereas adding the equal level predicate to chain logic results in a decidable extension of chain logic. This was shown in [Tho92]. The proof provides a reduction of the chain theory with equal level predicate of the infinite binary tree to the monadic second-order theory S1S of one successor which was already shown to be decidable by Büchi in [Büc62].

This thesis is concerned with an extension of this result in the following way. We consider decidability of chain logic over infinitely branching trees and additionally allow monadic second-order definable connections between siblings, i.e. the children of a parent node. As a background for this application we define an automaton model that recognizes relations of words over infinite alphabets. The new automaton model is defined as a finite synchronous multitape automaton with logical formulas as transition constraints. A similar automaton model was introduced in [Bès08]. Bès also defined synchronous multitape automata over infinite alphabets, but the transition constraints are restricted to first-order formulas only. We will show that the relations recognized by our automaton model are monadic second-order definable and show a general equivalence between monadic second-order definability and recognizability in case of infinite alphabets. Here we work in an abstract framework in which first-order logic (as in Bès) is replaced by any logic $\mathfrak{L}$ speaking over elements of the infinite alphabet.

This thesis is structured as follows. In the next chapter we introduce basic notations from mathematical logic and give a short introduction to first-order logic and some of its extensions, which we use in the subsequent chapters of this thesis.

In Chapter 3 we fix notations regarding finite automata. We introduce a new automaton model, denoted $\mathfrak{M}$-$\mathfrak{L}$-automaton when the alphabet structure is $\mathfrak{M}$ and the considered logic is $\mathfrak{L}$, which recognizes relations of words over both finite and infinite alphabets. We prove that the class of relations recognized by the automaton model is closed under Boolean operations and projection. Furthermore we investigate whether the decision problems are decidable.

In Chapter 4 we present a logical characterization of relations recognized by the automaton model using monadic second-order logic. We show the equivalence of monadic second-order logic interpreted over words over infinite alphabets and finite automaton recognizability.

In Chapter 5 we extend the results of the previous two chapters to the case of infinite words. First we introduce an automaton model that recognizes relations of infinite words over an infinite alphabet. We show that the equivalence of monadic second-order logic definability and recognizability is maintained over infinite words. Additionally, we provide a translation from this automaton model into an equivalent deterministic automaton model.

In Chapter 6 we present an application to monadic chain logic over infinitely branching trees with the equal level predicate. We consider an extension to chain logic by adding features that allow monadic second-order definable statements over the children of a parent node.

Finally, in the last chapter we summarize our results and give an outlook on further possible extensions based on the work of the previous chapter.

## Acknowledgments

I would like to thank Prof. Wolfgang Thomas for supervising and guiding me in writing this thesis.

I would also like to express my gratitude to Alex Spelten for her valuable comments and advice.

Also, I would like to thank Dr. Christof Löding for reviewing my thesis.

# Chapter 2

# Logical Formalisms

In this chapter we want to give an overview over first-order logic and some extensions. For a detailed introduction to mathematical logic the reader is referred to [EFT07].

## 2.1 Structures

**Definition 2.1** A *signature* $\tau$ is a set of relation and function symbols, where each symbol has a positive fixed finite arity. A function symbol with arity 0 is called constant symbol. A signature is called *relational* if it consists entirely of relation symbols.

In general, relation symbols are denoted by $P, Q, R, \ldots$, function symbols by $f, g, h, \ldots$. A mathematical structure consists of a non-empty domain and a set of finitary relations and functions on this domain.

**Definition 2.2** A *$\tau$-structure* $\mathfrak{M}$ is of the form $\mathfrak{M} = (\Sigma, R_1^{\mathfrak{M}}, \ldots, f_1^{\mathfrak{M}}, \ldots)$ where $\Sigma$ is a *non-empty* set, called *domain* of $\mathfrak{M}$ and

- for each $n$-ary relation symbol $R$ in $\tau$ $R^{\mathfrak{M}}$ is an $n$-ary relation on $\Sigma$,
- $f^{\mathfrak{M}}$ is an $n$-ary function on $\Sigma$ for each $n$-ary function symbol $f$ in $\tau$.

The relations and functions are the interpretations of the relation and function symbols in $\tau$. If the interpretation is clear, we also denote a $\tau$-structure $\mathfrak{M} = (\Sigma, R_1^{\mathfrak{M}}, \ldots, f_1^{\mathfrak{M}}, \ldots)$ by $\mathfrak{M} = (\Sigma, R_1 \ldots, f_1 \ldots)$.

A *relational structure* is a $\tau$-structure, where $\tau$ consists entirely of relation symbols.

In the next chapters, we will limit ourselves to relational structures. However, this is no restriction, because every structure has an equivalent relational variant, which we obtain in the following way:

**Definition 2.3** Given a $\tau$-structure $\mathfrak{M}$. We associate to $\mathfrak{M}$ a relational $\tau^r$-structure $\mathfrak{M}^r$ with domain $\Sigma^r = \Sigma$ and each $n$-ary function $f^{\mathfrak{M}} : \Sigma^n \to \Sigma$ is replaced by its graph $F^{\mathfrak{M}^r} := \{(a_1, \ldots, a_n, a_{n+1}) \in \Sigma^{n+1} : f^{\mathfrak{M}}(a_1, \ldots, a_n) = a_{n+1}\}$.

## 2.2 First-Order Logic

We now define the syntax and semantics of *first-order logic (FO)*. A first-order language is determined by a set of logical symbols and a fixed signature $\tau$.

**Definition 2.4** Let $\tau$ be a signature. We define the alphabet $\Sigma_\tau$ of a first-order language as follows:

- the relation and function symbols in $\tau$,
- a countably infinite set of variables $x, y, \ldots, x_1, \ldots$,
- the logical connectives $\neg, \wedge, \vee$, and $\rightarrow$,
- an equality symbol $=$,
- the quantifier symbols $\forall$ and $\exists$,
- bracket symbols (,).

Over this alphabet, FO-terms and FO-formulas can be constructed inductively.

**Definition 2.5** The $\tau$*-terms* are words over the alphabet $\Sigma_\tau$ which are inductively defined as follows:

(1) Any variable is a $\tau$-term.

(2) Let $t_1, \ldots, t_n$ be $\tau$-terms and $f$ a $n$-ary function symbol in $\tau$, then $f(t_1, \ldots, t_n)$ is a $\tau$-term

**Definition 2.6** The $\tau$*-formulas* in first-order logic are inductively defined as follows:

(1) If $t_1, t_2$ are $\tau$-terms, then $t_1 = t_2$ is a $\tau$-formula.

(2) If $t_1, \ldots, t_n$ are $\tau$-terms and $P$ is an $n$-ary relation symbol, then $P(t_1, \ldots, t_n)$ is a $\tau$-formula.

(3) If $\varphi$ is a $\tau$-formula, then $\neg\varphi$ is a $\tau$-formula.

(4) If $\varphi$ and $\psi$ are $\tau$-formulas, then $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$ and $(\varphi \rightarrow \psi)$ are $\tau$-formulas.

(5) If $\varphi$ is a $\tau$-formula and $x$ is a variable, then $\exists x\varphi$ and $\forall x\varphi$ are $\tau$-formulas.

A formula obtained from rules 1 and 2 is said to be atomic.

Usually it is more convenient to use infix notation for binary relation and function symbols, e.g. we write $x + y$ instead of $+(x, y)$. Bracket symbols can also be omitted if it increases the readability of a formula.

An *occurrence* of a variable $x$ in a formula $\varphi$ can either be *free* or *bounded* by a quantifier. It is bounded if $x$ occurs in a subformula of the form $\exists x\varphi$ or $\forall x\varphi$, otherwise the occurrence is free. The notation $\varphi(x_1, \dots, x_n)$ indicates that in the formula $\varphi$ at most the variables $x_1, \dots, x_n$ occur free. A *sentence* is a formula without free variables.

**Definition 2.7** We define the *satisfaction relation* $\models$ between a $\tau$-structure $\mathfrak{M}$ and a $\tau$-formula $\varphi$ as usual with the standard semantics for the connectives $\neg, \wedge, \vee$ and the quantifiers $\forall, \exists$. Let $\mathfrak{M}$ denote a $\tau$-structure with domain $\Sigma$ and let $\varphi(x_1, \dots, x_n)$ be a first-order $\tau$-formula with at most free variables $x_1, \dots, x_n$. We write

$$(\mathfrak{M}, k_1, \dots, k_n) \models \varphi(x_1, \dots, x_n)$$

with $k_1, \dots, k_n \in \Sigma$ if $\varphi$ is true in this semantics for the assignment of $x_i = k_i$ for $1 \le i \le n$. We also write $\mathfrak{M} \models \varphi[k_1, \dots, k_n]$.

**Definition 2.8** Let $\mathfrak{M}$ be a $\tau$-structure. The *first-order theory* of the structure $\mathfrak{M}$ is the set of all $\tau$-sentences that are satisfied in $\mathfrak{M}$. It is denoted by FO-Th($\mathfrak{M}$).

The FO-theory of a $\tau$-structure $\mathfrak{M}$ is *decidable* if and only if it is decidable for every sentence $\varphi$ whether $\mathfrak{M} \models \varphi$.

## 2.3 Extensions of First-Order Logic

In this section we mention some extensions of first-order logic.

*Monadic second-order* (MSO) logic is an extension to first-order logic by second order variables $X, Y, \dots, X_1, \dots$ which range over sets of elements. Additional atomic formulas are $X(x)$ with the intended meaning "x is an element of the set X".

*Weak monadic second order* (WMSO) logic is MSO logic where set quantifications are restricted to finite sets.

For the subsequent two extensions of FO logic, see [EF95] from which the following definitions are taken. Let $k \ge 1$ and $R$ is a $2k$-ary relation on a set $\Sigma$. The *transitive closure* TC($R$) is defined by

$$\begin{aligned}\mathrm{TC}(R) := \{(\overline{a}, \overline{b} \in \Sigma^k \times \Sigma^k) \mid \text{there exists } n > 0 \text{ and } \overline{e}_0, \dots, \overline{e}_n \in \Sigma^k \text{ such that} \\ \overline{a} = \overline{e}_0, \overline{b} = \overline{e}_n, \text{ and for all } i < n,\ (\overline{e}_i, \overline{e}_{i+1}) \in R\ \}.\end{aligned}$$

*Transitive Closure Logic* FO(TC) is obtained by closing FO under the transitive closure of definable relations. The class of *transitive closure formulas* is given by the usual rules for building first-order formulas and the new rule:

$[\mathrm{TC}_{\overline{x},\overline{y}}\varphi]_{\overline{s},\overline{t}}$ where the variables in $\overline{x},\overline{y}$ are pairwise disjoint and where the tuples $\overline{x},\overline{y},\overline{s}$, and $\overline{t}$ are of the same length, and $\overline{s},\overline{t}$ are tuples of terms.

The meaning of $[\mathrm{TC}_{\overline{x},\overline{y}}\varphi]_{\overline{s},\overline{t}}$ is given by $(\overline{s},\overline{t}) \in \mathrm{TC}(\{(\overline{x},\overline{y} \mid \varphi(\overline{x},\overline{y})\})$.

*First-order logic with counting quantifiers*, in short FO(C) is an extension of first-order logic by adding, for every $l \geq 1$, a new quantifier $\exists^{\geq l}$ with the intended meaning "there are at least $l$". A further restriction is the logic $\mathrm{FO(C)}^s$ which is a fragment of FO(C) with variables restricted to $x_1,\dots,x_s$.

## 2.4 Decidability Results

In this section we recall some well-known decidability results of first-order and monadic-second order theories.

A result basically due to Gödel [Göd31] is that the first-order theory of arithmetic $(\mathbb{N},+,\cdot)$ is undecidable, whereas two important fragments of the arithmetic are decidable. It is known that the first-order theory of $(\mathbb{N},+)$, often called Presburger arithmetic, is decidable [Pre29] and the first-order theory of $(\mathbb{N},\cdot)$, often called Skolem arithmetic, is decidable [Sko31].

The monadic second-order theory of $(\mathbb{N},+)$ is undecidable. Considering the weaker structure $(\mathbb{N},+1)$, it was proven by Büchi [Büc62] that the monadic second-order theory of $(\mathbb{N},+1)$, often called S1S (second-order theory of one successor), is decidable. Derived from this result were also the decidability of WS1S, i.e. the weak monadic second order theory of $(\mathbb{N},+1)$ and the decidability of the fist-order theory of $(\mathbb{R},+)$.

Additionally, the first-order theory of $(\mathbb{R},+,\cdot)$ is decidable, which has been proven in [Tar48] but not the monadic second-order theory of $(\mathbb{R},+,\cdot)$. This result goes back to Gödel.

# Chapter 3

# Automata and $\mathfrak{M}$-$\mathfrak{L}$-Automata

In this section, we introduce a new automaton model, called $\mathfrak{M}$-$\mathfrak{L}$-automaton, that recognizes relations of words over both finite and infinite alphabets, called $\mathfrak{M}$-$\mathfrak{L}$-recognizable relations. In $\mathfrak{M}$-$\mathfrak{L}$-automata, we specify the alphabet as the domain of a structure $\mathfrak{M}$ and use logical formulas in the language of a logic $\mathfrak{L}$ speaking over the structure $\mathfrak{M}$ to set constraints to the transitions. Furthermore, we prove some properties of $\mathfrak{M}$-$\mathfrak{L}$-recognizable relations. Beforehand, we recall some basic notions of finite automata theory.

## 3.1 Notation

An *alphabet* is a finite or infinite non-empty set $\Sigma$ of *symbols* or *letters*. A *word* over an alphabet $\Sigma$ is a finite sequence of symbols $a_1 \dots a_n$ with $a_i \in \Sigma$ for $1 \leq i \leq n$. Finite words are usually denoted by $u, v, w, \dots$ . We denote by $\Sigma^*$ (resp. $\Sigma^+$) the set of finite words (resp. finite non-empty words) over $\Sigma$. Given a word $w \in \Sigma^*$, we denote by $|w|$ the length of $w$. The empty word $\varepsilon$ has the length 0. A *language* $L$ over an alphabet $\Sigma$ is a subset of $\Sigma^*$.

We now define operations on words and languages for both finite and infinite alphabets. The *concatenation* of two words $u$ and $v$ is the word $u \cdot v$. Let $U \subseteq \Sigma^*$ and $V \subseteq \Sigma^*$ denote languages. The *concatenation* of $U$ and $V$ is the set $U \cdot V := \{u \cdot v \mid u \in U, v \in V\}$. The operator $\cdot$ is usually omitted. Let $L \subseteq \Sigma^*$ denote a language, given $L^0 = \{\varepsilon\}$, we define recursively the language $L^n = LL^{n-1}$. The *Kleene closure* of $L$ is the language $L^* := \bigcup_{n \geq 0} L_n$. Furthermore, the *intersection* of $U$ and $V$ is the set $U \cap V := \{w \mid w \in U, w \in V\}$ and the *complement* of $U$ is the set $\Sigma^* \setminus U$, often denoted by $\overline{U}$.

## 3.2 Finite Automata over Finite Alphabets

A language $L \subseteq \Sigma^*$ over a finite alphabet $\Sigma$ is said to be *regular* if $L$ is the language recognized by a non-deterministic finite automaton.

**Definition 3.1** A *non-deterministic finite automaton (NFA)* over an alphabet $\Sigma$ is of the form $\mathcal{A} = (Q, \Sigma, \Delta, q_0, F)$, where $Q$ is a finite set of states, $\Sigma$ is a finite alphabet, $\Delta \subseteq Q \times \Sigma \times Q$ is the transition relation, $q_0 \in Q$ is the initial state, and $F \subseteq Q$ is the set of final states. A *run* of $\mathcal{A}$ on a word $w = a_1 \ldots a_n$ is a sequence of states $\rho = \rho(0) \ldots \rho(|w|)$, such that $\rho(0) = q_0$ and for every $i$, with $0 \leq i < |w|$, $(\rho(i), a_{i+1}, \rho(i+1)) \in \Delta$. The run is *successful* if $\rho(|w|)$ is a final state. A word $w \in \Sigma^*$ is *accepted* by $\mathcal{A}$ if and only if there is a successful run of $\mathcal{A}$ on $w$. The language *recognized* by $\mathcal{A}$ is $L(\mathcal{A}) = \{w \in \Sigma^* \mid \mathcal{A} \text{ accepts } w\}$.

An extension to regular languages $L \subseteq \Sigma^*$ are *regular* or *automatic relations* of arity $n$ of words over finite alphabets, for $n \geq 1$. A *finite synchronous $n$-tape automaton* reads finite words over $\Sigma$ written on $n$-input tapes (cf. [EES69]). The automaton simultaneously reads the words on each input tape, i.e. the heads move at the same speed, processing one symbol per computation step. Hence, all words need to have the same length. Therefore we use a padding symbol in order to extend components if necessary.

**Definition 3.2** The *convolution of a tuple* $w = (u_1, \ldots, u_n) \in (\Sigma^*)^n$ with $u_i = u_{i1} \ldots u_{ik_i}$ is defined as $\langle w \rangle = \langle u_1, \ldots, u_n \rangle$ with $\ell = max\{|u_1|, \ldots, |u_n|\}$ as follows:

$$\langle w \rangle := \begin{bmatrix} u'_{11} \\ \vdots \\ u'_{n1} \end{bmatrix} \cdots \begin{bmatrix} u'_{1\ell} \\ \vdots \\ u'_{n\ell} \end{bmatrix} \in ((\Sigma \,\dot{\cup} \{\#\})^n)^*,$$

where $u'_{ij} = u_{ij}$ for $j \leq |u_i|$, otherwise $u'_{ij} = \#$. Note, that # must be a padding symbol not belonging to $\Sigma$. We will write $\Sigma_\#$ for $\Sigma \,\dot{\cup} \{\#\}$. Every $\langle w \rangle$ is word over the alphabet $(\Sigma_\#)^n$.

We define the *convolution of a relation* $R \subseteq (\Sigma^*)^n$ to be the language $L_R := \{\langle w \rangle \mid w \in R\}$.

Note, that the convolution of a language, i.e. a relation with arity 1, is the language itself.

**Definition 3.3** A *non-deterministic finite synchronous $n$-tape automaton* over an alphabet $\Sigma$ is a non-deterministic finite automaton over the alphabet $(\Sigma_\#)^n$. A relation $R \subseteq (\Sigma^*)^n$ is called *automatic* if its convolution $L_R$ is recognizable by a non-deterministic finite synchronous $n$-tape automaton.

## 3.3 $\mathfrak{M}$-$\mathfrak{L}$-Automata

The automaton model, called $\mathfrak{M}$-$\mathfrak{L}$-automaton, we will now define recognizes $n$-ary relations of finite words, for $n \geq 1$ over finite as well as infinite alphabets.

For this purpose, we use synchronous $n$-tape automata in combination with logical formulas as transition constraints. Recall, that to recognize $n$-ary relations of finite words, the automaton works on the convolution of a tuple of words, i.e. in every step an $n$-tuple of elements of the alphabet is processed. Therefore a logical formula has $n$ free variables to express properties of $n$-tuples of elements of the alphabet. We specify the alphabet as the domain of a

$\tau$-structure $\mathfrak{M}$, so we can express properties of elements from the $n$-tuple using a logic $\mathfrak{L}$ speaking over the structure $\mathfrak{M}$. For ease of presentation, we call these formulas $\mathfrak{M}$-$\mathfrak{L}$-formulas. Thus we can use an $\mathfrak{M}$-$\mathfrak{L}$-formula with $n$ free variables to describe the set of $n$-tuples of elements which lead from one state to another. So, when reading the convolution of a tuple of words, in each step the $n$ symbols currently read are assigned to the $n$ free variables. If the formula is true for this assignment, the transition can be executed via this $n$-tuple.

Since $\mathfrak{M}$-$\mathfrak{L}$-automata work on the convolution of a tuple of words as introduced in Definition 3.2, the logical formulas we use must be able to deal with the padding symbol #. Therefore, we define, in addition to a $\tau$-structure $\mathfrak{M}$ with domain $\Sigma$, the $\tau_\#$-structure $\mathfrak{M}_\#$. Let $\Sigma \uplus \{\#\}$ be the domain of $\mathfrak{M}_\#$ and $\tau_\# = \tau \uplus \{P_\#\}$, where $P_\#(x)$ holds if and only if $x = \#$. Each relational symbol of $\tau$ has the same interpretation in $\mathfrak{M}_\#$ as in $\mathfrak{M}$. Although, formally we use the structure $\mathfrak{M}_\#$, we still speak of $\mathfrak{M}$-$\mathfrak{L}$-automata and $\mathfrak{M}$-$\mathfrak{L}$-formula to simplify reading.

More formally, let $\mathfrak{M}$ denote a relational structure with domain $\Sigma$ and $\mathfrak{L}$ denotes a logic, which allows quantifications over elements, e.g FO or MSO logic. An $\mathfrak{M}$-$\mathfrak{L}$-*automaton* is a non-deterministic finite synchronous $n$-tape automaton, whose transitions are of the form $(p, \varphi, q)$, where $p, q$ are states of the automaton, and $\varphi(x_1, \ldots, x_n)$ is a $\mathfrak{M}$-$\mathfrak{L}$-formula with $n$ free variables. Given an $n$-tuple of elements, a transition $(p, \varphi, q)$ can be executed if $\varphi$ is satisfied in $\mathfrak{M}_\#$ for the assignment of the $n$ elements to the $n$ free variables.

**Definition 3.4 ($\mathfrak{M}$-$\mathfrak{L}$-automaton)** Let $\Sigma$ be an alphabet, let $\mathfrak{M}$ denote a relational structure with domain $\Sigma$, and let $\mathfrak{L}$ denote a logic. An $\mathfrak{M}$-$\mathfrak{L}$-automaton is of the form $\mathcal{A} = (Q, n, \Delta, \mathfrak{M}, \mathfrak{L}, q_0, F)$ where

- $Q$ is a finite set of states,
- $n \geq 1$ is the number of tapes,
- $\Delta \subseteq Q \times \mathcal{F}_n \times Q$ is the finite set of transitions, where $\mathcal{F}_n$ is the set of $\mathfrak{M}_\#$-$\mathfrak{L}$-formulas with $n$ free variables,
- $q_0 \in Q$ is the initial state,
- $F \subseteq Q$ is the set of terminal states.

**Definition 3.5** Given an $n$-tuple $w = (u_1, \ldots, u_n)$ of finite words over $\Sigma$ and an $\mathfrak{M}$-$\mathfrak{L}$-automaton $\mathcal{A}$, a *run* of $\mathcal{A}$ on the convolution $\langle w \rangle$ of the tuple $w$ is a sequence of states $\rho = \rho(0) \ldots \rho(|\langle w \rangle|)$ with $\rho(0) = q_0$, and for every $i < |\langle w \rangle|$ there exists an $\mathfrak{M}_\#$-$\mathfrak{L}$-formula $\varphi(x_1, \ldots, x_n)$ and $(\rho(i), \varphi, \rho(i+1)) \in \Delta$, such that the following holds:

$$\mathfrak{M}_\# \models \varphi[u'_{1(i+1)}, \ldots, u'_{n(i+1)}].$$

The run is *successful* if $\rho(|\langle w \rangle|)$ is a final state. The convolution $\langle w \rangle$, which is a word over $(\Sigma_\#)^n$, is *accepted* by $\mathcal{A}$ if there is a successful run of $\mathcal{A}$ on $\langle w \rangle$. We denote by $L(\mathcal{A})$ the set of words over $(\Sigma_\#)^n$ accepted by $\mathcal{A}$.

**Definition 3.6** Let $n \geq 1$. A relation $R \subseteq (\Sigma^*)^n$ is *$\mathfrak{M}$-$\mathfrak{L}$-recognizable* or *$\mathfrak{M}$-$\mathfrak{L}$-automatic* if and only if there exists an $\mathfrak{M}$-$\mathfrak{L}$-automaton $\mathcal{A}$ with $n$ tapes such that $L_R = L(\mathcal{A})$.

**Example 3.7** At first, we consider simple examples. Let $\mathfrak{M} = (\mathbb{N}, +)$ where $+$ denotes the graph of addition and $\mathfrak{L}$ is fixed as first-order logic speaking over $\mathfrak{M}$. The following relations are $\mathfrak{M}$-$\mathfrak{L}$-recognizable:

(1) The language $L = \{w \in \mathbb{N}^* \mid$ the symbols in $w$ are alternatively even and odd, starting with an even symbol$\}$ is $\mathfrak{M}$-$\mathfrak{L}$-recognizable. Consider the $\mathfrak{M}$-$\mathfrak{L}$-automaton with two states $q_0, q_1$, where $q_0$ is initial, both states are terminal and the set of transitions is $\{(q_0, \varphi_{even}, q_1), (q_1, \varphi_{odd}, q_0)\}$ where the first-order formulas are $\varphi_{even}(x) := \exists y\ y + y = x$ and $\varphi_{odd}(x) := \neg\varphi_{even}(x)$. The automaton is depicted in Figure 3.1.

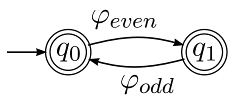


Figure 3.1: A simple $\mathfrak{M}$-$\mathfrak{L}$-automaton recognizing $L$

(2) In the previous example, we considered a language, now we consider a relation. In this case the automaton reads the convolution letter by letter, opposed to example (1). The relation $R \subseteq \mathbb{N} \times \mathbb{N}$ with $(u, v) \in R$ if there exists at least one position $i$, such that the $i$-th symbol of $u$ equals the $i$-th symbol of $v$. The following $\mathfrak{M}$-$\mathfrak{L}$-automaton recognizes $R$. The automaton has two states $q_0, q_1$, where $q_0$ is initial and $q_1$ terminal. The set of transitions is $\{(q_0, \varphi_{neq}, q_0), (q_0, \varphi_{eq}, q_1), (q_1, \varphi_t, q_1)\}$ with first-order formulas $\varphi_{eq}(x, y) := (x = y)$, $\varphi_{neq}(x, y) := (x \neq y)$ and $\varphi_t(x, y)$ expressing a statement that is always true. The automaton is depicted in Figure 3.2

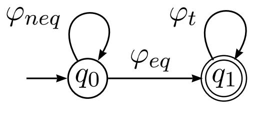


Figure 3.2: A simple $\mathfrak{M}$-$\mathfrak{L}$-automaton recognizing $R$

(3) The set of words over $\mathbb{N}$ that start with a 1 followed by arbitrarily many 0's. Consider an $\mathfrak{M}$-$\mathfrak{L}$-automaton with two states $q_0, q_1$, where $q_0$ is initial and $q_1$ terminal and two transitions $(q_0, \varphi_0, q_1), (q_0, \varphi_1, q_1)$ with first-order formulas $\varphi_0(x) := (x + x = x)$, and $\varphi_1(x) := \forall z(\neg\varphi_0(z) \to \exists y(x + y = z))$ expressing that $x = 0$, resp. $x = 1$.

**Example 3.8** Let us consider a structure with an uncountable domain. Let $\mathfrak{M} = (\mathbb{R}, +)$ and fix $\mathfrak{L} = \text{FO}$. The relation $R \subseteq \mathbb{R}^* \times \mathbb{R}^*$ defined by $(u, v) \in R$

if and only if for every $i$ the sum of the $i$-th symbol of $u$ and $v$ equals 0. The relation $R$ is recognized by an $\mathfrak{M}$-$\mathfrak{L}$-automaton with a single initial and terminal state $q$ and a single transition $(q, \varphi, q)$, where $\varphi(x, y) := \exists z(x+y = z \wedge z+z = z)$.

The following examples illustrate the limits of $\mathfrak{M}$-$\mathfrak{L}$-recognizable relations.

**Example 3.9** Consider any structure $\mathfrak{M}$ with an infinite domain $\Sigma$ and any logic $\mathfrak{L}$. Then, the relation $L = \{aa \mid a \in \Sigma\}$ is not $\mathfrak{M}$-$\mathfrak{L}$-recognizable. We prove this by contradiction. Assume $\mathcal{A} = (Q, n, \Delta, \mathfrak{M}, \mathfrak{L}, q_0, F)$ is an $\mathfrak{M}$-$\mathfrak{L}$-automaton that recognizes $L$. Since there are only finitely many transitions leading from the initial state $q_0$ to other states and infinitely many words that are accepted by $\mathcal{A}$, we can choose two distinct words $u, v \in L$, such that the runs on $u = aa$ and $v = bb$ starting from $q_0$ pass through the same state $q_1$ afterwards. This means, there exists a transition $(q_0, \varphi(x), q_1)$ such that the formula $\varphi$ is satisfied by both symbols $a$ and $b$, i.e. $\mathfrak{M}_\# \models \varphi[a]$ and $\mathfrak{M}_\# \models \varphi[b]$. Thus, we can build an accepting run on $ab \notin L$. This is a contradiction.

**Example 3.10** Let $\mathfrak{M} = (\mathbb{N}, <)$ denote a structure and let $\mathfrak{L}$ denotes any logic. Then the set of words over $\mathbb{N}$ such that each symbol is greater then the previous symbol is not $\mathfrak{M}$-$\mathfrak{L}$-recognizable. Let $L$ denote this language. Again, assume for a contradiction that the $\mathfrak{M}$-$\mathfrak{L}$-automaton $\mathcal{A}$ recognizes $L$. As is the previous example, since $L$ is infinite, we can pick two distinct words $u, v \in L$ such that the accepting runs on $u, v$ have a state in common. In particular, we can pick words $u = u_1 \dots u_\ell$ and $v = v_1 \dots v_\ell$ with $v_1 > u_2$ such that after reading $u_1$ and after reading $v_1$ the automaton is in the same state. Thus, we can build an accepting run on $v_1 u_2 \dots u_\ell \notin L$. Contradiction.

## 3.4 Closure Properties for $\mathfrak{M}$-$\mathfrak{L}$-recognizable Relations

In this section we show that the class of $\mathfrak{M}$-$\mathfrak{L}$-recognizable relations is closed under well known closure properties. As a preparation, we provide a transformation from non-deterministic to equivalent deterministic $\mathfrak{M}$-$\mathfrak{L}$-automata.

**Proposition 3.11** *A non-deterministic $\mathfrak{M}$-$\mathfrak{L}$-automaton can be transformed into an equivalent deterministic $\mathfrak{M}$-$\mathfrak{L}$-automaton.*

*Proof.* The determinization of non-deterministic $\mathfrak{M}$-$\mathfrak{L}$-automata needs more effort compared to the determinization of non-deterministic finite automata. In [Bès08], a proof is sketched. We will follow the proof idea, but provide a full proof. The construction is divided into three parts. At first, we need to find a suitable set of formulas we can use as transition labels in the complement automaton, such that only deterministic runs occur. Then, we replace the formulas in $\mathcal{A}$ with our new suitable set of formulas. Subsequently, we apply a powerset construction similar to the well-known powerset construction for non-deterministic automata on $\mathcal{A}$.

In the first step, we define a set of formulas, which we can use as transition labels in $\mathcal{A}'$. We want $\mathcal{A}'$ to be deterministic, thus the used formulas have to fulfill two properties. (a) No two formulas can be satisfied by the same symbol to obtain unambiguous runs and (b) also the set of formulas has to be complete such that in each state for each symbol there exists a transition labeled by a formula, such that the formula is satisfied by the symbol.

The desired formulas can be constructed as follows. Let $\varphi_1, \ldots, \varphi_m$ denote the formulas which appear in the transitions of $\mathcal{A}$. Consider, for every subset $J \subseteq \{1, \ldots, m\}$, the formula $\psi_J := \bigwedge_{i \in J} \varphi_i \wedge \bigwedge_{i \notin J} \neg\varphi_i$. Indeed, the demanded properties (a) and (b) are fulfilled:

(1) Consider two different sets $J$, $K \subseteq \{1, \ldots, m\}$ and corresponding formulas $\psi_J$ and $\psi_K$. Assume for a contradiction that there exists a symbol $a$ such that $\mathfrak{M}_\# \models \psi_J$ and $\mathfrak{M}_\# \models \psi_K$. Then, $\mathfrak{M}_\# \models \psi_J \wedge \psi_K$. The formulas differ at least in one member of the conjunction, say $\varphi_i$ is a subformula of $\psi_J$ and $\neg\varphi_i$ is a subformula of $\psi_J$. Hence, we have $\mathfrak{M}_\# \models \varphi_i \wedge \neg\varphi_i$ which is a contradiction.

(2) First, consider the case that there already exists a given formula $\varphi_i$ in $\mathcal{A}$ such that a symbol $a \in (\Sigma_\#)^n$ satisfies $\varphi_i$ in $\mathfrak{M}_\#$. Then, by construction of the formulas $\psi_J$ there exists a Boolean combination $\psi_J$, $i \in J$ of all formulas $\varphi_1, \ldots, \varphi_m$ such that $a$ satisfies $\psi_J$ in $\mathfrak{M}_\#$. Conversely, consider the case that a symbol $a \in (\Sigma_\#)^n$ satisfies none of the existing formulas $\varphi_i$, then $a$ satisfies the formula $\psi_J$ with $J = \emptyset$ which is a conjunction of all $\neg\varphi_i$.

The next steps are to replace the transitions in $\mathcal{A}$, such that the transition labels consist of the formulas $\psi_J$ and hereafter apply a powerset construction, similar to the classical case, which takes advantage of the newly introduced formulas. We replace every transition $(p, \varphi, q) \in \Delta$ with all transitions of the form $(p, \psi_J, q)$ where $i \in J$ and thus obtain $\mathcal{A}^\circ$ with a new transition relation $\Delta^\circ$. Let us now show, that $\mathcal{A}^\circ$ still recognizes $R$. Therefore we show that a transition from $p$ to $q$ in $\mathcal{A}^\circ$ is executable via a symbol $a$ if and only if there exists a transition from $p$ to $q$ in $\mathcal{A}$ that is executable via $a$. There are two cases we can distinguish:

(1) There exists a symbol $a \in (\Sigma_\#)^n$ such that the transition $(p, \varphi_i, q) \in \Delta$ of the unmodified automaton $\mathcal{A}$ is executable via $a$. Then $\mathfrak{M}_\# \models \varphi_i$, by construction there exists a formula $\psi_J$ where $i \in J \subseteq \{1, \ldots, m\}$ such that $\mathfrak{M}_\# \models \psi_J$ and $(p, \psi_J, q) \in \Delta^\circ$.

(2) There exists a symbol $a \in (\Sigma_\#)^n$ such that the transition $(p, \varphi_i, q) \in \Delta$ of the unmodified automaton $\mathcal{A}$ is not executable via $a$. Then $\mathfrak{M}_\# \not\models \varphi_i$, by construction every transition of the from $(p, \psi_J, q) \in \Delta'$ where $i \in J$ is not executable, because $\varphi_i$ is a subformula of $\psi_J$.

Thus, $\mathcal{A}^\circ$ still recognizes $R$.

Now we apply a powerset construction as follows. As above, let $\varphi_1, \ldots, \varphi_m$ denote the formulas which appear in the transitions of $\mathcal{A}$. Consider, for every subset $J \subseteq \{1, \ldots, m\}$, the formula $\psi_J := \bigwedge_{i \in J} \varphi_i \wedge \bigwedge_{i \notin J} \neg\varphi_i$. Let $\mathcal{A}^\circ = (Q, n, \Delta^\circ, \mathfrak{M}, \mathfrak{L}, q_0, F)$ be the $\mathfrak{M}$-$\mathfrak{L}$-automaton modified as described in the second step above. We obtain the $\mathfrak{M}$-$\mathfrak{L}$-automaton $\mathcal{A}' = (Q', n, \Delta', \mathfrak{M}, \mathfrak{L}, q_0', F')$, where

- $Q' = \mathcal{P}(Q)$ (powerset of $Q$),
- $q_0' = \{q_0\}$,
- $\Delta' \subseteq Q' \times \mathcal{F}_n \times Q'$, where $\mathcal{F}_n$ is the set of $\mathfrak{M}_\#$-$\mathfrak{L}$-formulas with $n$ free variables in which for each $P \subseteq Q$ and $J \subseteq \{1, \ldots, m\}$: $(P, \psi_J, S)$ with $S = \{s \mid p \in P : (p, \psi_J, s) \in \Delta^\circ\}$,
- $F' = \{P \subseteq Q \mid P \cap F \neq \emptyset\}$.

In the resulting deterministic $\mathfrak{M}$-$\mathfrak{L}$-automaton $\mathcal{A}'$ for every state $q$ and every formula $\psi_J$ there exists a single outgoing transition labeled $\psi_J$.

We show the equivalence between the $\mathfrak{M}$-$\mathfrak{L}$-automaton $\mathcal{A}^\circ$ and the deterministic $\mathfrak{M}$-$\mathfrak{L}$-automaton $\mathcal{A}'$.

Given a tuple $w$ of words over $\Sigma$, we show that $\mathcal{A}^\circ : q_0 \xrightarrow{\langle w \rangle} q$ if and only if $\mathcal{A}' : \{q_0\} \xrightarrow{\langle w \rangle} S,\ q \in S$ by induction over the length of $\langle w \rangle$.

If $|\langle w \rangle| = 0$, then $\langle w \rangle = \varepsilon$ and no transition is executed in $\mathcal{A}'$ nor in $\mathcal{A}^\circ$ and both automata stay in their initial states, obviously $q_0 \in \{q_0\}$.

For the induction step consider $|\langle w \rangle| = n$ with $\langle w \rangle = ua$ with $a \in (\Sigma_\#)^n$ and $u$ is a word over $(\Sigma_\#)^n$. In $A^\circ$ exists a run $q_0 \xrightarrow{\langle w \rangle} q$, i.e. a run $q_0 \xrightarrow{ua} q$. Then there exists a state $p \in Q$ such $\mathcal{A}^\circ : q_0 \xrightarrow{u} p$ and $\mathcal{A}^\circ : p \xrightarrow{a} q$. By induction hypothesis, there exists a run $\mathcal{A}' : \{q_0\} \xrightarrow{u} P,\ p \in P$. The existence of a run $p \xrightarrow{a} q$ in $\mathcal{A}^\circ$ implies the existence of a transition of the from $(p, \psi_J, q) \in \Delta^\circ$ in $\mathcal{A}^\circ$. By construction of $\Delta'$ in $\mathcal{A}'$ there exists a transition $(P, \psi_J, S) \in \Delta'$ with $q \in S$, because $(p, \psi_J, q) \in \Delta^\circ$. Thus, there exists a run $\mathcal{A}' : P \xrightarrow{a} S$. All in all, we obtain $\mathcal{A}' : \{q_0\} \xrightarrow{u} P \xrightarrow{a} S$. Therefore, there exists a run $\mathcal{A}' : \{q_0\} \xrightarrow{ua} S$ that is a run $\mathcal{A}' : \{q_0\} \xrightarrow{\langle w \rangle} S,\ q \in S$.

What is left is to prove that $L(\mathcal{A}^\circ) = L(\mathcal{A}')$. Therefore, we show for all tuples $w$ of words: $\langle w \rangle \in L(\mathcal{A}^\circ)$ iff $\langle w \rangle \in L(\mathcal{A}')$:

$$\begin{aligned}
\langle w \rangle \in L(\mathcal{A}^\circ) \quad & \text{iff } \exists q\ \mathcal{A}^\circ : q_0 \xrightarrow{\langle w \rangle} q \text{ and } q \in F \\
& \text{iff } \mathcal{A}' : \{q_0\} \xrightarrow{\langle w \rangle} S,\ q \in S \text{ and } S \in F', \text{ because } S \cap F \neq \emptyset \\
& \text{iff } \langle w \rangle \in L(\mathcal{A}')
\end{aligned}$$

□

Now we are ready to show that the class of $\mathfrak{M}$-$\mathfrak{L}$-recognizable relations is closed under well known closure properties.

**Proposition 3.12** *The class of $\mathfrak{M}$-$\mathfrak{L}$-recognizable relations is closed under*

(1) *union,* (2) *complementation,* (3) *intersection,*

(4) *projection,*

(5) *cylindrification.*

*Proof.*

(1) The closure under union is analogous to the classical construction for non-deterministic finite automata. Let $\mathcal{A}_1 = (Q_1, n, \Delta_1, \mathfrak{M}, \mathfrak{L}, q_0^1, F_1)$ and $\mathcal{A}_2 = (Q_2, n, \Delta_2, \mathfrak{M}, \mathfrak{L}, q_0^2, F_2)$ denote $\mathfrak{M}$-$\mathfrak{L}$-automata. The union automaton $\mathcal{A} = (Q, n, \Delta, \mathfrak{M}, \mathfrak{L}, q_0, F)$ merges $\mathcal{A}_1$ and $\mathcal{A}_2$ into a single automaton with $Q = Q_1 \uplus Q_2 \uplus \{q_0\}$ where $q_0$ is a new initial state, $\Delta = \Delta_1 \cup \Delta_2 \cup \{(q_0, \psi, p) \mid (q_0^1, \psi, p) \in \Delta_1 \text{ or } (q_0^2, \psi, p) \in \Delta_2\}$, and $F = F_1 \cup F_2$.

(2) For a given relation $R \subseteq (\Sigma^*)^n$ recognized by an $\mathfrak{M}$-$\mathfrak{L}$-automaton $\mathcal{A} = (Q, n, \Delta, \mathfrak{M}, \mathfrak{L}, q_0, F)$. The goal is to construct an $\mathfrak{M}$-$\mathfrak{L}$-automaton that recognizes $(\Sigma^*)^n \setminus R$. The idea is to determinize $\mathcal{A}$ and swap final states with non-final states and conversely as in the classical construction for complement automata. In Proposition 3.11, we presented a way to construct an equivalent deterministic $\mathfrak{M}$-$\mathfrak{L}$-automaton. Let $\mathcal{A}' = (Q', n, \Delta', \mathfrak{M}, \mathfrak{L}, q_0', F')$ denote the deterministic $\mathfrak{M}$-$\mathfrak{L}$-automaton. Now we replace $F'$ by $Q' \setminus F'$ in $\mathcal{A}'$ and thus obtain the complement automaton $\mathcal{A}''$ that recognizes $(\Sigma^*)^n \setminus R$.

(3) Regarding the closure under intersection, we do not give a direct construction for an $\mathfrak{M}$-$\mathfrak{L}$-automaton, but instead the closure under intersection is derived from the equivalence of $L \cap S$ to $\overline{\overline{L} \cup \overline{S}}$. The construction is possible, because we already proved the closure under union and complementation above.

(4) Projection. Let $R \subseteq (\Sigma^*)^n$ denote a relation recognized by an $\mathfrak{M}$-$\mathfrak{L}$-automaton $\mathcal{A}$. We replace every transition $(p, \varphi(x_1, \ldots, x_n), q)$ by $(p, \exists x_n \varphi(x_1, \ldots, x_n), q)$ in $\mathcal{A}$ and thus obtain an $\mathfrak{M}$-$\mathfrak{L}$-automaton $\mathcal{A}'$ that recognizes the projection of $R$ over the first $n$-1 components.

(5) Cylindrification. Let $R \subseteq (\Sigma^*)^n$ denote a relation recognized by an $\mathfrak{M}$-$\mathfrak{L}$-automaton $\mathcal{A}$ with $n$ tapes. Let $R' \subseteq (\Sigma^*)^{n+1}$ denote the cylindrification of $R$, which adds another component to $R$. It suffices to add another tape to $\mathcal{A}$, so that a head can read the new component. As the new component adds no information to $R$, the transitions remain unchanged.

□

## 3.5 Decision Problems for $\mathfrak{M}$-$\mathfrak{L}$-recognizable Relations

Regarding decision problems for $\mathfrak{M}$-$\mathfrak{L}$-recognizable relations, we shall see that the decidability of those problems is dependent on the decidability of the $\mathfrak{L}$-theory of $\mathfrak{M}$.

**Proposition 3.13** *The emptiness problem for $\mathfrak{M}$-$\mathfrak{L}$-recognizable relations is decidable if and only if the $\mathfrak{L}$-theory of $\mathfrak{M}$ is decidable.*

*Proof.* In order to decide the emptiness problem, we have to determine whether there exists a word which is the label of a successful path, i.e. a path from the initial state to a final state. Unlike in the classical case, the existence of a transition $(p, \varphi, q) \in \Delta$ in an $\mathfrak{M}$-$\mathfrak{L}$-automaton $\mathcal{A} = (Q, n, \Delta, \mathfrak{M}, \mathfrak{L}, q_0, F)$ does not necessarily imply the existence of an $n$-tuple of elements of $\Sigma_\#$ such that its elements satisfy $\varphi$ in $\mathfrak{M}_\#$. Since such transitions will never be executed by $\mathcal{A}$, they have to be removed before a reachability test can take place. Whether an $\mathfrak{M}$-$\mathfrak{L}$-formula $\varphi$ is satisfiable in $\mathfrak{M}_\#$ is decidable if and only if the $\mathfrak{L}$-theory of $\mathfrak{M}_\#$ is decidable. Since $\mathfrak{L}$-Th$(\mathfrak{M}_\#)$ reduces to $\mathfrak{L}$-Th$(\mathfrak{M})$, the emptiness problem is equivalent to the decidability of $\mathfrak{L}$-Th$(\mathfrak{M})$.

Let $\mathcal{A} = (Q, n, \Delta, \mathfrak{M}, \mathfrak{L}, q_0, F)$ be an $\mathfrak{M}$-$\mathfrak{L}$-automaton. We define the transition graph $G := (Q, E)$ where $E = \{(p, q) \in Q \times Q \mid \exists (a_1, \dots, a_n) \in (\Sigma_\#)^n : \mathfrak{M}_\# \models \varphi[a_1, \dots, a_n] \text{ and } (p, \varphi, q) \in \Delta\}$.

Thus, we have $L(\mathcal{A}) \neq \emptyset$ if and only if there is a path from $q_0$ to a final state $q$ in the transition graph $G$.

This can be checked by applying a graph search algorithm, e.g. apply depth-first search from $q_0$ in order to determine the set $Q_0$ of states reachable from $q_0$. Thus, $Q_0 \cap F \neq \emptyset$ if and only if $L(\mathcal{A}) \neq \emptyset$.

□

Regarding the examples given in the previous section, the emptiness problem was decidable for all examples.

As a consequence of Proposition 3.12 and Proposition 3.13 we get the following result.

**Proposition 3.14** *The inclusion- and the equivalence problem for $\mathfrak{M}$-$\mathfrak{L}$-recognizable relations are decidable if and only if the $\mathfrak{L}$-theory of $\mathfrak{M}$ is decidable.*

*Proof.* Given $\mathfrak{M}$-$\mathfrak{L}$-automata $\mathcal{A}$ and $\mathcal{B}$, we have to decide whether $L(\mathcal{A}) \subseteq L(\mathcal{B})$ resp. whether $L(\mathcal{A}) = L(\mathcal{B})$. Regarding the inclusion problem, $L(\mathcal{A}) \subseteq L(\mathcal{B})$ is equivalent to $L(\mathcal{A}) \cap \overline{L(\mathcal{B})} = \emptyset$. Due to the closure under union and complementation we can construct an $\mathfrak{M}$-$\mathfrak{L}$-automaton that recognizes $L(\mathcal{A}) \cap \overline{L(\mathcal{B})}$ and then apply the emptiness test, which is decidable if and only if the $\mathfrak{L}$-theory of $\mathfrak{M}$ is decidable.

Regarding the equivalence problem, $L(\mathcal{A}) = L(\mathcal{B})$ is true if and only if both $L(\mathcal{A}) \subseteq L(\mathcal{B})$ and $L(\mathcal{B}) \subseteq L(\mathcal{A})$ are true. Thus, $L(\mathcal{A}) = L(\mathcal{B})$ is decidable if and only if the $\mathfrak{L}$-theory of $\mathfrak{M}$ is decidable.

$\square$

# Chapter 4

# Logic and $\mathfrak{M}$-$\mathfrak{L}$-Automata

In this chapter, we connect $\mathfrak{M}$-$\mathfrak{L}$-automata and monadic second-order logic (MSO). Our main goal is to show that there exists a general equivalence between $\mathfrak{M}$-$\mathfrak{L}$-recognizable relations over both finite and infinite alphabets and MSO-definability.

Regarding finite alphabets, a general equivalence between languages recognized by finite automata and monadic second-order logic over words was found in the early 1960s. Büchi, Elgot and Trakhtenbrot proved that MSO-definable languages describe exactly the class of regular languages, a proof can be found in [Tho97].

**Theorem 1 ([Büc60],[Elg61],[Tra62])** *A language $L \subseteq \Sigma^+$ is regular if and only if it is MSO-definable. The transformation in both directions is effective.*

In the first section we introduce monadic second-order logic over words, suitable for finite and infinite alphabets. In the second section, we generalize the equivalence between recognizability and definability to any alphabet, finite or infinite.

## 4.1 Monadic Second-Order Logic over Words

In this section, we introduce a logical formalism that allows us to express properties of words over any alphabet, finite or infinite. A suitable logical formalism has to express relations between letter positions and properties of those letters.

In order to characterize a language over a finite or infinite alphabet $\Sigma$ by a logical formalism, we combine two logics. The logic MSO is suitable to describe relations between positions in a word, e.g. "position $x$ occurs before position $y$" or "the word is of even length". Of course, we also want to express properties that the letters on the positions should have. In MSO we can use predicates to describe properties of letters, for these predicates we make use of another logic $\mathfrak{L}$. The idea is that over any alphabet $\Sigma$, given a structure $\mathfrak{M}$ with domain $\Sigma$ and a logic $\mathfrak{L}$, we can describe properties of letters using $\mathfrak{M}$-$\mathfrak{L}$-formulas. Therefore, we introduce for any $\mathfrak{M}$-$\mathfrak{L}$-formula a new predicate that expresses "the letter at position $x$ has the property defined by the $\mathfrak{M}$-$\mathfrak{L}$-formula $\varphi$", i.e.

the symbol at position $x$ satisfies $\varphi$ in $\mathfrak{M}$. Thus, we combine the logics MSO and $\mathfrak{L}$ by adding to the MSO formalism unary predicates $P_\varphi$ for every $\mathfrak{M}$-$\mathfrak{L}$-formula $\varphi$. We call this new logical formalism $\mathfrak{M}$-$\mathfrak{L}$-MSO. We first define the syntax of $\mathfrak{M}$-$\mathfrak{L}$-MSO-formulas and then define the semantic.

**Definition 4.1 ($\mathfrak{M}$-$\mathfrak{L}$-MSO)** Given a structure $\mathfrak{M}$ with domain $\Sigma$ and a logic $\mathfrak{L}$. We define $\mathfrak{M}$-$\mathfrak{L}$-MSO as MSO-logic with signature $\tau = (<, S, (P_\varphi)_{\varphi \in \mathfrak{M}_\#\text{-}\mathfrak{L}}$, min, max$)$, where

- min, max are constant symbols,
- $<$ is the natural ordering relation symbol,
- $S$ is the successor relation symbol, and
- $P_\varphi$ is a unary relation symbol associated to every $\mathfrak{M}_\#$-$\mathfrak{L}$-formula $\varphi$.

To determine the semantics of $\mathfrak{M}$-$\mathfrak{L}$-MSO-formulas, we need a model to interpret them in. At first, we define how we represent a word as a model, called *word model* or *word structure.*

**Definition 4.2 (Word model)** Given a tuple $w = (u_1, \dots, u_n)$ of words over $\Sigma$, such that at least one of the $u_i$ is non-empty. We define the structure

$$\underline{w} = (\mathrm{dom}(w), \min, \max, <, S, (P_\varphi)_{\varphi \in \mathfrak{M}_\#\text{-}\mathfrak{L}})$$

where

- $\mathrm{dom}(w) = \{1, \dots, |\langle w \rangle|\}$ is a finite subset of $\mathbb{N}$, interpreted as positions in the word,
- $\min = 1$, $\max = |\langle w \rangle|$,
- $<$ is interpreted as the natural ordering relation restricted to $\mathrm{dom}(w)$,
- $S$ is interpreted as the successor relation restricted to $\mathrm{dom}(w)$,
- $P_\varphi = \{i \in \mathrm{dom}(w) \mid \mathfrak{M}_\# \models \varphi[u'_{1i}, \dots, u'_{ni}]\}$ for all $\mathfrak{M}$-$\mathfrak{L}$-formulas $\varphi$

The unary relations $P_\varphi$ are called letter predicates. So, $P_\varphi$ collects all letter positions of $w$ which carry a letter that satisfies the property described by $\varphi$.

To interpret the truth value of an $\mathfrak{M}$-$\mathfrak{L}$-MSO-formula $\psi(x_1, \dots, x_m, X_1, \dots, X_n)$ with free variables $x_1, \dots, x_m$ and $X_1, \dots, X_n$ we need a word model $\underline{w}$ and additionally $m$ positions $k_1, \dots, k_m$ and $n$ sets of positions $K_1, \dots, K_n$. Altogether, the complete model of a $\mathfrak{M}$-$\mathfrak{L}$-MSO-formula is defined as follows:

**Definition 4.3** Let $\psi(x_1, \ldots, x_m, X_1, \ldots, X_n)$ be an $\mathfrak{M}$-$\mathfrak{L}$-MSO-formula with free variables. Let $w$ be a tuple of words, $k_1, \ldots, k_m \in \mathrm{dom}(w)$ and $K_1, \ldots, K_n \subseteq \mathrm{dom}(w)$. We say

$$(\underline{w}, k_1, \ldots, k_m, K_1, \ldots, K_n) \models \varphi(x_1, \ldots, x_m, X_1, \ldots, X_n)$$

if $\psi$ holds in $\underline{w}$ for the assignment of $x_i = k_i$ and $X_j = K_j$ for $1 \leq i \leq m$ and $1 \leq j \leq n$. We also write:

$$\underline{w} \models \psi[k_1, \ldots, k_m, K_1, \ldots, K_n]$$

We now take a look at an example which illustrates the combination of MSO-logic and the logic $\mathfrak{L}$ speaking over a structure $\mathfrak{M}$.

**Example 4.4** Let $w = 010011$, $\mathfrak{M} = (\mathbb{N}, <)$ and $\mathfrak{L} = \mathrm{FO}$. We use the $\mathfrak{M}$-$\mathfrak{L}$-formulas $\varphi_0(x) = \neg(\exists y(y < x))$ and $\varphi_1(x) = \exists y(\varphi_0(y) \wedge \neg(\exists z(y < z \wedge z < x)))$ to express the letter properties that $x = 0$, resp. $x = 1$. Out of these $\mathfrak{M}$-$\mathfrak{L}$-formulas we define the predicates $P_{\varphi_0}$ and $P_{\varphi_1}$, where $P_{\varphi_0}(x)$ expresses that the symbol at position $x$ is 0, and $P_{\varphi_1}(x)$ expresses that the symbol at position $x$ is 1.

Now, we consider the following $\mathfrak{M}$-$\mathfrak{L}$-MSO-formulas, talking about properties of the positions in $w$, that use the predicates described above

- "positions $x_1, x_2$ are successive positions, both being 0"
  $\psi(x_1, x_2) = S(x_1, x_2) \wedge P_{\varphi_0}(x_1) \wedge P_{\varphi_0}(x_2)$
- "all positions in $X_1$ are 0 and at least one of those positions has a predecessor, which is 1"
  $\psi'(x_1, X_1) = \forall x(X_1(x) \rightarrow P_{\varphi_0}(x)) \wedge \exists y(x_1 < y \wedge P_{\varphi_1}(y) \wedge X_1(y))$
- "every position, which is 1 has a predecessor which is 0"
  $\psi'' = \forall x(\exists y(P_{\varphi_1}(x) \rightarrow (P_{\varphi_0}(y) \wedge y < x)))$

Then we have, for example:

$$(\underline{w}, 3, 4) \models \psi(x_1, x_2)$$

$$(\underline{w}, 1, 2) \not\models \psi(x_1, x_2)$$

$$(\underline{w}, 2, \{1, 3, 4\}) \models \psi'(x_1, X_1)$$

$$(\underline{w}, 5, \{1, 3, 4\}) \not\models \psi'(x_1, X_1)$$

$$\underline{w} \models \psi''$$

An $\mathfrak{M}$-$\mathfrak{L}$-MSO-sentence, i.e. formulas which do not have any free variables, is interpreted in a word model without e.g. any additional sets of positions. All tuples of words $w$ such that $\underline{w} \models \psi$ form the language defined by an $\mathfrak{M}$-$\mathfrak{L}$-MSO-sentence $\psi$.

**Definition 4.5** Let $\mathfrak{M}$ be a structure with domain $\Sigma$ and $\varphi$ be an $\mathfrak{M}$-$\mathfrak{L}$-MSO sentence. We set $L(\varphi) = \{w \in (\Sigma^*)^n \mid \underline{w} \models \varphi\}$ as the language defined by $\varphi$. We say a relation $R \subseteq (\Sigma^*)^n$ is $\mathfrak{M}$-$\mathfrak{L}$-*MSO definable* if there is a $\mathfrak{M}$-$\mathfrak{L}$-MSO sentence $\varphi$ with $L_R = L(\varphi)$.

**Example 4.6** Let $\mathfrak{M} = (\mathbb{N}, +)$ and fix $\mathfrak{L} = \text{FO}$. For comparison we use the languages and relations considered in Example 3.7.

(1) the set $L \subseteq \mathbb{N}^*$ of word whose symbols are alternatively even and odd, starting with an even symbol is $\mathfrak{M}$-$\mathfrak{L}$-MSO definable by the sentence $\forall x \forall y \big(S(x, y) \rightarrow (P_{\varphi_{even}}(x) \leftrightarrow P_{\varphi_{odd}}(y))\big) \wedge P_{\varphi_{even}}(min)$, where $\varphi_{even}(x) := \exists y\ y + y = x$ and $\varphi_{odd}(x) := \neg\varphi_{even}$.

(2) the relation $R \subseteq \mathbb{N}^* \times \mathbb{N}^*$ with $(u, v) \in R$ if there exists at least one position $i$, such that the $i$-th symbol of $u$ equals the $i$-th symbol of $v$ is $\mathfrak{M}$-$\mathfrak{L}$-MSO definable by the sentence $\exists i P_{\varphi_{eq}}(i)$, where $\varphi_{eq}(x, y) := (x = y)$.

(3) the set of words over that start with a 1 followed by arbitrarily many 0's is $\mathfrak{M}$-$\mathfrak{L}$-MSO definable by the sentence $P_{\varphi_1}(min) \wedge \forall x(min \neq x \rightarrow P_{\varphi_0}(x))$, where $\varphi_0(x)$ expresses that $x = 0$ and $\varphi_1(x)$ expresses that $x = 1$.

## 4.2 The Equivalence Theorem

At this point, we are ready to present a translation from any $\mathfrak{M}$-$\mathfrak{L}$-automaton to an $\mathfrak{M}$-$\mathfrak{L}$-MSO-formula and vice versa. Thereby, we show the equivalence between $\mathfrak{M}$-$\mathfrak{L}$-recognizable relations and $\mathfrak{M}$-$\mathfrak{L}$-MSO-definable relations.

First, we show the direction from a given $\mathfrak{M}$-$\mathfrak{L}$-automaton to a $\mathfrak{M}$-$\mathfrak{L}$-MSO-formula.

**Proposition 4.7** *Let $\mathcal{A}$ be an $\mathfrak{M}$-$\mathfrak{L}$-automaton, then there exists an $\mathfrak{M}$-$\mathfrak{L}$-MSO-sentence $\psi_{\mathcal{A}}$ with $L(\mathcal{A}) = L(\psi_{\mathcal{A}})$.*

*Proof.* The proof is a straightforward adaptation of the classical case for finite alphabets, shown by Büchi in [Büc60]. Additionally, we deal with relations over infinite alphabets. Given an $\mathfrak{M}$-$\mathfrak{L}$-automaton $\mathcal{A} = (\{1, \dots, m\}, n, \Delta, \mathfrak{M}, \mathfrak{L}, 1, F)$ we have to construct an $\mathfrak{M}$-$\mathfrak{L}$-MSO sentence $\psi_{\mathcal{A}}$ such that any word model $\underline{w}$ satisfies $\psi_{\mathcal{A}}$ if and only if $\mathcal{A}$ accepts the convolution $\langle w \rangle$ of the tuple $w$ of words. Accordingly, the sentence $\psi_{\mathcal{A}}$ has to express the existence of a successful run of $\mathcal{A}$. Therefore, we introduce $m$ sets $X_1, \dots, X_m$ such that $X_i$ encodes the set of position where a run in $\mathcal{A}$ passes through the state $i$. The purpose of $\psi$ is to set constraints on these sets such that they describe a successful run. There are four properties, which a sentence has to combine in a conjunction in order to express a successful run of $\mathcal{A}$:

- The automaton is only at one state at a time. Thus, the sets $X_1, \dots, X_m$ form a partition of $\text{dom}(w)$. The respective $\mathfrak{M}$-$\mathfrak{L}$-MSO-formula is given by

$$\psi_{Partition}(X_1, \dots, X_m) = \forall x \left( \bigvee_{i=1}^{m} X_i(x) \wedge \bigwedge_{i \neq j, 1 \leq i,j \leq m} \neg\big(X_i(x) \wedge X_j(x)\big)\right)$$

- The run starts in the initial state. This is easily expressed by $X_1(min)$.
- For all positions $x, y$ such that $y$ is successor of $x$ a transition is executed. We write

  $\forall x \forall y \ \big(S(x,y) \rightarrow \bigvee_{(i,\varphi,j)\in\Delta} (X_i(x) \wedge P_\varphi(x) \wedge X_j(y))\big)$
- The last executed transition, after reading the letter at position max, leads to a final state. This is expressed by

  $\bigvee_{\exists j\in F:(i,\varphi,j)\in\Delta} \big(X_i(max) \wedge P_\varphi(max))\big)$

Finally, we combine these formulas to a conjunction.

$$\begin{aligned}\psi_{\mathcal{A}} = \exists X_1 \dots \exists X_m \quad & \Big(\psi_{Partition}(X_1,\dots,X_m) \ \wedge \ X_1(min)\\ & \wedge \ \forall x \forall y \ \big(S(x,y) \rightarrow \bigvee_{(i,\varphi,j)\in\Delta} (X_i(x) \wedge P_\varphi(x) \wedge X_j(y))\big)\\ & \wedge \ \bigvee_{\exists j\in F:(i,\varphi,j)\in\Delta} \big(X_i(max) \wedge P_\varphi(max))\Big)\end{aligned}$$

Then $\mathcal{A}$ accepts $\langle w\rangle$ if and only if $\underline{w} \models \psi_{\mathcal{A}}$

□

To show the direction from a given $\mathfrak{M}$-$\mathfrak{L}$-MSO-formula to an $\mathfrak{M}$-$\mathfrak{L}$-automaton, we use a simpler variant of $\mathfrak{M}$-$\mathfrak{L}$-MSO-logic, in which first-order variables are omitted. Elements are represented by singletons.

**Definition 4.8** The language of $\mathfrak{M}$-$\mathfrak{L}$-MSO$_0$-logic is built up from atomic formulas of the form

- $X \subseteq Y$,
- $X \subseteq P_\varphi$,
- Sing($X$) meaning $X$ is a singleton,
- Succ($X, Y$) meaning $X, Y$ are singletons $\{x\}, \{y\}$ with $S(x, y)$,
- $X < Y$ meaning $X, Y$ are singletons $\{x\}, \{y\}$ with $x < y$.

as well as the connectives $\neg, \vee, \wedge$, and the set quantifiers $\exists$ and $\forall$.

**Remark 4.9** Every $\mathfrak{M}$-$\mathfrak{L}$-MSO-formula can be transformed to an equivalent $\mathfrak{M}$-$\mathfrak{L}$-MSO$_0$-formula.

*Proof.* We show this by induction over the structure of the given $\mathfrak{M}$-$\mathfrak{L}$-MSO-formula. We have to eliminate all occurrences of first-order variables, say $x, y$ are first-order variables and $Z$ is a set variable. In atomic formulas, we have to rewrite occurrences of first-variables as follows:

- "$x = y$" is rewritten as "$\mathrm{Sing}(X) \wedge \mathrm{Sing}(Y) \wedge X = Y$" with "$X = Y$" short for "$X \subseteq Y \wedge Y \subseteq X$".
- "$Z(x)$" is rewritten as "$\mathrm{Sing}(X) \wedge X \subseteq Z$".
- "$P_\varphi(x)$" is rewritten as "$\mathrm{Sing}(X) \wedge X \subseteq P_\varphi$"
- "$x < y$" is rewritten as "$\mathrm{Sing}(X) \wedge \mathrm{Sing}(Y) \wedge X < Y$"
- "$S(x, y)$" is rewritten as "$\mathrm{Sing}(X) \wedge \mathrm{Sing}(Y) \wedge \mathrm{Succ}(X, Y)$"

Let $\phi$ and $\psi$ be $\mathfrak{M}$-$\mathfrak{L}$-MSO-formulas. Assume $\phi'$ and $\psi'$ are the equivalent $\mathfrak{M}$-$\mathfrak{L}$-MSO$_0$-formulas to $\phi$ and $\psi$, respectively. For the remaining induction step, we have to consider the formulas "$\exists x\psi$" and "$\forall x\psi$" where $x$ is a free variable in $\psi$. Then $\exists x\psi$ is translated to $\exists X(\mathrm{Sing}(X) \wedge \psi')$ and $\forall x\psi$ is translated to $\forall X(\mathrm{Sing}(X) \wedge \psi')$. The cases $\neg\phi$, $\phi \wedge \psi$ and $\phi \vee \psi$ are translated to $\neg\phi'$, $\phi' \wedge \psi'$ and $\phi' \vee \psi'$, respectively.

□

**Example 4.10** The $\mathfrak{M}$-$\mathfrak{L}$-MSO-formula

$$\psi(Z) = \exists x(P_\varphi(x) \wedge \forall y(y < x \to Z(y)))$$

is rewritten as

$$\exists X(\mathrm{Sing}(X) \wedge X \subseteq P_\varphi \wedge \forall Y(\mathrm{Sing}(Y) \wedge Y < X \to Y \subseteq Z)).$$

Now that we have introduced $\mathfrak{M}$-$\mathfrak{L}$-MSO$_0$, we identify an $\mathfrak{M}$-$\mathfrak{L}$-MSO$_0$-formula with a language that shall be recognized by an equivalent $\mathfrak{M}$-$\mathfrak{L}$-automaton. Recall that an $\mathfrak{M}$-$\mathfrak{L}$-MSO$_0$-formula $\psi(X_1, \ldots, X_k)$ with free variables $X_1, \ldots, X_k$ is interpreted in a word model $\underline{w}$ and $k$ subsets $K_1, \ldots, K_k \subseteq \mathrm{dom}(w)$. To code such models as a single word that can be handled by an automaton we represent the sets $K_1, \ldots, K_k$ by bit vectors. For every position $i \in \mathrm{dom}(w)$ we represent the information whether $i \in K_1, i \in K_2, \ldots, i \in K_k$ by a bit vector with $k$ entries such that the $j$-th entry is 1 if and only if $i \in K_j$ for $1 \leq j \leq k$.

**Definition 4.11** A formula $\psi(X_1, \ldots, X_k)$ is interpreted in a model $(\underline{w}, K_1, \ldots, K_k)$ with $K_i \subseteq \mathrm{dom}(w)$, where $\underline{w}$ is the word model of a tuple $w = (u_1, \ldots, u_n) \in (\Sigma^*)^n$ of words. We represent the model by a word over the alphabet $(\Sigma_\#)^n \times \{0, 1\}^k$:

$$\begin{bmatrix} u'_{11} \\ \vdots \\ u'_{n1} \\ c_{11} \\ \vdots \\ c_{k1} \end{bmatrix} \begin{bmatrix} u'_{12} \\ \vdots \\ u'_{n2} \\ c_{12} \\ \vdots \\ c_{k2} \end{bmatrix} \cdots \begin{bmatrix} u'_{1\ell} \\ \vdots \\ u'_{n\ell} \\ c_{1\ell} \\ \vdots \\ c_{k\ell} \end{bmatrix} \in ((\Sigma_\#)^n \times \{0, 1\}^k)^*$$

and $K_i$ is the set of positions $j$ where $i$-th bit component $c_{ij} = 1$ iff $j \in K_i$.

Now, we can identify an $\mathfrak{M}$-$\mathfrak{L}$-MSO$_0$-formula $\psi(X_1,\ldots,X_k)$ with a language over the alphabet $(\Sigma_\#)^n \times \{0,1\}^k$.

**Example 4.12** Let $\mathfrak{M} = (\mathbb{N},+)$. Consider the $\mathfrak{M}$-$\mathfrak{L}$-MSO$_0$-formula $\psi(X_1,X_2) = \forall X\big(\mathrm{Sing}(X) \wedge (X \subseteq P_\varphi \leftrightarrow X \subseteq X_1) \wedge (\neg X \subseteq P_\varphi \leftrightarrow X \subseteq X_2)\big)$ and the $\mathfrak{M}$-$\mathfrak{L}$-formula $\varphi(x) = \exists y(y+y=x)$, where $\varphi(x)$ expresses that $x$ is an even number. Thus $\psi(X_1,X_2)$ expresses that in a word, every letter position such that the letter at this position is even is in $X_1$, otherwise if a letter is odd the respective position is in $X_2$. Let $w = 2\ 0\ 1\ 1\ 12$, then we have for example:

$$(\underline{w},\{1,2,5\},\{3,4\}) \models \psi(X_1,X_2)$$
$$(\underline{w},\{1,2,3\},\{3,4,5\}) \not\models \psi(X_1,X_2)$$

These two models are represented as follows, in the first case we have $K_1 = \{1,2,5\}, K_2 = \{3,4\}$; in the second case we have $K_1 = \{1,2,3\}, K_2 = \{3,4,5\}$.

$$\begin{array}{c} \\ K_1 \\ K_2 \end{array} \begin{bmatrix}2\\1\\0\end{bmatrix} \begin{bmatrix}0\\1\\0\end{bmatrix} \begin{bmatrix}1\\0\\1\end{bmatrix} \begin{bmatrix}1\\0\\1\end{bmatrix} \begin{bmatrix}12\\1\\0\end{bmatrix} \text{ satisfies } \psi$$
$$\begin{array}{c} \\ K_1 \\ K_2 \end{array} \begin{bmatrix}2\\1\\0\end{bmatrix} \begin{bmatrix}0\\1\\0\end{bmatrix} \begin{bmatrix}1\\1\\1\end{bmatrix} \begin{bmatrix}1\\0\\1\end{bmatrix} \begin{bmatrix}12\\0\\1\end{bmatrix} \text{ does not satisfy } \psi$$

Thus, $L(\psi)$ is the set of words over $\Sigma_\# \times \{0,1\}^2$, such that if the letter component is even only the first bit component is 1, otherwise only the second bit component is 1.

At this point, we have introduced all necessary notions to translate any given $\mathfrak{M}$-$\mathfrak{L}$-MSO-formula into an equivalent $\mathfrak{M}$-$\mathfrak{L}$-MSO-automaton.

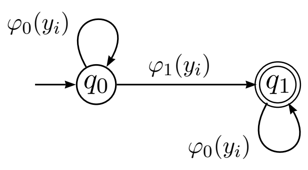
**Proposition 4.13** *Let $\varphi$ be an $\mathfrak{M}$-$\mathfrak{L}$-MSO sentence, then there exists an $\mathfrak{M}$-$\mathfrak{L}$-automaton $\mathcal{A}_\psi$ with $L(\psi) = L(\mathcal{A}_\psi)$.*

*Proof.* We construct for every $\mathfrak{M}$-$\mathfrak{L}$-MSO-formula an equivalent $\mathfrak{M}$-$\mathfrak{L}$-MSO$_0$-formula and proceed by induction over the structure of MSO$_0$-formulas. For any MSO$_0$-formula $\psi(X_1,\ldots,X_k)$ we have to construct an equivalent $\mathfrak{M}$-$\mathfrak{L}$-automaton $\mathcal{A}_\psi$ over $(\Sigma_\#)^n \times \{0,1\}^k$. Formally, every transition label $\varphi(x_1,\ldots,x_n,y_1,\ldots,y_k)$ has free variables $x_1,\ldots,x_n$ and free variables $y_1,\ldots,y_k$. Recall, that a representation of a model $(\underline{w},K_1,\ldots,K_k)$ is the input for the automaton. Therefore a word over $(\Sigma_\#)^n \times \{0,1\}^k$ is read, where the upper $n$ elements of each letter form the convolution $\langle w\rangle$ of $w$. These upper $n$ elements are assigned to the free variables $x_1,\ldots,x_n$. The lower $k$ elements are the bit vectors, which are assigned to the free variables $y_1,\ldots,y_k$.

Now, we specify $\mathfrak{M}$-$\mathfrak{L}$-automata for the atomic $\mathfrak{M}$-$\mathfrak{L}$-MSO$_0$-formulas, in the transitions of these automata, we use the $\mathfrak{M}$-$\mathfrak{L}$-formulas $\varphi_0(x)$ and $\varphi_1(x)$, where $\varphi_0(x)$ expresses that $x = 0$ and $\varphi_1(x)$ expresses that $x = 1$.

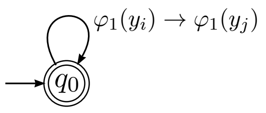
- $\text{Sing}(X_i)$:

  Given a non-empty word over $(\Sigma_\#)^n \times \{0,1\}^k$, the $\mathfrak{M}$-$\mathfrak{L}$-automaton checks that the $i$-th bit component is 1 exactly once.



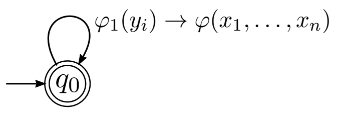
- $X_i \subseteq X_j$

  Given a non-empty word over $(\Sigma_\#)^n \times \{0,1\}^k$, the $\mathfrak{M}$-$\mathfrak{L}$-automaton checks that whenever the $i$-th bit component is 1, the $j$-th bit component is 1 as well.



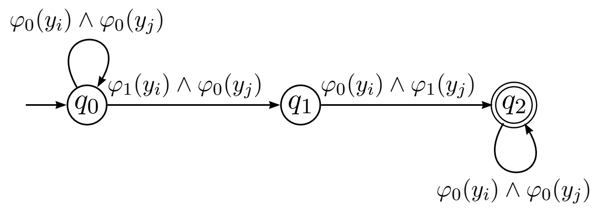
- $X_i \subseteq P_\varphi$

  Given a non-empty word over $(\Sigma_\#)^n \times \{0,1\}^k$, the $\mathfrak{M}$-$\mathfrak{L}$-automaton checks that whenever the $i$-th bit component is 1, the letter vector satisfies the $\mathfrak{M}$-$\mathfrak{L}$-formula $\varphi$, which defines the letter predicate $P_\varphi$.



- $\text{Succ}(X_i, X_j)$

  Given a non-empty word over $(\Sigma_\#)^n \times \{0,1\}^k$, the $\mathfrak{M}$-$\mathfrak{L}$-automaton checks that immediately after the $i$-th bit component is 1 the $j$-th bit component is 1 as well.


$\varphi_0(y_i) \wedge \varphi_0(y_j)$
$q_0$
$\varphi_1(y_i) \wedge \varphi_0(y_j)$
$q_1$
$\varphi_0(y_i) \wedge \varphi_1(y_j)$
$q_2$
$\varphi_0(y_i) \wedge \varphi_0(y_j)$

- $X_i < X_j$

  Given a non-empty word over $(\Sigma_\#)^n \times \{0,1\}^k$, the $\mathfrak{M}$-$\mathfrak{L}$-automaton works like the $\mathfrak{M}$-$\mathfrak{L}$-automaton specified for $\mathrm{Succ}(X_i, X_j)$, but after the $i$-th bit component is 1, the position where the $j$-th bit component is 1 may occur arbitrarily many positions later.

$$\begin{tikzcd}
{} \arrow[r] & q_0 \arrow[loop above, "\varphi_0(y_i) \wedge \varphi_0(y_j)"] \arrow[rr, "\varphi_1(y_i) \wedge \varphi_0(y_j)"] & & q_1 \arrow[loop below, "\varphi_0(y_i) \wedge \varphi_0(y_j)"] \arrow[rr, "\varphi_0(y_i) \wedge \varphi_1(y_j)"] & & \boxed{q_2} \arrow[loop below, "\varphi_0(y_i) \wedge \varphi_0(y_j)"]
\end{tikzcd}$$

For the induction step, it suffices to consider the connectives $\neg$ and $\vee$, and the existential set quantifier $\exists$, since the connectives $\wedge, \rightarrow$, and the universal set quantifier $\forall$ are definable in terms of the functionally complete set of Boolean operators $\{\neg, \vee\}$ and the existential set quantifier $\exists$.

Consider the $\mathfrak{M}$-$\mathfrak{L}$-MSO$_0$-formulas $\psi_1(X_1, \ldots, X_k)$ and $\psi_2(X_1, \ldots, X_k)$ and the equivalent $\mathfrak{M}$-$\mathfrak{L}$-automata $\mathcal{A}_{\psi_1}$, resp. $\mathcal{A}_{\psi_2}$. For the negation, we can construct an equivalent $\mathfrak{M}$-$\mathfrak{L}$-automaton for $\neg\psi_1$ by constructing the complement automaton of $\mathcal{A}_{\psi_1}$. Regarding the connective $\vee$, the formula $\psi_1 \vee \psi_2$ is equivalent to the $\mathfrak{M}$-$\mathfrak{L}$-automaton that recognizes $L(\mathcal{A}_1) \cup L(\mathcal{A}_2)$, which we obtain by constructing the union automaton of $\mathcal{A}_{\psi_1}$ and $\mathcal{A}_{\psi_2}$. At last we handle the existential quantifier. Consider the $\mathfrak{M}$-$\mathfrak{L}$-MSO$_0$-formula $\psi' = \exists X_k \psi(X_1, \ldots, X_k)$. To the formula $\psi(X_1, \ldots, X_k)$ we construct an equivalent $\mathfrak{M}$-$\mathfrak{L}$-automaton $\mathcal{A}_\psi$. Let $L \subseteq ((\Sigma_\#)^n \times \{0,1\}^k)^+$ be the language defined by $\psi(X_1, \ldots, X_k)$ and recognized by $\mathcal{A}_\psi$ and $L' \subseteq ((\Sigma_\#)^n \times \{0,1\}^{k-1})^+$ denotes the projection of $L$ recognized by the projection automaton $\mathcal{A}'$. The projection automaton is constructed from the $\mathfrak{M}$-$\mathfrak{L}$-automaton for $\psi(X_1, \ldots, X_k)$ by replacing every transition $(p, \varphi(x_1, \ldots, x_n, y_1, \ldots, y_k), q)$ by $(p, \exists y_k \varphi(x_1, \ldots, x_n, y_1, \ldots, y_k), q)$. Then the formula $\psi'$ defines the language $L'$.

We show $L(\mathcal{A}') = L(\psi')$. Given a model $(\underline{w}, K_1, \ldots, K_{k-1})$, we represent the model by $b_1 b_2 \ldots b_\ell$

$$b_1 b_2 \ldots b_\ell := \begin{bmatrix} u'_{11} \\ \vdots \\ u'_{n1} \\ c_{11} \\ \vdots \\ c_{(k-1)1} \end{bmatrix} \begin{bmatrix} u'_{12} \\ \vdots \\ u'_{n2} \\ c_{12} \\ \vdots \\ c_{(k-1)2} \end{bmatrix} \cdots \begin{bmatrix} u'_{1\ell} \\ \vdots \\ u'_{n\ell} \\ c_{1\ell} \\ \vdots \\ c_{(k-1)\ell} \end{bmatrix} \in ((\Sigma_\#)^n \times \{0,1\}^{k-1})^+$$

as defined in Definition 4.11.

$b_1 b_2 \dots b_\ell$ satisfies $\exists X_k \psi(X_1, \dots, X_k)$

iff there exists $K_k \subseteq \mathrm{dom}(w)$, such that
$(\underline{w}, K_1, \dots, K_{k-1}, K_k) \models \psi(X_1, \dots, X_k)$

iff there exists $c_{k1}, \dots, c_{k\ell} \in \{0, 1\}$, such that
$\begin{bmatrix} b_1 \\ c_{k1} \end{bmatrix} \begin{bmatrix} b_2 \\ c_{k2} \end{bmatrix} \cdots \begin{bmatrix} b_\ell \\ c_{k\ell} \end{bmatrix}$ satisfies $\psi(X_1, \dots, X_k)$

iff $b_1 b_2 \dots b_\ell$ is the projection of some $u \in L(\mathcal{A})$

iff $b_1, \dots, b_\ell \in L(\mathcal{A}')$.

This completes the proof of Proposition 4.13.

□

We presented a way to translate any given $\mathfrak{M}$-$\mathfrak{L}$-automaton to an $\mathfrak{M}$-$\mathfrak{L}$-MSO-formula and vice versa. Combining Proposition 4.7 and Proposition 4.13 we immediately obtain the following result.

**Theorem 2** *A relation $R \subseteq (\Sigma^*)^n$ with $n \geq 1$ is $\mathfrak{M}$-$\mathfrak{L}$-MSO definable if and only if it is $\mathfrak{M}$-$\mathfrak{L}$-recognizable. The transformation in both directions is effective.*

As a consequence of Theorem 2 we derive the following theorem.

**Theorem 3** *Satisfiability and equivalence of $\mathfrak{M}$-$\mathfrak{L}$-MSO-formulas over word models are decidable problems if and only if the $\mathfrak{L}$-theory of the structure $\mathfrak{M}$ is decidable.*

*Proof.* Given $\mathfrak{M}$-$\mathfrak{L}$-MSO sentences $\psi$ and $\phi$, construct corresponding $\mathfrak{M}$-$\mathfrak{L}$-automata $\mathcal{A}_\psi$ and $\mathcal{A}_\phi$, respectively.

- $\psi$ is satisfiable iff $L(\psi) \neq \emptyset$ iff $L(\mathcal{A}_\psi) \neq \emptyset$
- $\psi \equiv \phi$ iff $L(\psi) = L(\phi)$ iff $L(\mathcal{A}_\psi) = L(\mathcal{A}_\phi)$

Recall, as stated in Proposition 3.13 and Proposition 3.14, the emptiness problem and the equivalence problem is decidable if and only if the $\mathfrak{L}$-theory of the structure $\mathfrak{M}$ is decidable.

□

# Chapter 5

# Extension to Büchi Automata

The purpose of this chapter is to extend the results of the previous chapter to infinite words over infinite alphabets. First, we introduce $\mathfrak{M}$-$\mathfrak{L}$-Büchi automata, an automaton model that recognizes relations of $\omega$-words over both finite and infinite alphabets. Thereafter, we show the equivalence between MSO-logic over $\omega$-words and $\mathfrak{M}$-$\mathfrak{L}$-Büchi automata.

## 5.1 Notation

A *$\omega$-word* over an alphabet $\Sigma$ is a infinite sequence $\alpha = \alpha(0)\alpha(1)\ldots$ with $\alpha(i) \in \Sigma$ for all $i \in \mathbb{N}$. Usually, $\omega$-words are denoted by $\alpha, \beta, \gamma \ldots$ . We denote by $\Sigma^\omega$ the set of $\omega$-words over $\Sigma$. An *$\omega$-language* $L$ is a subset of $\Sigma^\omega$.

The *concatenation* of the word $u$ and the $\omega$-word $\alpha$ is the $\omega$-word $u\alpha$. Let $U \subseteq \Sigma^*$ denote a language and $L \subseteq \Sigma^\omega$ denotes an $\omega$-language. The *concatenation* of $U$ and $L$ is defined to be the set $UL := \{u\alpha \mid u \in U, \alpha \in L\}$. Furthermore, to the language $U \subseteq \Sigma^*$ we define the $\omega$-language $U^\omega := \{\alpha \in \Sigma^\omega \mid \alpha = u_1u_2\ldots \text{ with } u_i \in U\}$.

## 5.2 Definition of $\mathfrak{M}$-$\mathfrak{L}$-Büchi Automata

Finite $\omega$-automata are finite automata equipped with an acceptance condition for infinite words. There are different possibilities to define the acceptance condition. In this thesis, we consider the *Büchi condition* [Büc62] and the *Muller condition*[Mul63], the respective $\omega$-automata are called *Büchi automata* and *Muller automata*. In this section, we recall the definition of Büchi automata, introduce $\mathfrak{M}$-$\mathfrak{L}$-Büchi automata, and in Section 5.5 we recall the definition of Muller automata and introduce $\mathfrak{M}$-$\mathfrak{L}$-Muller automata.

**Definition 5.1** A *non-deterministic Büchi automaton* is of the form

$$\mathcal{B} = (Q, \Sigma, \Delta, q_0, F)$$

with a finite set of states $Q$, a finite alphabet $\Sigma$, a transition relation $\Delta \in Q \times \Sigma \times Q$, an initial state $q_0 \in Q$, and a set of final states $F$.

**Definition 5.2** Let $\mathcal{B} = (Q, \Sigma, q_0, \Delta, F)$ be a Büchi automaton. A *run* of $\mathcal{B}$ on an $\omega$-word $\alpha$ is an infinite sequence of states $\rho = \rho(0)\rho(1)\dots$ with $\rho(0) = q_0$ and $(\rho(i), \alpha(i), \rho(i+1)$ for $i \geq 0$. A run $\rho$ is *successful* if there exist infinitely many $i$ such that $\rho(i) \in F$. We say, $\mathcal{B}$ *accepts* $\alpha$ if and only if there exists a successful run of $\mathcal{B}$ on $\alpha$.

**Definition 5.3** The $\omega$-language *recognized* by a Büchi-automaton $\mathcal{B}$ is defined to be the set $L(\mathcal{B}) = \{\alpha \in \Sigma^\omega \mid \mathcal{B} \text{ accepts } \alpha\}$. An $\omega$-language is called *Büchi recognizable* if a corresponding Büchi automaton $\mathcal{B}$ with $L = L(\mathcal{B})$ exists.

Let us extend this automaton model such that it recognizes relations of $\omega$-words over finite and infinite alphabets. Again, in order to recognize relations of $\omega$-words, we define the *convolution of a tuple of $\omega$-words.* Compared to the convolution of a tuple of finite words, we can omit a padding symbol, since all words are of infinite length. Also, the use of an extended structure is no longer necessary.

**Definition 5.4** The *convolution of a tuple of $\omega$-words* $\alpha = (\alpha_1, \dots, \alpha_n) \in (\Sigma^\omega)^n$ with $\alpha_i = \alpha_i(0)\alpha_i(1)\dots$ is defined as follows:

$$\langle\alpha\rangle := \begin{bmatrix}\alpha_1(0)\\ \vdots \\ \alpha_n(0)\end{bmatrix}\begin{bmatrix}\alpha_1(1)\\ \vdots \\ \alpha_n(1)\end{bmatrix}\cdots \quad \in (\Sigma^n)^\omega$$

We define the *convolution of a relation $R \subseteq (\Sigma^\omega)^n$ of $\omega$-words* to be the $\omega$-language $L_R := \{\langle\alpha\rangle \mid \alpha \in R\}$.

An $\mathfrak{M}$-$\mathfrak{L}$-*Büchi automaton* is of the same form as an $\mathfrak{M}$-$\mathfrak{L}$-automaton, we just change the acceptance condition to Büchi acceptance.

**Definition 5.5** Let $\Sigma$ be an alphabet and let $\mathfrak{M}$ denote a structure with domain $\Sigma$. An $\mathfrak{M}$-$\mathfrak{L}$-*Büchi automaton* is of the form

$$\mathcal{B} = (Q, n, \Delta, \mathfrak{M}, \mathfrak{L}, q_0, F)$$

with a finite set of states $Q$, $n \geq 1$ tapes, a transition relation $\Delta \subseteq Q \times \mathcal{F}_n \times Q$, where $\mathcal{F}_n$ is the set of $\mathfrak{M}$-$\mathfrak{L}$-formulas with $n$ free variables, an initial state $q_0 \in Q$, and a set of terminal states $F \subseteq Q$.

**Definition 5.6** Let $\mathcal{B} = (Q, n, \Delta, \mathfrak{M}, \mathfrak{L}, q_0, F)$ be an $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton and let $\alpha = (\alpha_1, \dots, \alpha_n)$ be an $n$-tuple of $\omega$-words over $\Sigma$. A *run* of $\mathcal{B}$ on the convolution $\langle\alpha\rangle$ is an infinite sequence of states $\rho = \rho(0)\rho(1)\dots$ with $\rho(0) = q_0$ and for every $i \geq 0$ exists an $\mathfrak{M}$-$\mathfrak{L}$-formula $\varphi(x_1, \dots, x_n)$ and a transition $(\rho(i), \varphi, \rho(i+1))$ such that the following holds:

$$\mathfrak{M} \models \varphi[\alpha_1(i), \dots, \alpha_n(i)]$$

A run $\rho$ of $\mathcal{B}$ on $\langle\alpha\rangle$ is *successful* if there exist infinitely many $i$ such that $\rho(i) \in F$. We say, $\mathcal{B}$ *accepts* $\langle\alpha\rangle$ if there exists a successful run of $\mathcal{B}$ on $\langle\alpha\rangle$. We denote by $L(\mathcal{B})$ the set of $\omega$-words over $\Sigma^n$ accepted by $\mathcal{B}$.

**Definition 5.7** Let $n \geq 1$. A relation $R \subseteq (\Sigma^\omega)^n$ of $\omega$-words is said to be *$\mathfrak{M}$-$\mathfrak{L}$-Büchi recognizable* if there exists an $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton $\mathcal{B}$ with $n$ tapes such that $L_R = L(\mathcal{B})$.

**Proposition 5.8** *The emptiness problem for $\mathfrak{M}$-$\mathfrak{L}$-Büchi recognizable relations is decidable if and only if the $\mathfrak{L}$-theory of $\mathfrak{M}$ is decidable.*

*Proof.* The proof is similar to the proof of the decidability of the emptiness problem for $\mathfrak{M}$-$\mathfrak{L}$-recognizable relations in Proposition 3.13. Again, we omit all transitions which are labeled by a non-satisfiable formula $\varphi$ and afterward apply a standard non-emptiness test for Büchi automata.

Let $\mathcal{B} = (Q, n, \Delta, \mathfrak{M}, \mathfrak{L}, q_0, F)$ denote an $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton. We define the transition graph $G := (Q, E)$ where $E = \{(p, q) \in Q \times Q \mid \exists (a_1, \dots, a_n) \in \Sigma^n : \mathfrak{M} \models \varphi[a_1, \dots, a_n] \text{ and } (p, \varphi, q) \in \Delta\}$.

We have $L(\mathcal{B}) = \emptyset$ if and only if there is a strongly connected component (SCC) in $G$ reachable from $q_0$ which contains a final state. This can be checked by applying depth-first search from $q_0$ in order to determine the set $Q_0$ of states reachable from $q_0$. Then apply Tarjan's SCC-algorithm on $Q_0$ to obtain a list of all SCCs over $Q_0$. If at least one of those SSCs contains a final state $L(\mathcal{B}) \neq \emptyset$.

□

## 5.3 The Equivalence Theorem for $\mathfrak{M}$-$\mathfrak{L}$-Büchi Automata

In [Büc62], Büchi showed that the results of Theorem 1 are also applicable to infinite words. Equally, we can extend the result of Theorem 2 to the case of relations over infinite words, i.e. we present a way to translate $\mathfrak{M}$-$\mathfrak{L}$-Büchi automata to $\mathfrak{M}$-$\mathfrak{L}$-MSO-formulas and vice versa.

Analogously, to a word model for a tuple $w$ of finite words as given in Definition 4.2, we define a word model for a tuple $\alpha$ of $\omega$-words. Given a tuple $\alpha$ of $\omega$-words over a finite or infinite alphabet, the constant max is omitted, since the domain of an $\omega$-word is always $\mathbb{N}$.

**Definition 5.9 (Word model)** Given a tuple $\alpha = (\alpha_1, \dots, \alpha_n)$ of words over $\Sigma$. We define the structure

$$\underline{\alpha} = (\mathbb{N}, \min, <, S, (P_\varphi)_{\varphi \in \mathfrak{M}\text{-}\mathfrak{L}})$$

To speak over word models of $\omega$-words, we use $\mathfrak{M}$-$\mathfrak{L}$-MSO as defined in Section 4.1 and just omit the constant symbol max.

The proof of the equivalence between $\mathfrak{M}$-$\mathfrak{L}$-Büchi recognizable languages and $\mathfrak{M}$-$\mathfrak{L}$-MSO definable $\omega$-languages is mostly analogous to the proof of Theorem 2. The main difficulty lies in proving the closure of $\mathfrak{M}$-$\mathfrak{L}$-Büchi recognizable languages under complementation. There are two different approaches; one is to use a special representation of $\mathfrak{M}$-$\mathfrak{L}$-Büchi recognizable languages, the other approach is to switch to another class of $\omega$-automaton, namely deterministic $\mathfrak{M}$-$\mathfrak{L}$-Muller automaton, where closure under complementation is easily achieved by replacing the system of final states by its complement. In this section, we show the closure under complementation by following Büchis original approach used in [Büc62], a way to translate $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton to deterministic $\mathfrak{M}$-$\mathfrak{L}$-Muller automaton is presented in Section 5.5.

First, we consider the translation from any given $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton to an $\mathfrak{M}$-$\mathfrak{L}$-MSO-formula.

**Proposition 5.10** *Let $\mathcal{B}$ be an $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton, then there exists an $\mathfrak{M}$-$\mathfrak{L}$-MSO sentence $\psi$ with $L(\mathcal{B}) = L(\psi)$.*

*Proof.* Given an $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton $\mathcal{B} = (\{1, \dots, m\}, n, \Delta, \mathfrak{M}, \mathfrak{L}, q_0, F)$ we have to construct an $\mathfrak{M}$-$\mathfrak{L}$-MSO sentence $\psi_\mathcal{B}$ with $L(\mathcal{B}) = L(\psi_\mathcal{B})$. The translation is mostly done as in Proposition 4.7, only the last property of the conjunction has to be changed to match the Büchi acceptance condition. Therefore, a corresponding $\mathfrak{M}$-$\mathfrak{L}$-MSO-formula has to express that again and again a final state is visited. Consider the following formula that expresses the desired property:

$$\forall x \exists y (x < y \wedge \bigvee_{i \in F} X_i(y))$$

Altogether, we obtain the $\mathfrak{M}$-$\mathfrak{L}$-MSO sentence

$$\psi_{Partition}(X_1, \dots, X_m) = \forall x \, \Big( \textstyle\bigvee_{i=1}^m X_i(x) \wedge \bigwedge_{i \neq j, 1 \leq i,j \leq m} \neg \big(X_i(x) \wedge X_j(x)\big) \Big)$$

$$\begin{aligned}
\psi_\mathcal{B} = \exists X_1 \dots \exists X_m \quad & \Big( \psi_{Partition}(X_1, \dots, X_m) \wedge X_1(min) \\
& \wedge \forall x \forall y \, \big( S(x,y) \to \textstyle\bigvee_{(i,\varphi,j) \in \Delta} (X_i(x) \wedge P_\varphi(x) \wedge X_j(y)) \big) \\
& \wedge \forall x \exists y (x < y \wedge \textstyle\bigvee_{i \in F} X_i(y)) \Big)
\end{aligned}$$

Then $\mathcal{B}$ accepts $\langle \alpha \rangle$ if and only if $\underline{\alpha} \models \psi_\mathcal{B}$.

□

To show the direction from a given $\mathfrak{M}$-$\mathfrak{L}$-MSO-formula to an $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton, we prove some related results beforehand.

**Proposition 5.11** *Let $U \subseteq (\Sigma^n)^*$ denote an $\mathfrak{M}$-$\mathfrak{L}$-recognizable language, let $K, L \subseteq (\Sigma^n)^\omega$ denote $\mathfrak{M}$-$\mathfrak{L}$-Büchi recognizable $\omega$-languages, then*

(1) *$U^\omega$ is $\mathfrak{M}$-$\mathfrak{L}$-Büchi recognizable.*

(2) *$U \cdot K$ is $\mathfrak{M}$-$\mathfrak{L}$-Büchi recognizable.*

(3) *$K \cup L$ is $\mathfrak{M}$-$\mathfrak{L}$-Büchi recognizable.*

*Proof.* For (1), consider an $\mathfrak{M}$-$\mathfrak{L}$-automaton $\mathcal{A} = (Q, n, \Delta, \mathfrak{M}, \mathfrak{L}, q_0, F)$ that recognizes $U$. The idea is to let a run directly continue from the initial state after a final state is visited. Therefore, transitions leading to $F$ should return to $q_0$ instead and $q_0$ serves a as final state. To avoid problems if a return to $q_0$ is already possible, we transform $\mathcal{A}$ as follows, such that the initial state has no incoming transitions.

- Introduce a new initial state $q_0'$ with transition $(q_0', \varphi, q)$ for all transition $(q_0, \varphi, q)$. The state $q_0'$ is added to the final states $F$ if $q_0$ is a final state.

Now that there are no incoming transitions in the initial state in $\mathcal{A}$, we construct an $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton $\mathcal{B}$ that recognizes $U^\omega$ as follows

- Every transition $(p, \varphi, q)$ with $q \in F$ is replaced by $(q, \varphi, q_0)$.
- Fix the set of final states $F$ to $q_0$ only.

For (2), let $\mathcal{B}_1 = (Q_1, n, \Delta_1, \mathfrak{M}, \mathfrak{L}, q_0^1, F_1)$ and $\mathcal{B}_2 = (Q_2, n, \Delta_2, \mathfrak{M}, \mathfrak{L}, q_0^2, F_2)$ be $\mathfrak{M}$-$\mathfrak{L}$-Büchi automata that recognize $K$ and $L$, respectively. The idea of the construction is that every transition leading to a final state in $\mathcal{B}_1$ should also lead to the initial state $q_0^2$ in $\mathcal{B}_2$. Additionally, if $q_0^1$ is final, it should be possible that a run begins in $q_0^2$. Consider the $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton $\mathcal{B} = (Q_1 \cup Q_2, n, \Delta, \mathfrak{M}, \mathfrak{L}, q_0^1, F)$ with

- $\Delta = \Delta_1 \cup \Delta_2$ and in addition
- for every transition $(p, \varphi, q) \in \Delta_1$ with $q \in F$ add the transition $(p, \varphi, q_0^2)$
- if $q_0^1 \in F$, then for every transition $(q_0^2, \varphi, q) \in \Delta_2$ add the transition $(q_0^1, \varphi, q)$.

For (3) we can use the same construction for union automata as for $\mathfrak{M}$-$\mathfrak{L}$-automata, as detailed in Section 3.4.

□

**Proposition 5.12** *The class of $\mathfrak{M}$-$\mathfrak{L}$-Büchi recognizable relations is closed under complementation.*

*Proof.* Let $\mathcal{B} = (Q, n, \Delta, \mathfrak{M}, \mathfrak{L}, q_0, F)$ be the $\mathfrak{M}$-$\mathfrak{L}$-Büchi-automaton that is to be complemented. The construction is divided into 6 parts.

(1) We introduce a finite set of types or transition profiles, and assign a type to each finite word. The types completely determine the behavior of $\mathcal{B}$ on a finite word.

(2) Every infinite word can be divided into a sequence of finite words; the resulting type sequence $t_0 t_1 t_2 \dots$ determines the behavior of $\mathcal{B}$ on an infinite word, thus the type sequence is enough to decide whether an infinite word is accepted or not.

(3) For each type $t$ the set of words $U_t$ with this type form an $\mathfrak{M}$-$\mathfrak{L}$-Büchi recognizable language.

(4) Every infinite word can be divided into a sequence of finite words using only two different types, such that its type sequence is of the simple form $t_0 t_1^\omega$.

(5) This simple type sequence is used to derive a representation of the complement as $\bigcup U_{t_0} U_{t_1}^\omega$

(6) Finally, we can construct a complement automaton based on the representation, since every component of it is $\mathfrak{M}$-$\mathfrak{L}$-Büchi recognizable.

In step (1), we assign to every finite word a type. Therefore we use the notion of *transition profiles.* For a finite word $u \in (\Sigma^n)^*$ and $p, q \in Q$ we write:

- $\mathcal{B} : p \xrightarrow{u} q$ if there is a run on $u$ from $p$ to $q$ in $\mathcal{B}$.
- $\mathcal{B} : p \xrightarrow[F]{u} q$ if there is a run on $u$ from $p$ to $q$ in $\mathcal{B}$ that visits an accepting state.

A transition profile $t(u)$ of a finite word $u \in (\Sigma^n)^*$ is a directed graph with vertex set $Q$ and edges as follows:

- There is an edge from $p$ to $q$ if $\mathcal{B} : p \xrightarrow{u} q$;
- An edge from $p$ to $q$ is labeled $F$ if $\mathcal{B} : p \xrightarrow[F]{u} q$.

In step (2), we use transition profiles to determine whether an $\omega$-word is accepted by $\mathcal{B}$. A *factorization* of an $\omega$-word $\alpha$ is a division into an infinite sequence of finite words $u_0, u_1, u_2, \dots$ such that $\alpha = u_0 u_1 u_2 \dots$. A factorization $u_0, u_1, u_2, \dots$ of an $\omega$- word $\alpha$ induces a transition profile sequence $t(u_0)t(u_1)t(u_2)\dots$. This sequence is sufficient to decide whether $\mathcal{B}$ accepts $\alpha$ or not.

An infinite sequence $t_0 t_1 t_2 \dots$ of transition profiles is called *accepting* if there is a sequence of states $q_0 q_1 q_2 \dots$ such that $q_0$ is the initial state and for every $i$ there is an edge in the transition profile from $q_i$ to $q_{i+1}$ and infinitely many of these edges are labeled $F$. In this case, the sequence $q_0 q_1 q_2 \dots$ is the path of an accepting run in an $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton. Thus, a transition profile sequence of an $\omega$-words $\alpha$ contains all necessary information to decide whether $\mathcal{B}$ accepts $\alpha$ or not.

**Remark 5.13** Let $\alpha \in (\Sigma^n)^\omega$ and $\alpha = u_0 u_1 u_2 \dots$ is an arbitrary factorization of $\alpha$ into finite words. Then $\alpha \in L$ if and only if $t(u_0)t(u_1)t(u_2)\dots$ is an accepting sequence of transition profiles.

Note that there are only finitely many different transition profiles, since there are only finitely many possibilities to have edges between two states of a given vertex set $Q$.

In step (3) we construct for a transition profile $t$ an $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton that recognizes the set $U_t$ of all finite words with type $t$.

Given an $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton $\mathcal{B}$ over $\Sigma^n$, let $TP$ denote the set of all transition profiles of $\mathcal{B}$, and let $NTP$ denote the set of non-accepting pairs of transition profiles. Formally, $NTP = \{(t_0, t_1) \in TP \times TP \mid t_0 t_1^\omega \text{ is non-accepting}\}$. For a transition profile $t \in TP$ define the set $U_t = \{u \in (\Sigma^n)^+ \mid t(u) = t\}$ of finite words with type $t$.

In the next steps, we see how we can combine different sets $U_t$ to a representation of the complement. Accordingly, we can construct the complement automaton as a combination of different automata recognizing the sets $U_t$. Note that the original automaton $\mathcal{B}$ does not necessarily read every possible input letter, i.e. there might be a letter such that none of the formulas on the transitions is satisfied. However, the complement automaton has to be able to read every possible input letter. Thus each of the automata recognizing the sets $U_t$ has to be able to read each input letter. Therefore we introduce a new set of formulas such that every possible input is covered by a formula. Recall the set of formulas we introduced for the powerset construction for $\mathfrak{M}$-$\mathfrak{L}$-automata in Section 3.4. These formulas fulfill our purpose, namely that for each input letter a formula exists such that the formula is satisfied by the letter. Let $\varphi_1, \dots, \varphi_m$ denote the formulas which occur in the transitions of $\mathcal{B}$. For every subset $J \subseteq \{1, \dots, m\}$, we introduce the formula $\psi_J := \bigwedge_{i \in J} \varphi_i \wedge \bigwedge_{i \notin J} \neg\varphi_i$.

We define the *transition profile automaton (TPA) for* $\mathcal{B}$, the TPA is itself an $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton with the following properties:

- $Q = TP$
- $q_0 = t(\varepsilon)$ is the initial state
- $Q \times \mathcal{F}_J \times Q$, where $\mathcal{F}_J$ is the set of all formulas $\Psi_J$
  Let $u \in (\Sigma^n)^+$ and $a = (a_1, \dots, a_n) \in \Sigma^n$. The transitions are of the form $\delta(t(u), \psi_J, t(ua))$, where $\mathfrak{M} \models \psi_J[a_1, \dots, a_n]$

Now we can use the TPA to construct $\mathfrak{M}$-$\mathfrak{L}$-Büchi automata that recognize the sets $U_t$, respectively. For each transition profile $t$ the TPA *recognizes* $U_t$ if $t$ is the only final state. The transitions are constructed such that after reading a input word $u \in (\Sigma^n)^+$ the automaton is in the state $t(u)$. Thus, we can define $t$ as final state to recognize $U_t$.

In step (4), we show that every infinite word can be cut into finite words, such that the resulting type sequence is a periodic factorization of the form $t_0 t_1^\omega$. This can be proven using a theorem of Ramsey.

**Theorem 4 (Ramsey [Ram30])** *Let $X$ be a countably infinite set such that each undirected edge $\{x, y\}$ over $X$ has a color from a finite set $C$ of colors. Then there is an infinite subset $Y$ of $X$ such that all edges over $Y$ have the same color.*

**Lemma 5.14** *For each $\alpha \in (\Sigma^n)^\omega$ there is a factorization $\alpha = u_0 u_1 u_2 \dots$ with $t(u_i) = t(u_j)$ for all $i, j \geq 1$.*

*Proof.* Let $\mathbb{N}$ denote the positions in $\alpha$. Let $X = \mathbb{N}$. Let the colors $C = TP$, where $TP$ denotes the set of transition profiles of the original automaton $\mathcal{B}$. Every edge $\{i, j\}$ is colored by its corresponding transition profile $t(\alpha[i, j])$, for $i < j$. By Ramsey's Theorem there exists a set of positions $Y = \{i_1, i_2, i_3, \dots\}$ such that there is a sequence $i_1 < i_2 < i_3 \dots$ with $t(\alpha[i_j, i_k]) = t(\alpha[i_\ell, i_m])$ for all $1 \le j < k, \ell < m$.

Hence, choosing $u_0 = \alpha[0, i_1]$ and $u_j = \alpha[i_j, i_{j+1}]$ for $j \ge 1$ yields a periodic factorization.

□

In step (5), we use the result from step (4) to derive a simple representation of the complement. Consider the $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton $\mathcal{B}$, let $TP$ denote the set of transition profiles of $\mathcal{B}$. As stated by Lemma 5.14, every infinite word has a type sequence of the form $t_0 t_1^\omega$ with $t_0, t_1 \in TP$. Thus, it is sufficient to find the set of pairs $(t_0, t_1)$ of transitions profiles, for which the sequence $t_0 t_1^\omega$ is non-accepting. Thus, we can represent the complement as follows.

**Lemma 5.15** *The complement of $L(\mathcal{B})$ is*

$$(\Sigma^n)^\omega \setminus L(\mathcal{B}) = \bigcup_{(t_0,t_1) \in NTP} U_{t_0} U_{t_1}^\omega$$

.

Finally, in step (6) we construct the complement automaton of $\mathcal{B}$. As a result of Lemma 5.15 in step (3), it suffices to construct for each pair $(t_0, t_1) \in NTP$ $\mathfrak{M}$-$\mathfrak{L}$-Büchi automata, which recognize $U_{t_0} U_{t_1}^\omega$ and construct the union automaton. The constructions for the respective automata are given in Proposition 5.11.

□

As a consequence of Proposition 5.11 and Proposition 5.12 we obtain the following result.

**Proposition 5.16** *The inclusion- and the equivalence problem for $\mathfrak{M}$-$\mathfrak{L}$-Büchi recognizable relations are decidable if and only if the $\mathfrak{L}$-theory of $\mathfrak{M}$ is decidable.*

Now, we are ready to give a translation from any $\mathfrak{M}$-$\mathfrak{L}$-MSO sentence to an equivalent $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton.

**Proposition 5.17** *Let $\psi$ be an $\mathfrak{M}$-$\mathfrak{L}$-MSO sentence, then there exists an $\mathfrak{M}$-$\mathfrak{L}$-Büchi-automaton $\mathcal{B}$ with $L(\psi) = L(\mathcal{B})$.*

*Proof.* Given an $\mathfrak{M}$-$\mathfrak{L}$-MSO sentence $\psi$, the idea to proceed by induction over the structure of $\psi$. Since we want to avoid handling first-order variables, we switch to $\mathfrak{M}$-$\mathfrak{L}$-MSO$_0$ and construct an equivalent $\mathfrak{M}$-$\mathfrak{L}$-MSO$_0$ sentence $\phi$ and proceed by induction over the structure of $\phi$. For the induction basis, we consider the atomic formulas $X_i \subseteq X_j$, $\mathrm{Sing}(X_i)$, $\mathrm{Succ}(X_i, X_j)$, $X_i < X_j$ and

$X_i \subseteq P_\varphi$ and specify $\mathfrak{M}$-$\mathfrak{L}$-Büchi automata that recognize the sets of $\omega$-words defined by these formulas. These automata are of the same form as the $\mathfrak{M}$-$\mathfrak{L}$-automata given in the proof of Proposition 4.13. For the induction step it suffices to consider the connectives $\vee, \neg$ and the existential set quantifier $\exists$, because $\wedge, \rightarrow$ and $\forall$ are definable in terms of them. Given the formulas $\psi_1$ and $\psi_2$, let $\mathcal{B}_1$ and $\mathcal{B}_2$ denote the respective corresponding $\mathfrak{M}$-$\mathfrak{L}$-Büchi automata. For the disjunction $\psi_1 \vee \psi_2$ we can use the same construction we used in Proposition 4.13, the same holds for the existential set quantifier $\exists$. Regarding the negation, we have seen in Proposition 5.12 how to construct the complement automaton of $\mathcal{B}_1$.

□

As a consequence of Proposition 5.10 and Proposition 5.17, we obtain the following result.

**Theorem 5** *A relation $R \subseteq (\Sigma^\omega)^n$ with $n \geq 1$ of $\omega$-words is $\mathfrak{M}$-$\mathfrak{L}$-MSO definable if and only if it is $\mathfrak{M}$-$\mathfrak{L}$-Büchi-recognizable. The transformation in both directions is effective.*

## 5.4 Consequences and Applications in Model Checking

As a consequence of Theorem 2 we derive the following theorem.

**Theorem 6** *Satisfiability and equivalence of $\mathfrak{M}$-$\mathfrak{L}$-MSO-formulas over word models are decidable problems if and only if the $\mathfrak{L}$-theory of the structure $\mathfrak{M}$ is decidable.*

*Proof.* Given $\mathfrak{M}$-$\mathfrak{L}$-MSO sentences $\psi$ and $\phi$ and construct corresponding $\mathfrak{M}$-$\mathfrak{L}$-Büchi automata $\mathcal{B}_\psi$ and $\mathcal{B}_\phi$, respectively.

- $\psi$ is satisfiable iff $L(\psi) \neq \emptyset$ iff $L(\mathcal{B}_\psi) \neq \emptyset$
- $\psi \equiv \phi$ iff $L(\psi) = L(\phi)$ iff $L(\mathcal{B}_\psi) = L(\mathcal{B}_\phi)$

Recall, as stated in Proposition 5.8 and Proposition 5.16, the emptiness problem and the equivalence problem is decidable if and only if the $\mathfrak{L}$-theory of the structure $\mathfrak{M}$ is decidable.

□

A well-known application of the connection between automata and logic is the verification of specifications of finite-state systems modeled by finite automata ("model checking").

The *model checking problem* is the following: Given a system and a specification of the behavior of the system, does every run of the system satisfy the specification?

We introduce a structure for a finite system model, where infinitely many actions may lead from one state to another, while the state set remains finite. The set of actions, which lead from one state to another are specified via logical formulas. We call this form of transitions *bundled transitions* because one formula can cover a (finite or infinite) set of actions.

**Definition 5.18** A *transition system with bundled transitions* over a structure $\mathfrak{M}$ with logic $\mathfrak{L}$ is of the form

$$\mathcal{K} = (S, (E_\varphi)_{\varphi \in \mathfrak{M}\text{-}\mathfrak{L}})$$

with a finite set $S$ of states and binary transition relations $E_\varphi \subseteq V \times V$ for all $\mathfrak{M}$-$\mathfrak{L}$-formulas $\varphi$ with $n \geq 1$ free variables.

**Definition 5.19** Let $\mathcal{K} = (S, (E_\varphi)_{\varphi \in \mathfrak{M}\text{-}\mathfrak{L}})$ denote a transition system with bundled transitions and $s \in S$ a state of $\mathcal{K}$. A *path* through $(\mathcal{K}, s)$ is a sequence $s_0 s_1 s_2 \ldots$ where $s_0 = s$ and $(s_i, s_{i+1}) \in E_\varphi$ for $i \geq 0$. Corresponding *label sequences* are all $\omega$-words of the form $\alpha = \alpha(0)\alpha(1)\alpha(2)\ldots$ where $(s_i, s_{i+1}) \in E_\varphi$ and $\alpha(i)$ satisfies $\varphi$. All label sequences through $(\mathcal{K}, s)$ define an $\omega$-language.

We use the logic $\mathfrak{M}$-$\mathfrak{L}$-MSO to specify the constrains of a transition system with bundled transitions.

**Definition 5.20** A transition system $(\mathcal{K}, s)$ with initial state $s$ satisfies an $\mathfrak{M}$-$\mathfrak{L}$-MSO formula $\psi$ if all $\omega$-words in the $\omega$-language defined by all label sequences through $(\mathcal{K}, s)$ satisfy $\psi$.

A transition system with bundled transitions and initial state is easily transformed into an $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton. Given $(\mathcal{K}, s)$ with $\mathcal{K} = (S, (E_\varphi)_{\varphi \in \mathfrak{M}\text{-}\mathfrak{L}})$, construct an $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton $\mathcal{B}_{\mathcal{K},s} = (S, n, \Delta, \mathfrak{M}, \mathfrak{L}, s, S)$ where every state is a final state with $(s, \varphi, s') \in \Delta$ iff $(s, s') \in E_\varphi$ for all $s, s' \in S$.

We can now state the model checking problem using transition system with bundled transitions and $\mathfrak{M}$-$\mathfrak{L}$-MSO as follows:

**Model checking problem:** Given a transition system with bundled transitions with initial state $(\mathcal{K}, s)$ and an $\mathfrak{M}$-$\mathfrak{L}$-MSO-formula $\psi$, decide whether $(\mathcal{K}, s)$ satisfies $\psi$.

To solve the model checking problem, we check for *error scenarios*, i.e. we ask whether there is a label sequence through $(\mathcal{K}, s)$ which does not satisfy $\psi$. The solution to this problem can be computed algorithmically. First, we represent $(\mathcal{K}, s)$ by an $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton $\mathcal{B}_{\mathcal{K},s}$ which recognizes all label sequences through $(\mathcal{K}, s)$, using the construction from above. Secondly, we construct an automaton $\mathcal{B}_{\neg\psi}$, which recognizes all label sequences, which do not satisfy $\psi$, using Theorem 5. Hence, the error scenarios are exactly those label sequences which are accepted by both automata. The construction of an $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton which recognizes $L(\mathcal{B}_{\mathcal{K},s}) \cap L(\mathcal{B}_{\neg\psi})$ is possible, since we showed in the proof of Theorem 5 the closure under union and complementation for $\mathfrak{M}$-$\mathfrak{L}$-Büchi recognizable languages, thus the closure under intersection follows as a direct consequence. If $L(\mathcal{B}_{\mathcal{K},s}) \cap L(\mathcal{B}_{\neg\psi})$ is empty, no label sequence violates the specification $\psi$. So, given a system $(\mathcal{K}, s)$ and a formula $\psi$, to solve the model checking problem we have to do the following:

(1) Construct the $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton $\mathcal{B}_{\mathcal{K},s}$

(2) Construct the $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton $\mathcal{B}_{\neg\psi}$

(3) Construct the $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton $\mathcal{B}$ that recognizes $L(\mathcal{B}_{\mathcal{K},s}) \cap L(\mathcal{B}_{\neg\psi})$

(4) Apply the non-emptiness test on $\mathcal{B}$; if $L(\mathcal{B}) = \emptyset$, then $(\mathcal{K}, s)$ does not satisfy $\psi$, otherwise $(\mathcal{K}, s)$ satisfies $\psi$.

Regarding the emptiness test in (4), we recall, as stated in Proposition 5.8, that the emptiness problem is only decidable if and only if the $\mathfrak{L}$-theory of the structure $\mathfrak{M}$ is decidable.

## 5.5 Determinization of $\mathfrak{M}$-$\mathfrak{L}$-Büchi Automata

The purpose of this section is to transform non-deterministic $\mathfrak{M}$-$\mathfrak{L}$-Büchi automata into equivalent deterministic $\mathfrak{M}$-$\mathfrak{L}$-Muller automata. Another well-known way to determize Büchi automata is to employ a change of the acceptance condition towards Muller automata. The Muller acceptance condition is more restricted as the Büchi condition as it specifies which states should be visited infinitely often, and in contrast to the Büchi condition, which not. A key theorem of the theory of finite $\omega$-automata is McNaughton's Theorem:

**Theorem 7 ([McN66])** *A Büchi automaton can be transformed effectively into an equivalent deterministic Muller automaton.*

McNaughton's Theorem was also proven by Safra in [Saf88], for a comprehensive proof of McNaughton's Theorem using Safra's construction see e.g. [Tho97, Rog02]. First, we introduce deterministic $\mathfrak{M}$-$\mathfrak{L}$-Muller automata and then extend the result of McNaughton's Theorem to $\mathfrak{M}$-$\mathfrak{L}$-Büchi and $\mathfrak{M}$-$\mathfrak{L}$-Muller automata following Safra's construction.

**Definition 5.21** Let $\mathrm{Inf}(\rho) = \{q \in Q \mid q \text{ occurs infinitely often in } \rho\}$.

**Definition 5.22** A *finite deterministic Muller automaton* is of form

$$\mathcal{M} = (Q, \Sigma, \delta, q_0, \mathcal{F})$$

with a finite set of states $Q$, a finite alphabet $\Sigma$, a transition function $\delta : Q \times \Sigma \to Q$, an initial state $q_0 \in Q$ for $1 \leq i \leq k$, and a system of final state sets $\mathcal{F} = \{F_1, \ldots, F_k\}$ with $F_i \subseteq Q$. A *run* $\rho$ is *successful* iff $\mathrm{Inf}(\rho) \in \mathcal{F}$.

Now, we extend this automaton model such that it recognizes relations of $\omega$-words over finite and infinite alphabets. As in the case of $\mathfrak{M}$-$\mathfrak{L}$-Büchi automata, deterministic $\mathfrak{M}$-$\mathfrak{L}$-Muller automata work on the convolution of a tuple of $\omega$-words as defined in Definition 5.4.

**Definition 5.23** Let $\Sigma$ be an alphabet (finite or infinite) and let $\mathfrak{M}$ denote a structure with domain $\Sigma$. A *finite deterministic $\mathfrak{M}$-$\mathfrak{L}$-Muller automaton* is of the form

$$\mathcal{M} = (Q, n, \delta, \mathfrak{M}, \mathfrak{L}, q_0, \mathcal{F})$$

with a finite set of states $Q$, $n \geq 1$ tapes, a transition function $\delta : Q \times \mathcal{F}_n \to Q$, where $\mathcal{F}_n$ is the set of $\mathfrak{M}$-$\mathfrak{L}$-formulas with $n$ free variables, an initial state $q_0 \in Q$, and a system of final state sets $\mathcal{F} = \{F_1, \ldots, F_k\}$ with $F_i \subseteq Q$ for $1 \leq i \leq k$. A *run* $\rho$ is *successful* iff $\mathrm{Inf}(\rho) \in \mathcal{F}$.

Let us take a look at an example, which illustrates that the powerset construction applied to a non-deterministic $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton does not always result in an equivalent deterministic $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton recognizing the same relation.

**Example 5.24** Given a structure $\mathfrak{M} = (\mathbb{N}, +)$ and $\mathfrak{L} = \mathrm{FO}$. The set of $\omega$-words over $\mathbb{N}$ such that from some point onwards only 0's occur is recognized by the $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton depicted in Figure 5.1 with first-order formulas $\varphi_1(x) := (x = x)$ and $\varphi_2(x) := (x + x = x)$.

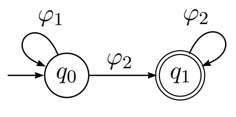


Figure 5.1: A non-deterministic $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton.

Now we apply the powerset construction for infinite alphabets that we introduced in Section 3.4 to the $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton. Recall that in the powerset construction for infinite alphabets we have to construct a set of formulas such that the resulting powerset automaton is deterministic. Therefore we used all Boolean combinations of the formulas that appeared in the original automaton. In this case we obtain the formulas $\psi_1 := (\varphi_1 \wedge \varphi_2), \psi_2 := (\varphi_1 \wedge \neg\varphi_2), \psi_3 := (\neg\varphi_1 \wedge \varphi_2)$ and $\psi_4 := (\neg\varphi_1 \wedge \neg\varphi_2)$. The resulting automaton is depicted in Figure 5.2. As we can see, the automaton accepts any $\omega$-word with infinitely many occurrences of 0's, thus in particular including $\omega$-words with infinitely many occurrences of other letters. As a conclusion we see that the powerset construction is not an appropriate method to construct equivalent deterministic $\mathfrak{M}$-$\mathfrak{L}$-Büchi automata.

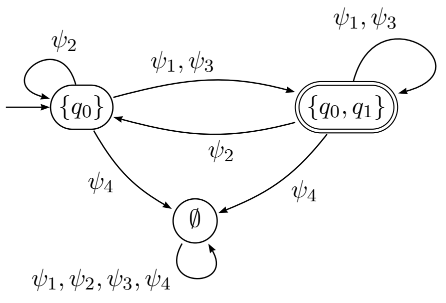


Figure 5.2: The powerset construction for infinite alphabets applied to the $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton from Figure 5.1.

Now, we present a transformation from $\mathfrak{M}$-$\mathfrak{L}$-Büchi automata to deterministic $\mathfrak{M}$-$\mathfrak{L}$-Muller automata yielding equivalent deterministic $\mathfrak{M}$-$\mathfrak{L}$-Muller automata.

**Theorem 8** *An $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton can be transformed effectively into an equivalent deterministic $\mathfrak{M}$-$\mathfrak{L}$-Muller automaton.*

We shall use Safra's construction with only a slight adaptation to deal with infinite alphabets. Before proving the theorem let us take a look at why the powerset construction fails in case of $\mathfrak{M}$-$\mathfrak{L}$-Büchi automata and informally describe Safra's construction.

In the powerset construction (for both finite automata and $\mathfrak{M}$-$\mathfrak{L}$-automata), the powerset automaton uses macrostates, i.e. subsets of states in the automaton, that contain the momentarily reachable states of the original non-deterministic automaton. Let $\mathcal{B}$ be a non-deterministic $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton and let $\mathcal{B}'$ be the corresponding powerset automaton. An accepting run in the powerset automaton passes infinitely often trough a macrostate which contains some final state from $\mathcal{B}$. If $\mathcal{B}'$ is equivalent to $\mathcal{B}$ we can follow the accepting run in $\mathcal{B}$. Therefore we would have to pick a sequence of states from the encountered macrostates such that this sequence forms an accepting run in $\mathcal{B}$. This might be impossible, since for a run on a finite word ending on a final state in $\mathcal{B}$, there might not be an $\omega$-word such that the run can be continued successfully from the final state. Whereas the same finite word leads to a macrostate $R$ in the powerset automaton which, although it contains a final state and thus counts for accepting, can only be successfully continued in the original automaton $\mathcal{B}$ starting from some other non-final state in $q \in R$.

Safra's construction can be seen as a generalized powerset construction which overcomes this weakness by treating occurrences of final states separately. In this construction, states of the automaton are no longer macrostates but sets of macrostates, organized in a tree structure called Safra tree, whose nodes are labeled with sets of states of the $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton. Following a run on an $\omega$-word in a non-deterministic $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton $\mathcal{B}$, the root of a Safra tree collects all momentarily reachable states of $\mathcal{B}$. If final states are encountered in any macrostate, they are split off and introduced as a new child node. This allows us to keep track of how a run is continued after it passed through a final state. To proceed to the next Safra tree in the run we have to compute the sets of states reachable via the next input letter. Therefore in Safra's construction the classical powerset construction is applied to each node of a Safra tree. At this point we need a minor adjustment of Safra's construction. Since we have to deal with infinite alphabets, we use the powerset construction adapted for infinite alphabets as introduced in Section 3.4 instead of the classical powerset construction. We proceed in the same way as in Safra's construction. Safra trees constructed this way will lead to unbounded trees. In order to use these trees as states for a deterministic automaton, we need a finite set of Safra trees. Safra trees of bounded size are obtained by performing two merge operations: States occurring in several brother macrostates only remain in the oldest brother. Nodes with empty macrostates are deleted, except the root node. If the union of brother macrostates equals the macrostate of their parents, then all child nodes and its descendants are deleted and the former parent is marked "!".

### 5.5.1 Preparations

We recall some known results from Safra's construction that we shall use in the proof of Theorem 8.

**Definition 5.25** A *Safra tree* over a set of states $Q$ is an ordered directed finite tree with node names in $\{1, \ldots, 2|Q|\}$, where each node is labeled by a non-empty *macrostate*, i.e. a subset $R \subseteq Q$. The label $R = \emptyset$ is only allowed in the root node. Further, a node can be marked "!", then the label of a node is the pair $(R, !)$. Safra trees satisfy two conditions:

(1) Brother macrostates are disjoint.

(2) The union of brother macrostates is a proper subset of their parent macrostate.

**Remark 5.26 ([Rog02])** The number of nodes in a Safra tree over $Q$ is bounded by $|Q|$. The height of a Safra tree is at most $|Q| - 1$.

Let $P \overset{w}{\rightsquigarrow} R$ with $P, R \subseteq Q$ denote that $\forall r \in R : \exists\ p \in P$ such that $\mathcal{B} : p \xrightarrow{w} r$.

**Remark 5.27 ([Tho97])** Let

$$\begin{tikzcd}
R_0 \arrow[r, squiggly, "u_1"] & P_1 \arrow[r, squiggly, "v_1"] \arrow[d, phantom, "\cup\!|"] & R_1 \arrow[r, squiggly, "u_2"] \arrow[d, phantom, "||"] & P_2 \arrow[r, squiggly, "v_2"] \arrow[d, phantom, "\cup\!|"] & R_2 \arrow[d, phantom, "||"] \arrow[r, phantom, "\ldots"] & P_i \arrow[r, squiggly, "v_i"] \arrow[d, phantom, "\cup\!|"] & R_i \arrow[d, phantom, "||"] \\
 & F_1 \arrow[r, squiggly, "v_1"] & Q_1 & F_2 \arrow[r, squiggly, "v_2"] & Q_2 & F_i \arrow[r, squiggly, "v_i"] & Q_i
\end{tikzcd}$$

where $F_i$ is the subset set of all final states of $P_i$. Then for all $r \in R_i$ there exists a $p \in R_0$, such that $\mathcal{B}$ reaches from $p$ via input $u_1v_1u_2v_2 \ldots u_iv_i$ state $r$ with at least $i$ visits in final states.

**Remark 5.28 ([Tho97])** Let $R_0 \overset{u_1v_1}{\rightsquigarrow} R_1! \overset{u_2v_2}{\rightsquigarrow} R_2! \ldots R_i! \overset{u_{i+1}v_{i+1}}{\rightsquigarrow} \ldots$ as defined in Remark 5.27 starting in $R_0 = \{q_0\}$. Then there is a successful run of the non-deterministic $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton $\mathcal{B}$ on $u_1v_1u_2v_2 \ldots$.

### 5.5.2 Safra's Construction Adapted to Infinite Alphabets

To define a deterministic $\mathfrak{M}$-$\mathfrak{L}$-Muller automaton we need a suitable set of formulas; a complete set of formulas such that only deterministic runs on every input are possible. Therefore we rely on the formulas introduced in the construction of deterministic $\mathfrak{M}$-$\mathfrak{L}$-automata as detailed in Section 3.4. Given the $\mathfrak{M}$-$\mathfrak{L}$-Büchi automaton $\mathcal{B} = (Q, n, \Delta, \mathfrak{M}, \mathfrak{L}, q_0, F)$, let $\varphi_1, \ldots, \varphi_m$ denote the formulas which appear in the transitions of $\mathcal{B}$. Consider, for every subset $J \subseteq \{1, \ldots, m\}$, the formula $\psi_J := \bigwedge_{i \in J} \varphi_i \wedge \bigwedge_{i \notin J} \neg\varphi_i$ and replace every transition $(p, \varphi_i, q) \in \Delta$ with all transitions of the form $(p, \psi_J, q)$ where $i \in J$.

Now, we define the $\mathfrak{M}$-$\mathfrak{L}$-Muller automaton $\mathcal{M} = (Q', n, \delta, \mathfrak{M}, \mathfrak{L}, q'_0, \mathcal{F})$ where

- $Q'$ is the set of Safra trees over $Q$.
- $q'_0$ is the Safra tree consisting of a root node labeled $\{q_0\}$ only.
- The value of the transition function $\delta(s, \psi)$ for a given Safra tree $s$ and a formula $\psi_J$ as above is computed in 6 steps.

(1) Remove all marks "!" in the Safra tree $s$.

(2) For every node $v$ with label $R$ such that $R \cap F \neq \emptyset$, create a new node $v'$ and add $v'$ as youngest child node labeled with $R \cap F$. Choose a free number from $\{1, \ldots, 2|Q|\}$ as node name for $v'$. Since at most one child node is introduced for every node, the set $\{1, \ldots, 2|Q|\}$ is sufficient for denoting node names, as the number of node trees is bounded by $|Q|$.

(3) Apply the powerset construction to each node label, i.e. replace each node label $R$ by the set $\{q \in Q \mid \exists r \in R : (r, \psi_J, q) \in \Delta\}$.

(4) *horizontal merge:* For every node $v$ with label $R$ and state $q \in R$, such that $q$ also occurs in an older brother node, remove all occurrences of $q$ in $v$ and its subtree.

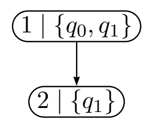
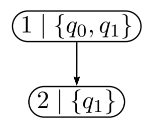
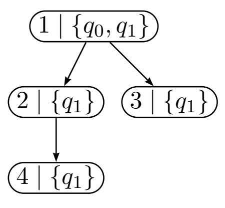
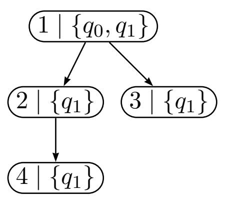
(5) Remove all nodes with empty labels (except the root node).

(6) *vertical merge:* For every node $v$ whose label is equal to the union of the labels of its sons, remove all descendants of $v$ and mark $v$ with "!"

- A set $S \subseteq Q'$ of Safra trees is in $\mathcal{F}$ if some node name appears in each Safra tree $s \in S$, and in at least one $s$ the label of this node is marked "!".

Note, that step (3) differs from the classical Safra construction, since we used the powerset construction adapted to infinite alphabet as introduced in Section 3.4.

Exemplary computation of $\delta(s, \psi_1)$:

| Safra tree s | Step 1 | **Step 2** | Step 3 |
|---|---|---|---|
|  |  |  |  |

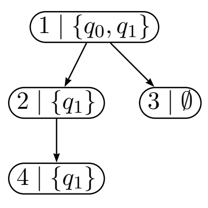
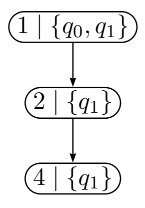
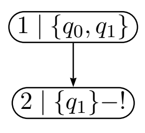
| **Step 4** | **Step 5** | **Step 6** |
|---|---|---|
|  |  |  |

Figure 5.3: Steps from Safra's construction.

**Example 5.29** We apply Safra's construction to the non-deterministic 𝔐-𝔏-Büchi automaton from Example 5.24. The resulting automaton is depicted in Figure 5.4. The employed formulas $\psi_1, \psi_2, \psi_3, \psi_4$ in the transitions are defined as in Example 5.24. Figure 5.3 gives an example on how to compute the transition function $\delta$: The Safra tree is depicted after the execution of each computation step. If a step modifies the Safra tree, it is highlighted.

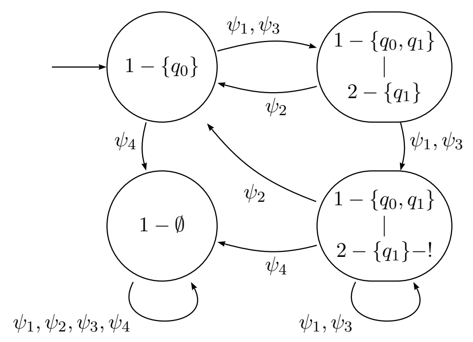


Figure 5.4: Safra's construction applied on the automaton of Figure 5.1.

Indeed, this automaton accepts the $\omega$-language defined in Example 5.24, e.g. the $\omega$-word $(10)^\omega$ passes infinitely often trough the initial state, which is not included in the system of final states. Thus we can see that Safra's construction applied on the automaton of Figure 5.1 yields an equivalent 𝔐-𝔏-Muller automaton, whereas the powerset construction fails in this case.

With the presented construction and the results from the previous section, we are ready to prove the correctness of Safra's construction adapted to infinite alphabets.

*Proof of Theorem 8.* We have to show that $L(\mathcal{B}) = L(\mathcal{M})$.

First, we show $L(\mathcal{B}) \supseteq L(\mathcal{M})$. Let the constructed 𝔐-𝔏-Muller automaton $\mathcal{M}$ accept $\alpha$. Since $\mathcal{M}$ is deterministic there exists a unique run $\rho'$ on $\alpha$, i.e. $\mathrm{Inf}(\rho') \in \mathcal{F}$. Therefore there exist some node name $v$ such that $v$ appears in all Safra trees of $\mathrm{Inf}(\rho')$. Consequently, $v$ appears in every Safra tree — from a certain point on —and is marked "!" infinitely often. Say $s_1, s_2, \ldots$ are the Safra trees that occur in the run $\rho'$, with $s_i < s_j$ for $i < j$ with $i, j \in \mathbb{N}$ and $v$ appears — from a certain point on — in each tree and is marked "!". Let $R_1!, R_2!, \ldots$ denote their labels.

Then $R_0 \overset{u_1v_1}{\rightsquigarrow} R_1! \overset{u_2v_2}{\rightsquigarrow} R_2! \ldots R_i! \overset{u_{i+1}v_{i+1}}{\rightsquigarrow} \ldots$ with $R_0 = \{q_0\}$ is a macrostate sequence as defined in Remark 5.27: In case a node $v$ carries the marker "!", in the Safra construction, we had to apply step 6. Therefore, it is necessary that $v$ was a parent node, which implies that at some point a non-empty subset of final states was split off in step 2 of the Safra construction. Hence, then as stated by Remark 5.28, $\mathcal{B}$ accepts $\alpha$.

Now we show $L(\mathcal{B}) \subseteq L(\mathcal{M})$. Let $\alpha \in L(\mathcal{B})$. We have to prove the existence of a run $\rho'$ on $\alpha$ of the constructed 𝔐-𝔏-Muller automaton such that there is a node $v$ in the Safra trees of $\rho'$ that — from a certain point on — is a node of all

Safra trees in $\rho'$ and is marked "!" again and again. These two requirements satisfy the Muller acceptance condition, because $v$ is marked infinitely often in $\rho'$ and since $Q'$ is a finite set, there exists some Safra tree in $\mathrm{Inf}(\rho')$ with $v$ marked "!" which occurs infinitely often in $\rho$. Also, any Safra tree without $v$ is not in $\mathrm{Inf}(\rho')$, because from a certain point on $v$ is a node of all Safra trees in $\rho'$. Thus, Safra trees without $v$ occur only finitely many times. Then $\rho'$ is accepting and we obtain $\alpha \in L(\mathcal{M})$.

Say $q \in F$ occurs infinitely often in the run $\rho$ on $\alpha$ of $\mathcal{B}$. As there exists an accepting run of $\mathcal{B}$, the root of all Safra trees is non-empty, since the root macrostate of the $i$-th Safra tree in $\rho'$ contains $\rho(i)$.

In the simplest case, the root is marked "!" infinitely often, then the proof is complete.

If not, we have to consider another candidate for $v$. Then — from a certain point on — the state $q$ appears in the macrostate of the youngest son, because $q$ is a final state and was passed onto a child node in step 2 of Safra's construction or it appears in an older brother of the youngest son, due to horizontal merge operations. This can only happen finitely many times, since brother macrostate are disjoint and a proper subset of their parent macrostate. Therefore every node in a Safra tree has only finitely many sons as stated by Remark 5.26. Eventually, $q$ stays in the macrostate of some fixed node, this node will never be deleted by step 6 of Safra's construction, because otherwise the root would be marked "!" again. If this node is marked "!" infinitely often, the proof is complete. Otherwise, continue with this son in the same way as with the root above. Eventually, we must find a node which is marked "!" infinitely often in which $q$ occurs infinitely often, since the height of Safra trees is bounded, as we have seen in Remark 5.26. Thus, $\mathcal{M}$ accepts $\alpha$.

□

# Chapter 6

# Application to Monadic Chain Logic

In this chapter we present an application to monadic chain logic. Over structures where the notion of chains is meaningful, like tree structures, chain logic is a fragment of monadic second order logic in which set quantifications are restricted to chains. In [Tho92] it was shown that an extension of chain logic over the infinite binary tree by the equal level predicate that connects two vertices on the same tree level is decidable. We will further extend this result to infinitely branching trees. Moreover we add to this logical framework a connection between sibling sets, i.e. the children of a particular node. First, we recall the proof of the decidability of chain logic with the equal level predicate over the infinite binary tree.

## 6.1 Chain Logic over the Infinite Binary Tree $T_2$

Let us consider the tree structure $T_2 = (\{0,1\}^*, S_0, S_1, E)$ which describes the infinite binary tree with the two successor relations $S_0$, $S_1$, and the equal level predicate $E$. The successor relation $S_0(x,y)$ (respectively $S_1(x,y)$) holds if and only if $y$ directly succeeds $x$ following the left (respectively right) branch. The binary equal level predicate holds if and only if two vertices of a tree are on the same tree level. More formally we have:

- $S_0(u,v) :\Leftrightarrow v = u0$, $S_1(u,v) :\Leftrightarrow v = u1$,
- $E(u,v) :\Leftrightarrow |u| = |v|$.

An infinite *path* is a infinite sequence $v_0, v_1, \ldots$ of finite bit words, starting at the root $\varepsilon$. Every $v_{i+1}$ is the successor of $v_i$ in terms of $S_0, S_1$. A *chain* is a subset of a path, i.e. a linearly ordered subset of the universe.

Referring to chains, we can speak about the structure $T_2$ with equal level predicate using a fragment of MSO-logic, in which monadic second order quantifications are restricted to chains. We call this system *chain$_E$-logic.*

**Theorem 9 ([Tho92] )** *The chain$_E$-theory of $T_2 = (\{0,1\}^*, S_0, S_1, E)$ with equal level predicate is decidable.*

We claim that there is a reduction of the chain$_E$-theory of $T_2$ to the monadic second-order theory of $(\mathbb{N}, +1)$, i.e. to the monadic theory S1S of one successor which is decidable, see [Büc62].

For the proof it is convenient to use a simpler variant of chain logic of the same expressiveness, which replaces first-order variables by second order variables ranging over singletons. We call this variant chain$_{0,E}$-logic, which has the following atomic formulas:

- $X \subseteq Y$ ("$X$ is a subset of $Y$"),
- $\mathrm{Sing}(X)$ ("$X$ is a singleton"),
- $S_0(X,Y)$ (resp. $S_1(X,Y)$) ("$X, Y$ are singletons $\{x\}, \{y\}$ and we have $S_0(x,y)$ (resp. $S_1(x,y)$)"),
- $E(X,Y)$ ("$X, Y$ are singletons $\{x\}, \{y\}$ and $E(x,y)$ holds").

Every chain$_E$-formula can easily be rewritten as a chain$_{0,E}$-formula. This is done analogous to the translation of MSO-formulas to MSO$_0$-formulas as presented in Section 4.2.

As a preparation let us encode a chain $C$ in $(\{0,1\}^*, S_0, S_1, E)$ by a pair $(\alpha_C, \beta_C)$ of $\omega$-words over $\{0,1\}$. We only allow infinite paths. The idea is that the sequence $\alpha_C = \alpha_C(0)\alpha_C(1)\ldots$ codes the path of which $C$ is a subset, the sequence $\beta_C = \beta_C(0)\beta_C(1)\ldots$ codes membership in $C$ along the path. We can distinguish between three cases.

(1) $C = \emptyset$, then $\alpha_C = \beta_C = 0^\omega$.

(2) $C$ is finite, let $C = \{w_1, \ldots, w_k\}$, $w_i \in \{0,1\}^*$ and $\varepsilon \leq w_1 < \cdots < w_k$, then we define $\alpha_C = w_k 0^\omega$; this is the leftmost path of which $C$ is a subset. We define $\beta_c(i) = 1$ if and only if $\alpha_C(0)\ldots\alpha_C(i-1) \in C$.

(3) $C$ is infinite, let $C = \{w_1, w_2, \ldots\}$, then $\alpha_C$ is the common continuation of all $w_i$ and $\beta_C$ is defined as above. Then every $w_i$ is a prefix of $\alpha_C$ as in (2).

Now that we have established a coding of chains $C$ by pairs $(\alpha_C, \beta_C)$, we will use this correspondence to rewrite chain$_E$-formulas speaking about the structure $T_2$ as equivalent S1S-formulas.

As S1S-formulas are evaluated over the structure $(\mathbb{N}, +1)$ we identify a pair $(\alpha_C, \beta_C)$ with sets of natural numbers in the following way: Given an $\omega$-word $\alpha \in \{0,1\}^\omega$ we can identify $\alpha$ with a set $X$ such that $j \in X \Leftrightarrow \alpha(j) = 1$.

**Lemma 6.1** *Any chain$_{0,E}$-formula $\varphi(X_1, \ldots, X_n)$ can be rewritten as an S1S-formula $\varphi'(Y_1, Z_1, \ldots, Y_n, Z_n)$ such that for all chains $C_1, \ldots, C_n$ and the corresponding pairs $(\alpha_{C_1}, \beta_{C_1}), \ldots, (\alpha_{C_n}, \beta_{C_n})$ we have:*

$$(*)\qquad \begin{array}{c} (\{0,1\}^*, S_0, S_1, E) \models \varphi[C_1, \ldots, C_n] \\ \textit{if and only if } (\mathbb{N}, +1) \models \varphi'[\alpha_{C_1}, \beta_{C_1}, \ldots, \alpha_{C_n}, \beta_{C_n}]. \end{array}$$

*Proof.* By induction over chain$_0$-formulas we show that for any chain$_0$-formula $\varphi(X_1, \ldots, X_n)$ there is an S1S-formula $\varphi'(Y_1, Z_1, \ldots, Y_n, Z_n)$ such that $(*)$ holds. For the induction base we have to consider the atomic formulas $\mathrm{Sing}(X)$, $X_i \subseteq X_j$, $S_0(X_i, X_j)$, $S_1(X_i, X_j)$ and $E(X_i, X_j)$. The formulas are translated as follows:

- $\mathrm{Sing}(X)$

  A formula has to express that $Z$ is a singleton. This is expressed by $\varphi_{Sing}(Y, Z) := \exists x(Z(x) \wedge \forall x \neq y \rightarrow \neg Z(y))$

- $X_i \subseteq X_j$ is translated into $\varphi_{\subseteq}(Y_i, Z_i, Y_j, Z_j) :=$

$$\underbrace{\exists i\big(\varphi_{max}(i, Z_i)}_{\text{“}X_i\text{ is a finite chain”}} \wedge \underbrace{\forall j(Z_i(j) \rightarrow Z_j(j)))}_{\text{“}X_i \subseteq X_j\text{”}}$$

$$\underbrace{\wedge\ \forall j(j < i \rightarrow (Y_i(j) \leftrightarrow Y_j(j)))\big)}_{\text{“first part of path of }X_j\text{ corresp. to path of }X_i\text{”}}$$

$$\vee\ \big(\underbrace{\forall i \exists j(i < j \wedge Z_i(j))}_{\text{“}X_i\text{ is infinite“}} \wedge \underbrace{\forall j(Z_i(j) \rightarrow Z_j(j))}_{\text{“}X_i \subseteq X_j\text{”}} \wedge \underbrace{\forall j(Y_i(j) \leftrightarrow Y_j(j))\big)}_{\text{“path of }X_j\text{ corresp. to path of }X_i\text{”}}$$

  with $\varphi_{max}(x, X) := \forall y(X(y) \wedge x \neq y \rightarrow y < x)$ expressing that $x$ is the maximal element in $X$.

- $S_0(X_i, X_j)$

  For singletons $X_i = \{x_i\}$ at tree level $x$ and $X_j = \{x_j\}$ at tree level $y$, a formula has to express that they are successors on the same path. A respective formula is given by

$$\exists x \exists y\big(Z_i(x) \wedge Z_j(y) \wedge x + 1 = y \wedge \forall j(j < x \rightarrow (Y_i(j) \leftrightarrow Y_j(j)))\big)$$

  and that the last letter of $x_j$ is 0. This easily expressed by $\neg Y_j(y)$.

  An overall translation yields

$$\varphi_{S_0}(Y_i, Z_i, Y_j, Z_j) := \varphi_{Sing}(Y_i, Z_i) \wedge \varphi_{Sing}(Y_j, Z_j)\ \wedge$$

$$\exists x \exists y\big(Z_i(x) \wedge Z_j(y) \wedge x + 1 = y \wedge \forall j(j < x \rightarrow (Y_i(j) \leftrightarrow Y_j(j)) \wedge \neg Y_j(y)\big)$$

- $S_1(X_i, X_j)$

  A formula differs from the case $S_0(X_i, X_j)$ only that it has to express that the last letter of $x_j$ is 1. A translation is given by

$$\varphi_{S_1}(Y_i, Z_i, Y_j, Z_j) := \varphi_{Sing}(Y_i, Z_i) \wedge \varphi_{Sing}(Y_j, Z_j)\ \wedge$$

$$\exists x \exists y\big(Z_i(x) \wedge Z_j(y) \wedge x + 1 = y \wedge \forall j(j < x \rightarrow (Y_i(j) \leftrightarrow Y_j(j)) \wedge Y_j(y)\big)$$

- $E(X_i, X_j)$

  For singletons $X_i = \{x_i\}$ and $X_j = \{x_j\}$ a formula hat to express that they are on the same tree level $x$. A translation yields

  $$\varphi_E(Y_i, Z_i, Y_j, Z_j) := \exists x\big(Z_i(x) \land Z_j(x) \land \forall y(x \neq y \to \neg Z_i(y) \land Z_j(y))\big)$$

For the induction step it suffices to consider the connectives $\neg, \lor$ and the set quantifier $\exists$, since $\land, \to$ and $\forall$ are definable in terms of them. Let $\varphi, \psi$ be chain$_0$-formulas, by induction hypothesis the S1S-formulas $\varphi', \psi'$ are the respective translations. Then $\neg\varphi$ is translated to $\neg\varphi'$, the disjunction $\varphi \lor \psi$ is translated to $\varphi' \lor \psi'$. The formula $\exists X_i \varphi$ where $X_i$ is a free variable in $\varphi$ is translated to $\exists Y_i \exists Z_i \varphi'$.

□

This transformation applied to sentences yields a reduction of the chain$_E$-theory of $T_2 = (\{0,1\}^*, S_0, S_1, E)$ to S1S which is decidable.

## 6.2 Chain Logic over $T_{\mathbb{N}}$ with MSO on Siblings

In this section we extend the result of the previous section to the infinitely branching tree over $\mathbb{N}$. We will use the prefix relation instead of direct successor relations and moreover introduce predicates $\underline{P_i}$ which connect siblings, i.e. the children of a parent node.

Consider the tree structure $T_{\mathbb{N}} = (\mathbb{N}^*, \preceq, \underline{P_1}, \ldots, \underline{P_k}, E)$ where

- $u \preceq v :\Leftrightarrow u$ is a prefix of $v$;
- $\underline{P_i}(u_1, \ldots, u_k) :\Leftrightarrow u_1, \ldots, u_k$ are siblings $ua_1, \ldots, ua_k$, $u \in \mathbb{N}^*$ such that $(\mathbb{N}, +1) \models \varphi_i[a_1, \ldots, a_k]$ for $i \in \{1, \ldots, k\}$.

The new predicates $\underline{P_i}$ allow us to express S1S-definable properties on the sibling set. The meaning of each predicate $\underline{P_i}$ is defined by an S1S-formula $\varphi_i$. Speaking over chains, we will call chain$_E$-logic extended by these predicates chain$_E^{sib}$-logic. Alternatively, it is also possible to define infinitely many predicates $\underline{P_\varphi}$ for each possible S1S-formula $\varphi$. Both variants are equally expressive. Let us take a look at an example that illustrates a possible usage of this extension of chain$_E$-logic.

**Example 6.2** For instance, we can express that there is a chain, such that the last letters of its elements are in alternation even or odd. A respective chain$_E^{sib}$-formula is given by

$$\exists X \forall x \forall y\big(X(x) \land X(y) \land x \neq y \land x \preceq y \land \neg\exists z(X(z) \land x \preceq z \land z \preceq y \land x \neq z))$$
$$\to (\underline{P}_{even}(x) \leftrightarrow \underline{P}_{odd}(y))\big)$$

where $\varphi_{even}(u) = \forall X\big(X(0) \land \forall y(X(y) \to X(y+1+1)) \to X(u)\big)$, $\varphi_{odd}(u) = \neg\varphi_{even}(u)$. The formulas $\varphi_{even}(u)$ and $\varphi_{odd}(u)$ speak over the properties of siblings, expressing that and element $u$ is even or odd, respectively.

Our goal is to show that $\text{chain}_E^{sib}$-logic is a decidable extension of $\text{chain}_E$-logic.

**Theorem 10** *The $\text{chain}_E^{sib}$-theory of $T_{\mathbb{N}} = (\mathbb{N}^*, \preceq, \underline{P_1} \ldots, \underline{P_k}, E)$ is decidable.*

The proof is similar to the proof of Theorem 9 presented in the previous section. We claim that there is a reduction from the $\text{chain}_E^{sib}$-theory of $T_{\mathbb{N}}$ to the $\mathfrak{M}$-$\mathfrak{L}$-MSO theory where we choose $\mathfrak{M}$ as $(\mathbb{N}, +1)$ and $\mathfrak{L}$ as MSO. Thus an $\mathfrak{M}$-$\mathfrak{L}$-formula is an S1S-formula. Therefore, to increase readability, we will speak of $\text{MSO}^{\text{S1S}}$-formulas instead of $(\mathbb{N}, +1)$-MSO-MSO-formulas in this section.

As before, we use a simpler variant of $\text{chain}_E^{sib}$-logic, in which first-order variables are eliminated. We will refer to this variant as $\text{chain}_{0,E}^{sib}$-logic with atomic formulas $\text{Sing}(X)$, $X \subseteq Y$, $X \preceq Y$, $\underline{P_i}(X_1, \ldots, X_n)$ and $E(X, Y)$. Every $\text{chain}_E^{sib}$-formula can easily be rewritten as an equivalent $\text{chain}_{0,E}^{sib}$-formula analogous to the procedure as presented in Remark 4.9.

Similar to the previous section, we encode a chain $C$ in $T_{\mathbb{N}}$ by a pair $(\alpha_C, \beta_C) \in (\mathbb{N}^\omega \times \mathbb{B}^\omega)$ of $\omega$-words. The sequence $\alpha_C$ represents the leftmost infinite path of which $C$ is a subset, it can be interpreted as a sequence of "directions". Whereas the 0-1-sequence $\beta_C$ encodes membership in $C$ along the path as defined in the previous case of the infinite binary tree.

Now, we provide a translation from the $\text{chain}_{0,E}^{sib}$-theory into the $\text{MSO}^{\text{S1S}}$-theory.

For the corresponding pair $(\alpha_C, \beta_C)$ of a chain $C$ there exists a word model $\underline{\alpha}$ as introduce in Definition 5.9. For improved readability, we write $\underline{\alpha}(\alpha_C, \beta_C)$ instead of $\underline{\alpha}$.

**Lemma 6.3** *Any $\text{chain}_{0,E}^{sib}$-formula $\varphi(X_1, \ldots, X_n)$ can be rewritten as an $\text{MSO}^{\text{S1S}}$-formula $\varphi'$ such that for all chains $C_1, \ldots, C_n$ and the corresponding pairs $(\alpha_{C_1}, \beta_{C_1}), \ldots, (\alpha_{C_n}, \beta_{C_n})$ we have:*

$$(*) \qquad \begin{array}{c} T_{\mathbb{N}} \models \varphi[C_1, \ldots, C_n] \\ \textit{if and only if } \underline{\alpha}(\alpha_{C_1}, \beta_{C_1}, \ldots, \alpha_{C_n}, \beta_{C_n}) \models \varphi'. \end{array}$$

*Proof.* By induction on the structure of $\text{chain}_{0,E}^{sib}$-formulas we show that for any $\text{chain}_{0,E}^{sib}$-formula $\varphi(X_1, \ldots, X_n)$ there is an $\text{MSO}^{\text{S1S}}$-formula $\varphi'$ such that $(*)$ holds. First we define two predicates, which we shall use in the translated formulas. The predicate $P_{member,i}(x)$ is defined by the S1S-formula $\varphi_{member,i}(x_1, y_1, \ldots, x_n, y_n) := (y_i = 1)$. At position $x$ we determine the choice of the free variables $x_1, y_1, \ldots, x_n, y_n$ in $\varphi_{member,i}$. Thus, $P_{member,i}(x)$ expresses that at position $x$ is an element of a chain $C_i$. In addition, we define the predicate $P_{eqDir_{i,j}}(x)$ by the S1S-formula $\varphi_{eqDir_{i,j}}(x_1, y_1, \ldots, x_n, y_n) := (x_i = x_j)$. The predicate $P_{eqDir_{i,j}}(x)$ expresses that at position $x$ the paths of which the chains $C_i, C_j$ are a subset take the same directions. The atomic formulas are translated as follows:

- $\text{Sing}(X_i)$

Given a chain $C_i$, a formula has to express that the 0-1-sequence $\beta_{C_i}$ that encodes the membership in $C_i$ along the path equals 1 exactly once. This is expressed by

$$\varphi_{Sing_{X_i}} := \exists x\big(P_{member,i}(x) \wedge \forall y(x \neq y \rightarrow \neg P_{member,i}(y)\big)$$

- $X_i \subseteq X_j$

  Given chains $C_i$ and $C_j$, there are two properties a formula has to express:

  - If $C_i$ is a finite chain, then we have to express that the first part of the path of $C_j$ corresponds to the path of $C_i$. This is expressed by

    $$\exists x\big(P_{member,i}(x) \wedge \forall y(x < y \rightarrow \neg P_{member,i}(y)) \wedge \forall j(j < x \rightarrow P_{eqDir_{i,j}}(j))\big)$$

  - If $C_i$ is infinite, the path of $C_j$ always corresponds to the path of $C_i$. This is expressed by

    $$\big(\forall j\ P_{eqDir_{i,j}}(j)\big)$$

  - Furthermore, a formula has to express that every member of $C_i$ is also a member of $C_j$. This is expressed by

    $$\forall x(P_{member,i}(x) \rightarrow P_{member,j}(x))$$

  A combination yields

  $$\begin{gathered}\varphi_{X_i \subseteq X_j} := \exists x\big(P_{member,i}(x) \wedge \forall y(x < y \rightarrow \neg P_{member,i}(y)) \\ \wedge \forall j(j < x \rightarrow P_{eqDir_{i,j}}(j))\big) \vee \big(\forall j\ P_{eqDir_{i,j}}(j)\big) \\ \wedge \forall x(P_{member,i}(x) \rightarrow P_{member,j}(x))\end{gathered}$$

- $X_i \preceq X_j$

  Given singletons $C_i$ and $C_j$, a formula has to express that the first part of the path of $C_j$ corresponds to the path of $C_i$. A respective formula is given by $\varphi_{X_i \preceq X_j} :=$

  $$\exists x \exists y\big(x \leq y \wedge P_{member,i}(x) \wedge P_{member,j}(y) \rightarrow \forall j(j < x \rightarrow P_{eqDir_{i,j}}(j))\big)$$

- $E(X_i, X_j)$

  Given singletons $C_i$ and $C_j$, a formula has to express that there exists a position where both $\beta_{C_i}$ and $\beta_{C_j}$ equal 1 and otherwise both equal 0. A respective formula is given by $\varphi_{E(X_i,X_j)} :=$

  $$\exists x\big(P_{member,i}(x) \wedge P_{member,j}(x) \wedge \forall y(x \neq y \rightarrow \neg P_{member,i}(y) \wedge \neg P_{member,j}(y)\big)$$

- $\underline{P_i}(X_1, \ldots, X_k)$

  Given singletons $C_1, \ldots, C_k$, a formula has to express two properties :

- First of all, a formula has to express that the singletons are siblings. Therefore we introduce the formula $\varphi_{Sibling_{i,j}}(x) := \forall y\big(y < x \to P_{eqDir_{i,j}}(y)\big)$ which expresses that $C_i$ and $C_j$ are siblings.
- The meaning of $\underline{P_i}$ is defined by the S1S-formula $\varphi_i$, thus the formula has to express that the siblings fulfill the property. Therefore we introduce the predicate $P_{\varphi_i}$ which is also defined by $\varphi_i(x_1, \ldots, x_k)$.

An overall translation yields:

$$\varphi_{P_i} := \exists x\big( \bigwedge_{i\in\{1,\ldots,k\}} P_{member,i}(x) \wedge \bigwedge_{i\in\{2,\ldots,k\}} \varphi_{Sibling_{1,i}}(x) \wedge P_{\varphi_i}(x)\big)$$

For the induction step we consider only $\neg, \vee$ and $\exists$, since $\wedge, \to$ and $\forall$ are definable in terms of them. The cases $\neg$ and $\vee$ are straightforward. Let $\varphi$ and $\psi$ be chain$^{sib}_{0,E}$-formulas, and by induction hypothesis $\varphi'$ and $\psi'$ are the translations, respectively. Then $\neg\varphi$ is translated to the MSO$^{\text{S1S}}$-formula $\neg\varphi'$, and $\varphi \vee \psi$ is translated to the MSO$^{\text{S1S}}$-formula $\varphi' \vee \psi'$.

We now turn to a preparation to handle the existential set quantifier $\exists$. In an MSO$^{\text{S1S}}$-formula we group single appearances of predicates referring to the same position $x$ together to avoid unnecessary repetition. This can easily be done as predicates are defined by formulas. We introduce a new formula as a corresponding Boolean combination of the formulas of the previous appearances of predicates referring to $x$. We illustrate this in the following example.

**Example 6.4** Consider the formula

$$\exists x \exists y(P_1(x) \wedge \neg P_2(x) \to S(x, y) \wedge P_2(y) \wedge P_3(y)).$$

The employed predicates $P_1, P_2$, and $P_3$ are defined by the formulas $\varphi_1, \varphi_2$, and $\varphi_3$, respectively. We introduce two new predicates $Q(x)$ and $Q'(x)$. The predicate $Q(x)$ defined by the formula $\varphi_i := (\varphi_1 \wedge \neg\varphi_2)$, the predicate $Q'(x)$ defined by the formula $\varphi_j := (\varphi_2 \wedge \varphi_3)$. For instance, referring to position $x$, we replace $P_1(x) \wedge \neg P_2(x)$ by $Q(x)$. Altogether we obtain

$$\exists x \exists y\big(Q(x) \to S(x, y) \wedge Q'(y)\big)$$

It is easy to see that this formula is equivalent to the formula above.

Now we can derive a translation for the chain$^{sib}_{0,E}$-formula $\exists X_n \varphi(X_1, \ldots, X_n)$. By induction hypothesis, we can construct an equivalent MSO$^{\text{S1S}}$-formula to $\varphi$. Let $\varphi'$ denote this formula. In $\varphi'$, we replace any Boolean combination of predicates that refer to a fixed position $x$, by a single predicate $Q(x)$ as described above. Modified in this way, the structure of $\varphi'$ allows us to make a simple further modification such that the resulting formula is equivalent to $\exists X_n \varphi(X_1, \ldots, X_n)$. Recall that each predicate $Q(x)$ is defined by a formula $\varphi(x_1, y_1, \ldots, x_n, y_n)$, we redefine $Q(x)$ by the formula $\exists x_n \exists y_n \varphi(x_1, y_1, \ldots, x_n, y_n)$. Thereby we obtain an equivalent MSO$^{\text{S1S}}$-formula to $\exists X_n \varphi(X_1, \ldots, X_n)$.

Let us turn to the correctness of the construction: The chain $C_n$ induces the choice of the variables $x_n, y_n$ in every formula $Q_J$ at every position in the corresponding word model. Vice versa, the choice of the variables $x_n, y_n$ for every position induces sequences $\alpha_{C_n}, \beta_{C_n}$ which in turn induce a chain $C_n$.

□

Applied to a chain$_{0,E}^{sib}$ sentence $\varphi$, this transformation yields an equivalent MSO$^{\mathrm{S1S}}$-sentence $\varphi'$. Thus, we obtain a reduction of the chain$_{0,E}^{sib}$-theory of $T_{\mathbb{N}}$ to the MSO$^{\mathrm{S1S}}$-theory. This theory is decidable, as a consequence of Theorem 6 that states that satisfiability of $\mathfrak{M}$-$\mathfrak{L}$-MSO sentences is decidable if and only if the $\mathfrak{L}$-theory of $\mathfrak{M}$ is decidable. In this case the $\mathfrak{L}$-theory of $\mathfrak{M}$ is the logical system S1S which is decidable. This completes the proof of Theorem 10.

## 6.3 Chain Logic over Tree Iterations

**Definition 6.5** Let $\mathfrak{M} = (\Sigma, R_1, \dots, R_m)$ be a structure. We define the tree iteration

$$\mathfrak{M}^* = (\Sigma^*, \preceq, \underline{R_1}, \dots, \underline{R_m})$$

where, for every relation $R_i$ with arity $n_i$,

$$\underline{R_i}(ua_1, \dots, ua_{n_i}) :\Leftrightarrow u \in \Sigma^*,\ a_1, \dots, a_{n_i} \in \Sigma \text{ and } R_i(a_1, \dots, a_{n_i})$$

Additionally, we add the equal level predicate $E$ and define the tree iteration

$$\mathfrak{M}^*_E = (\mathfrak{M}^*, E) = (\Sigma^*, \preceq, \underline{R_1}, \dots, \underline{R_m}, E).$$

We mention two related results. In [KL06] Kuske and Lorey proved the following result.

**Theorem 11 ([KL06])** *The chain-theory of* $\mathfrak{M}^* = (\Sigma^*, \preceq, \underline{R_1}, \dots, \underline{R_m})$ *is decidable if the FO-theory of* $\mathfrak{M}$ *is decidable.*

This result was further extended in [Bès08], where the structure $\mathfrak{M}^*$ was replaced by the structure

$$\hat{\mathfrak{M}}^*_E = (\Sigma^*, \preceq, \underline{\hat{R}_1}, \dots, \underline{\hat{R}_m}, E)$$

where, for every relation $R_i$ with arity $n_i$,

$$\underline{\hat{R}_i}(u_1a_1, \dots, u_{n_i}a_{n_i}) :\Leftrightarrow \text{if all } u_j\text{'s have the same length and } R_i(a_1, \dots, a_{n_i}).$$

In $\hat{\mathfrak{M}}^*_E$ the equal level predicate was added and furthermore a predicate $\underline{\hat{R}_i}$ is defined over an entire tree level and is not restricted to a sibling set. It should be noted that every predicate $\underline{R_i}$ can be defined in $\hat{\mathfrak{M}}^*_E$. Bès showed the following result.

**Theorem 12 ([Bès08])** *The chain*$_E$*-theory of* $\hat{\mathfrak{M}}^*_E = (\Sigma^*, \preceq, \underline{\hat{R}_1}, \dots, \underline{\hat{R}_m}, E)$ *is decidable if the FO-theory of* $\mathfrak{M}$ *is decidable.*

We now turn to an extension of chain-logic by MSO-definable properties over siblings.

**Definition 6.6** An MSO-expansion of a structure $\mathfrak{M} = (\Sigma, R_1, \ldots, R_m)$ is defined by MSO-formulas $\varphi_1, \ldots, \varphi_k$ speaking over the structure $\mathfrak{M}$. We define the structure

$$\mathfrak{M}^*_{MSO} = (\Sigma^*, \preceq, \underline{R_1}, \ldots, \underline{R_m}, \underline{P_1}, \ldots, \underline{P_k})$$

where for every $i \in \{1, \ldots, k\}$, $\underline{P_i}(ua_1, \ldots, ua_{n_i}) :\Leftrightarrow u \in \Sigma^*$, $a_1, \ldots, a_{n_i} \in \Sigma$ and $\mathfrak{M} \models \varphi_i[a_1, \ldots, a_{n_i}]$.

In an MSO-expansion a predicate $\underline{P_i}$ allows us to express MSO-definable properties on a sibling set, because we can specify the meaning of $\underline{P_i}(X_1, \ldots, X_k)$ by any MSO-formula $\varphi_i$ speaking over the structure $\mathfrak{M}$. Recall, that the predicates $\underline{R_i}$ also connect siblings, but can only express that for a sibling set that the relation $R_i$ holds. Thus, predicates $\underline{P_i}$ are more expressive.

**Remark 6.7** When we consider the MSO-expansion of a structure $\mathfrak{M} = (\Sigma, R_1, \ldots, R_m)$ the predicates $\underline{R_1}, \ldots, \underline{R_m}$ are special cases of predicates $\underline{P_i}$. Every predicate $\underline{R_i}(x_1, \ldots, x_n)$ is equivalent to the predicate $\underline{P_{R_i}}$ defined by the formula $R_i(x_1, \ldots, x_n)$.

**Theorem 13** *The chain$^{sib}_E$-theory of $\mathfrak{M}^*_{MSO,E} = (\Sigma^*, \preceq, \underline{R_1}, \ldots, \underline{R_m}, \underline{P_1}, \ldots, \underline{P_k}, E)$ with equal level predicate is decidable if the MSO-theory of $\mathfrak{M}$ is decidable.*

We claim that there is a reduction from the chain$^{sib}_E$-theory of $\mathfrak{M}^*_{MSO,E}$to the $\mathfrak{M}$-$\mathfrak{L}$-MSO-theory where we fix $\mathfrak{L}$ as MSO-logic. We call this logical framework MSO$^{\mathfrak{M}}$-logic. As in the previous section, we use chain$^{sib}_{0,E}$-logic in which only second-order variables occur with atomic formulas $\text{Sing}(X), X \subseteq Y, X \preceq Y, \underline{R_i}(X_1, \ldots, X_k)$, $\underline{P_j}(X_1, \ldots, X_k)$, and $E(X, Y)$.

Again, we encode a chain $C$ by a pair $(\alpha_C, \beta_C)$ of $\omega$-words. Since we speak over structures $\mathfrak{M}^*_{MSO,E}$we have to consider any domain of the base structure $\mathfrak{M}$. This means the domain does not necessarily contain the symbols $0, 1$. Therefore we have to alter our coding of a chain $C$ slightly. We keep the pair $(\alpha_C, \beta_C)$ of $\omega$-words over $\Sigma$ but redefine $\beta_C$ only, thus:

- $\alpha_C$ encodes the infinite path of which $C$ is a subset.
- $\beta_C$ encodes membership in $C$ along the path where $\alpha_C(i) = \beta_C(i)$ iff $\alpha_C(0) \ldots \alpha_C(i-1) \in C$.

Now we provide a translation from the chain$^{sib}_{0,E}$-theory of $\mathfrak{M}^*_{MSO,E}$to the MSO$^{\mathfrak{M}}$-theory.

**Lemma 6.8** *Any chain$^{sib}_{0,E}$-formula $\varphi(X_1, \ldots, X_n)$ over $\mathfrak{M}^*_{MSO,E} = (\Sigma^*, \preceq, \underline{R_1}, \ldots, \underline{R_m}, \underline{P_1}, \ldots, \underline{P_k}, E)$ can be rewritten as an MSO$^{\mathfrak{M}}$-formula $\varphi'$ such that for all chains $C_1, \ldots, C_n$ and the corresponding pairs $(\alpha_{C_1}, \beta_{C_1}), \ldots, (\alpha_{C_n}, \beta_{C_n})$ we have:*

$$(*)\qquad \begin{array}{c}\mathfrak{M}^*_E \models \varphi[C_1,\dots,C_n]\\ \textit{if and only if } \underline{\alpha}(\alpha_{C_1},\beta_{C_1},\dots,\alpha_{C_n},\beta_{C_n}) \models \varphi'.\end{array}$$

*Proof.* By induction over the structure of $\text{chain}^{sib}_{0,E}$-formulas over $\mathfrak{M}^*_E$ we show that for any $\text{chain}^{sib}_{0,E}$-formula $\varphi(X_1,\dots,X_n)$ over $\mathfrak{M}^*_E$ there is an $\text{MSO}^{\mathfrak{M}}$-formula $\varphi'$ such that $(*)$ holds.

For the induction basis we have to consider the atomic formulas $\text{Sing}(X)$, $X_i \subseteq X_j$, $X_i \preceq X_j$, $\underline{R_i}(X_1,\dots,X_n)$, $\underline{P_i}(X_1,\dots,X_k)$, $E(X_i,X_j)$ and $\underline{P_i}(X_1,\dots,X_k)$. Note that these formulas are also the same formulas we considered over $T_{\mathbb{N}} = (\mathbb{N}^*, \preceq, \underline{P_1},\dots,\underline{P_k}, E)$, the only addition is $\underline{R_i}(X_1,\dots,X_n)$. Due to this similarity the translation of the atomic formulas requires only a slight adaption of the translations presented in Lemma 6.3. Recall that a chain $C$ is encoded by a pair $(\alpha_C,\beta_C)$; in case of the structure $T_{\mathbb{N}}$ the sequence $\beta_C$ was defined over $\{0,1\}$, in this more general case $\beta_C$ is defined over $\Sigma$. We introduced two predicates $P_{member,i}(x)$ and $P_{eqDir_{i,j}}(x)$ with the meaning at position $x$ is an element of chain $C_i$ and that at position $x$ the paths of $C_i, C_j$ take the same direction. We have to redefine the predicate $P_{member,i}(x)$ since the membership is coded no longer by a 0-1-sequence. Thus we redefine $P_{member,i}(x)$ by the formula $\varphi_{member_i}(x_1,y_1,\dots,x_n,y_n) := (x_i = y_i)$. Now we can use the same translations for the atomic formulas as presented in the previous section. What is left is to construct a translation for the atomic formula $\underline{R_i}(X_1,\dots,X_n)$. A respective formula is given by

$$\varphi_{\underline{R_i}} := \exists x\big(\bigwedge_{i\in\{1,\dots,k\}} P_{member,i}(x) \wedge \bigwedge_{i\in\{2,\dots,k\}} \varphi_{Sibling_{1,i}}(x) \wedge P_{R_i}(x)\big)$$

In the induction step we can proceed in the same way as presented in the induction step in Lemma 6.3 in the previous section.

□

Applying the translation to a sentence yields an equivalent $\text{MSO}^{\mathfrak{M}}$ sentence. By Theorem 6 we obtain that the satisfiability of an $\text{MSO}^{\mathfrak{M}}$ sentence is decidable if the MSO-theory of $\mathfrak{M}$ is decidable. Thus we obtain a reduction of the $\text{chain}^{sib}_E$-theory of $\mathfrak{M}^* = (\Sigma^*, \preceq, \underline{R_1},\dots,\underline{R_m},\underline{P_1},\dots,\underline{P_k})$ which is decidable if the MSO-theory of $\mathfrak{M}$ is decidable. This completes the poof of Theorem 13.

Let us take a look at how we can define the previous considered cases as tree iterations: The tree structure $T_{\mathbb{N}} = (\mathbb{N}^*, \preceq, \underline{P_1},\dots,\underline{P_k}, E)$ is the MSO-expansion of the base structure $\mathfrak{N} = (\mathbb{N}, +1)$. For the infinite binary tree $T_2 = (\{0,1\}^*, S_0, S_1, E)$ we choose $\mathfrak{M}_{\mathfrak{B}} = (\{0,1\}, 0, 1)$ as base structure. The corresponding tree iteration is given by $\mathfrak{M}^*_{\mathfrak{B},E} = (\{0,1\}^*, \preceq, \underline{R_0}, \underline{R_1}, E)$. The relations $S_0(x,y)$ and $S_1(x,y)$ are definable in $\mathfrak{M}^*_{\mathfrak{B},E}$. For instance, we can define $S_0(x,y) := x \neq y \wedge x \preceq y \wedge \neg\exists z(x \preceq z \wedge z \preceq y \wedge x \neq z) \wedge \underline{R_0}(y)$. We can define the relation $S_1(x,y)$ analogously.

# Chapter 7

# Conclusion

In this work we focused on the connection between finite automata over infinite alphabets and logic. We have generalized the equivalence of finite automata recognizability and monadic second-order logic definability to the case of infinite alphabets.

In Chapter 3 we have introduced an automaton model that recognizes relations of words over infinite alphabets. We defined the automaton model in the form of synchronous $n$-tape automaton with transitions specified by formulas in a suitable logical framework. Thus we were able to provide an automaton model with good closure and decision properties. We proved that the class of recognizable relations by this automaton model is closed under Boolean operations and projection. Regarding decision problems, such as the emptiness problem the decidability depends on the decidability of the logical formalism used in the transitions.

In Chapter 4 we focused on the connection between the automaton model and monadic second order logic. We showed the general equivalence of monadic second-order interpreted over words over infinite alphabets and finite automaton recognizability. Therefore we provided a translation from formulas to automata and vice versa.

In Chapter 5 we have seen that the same automaton model equipped with Büchi acceptance recognizes relations of $\omega$-words over infinite alphabets. We were able to extend the previous results to the case of relations of $\omega$-words and also showed that monadic second-order logic over relations of $\omega$-words has the same expressiveness as the automaton model. Furthermore, we have seen how we can obtain an equivalent deterministic automaton model if we use Muller acceptance instead of Büchi acceptance.

In chapter 6 we introduced an extension of monadic chain logic that allows to express monadic second-order properties on the sibling set, i.e. direct successors of a vertex. We have shown that this form of chain logic over a tree iteration $\mathfrak{M}^*$ is decidable if the monadic second-order theory of the base structure $\mathfrak{M}$ is decidable.

For a possible extension of the present results we mention two directions. The first would enable to express monadic second-order properties not only on the sibling set, i.e. direct successors of a vertex, but on a whole level of the tree. This extension is however too strong to allow decidability — a proof idea is sketched in [Tho09] since weak monadic second-order logic with equal level can then be expressed, which is known to undecidable.

A second approach is to extend the (weak) tree model $\mathfrak{M}^*$ as presented in Chapter 6 by the "strong tree iteration" in which a stronger connection between tree level is possible, thus a stronger connection between successive elements of chains. In particular, equality of successive elements becomes expressible (which is not covered be our results see Example 3.9). This extension would have to use Muchnik's theorem of tree iterations where the unary "clone predicate" is included, which states that the last two letters of a word are the same (see [BB02]).

# Bibliography

[BB02] D. Berwanger and A. Blumensath. The monadic theory of tree-like structures. In E. Grädel, W. Thomas, and T. Wilke, editors, *Automata, Logics, and Infinite Games*, volume 2500 of *Lecture Notes in Computer Science*, pages 43–60. Springer, 2002.

[Bès08] A. Bès. An Application of the Feferman-Vaught Theorem to Automata and Logics for Words over an Infinite Alphabet. In *Logical Methods in Computer Science*, volume 4, pages 1–23, 2008.

[Büc60] J.R. Büchi. Weak second-order arithmetic and finite automata. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, 6(1-6):66–92, 1960.

[Büc62] J.R. Büchi. On a decision method in restricted second order arithmetic. In *Logic, Methodology and Philosophy of Science: Proceedings of the 1960 International Congress*, pages 1–11. Stanford Univ. Press, 1962.

[EES69] S. Eilenberg, C.C. Elgot, and J.C. Shepherdson. Sets recognized by n-tape automata. *Journal of Algebra*, 13(4):447–464, 1969.

[EF95] H.-D. Ebbinghaus and J. Flum. *Finite model theory.* Perspectives in Mathematical Logic. Springer, 1995.

[EFT07] H.-D. Ebbinghaus, J. Flum, and W. Thomas. *Einführung in die mathematische Logik.* Spektrum Akademischer Verlag, Heidelberg, 5 edition, 2007.

[Elg61] C.C. Elgot. Decision problems of finite automata design and related arithmetics. *Trans. Amer. Math. Soc*, 98:21–51, 1961.

[Göd31] K. Gödel. Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme, I. *Monatshefte für Mathematik und Physik*, 38:173–198, 1931.

[KL06] D. Kuske and M. Lohrey. Monadic chain logic over iterations and applications to pushdown systems. In *Logic in Computer Science, 2006*, pages 91–100. IEEE Computer Society, 2006.

[McN66] R. McNaughton. Testing and generating infinite sequences by a finite automaton. *Information and control*, 9(5):521–530, 1966.

[Mul63] D.E. Muller. Infinite sequences and finite machines. In *Proceedings of the Fourth Annual Symposium on Switching Circuit Theory and Logical Design*, pages 3–16. IEEE, 1963.

[Pre29] M. Presburger. Über die Vollständigkeit eines gewissen Systems der Arithmetik ganzer Zahlen, in welchem die Addition als einzige Operation hervortritt. In *Comptes Rendus du I congres de Mathématiciens des Pays Slaves*. Warszawa, 1929.

[Rab69] M.O. Rabin. Decidability of second-order theories and automata on infinite trees. *Trans. Amer. Math. Soc*, 141(1):1–35, 1969.

[Ram30] F.P. Ramsey. On a problem of formal logic. *Proceedings of the London Mathematical Society*, 2(1):264, 1930.

[Rog02] M. Roggenbach. Determinization of Büchi-Automata. In E. Grädel, W. Thomas, and T. Wilke, editors, *Automata, Logics, and Infinite Games*, volume 2500 of *Lecture Notes in Computer Science*, pages 43–60. Springer, 2002.

[Saf88] S. Safra. On the complexity of omega-automata. In *Proc. 29th IEEE Symp. Found. of Comp. Sci*, pages 319–327, 1988.

[Sko31] T. Skolem. Über einige Satzfunktionen in der Arithmetik. *Skrifter Vitenskapsakadetiet i Oslo, I*, 7:1–28, 1931.

[Tar48] A. Tarski. *A Decision Method for Elementary Algebra and Geometry.* The RAND Corporation, Santa Monica, California, 1948. edited by J.C.C. McKinsey.

[Tho87] W. Thomas. On chain logic, path logic, and first-order logic over infinite trees. In *Proceedings, Symposium on Logic in Computer Science*, pages 245–256. IEEE Computer Society, 1987.

[Tho90] W. Thomas. Automata on infinite objects. In *Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics (B)*, pages 133–192. Elsevier and MIT Press, 1990.

[Tho92] W. Thomas. Infinite trees and automation-definable relations over omega-words. *Theor. Comput. Sci.*, 103(1):143–159, 1992.

[Tho97] W. Thomas. Languages, automata, and logic. In G. Rozenberg and A. Salomaa, editors, *Handbook of formal languages, vol. 3*, pages 389–455. Springer, New York, 1997.

[Tho09] W. Thomas. Path logics with synchronization. In K. Lodaya, M.Mukund, and R.Ramanujam, editors, *Perspectives in Concurrency Theory*, IARCS-Universities, pages 469–481. Universities Press, 2009.

[Tra62] B.A. Trakhtenbrot. Finite automata and monadic second order logic. *Siberian Math. J*, 3:101–131, 1962.