RHEINISCH-
WESTFÄLISCHE
TECHNISCHE
HOCHSCHULE
AACHEN

Bachelor Thesis

Finite Automata with Infinite Structures as Alphabet: Closure Properties and Decidability Results

Christopher Czyba
Matrikelnummer: 300760

March 10, 2015

Gutachter:
Prof. Dr. Wolfgang Thomas
Priv.-Doz. Dr. Christof Löding

Betreuer:
Wolfgang Thomas

Abstract

In this thesis, we will give an overview over an existing automata model for infinite alphabets which uses an infinite structure and logic to retain a finite description of the automaton. We will then extend the model and show that the usual closure properties still hold. We will also look into the non-emptiness problem and other closely related problems for the extended model and present decidability results.

Erklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit selbständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten und nicht veröffentlichten Schriften entnommen sind, sind als solche kenntlich gemacht.

Aachen, March 10, 2015

(Christopher Czyba)

Contents

1 Introduction

Current automata models are restricted to a finite alphabet. For computer scientists however, infinite domains, like the natural or rational numbers, are frequently used and therefore some of the most important domains.

Infinite alphabets have one critical restriction compared to finite alphabets. It is in general impossible to represent an subset as a finite list. Transitions of automata over a finite alphabet use such lists. Since we still need a finite description of the automaton, we need to express those subsets in a different form.

In order to keep a finite description of an automaton, we will use an arbitrary structure $\mathcal{M} = (M, R_1..., R_r, F_1, ..., F_f)$ where M is the, generally infinite, domain of $\mathcal{M}$, R_i relations and F_i functions on the domain M. Furthermore we require a logic $\mathcal{L}$ by which we specify properties of M or, if we build words over the tuple alphabet M^n, properties of tuples from M.

The classical approach uses unary relations $P_a(x)$, which express that for position x there is the letter a. We will replace those by predicates P_φ with the semantic being $P_\varphi(x)$ iff x satisfies the property described by the $\mathcal{M}$-$\mathcal{L}$-formula φ. Together $\mathcal{M}$ and $\mathcal{L}$ form a tuple $(\mathcal{M}, \mathcal{L})$, which we will call alphabet frame, as it was named in [Spe13].

In the following we will introduce automata using such an alphabet frame as alphabet and as a part of the transition relation. For those models, we will examine the decidability of the non-emptiness and other decision problems as well as the closure under union, intersection and complementation.

2 $\mathcal{M}$-$\mathcal{L}$-Automata

In this section we will introduce the basic model which we will use and extend throughout this work. It is based on the approach mentioned (for special cases) in [Bès08] and was later refined in [Spe13]. Since it is very similar to Nondeterministic Finite Automata or Nondeterministic Büchi Automata, we will only introduce the model, state some of the results and the basic ideas on how to prove them. Finally, we will mention some problems that occur due to the infinite alphabet, which have to be addressed differently.

2.1 Definitions

Definition 2.1. Let $(\mathcal{M}, \mathcal{L})$ be an alphabet frame where $\mathcal{M}$ is a structure with the domain M. An $\mathcal{M}$-$\mathcal{L}$-Automaton over finite words of n-tuples of M-elements is of the form

$$\mathfrak{A} = (Q, M^n, q_0, \Delta, F)$$

where

- Q is a finite set of states
- M^n is the input alphabet
- $q_0 \in Q$ is the initial state
- $\Delta \subseteq Q \times \Psi_n \times Q$ is the finite transition relation, where Ψ_n contains all $\mathcal{M}$-$\mathcal{L}$-formulas with n free variables.
- and finally $F \subseteq Q$ is the set of accepting states.

Given a word $w = w_1...w_m$ where each $w_i = \begin{pmatrix} w_{i,1} \\ \vdots \\ w_{i,n} \end{pmatrix} \in M^n$. Then a run ρ of $\mathfrak{A}$ on the word w is given by the finite sequence of states with $\rho = \rho(0)...\rho(m)$ with $\rho(0) = q_0$, such that for every $0 < i \leq m$ exists an $\mathcal{M}$-$\mathcal{L}$-formula $\varphi(x^1, ..., x^n)$ and a transition $(\rho(i-1), \varphi, \rho(i)) \in \Delta$ satisfying

$$\mathcal{M} \models \varphi[w_i(1), ..., w_i(n)].$$

where $w_i(j)$ is the j-th element of w_i. A run ρ of $\mathfrak{A}$ on w is successful if $\rho(m) \in F$. We will say $\mathfrak{A}$ accepts w, if there is a successful run. Furthermore we denote $L(\mathfrak{A})$ the set of finite words over M^n accepted by $\mathfrak{A}$.

If we use them to recognize infinite words over M^n, we will use it as a Büchi Automaton as follows:

Definition 2.2. An $\mathcal{M}$-$\mathcal{L}$-Büchi-Automaton over n-tuples for an alphabet frame $(\mathcal{M}, \mathcal{L})$ where $\mathcal{M}$ has the domain M is defined as

$$\mathfrak{A} = (Q, M, q_0, \Delta, F)$$

where the respective parts are as in the previous definition. The difference is in the acceptance, which is now defined over ω-words: If $\alpha = \alpha_1\alpha_2...$ is an ω-word over M^n, with each $\alpha_i = \begin{pmatrix} \alpha_{i,1} \\ \vdots \\ \alpha_{i,n} \end{pmatrix} \in M^n$, a run of $\mathfrak{A}$ is an infinite sequence of states $\rho = \rho(0)\rho(1)...$ with $\rho(0) = q_0$, s.t. for every $i > 0$ there exists an $\mathcal{M}$-$\mathcal{L}$-formula $\varphi(x^1, ..., x^n)$ and a transition $(\rho(i-1), \varphi, \rho(i)) \in \Delta$ satisfying

$$\mathcal{M} \models \varphi[\alpha_i(1), ..., \alpha_i(n)]$$

where $\alpha_i(j)$ is the j-th elements of α_i. A run ρ of $\mathfrak{A}$ on α is successful if $\forall i \exists j > i$ such that $\rho(j) \in F$, which we will call the Büchi-acceptance condition, cf. [Tho90]. As with $\mathcal{M}$-$\mathcal{L}$-Automata, we say $\mathfrak{A}$ accepts a word α if its run on $\mathfrak{A}$ is successful. Similarly $L(\mathfrak{A})$ is the set of infinite words α which are accepted by $\mathfrak{A}$.

As you can see, the only real difference between Nondeterministic Finite Automata, resp. Nondeterministic Büchi Automata, is that the transition relation now uses $\mathcal{M}$-$\mathcal{L}$-formulas to describe a set of letters which are allowed to take the transition.

2.2 Non-emptiness Problem

Theorem 2.3. Let $(\mathcal{M}, \mathcal{L})$ be an alphabet frame. If the logic $\mathcal{L}$ for the structure $\mathcal{M}$ is decidable then

- The non-emptiness problem for $\mathcal{M}$-$\mathcal{L}$-Automata is decidable as well.
- The non-emptiness problem for $\mathcal{M}$-$\mathcal{L}$-Büchi-Automata is decidable as well.

The proofs are straightforward. The decidability is required to check whether there exists a letter which can satisfy a transition or not. After removing all unsatisfiable transitions the procedure is the same as for Nondeterministic Finite Automata or respectively Nondeterministic Büchi Automata. Details on the proof can be found in [Spe13].

While we could cover the remaining closure properties here, since we will extend this automata model in the next sections in a way such that $\mathcal{M}$-$\mathcal{L}$-Automata will be considered a special case of that model, we will not cover those properties here but in the latter sections of this work. Instead we will mention one shortcoming of $\mathcal{M}$-$\mathcal{L}$-Automata.

2.3 Shortcomings of $\mathcal{M}$-$\mathcal{L}$-Automata

For Finite Automata or Büchi Automata, we can allow the automaton to look a finite amount of letters back without changing the expressiveness of the model, since a finite look-back for a finite alphabet are still finitely many possibilities. They can simply be encoded in the state set. For $\mathcal{M}$-$\mathcal{L}$-Automata that is not possible anymore. Due to the input alphabet is infinite,

they can not encode the look-back in the state set, since that would lead to an infinite description of the automaton. In fact, consider the language $L = \{w \in \mathbb{N}^* \mid w = nn, n \in \mathbb{N}\}$. It is impossible to recognize this language with $\mathcal{M}$-$\mathcal{L}$-Automata, since they would need an infinite state space. In order to address that problem, we will extend the $\mathcal{M}$-$\mathcal{L}$-Automata in the next section by allowing a look-back of one character.

The concept of allowing a look-back is a more generalized approach of the idea of a 'clone predicate' C, which is true if the letter at position s coincides with the previous letter. By using this special predicate, we would be able to recognize the language L. That approach however, we would still be unable to recognize languages like $\{n(n+1) \mid n \in \mathbb{N}\}$, which is why we will allow look-back for arbitrary formulas.

The idea of such a clone predicate appeared in the context of 'tree iteration' and was first introduced by Muchnik (see [Sem84]). The predicate is used to define unraveled structures. For a relational structure $\mathcal{M} = (M, R_1, ..., R_n)$, the unraveled structure $\mathcal{M}^*$ is defined as $\mathcal{M}^* = (M^*, S^M, C^M, R_1^+, ..., R_n^+)$, where

- $S^M(x, xd)$ for every $w \in M^*, d \in M$,
- $C^M(xdd)$ for all $x \in M^*, d \in M$,
- and finally $R_i^+(xd_1, ..., xd_k)$ for $x \in M^*, d \in M$, if $R_i(d_1, ..., d_k)$,

cf. [Tho97]. Muchnik's Theorem states that the MSO-theory of $\mathcal{M}^*$ is decidable if the MSO-theory of $\mathcal{M}$ is decidable and was proven by Walukiewicz (see [Wal96]).

3 Strong Automata

As already mentioned, the model we introduced in the previous chapter is severely restricted, because it is not able to compare and modify successive letters. For example, it is impossible to recognize the language $L = \{01 \cdots n \mid n \in \mathbb{N}\}$, if we use $((\mathbb{N}, +1, 0), \text{FO-logic})$ as alphabet frame. To address that problem we will introduce a new model of automaton, the Strong Automaton, as done in [Spe13].

Strong Automata will feature transitions labels of the form $\psi(x_1, .., x_n, y_1, .., y_n)$, where x_i is the i-th component of the previously read letter and y_i is the i-th the component of the current letter. Therefore they can compare successive letters and, for example, accept the language L. We will show some decidability results for this type of automaton in this chapter. Before we cover that, however, we will begin with the definition of the automaton model and give some examples.

3.1 Definitions

Definition 3.1. Let $(\mathcal{M}, \mathcal{L})$ be an alphabet frame where M is the domain of $\mathcal{M}$. A Strong Automaton over finite words of n-tuples of M-elements is defined as

$$\mathfrak{A} = (Q, M^n, q_1, \Delta, F)$$

where

- Q is a finite set of states
- the input alphabet is M^n, where M is the domain of $\mathcal{M}$
- $q_1 \in Q$ is the initial state
- $\Delta \subseteq Q \times (\Psi_n \cup \Psi_{2n}) \times Q$ is the finite transition relation where Ψ_n and Ψ_{2n} are sets of $\mathcal{M}$-$\mathcal{L}$-formulas with n respectively $2n$ free variables.
- and lastly $F \subseteq Q$ is the set of accepting states.

A run ρ of $\mathfrak{A}$ on the word $w = a_1...a_m$ with $a_i = \begin{pmatrix} a_{i,1} \\ \vdots \\ a_{i,n} \end{pmatrix} \in M^n$ is a finite sequence $\rho = \rho(0)...\rho(m)$ where

- $\rho(0) = q_1$
- $\rho(1) = q$, where $(q_1, \psi_{q_1,q}(x_1, .., x_n), q) \in \Delta$ and $\mathcal{M} \models \psi_{q_1,q}[a_1(1), .., a_1(n)]$
- $\rho(i) = q$ if $\rho(i-1) = p$, $(p, \psi_{q,p}(x_1, .., x_n, y_1, .., y_n), q) \in \Delta$ and $\mathcal{M} \models \psi_{p,q}[a_{i-1}(1), .., a_{i-1}(n), a_i(1), .., a_i(n)]$ for $1 < i \leq m$

where $a_i(j)$ is the j-th component of letter a_i. We call such a run ρ of $\mathfrak{A}$ for the word w successful if $\rho(m) \in F$. We say $\mathfrak{A}$ accepts w if it has a successful run. Furthermore we define $L(\mathfrak{A})$ as the set of finite words over M^n which are accepted by $\mathfrak{A}$.

With Strong Automata you can, as already mentioned, recognize the language $L = \{01 \cdots n \mid n \in \mathbb{N}\}$ for the alphabet frame $((\mathbb{N}, +1, 0), \text{FO logic})$. For example the automaton

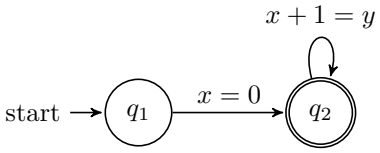

Figure 3.1: Strong Automaton recognizing $\{01 \cdots n \mid n \in \mathbb{N}\}$

accepts the language L.

As with $\mathcal{M}$-$\mathcal{L}$-Automata, we can also use the Strong Automata to recognize infinite words.

Definition 3.2. Let $(\mathcal{M}, \mathcal{L})$ be am alphabet frame where M is the domain of $\mathcal{M}$. A Strong Büchi Automaton over infinite words of n-tuples of M-elements is defined as

$$\mathfrak{A} = (Q, M^n, q_1, \Delta, F)$$

where the semantics remain the same as for Strong Automata.

A run ρ of $\mathfrak{A}$ on the word $\alpha = \alpha_1...$ with $\alpha_i = \begin{pmatrix} \alpha_{i,1} \\ \vdots \\ \alpha_{i,n} \end{pmatrix} \in M^n$ is an infinite sequence $\rho = \rho(0)\rho(1)...$ where

- $\rho(0) = q_1$
- $\rho(1) = q$, where $(q_1, \psi_{q_1,q}(x_1, .., x_n), q) \in \Delta$ and $\mathcal{M} \models \psi_{q_1,q}[\alpha_1(1), .., \alpha_1(n)]$
- $\rho(i) = q$ if $\rho(i-1) = p$, $(p, \psi_{q,p}(x_1, .., x_n, y_1, .., y_n), q) \in \Delta$ and $\mathcal{M} \models \psi_{p,q}[\alpha_{i-1}(1), .., \alpha_{i-1}(n), \alpha_i(1), .., \alpha_i(n)]$ for $1 < i$

where $\alpha_i(j)$ is the j-th component of α_i). We call such a run ρ of $\mathfrak{A}$ for the infinite word α successful if ρ fulfills the Büchi-acceptance condition. We say $\mathfrak{A}$ accepts α if it has a successful run. Furthermore we define $L(\mathfrak{A})$ as the set of infinite words over M^n which are accepted by $\mathfrak{A}$.

A Strong Büchi Automaton can, for example, recognize the language $L' = \{\alpha \mid \alpha \in \Sigma^\omega \text{ and } \alpha(i) + 2 = \alpha(i+1) \text{ for infinitely many } i \in N\}$. An automaton recognizing L is illustrated in Figure 3.2.

In the latter portions of this work, we will extend the model of Strong Automata and Strong Büchi Automata again by allowing them to look an arbitrary but fixed amount of letters back. We will cover the union, closure and intersection only for the more generalized model. Since that model is a generalization of this model, the results can be applied to this model as well. There is a difference in the decidability of the non-emptiness problem however. While it is decidable for Strong Automata and Strong Büchi Automata, given certain restrictions, it will be undecidable for an arbitrary but fixed look-back.

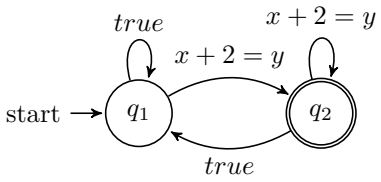

Figure 3.2: Strong Automaton recognizing L'

3.2 Non-emptiness Problem for words of dimension 1

While there is a proof in [Spe13] for this problem, we will give an alternate and somewhat more comprehensible proof that the the non-emptiness problem for words of dimension 1 is decidable, if the MSO-theory of a structure used structure $\mathcal{M}$ is decidable. For that, let $(\mathcal{M}, \text{MSO-logic})$ be an alphabet frame, where $\mathcal{M}$ is an arbitrary relational structure with decidable MSO-theory.

First we will transform the structure $\mathcal{M}$ into a new structure $\mathcal{N}$ with the following properties:

Definition 3.3. Let $\mathcal{M} = (M, R_1, ..., R_n)$ be a relational structure. For any given $m \geq 1$, the structure $\mathcal{N} = (M \times \{1, ..., m\}, R_1'.., R_n', Id, K_1...K_m)$, where

- $R_i' = \{((a_1, j_1), ...(a_{r_i}, j_{r_i})) \mid a_j \in M, j_k \in \{1, ..., m\} \wedge (a_1, ..., a_{r_i}) \in R_i\}$ if R_i is of arity r_i
- $Id = \{((a, i), (a, j)) \mid a \in M, i, j \in \{1, ..., m\}\}$
- $K_i = \{(a, i) \mid a \in M\}$,

is called m-extension of $\mathcal{M}$.

We will use this structure to construct a $\mathcal{N}$-MSO-formula such that the constructed formula will only be satisfiable if the language of the corresponding Strong (Büchi) Automaton is nonempty. For that we need to show that MSO-theory of $\mathcal{N}$ is decidable if MSO-theory of $\mathcal{M}$ is decidable. Furthermore, we need to model each transition of the input Automaton in the new structure $\mathcal{N}$, s.t.

$$\mathcal{N} \models \varphi'[a_1, a_2] \Leftrightarrow a_1 = (a, i) \wedge a_2 = (b, j) \wedge \mathcal{M} \models \varphi[a, b]$$

Both preliminaries for the main proof will be shown by induction over the structure of MSO-formulas. In order to simplify those proofs, we will first introduce MSO_0-logic, which eliminates all first-order variables by replacing them through singletons. Furthermore, since each function can be transformed into relations, we will assume that all structures only contain relations and no functions. Moreover, we will use $\underline{k}$ as a shorthand for $\{1, ..., k\}$.

Definition 3.4. Given a relational structure $\mathcal{M} = (M, R_1, ..., R_n)$, the MSO_0 logic of $\mathcal{M}$ is built up from the atomic formulas

- $X \subseteq Y$
- $Sing(X)$

- If R_i is of arity r_i, then $R_i(X_1, ..., X_{r_i})$ iff $X_1 = \{x_1\}, ..., X_{r_i} = \{x_{r_i}\}$ s.t. $R_i(x_1, ..., x_{r_i})$

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

Remark 3.5. Each MSO-formula can be transformed into an equivalent MSO_0-formula.

Proof. The translation of MSO-formula into equivalent MSO_0-formulas can easily be shown by an induction over $\mathcal{M}$-formulas.

First we replace all atomic formulas as follows:

- $x = y$ by $Sing(X) \wedge Sing(Y) \wedge X \subseteq Y$
- $R_i(x_1, ..., x_{r_i})$ by $\bigwedge_{i \in \underline{r_i}} Sing(X_i) \wedge R_i(X_1, ..., X_{r_i})$

In the induction step the Boolean connectives are used as usual. For the quantifiers over first order variables will be replaced by a the same quantifier over a set, enforce that it is a singleton with *Sing* and finally use it in the transformed sub-formula. Note that any MSO-formula can be brought into this form by replacing first-order variables with a set and adding the conditions that these sets are singletons. □

Lemma 3.6. *If $\mathcal{M} = (M, R_1, ..., R_n)$ is a relational structure whose MSO-theory is decidable then the MSO-theory of the structure $\mathcal{N}$, where $\mathcal{N}$ is the m-extension of $\mathcal{M}$, is decidable as well for all $m \in \mathbb{N}$.*

Proof. We will prove the Lemma by showing that each MSO_0-formula $\varphi(X_1, .., X_k)$ over the structure $\mathcal{N}$ is satisfiability-equivalent to an MSO_0-formula $\varphi(X_1^1, ..., X_1^m, ..., X_k^m)$ over the structure $\mathcal{M}$.

First we will go over the atomic formulas:

Case: $\varphi(X_1, X_2) = X_1 \subseteq X_2$
This formula is satisfiability-equivalent to

$$\varphi'(X_1^1, ...X_2^m) = \bigwedge_{i=1}^{m} X_1^i \subseteq X_2^i$$

If $\mathcal{N} \models \varphi[A_1, A_2]$, then it follows that $\mathcal{M} \models \varphi'[A_1^1, ...A_2^m]$ for $A_i^j = \{a \mid a \in M \wedge (a, j) \in A_i\}$ for $i \in \{1, 2\}, j \in \underline{m}$, since each set represents a subset of A_i, which only contains those elements for a certain second component. For the other direction there exist the sets $A_i = \{(a, j) \mid a \in A_i^j\}$. Since all sets $A_1^j \subseteq A_2^j$ for $j \in \underline{m}$, for the constructed sets the relation $A_1 \subseteq A_2$ will hold as well.

Case: $\varphi(X) = Sing(X)$
This formula can be expressed in $\mathcal{M}$ with the formula $\varphi'(X^1, ..., X^m)$ as

$$(\bigvee_{i=1}^{m} Sing(X^i)) \wedge \bigwedge_{i,j \in \underline{m}, i<j} (\neg Sing(X^i) \vee \neg Sing(X^j)) \wedge \bigwedge_{i=1}^{m} (\neg Sing(X^i) \rightarrow \forall Z X^i \subseteq Z)$$

The disjunction enforces that at least one set is a singleton, while the second part enforces that there is at most one set which is a singleton. The last part ensures that all other sets are empty.

If $\mathcal{N} \models \varphi[A]$, it follows that A contains exactly one element (a, i) for some $a \in M$, $i \in \underline{m}$. So the formula $\varphi'(A^1, ..., A^m)$ can be fulfilled by $A_i = \{x\}$ and $A_j = \emptyset$ for all $j \in \underline{m}, i \neq j$. Since these sets exist in the structure $\mathcal{M}$, it follows that $\mathcal{M} \models \varphi'[A^1, ..., A^m]$.

If we assume $\mathcal{M} \models \varphi'[A^1, ..., A^m]$, then A^i is a singleton for some $i \in \underline{m}$, while all other sets are empty. Let A^i be $\{a\}$ for some $a \in M$, then the set $A = \{(a, i)\}$ exists in $M \times \underline{m}$, which is a singleton and fulfills *varphi*. Therefore $\mathcal{N} \models \varphi[A]$

Since we will frequently need a formula to express that exactly one set is a singleton, we will define $\vartheta_n(Y^1, ..., Y^n)$ as

$$(\bigvee_{i=1}^{n} Y^i) \wedge \bigwedge_{i,j \in \underline{n}, i<j} (\neg Sing(Y^i) \vee \neg Sing(Y^j)) \wedge \bigwedge_{i=1}^{n} (\neg Sing(X^i) \rightarrow \forall Z X^i \subseteq Z)$$

Case: $\varphi(X_1, X_2) = Id(X_1, X_2)$
This formula is satisfiability-equivalent to

$$\varphi'(X_1^1, ..., X_2^m) = \vartheta_m(X_1^1, ..., X_1^m) \wedge \vartheta_m(X_2^1, ..., X_2^m) \wedge \bigwedge_{i,j \in \underline{m}} ((Sing(X_1^i) \wedge Sing(X_2^j) \rightarrow (X_1^i \subseteq X_2^j \wedge X_2^j \subseteq X_1^i))$$

The first part checks that exactly two singletons exist and that they are in different groups. The second part then ensures the equality of both singleton sets.

If $\mathcal{N} \models \varphi[A_1, A_2]$ then we know there exist some $A_1 = \{(a, i)\}$ and $A_2 = \{(a, j)\}$ for some $a \in M, i, j \in \underline{m}$. But then there exist the sets $A_1^i = \{a\}$ and $A_2^j = \{a\}$, while all other free variables in $\varphi'(X_1^1, ..., X_2^m)$ must be empty. And if those sets exist, the formula can be fulfilled by choosing them like that. Therefore $\mathcal{M} \models \varphi'[A_1^1, ..., A_2^m]$.

The other direction is easily shown, since some $A_1^i = \{a\}$ and $A_2^j = \{a\}$ for some $a \in M, i, j \in \underline{m}$. Then there exists $A_1 = \{(a, i)\}$ and $A_2 = \{(a, j)\}$, which fulfill $\varphi(X_1, X_2)$ by definition of Id. Therefore $\mathcal{N} \models \varphi[A_1, A_2]$.

Case: $\varphi(X_1, ..., X_{r_i}) = R_i'(X_1, ..., X_{r_i})$, for $i \in \underline{n}$ and A_i' being of arity r_i
This formula is satisfiability-equivalent to

$$\varphi'(X_1^1, ..., X_{r_i}^m) = \bigwedge_{j \in \underline{r_i}} \vartheta_m(X_j^1, ..., X_j^m) \wedge \bigwedge_{i_1 .. i_{r_i} \in \underline{m}} ((\bigwedge_{j \in \underline{r_i}} Sing(X_j^{i_j})) \rightarrow R_i(X_1^{i_1}, .., X_{r_i}^{i_{r_i}}))$$

The first part checks that for each group there exist exactly one singleton set. The latter part then checks that those singleton sets, fulfill the original relation.

If $\mathcal{N} \models \varphi[A_1, ..., A_{r_i}]$, then there exist singleton sets $A_j = \{(a_j, k_j)\}$ for $j \in \underline{r_i}, a_j \in M, k_j \in \underline{m}$. Therefore, by definition of R_i', we know $R_i(a_1, ..., a_{r_i})$. Then the formula φ' can be fulfilled by $\mathcal{N}$ with the sets $A_j^{k_j} = \{a_j\}$, while all other sets are empty. Therefore $\mathcal{M} \models \varphi[A_1^1, ..., A_{r_i}^m]$.

If $\mathcal{M} \models \varphi[A_1^1, ..., A_{r_i}^m]$, then the formula can be fulfilled by $A_j^{k_j} = \{a_j\}$ for $\forall j \in \underline{r_i}$ and $k_j \in \underline{m}, a_j \in M$, while the remaining free sets are empty. Then the free sets in φ can be chosen as $A_j = \{(a_j, k_j)\}$, which, due to the definition of R_i', fulfill the formula φ. Therefore

$\mathcal{N} \models \varphi[A_1, ..., A_{r_i}]$.

Case: $\varphi_i(X) = K_i(X)$ for $i \in \underline{m}$
This formula is satisfiability-equivalent to

$$\varphi_i'(X^1, ..., X^m) = \vartheta_m(X^1, ..., X^m) \wedge Sing(X^i)$$

If $\mathcal{N} \models \varphi_i[A]$, then $A = \{(a, i)\}$ for some $a \in M$. So for $A^i = \{a\}$ and $A^j = \emptyset \, \forall j \neq i$, $M \models \varphi_i'[A^1, ..., A^n]$ must hold.

For the other direction, only the set A^i will be nonempty. Furthermore it must be a singleton, therefore $A^i = \{a\}$ for some $a \in M$. By choosing $A = \{(a, i)\}$, $\mathcal{N} \models \varphi[A]$ must also hold, since A is a singleton.

With this we looked at all atomic formulas of the structure $\mathcal{N}$. Since for every formula there exists an equivalent formula which only uses $\exists$, $\wedge$ and $\neg$ it suffices to show

- $\varphi(X_1, .., X_{(k-1)}) \; = \; \exists X_k \varphi_1(X_1, ..., X_k)$
- $\varphi(X_1, .., X_{k_1}, Y_1, .., Y_{k_2}) = \varphi_1(X_1, ..., X_{k_1}) \wedge \varphi_2(Y_1, ..., Y_{k_2})$
- $\varphi(X_1, ..., X_k) = \neg \varphi'(X_1, ..., X_k)$

in the induction step.

For the induction hypothesis (IH) we can assume the existence of satisfiability-equivalent formulas $\varphi_i(X_1, .., X_n)$ and $\varphi_i'(X_1^1, ..., X_n^m)$ for $i \in \{1, 2\}$, s.t.

$$\begin{array}{l} \mathcal{N} \models \varphi_i[A_1, .., A_n] \text{ for some } A_1, ..A_n \subseteq M \times \underline{m} \\ \Leftrightarrow \mathcal{M} \models \varphi_i'[A_1^1, ..., A_n^m] \text{ for some } A_1^1, ..A_n^m \subseteq M \end{array}$$

Case $\varphi(X_1, .., X_{k_1}, Y_1, .., Y_{k_2}) = \varphi_1(X_1, ..., X_{k_1}) \wedge \varphi_2(Y_1, ..., Y_{k_2})$:

$$\begin{array}{ll} & \mathcal{N} \models \varphi[A_1, .., A_{k_1}, B_1, .., B_{k_2}] \\ \Leftrightarrow & \mathcal{N} \models \varphi_1[A_1, .., A_{k_1}] \wedge \varphi_2[B_1, .., B_{k_2}] \\ \Leftrightarrow & \mathcal{N} \models \varphi_1[A_1, .., A_{k_1}] \text{ and } \mathcal{N} \models \varphi_2[B_1, .., B_{k_2}] \\ \overset{IH}{\Leftrightarrow} & \mathcal{M} \models \varphi_1'[A_1^1, .., A_{k_1}^m] \text{ and } \mathcal{M} \models \varphi_2'[B_1^1, .., B_{k_2}^m] \\ \Leftrightarrow & \mathcal{M} \models \varphi_1'[A_1^1, .., A_{k_1}^m] \wedge \varphi_2'[B_1^1, .., B_{k_2}^m] \\ \Leftrightarrow & \mathcal{M} \models \varphi'[A_1^1, .., A_{k_1}^m, B_1^1, .., B_{k_2}^m] \end{array}$$

Case $\varphi(X_1, ..., X_k) = \neg \varphi'(X_1, ..., X_k)$:

$$\begin{array}{ll} & \mathcal{N} \models \varphi[A_1, .., A_{k_1}] \\ \Leftrightarrow & \mathcal{N} \models \neg \varphi_1[A_1, .., A_{k_1}] \\ \Leftrightarrow & \mathcal{N} \not\models \varphi_1[A_1, .., A_{k_1}] \\ \overset{IH}{\Leftrightarrow} & \mathcal{M} \not\models \varphi_1'[A_1^1, .., A_{k_1}^m] \\ \Leftrightarrow & \mathcal{M} \models \neg \varphi_1'[A_1^1, .., A_{k_1}^m] \\ \Leftrightarrow & \mathcal{M} \models \varphi'[A_1^1, .., A_{k_1}^m] \end{array}$$

and the final case $\varphi(X_1, .., X_{(k-1)}) \; = \; \exists X_k \varphi_1(X_1, ..., X_k)$:

If $\mathcal{N} \models \varphi[A_1, .., A_{(k-1)}]$ there exists some $A_k \subseteq M \times \underline{m}$, s.t. $\mathcal{N} \models \varphi_1[A_1, .., A_{(k-1)}, A_k]$. By using the induction hypothesis, it follows that $\mathcal{M} \models \varphi_1'[A_1^1, .., A_k^m]$. Since $X_k^1, .., X_k^m$

can be chosen freely, that formula is equivalent to $\varphi'(X_1^1, .., X_{k-1}^m) = \exists X_k^1 .. \exists X_k^m (X_1^1, .., X_k^m)$. And by assigning $X_k^1, .., X_k^m$ the values $A_k^1, .., A_k^m$ it follows that $\mathcal{M} \models \varphi'[A_1^1, .., A_{k-1}^m]$.

□

Now, since we have shown that the MSO-theory of the m-extension of $\mathcal{M}$ is decidable if the MSO-theory of $\mathcal{M}$ is decidable. Now we need to show the following theorem in order to transform the original transition formulas into formulas in the new structure $\mathcal{N}$, s.t. they only consider the M-component of the element and ignore the second.

Lemma 3.7. *Let $\mathcal{N}$ be the m-extension of $\mathcal{M}$. For each formula $\varphi(X_1, .., X_n)$ over the signature $(R_1, .., R_n)$ there exists a formula $\varphi'(X_1', .., X_n')$ over the signature $(R_1', .., R_n', Id, K_1, .., K_m)$ s.t.*

$$\begin{gathered}\mathcal{M} \models \varphi[A_1, .., A_n] \text{ for } A_1, .., A_n \subseteq M \\ \Leftrightarrow \mathcal{N} \models \varphi'[A_1', .., A_n'] \text{ for some } A_1', .., A_n' \subseteq M \times \underline{m} \text{ for} \\ \pi(A_i') = A_i \forall i \in \underline{n} \text{ and } \pi(X) = \{a \mid (a, i) \in X\}\end{gathered}$$

Proof. Again this will be shown by an induction over the structure of MSO_0-formulas of the structure $\mathcal{M}$. We will skip the atomic case for the relations, since they follow directly from the definition of their counterparts in $\mathcal{N}$. We will skip the singleton case as well, since it is easy to see the immediate equivalence. The last remaining atomic case is the subset.

Case: $\varphi(X_1, X_2) = X_1 \subseteq X_2$

This can be handled by

$$\begin{gathered}\varphi'(X_1', X_2') = \exists Z_1 \exists Z_2 (Z_1 \subseteq Z_2 \wedge \\ \bigwedge_{i=1}^{2} \forall Z((Sing(Z) \wedge Z \subseteq Z_i) \leftrightarrow (K_1(Z) \wedge \exists Z'(Id(Z, Z') \wedge Z' \subseteq X_i'))))\end{gathered}$$

We can't simply ignore the second component as the projection π does, therefore we will project them onto the elements where the second component is always 1 instead. By doing so we reach the same result as π and compare these sets.

$\mathcal{M} \models \varphi[A, B]$. Then we know $A \subseteq B$. By choosing $A' = \{(a, 1) \mid a \in A\}$ and $B' = \{(b, 1) \mid b \in B\}$ the subset relation will still hold for A' and B'. And since $Z_1 = \{(a, 1) \mid (a, i) \in A'\}$ and $Z_2 = \{(a, 1) \mid (a, i) \in B'\}$, $Z_1 \subseteq Z_2$ will hold as well. Therefore $\mathcal{N} \models \varphi'[A', B']$

If $\mathcal{N} \models \varphi'[A', B']$ there exist sets Z_1 and Z_2, s.t. $Z_1 \subseteq Z_2$ and $Z_1 = \{(a, 1) \mid \exists j \in \underline{m}, (a, j) \in A'\}$ and $Z_2 = \{(a, 1) \mid \exists j \in \underline{m}, (a, j) \in B'\}$. By choosing $A = \{a \mid (a, 1) \in Z_1\}$ and $B = \{b \mid (b, 1) \in Z_2\}$ the subset relation will also hold for A and B. Therefore $\mathcal{M} \models \varphi[A, B]$

With this we finished the atomic cases and can start with the boolean connectives $\neg$, $\wedge$ and the existential set quantifier $\exists$. However, since $\neg$ and the $\wedge$ will use very similar arguments as in the previous proof, we will omit them and concentrate on the existential quantifier. We will assume that the claim holds for $\varphi(X_1.., X_n)$ as the induction hypothesis.

Case $\psi(X_1, .., X_{n-1}) = \exists X_n \varphi(X_1, .., X_n)$: This can be modeled by

$$\psi'(X_1, .., X_{n-1}) = \exists X_n \varphi(X_1, .., X_n).$$

Then

$$\begin{aligned}
&\mathcal{M} \models \psi[A_1, .., A_{n-1}] \\
\Leftrightarrow &\exists A_n \text{ s.t. } \mathcal{M} \models \varphi[A_1, .., A_n] \\
\overset{IH}{\Leftrightarrow} &\exists A'_1, .., A'_n \text{ s.t. } \mathcal{N} \models \varphi'[A'_1, .., A'_n] \\
\Leftrightarrow &\mathcal{N} \models \psi'[A'_1, .., A'_{n-1}]
\end{aligned}$$

With this we have shown, that we can transform each MSO-formula in $\mathcal{M}$ into an equivalent MSO-formula in $\mathcal{N}$. Furthermore the structures created this way can only be satisfied if the projection of their input parameters onto their first components satisfy the original formula in $\mathcal{M}$.

□

With both lemmas we can show the main result of this section, the decidability of the non-emptiness problem for Strong Automata and Strong Büchi Automata.

Theorem 3.8. *If the MSO-Theory of $\mathcal{M}$ is decidable, then the following two theorems hold:*

1. *The non-emptiness problem for Strong Automata over the alphabet frame ($\mathcal{M}$, MSO) is decidable*
2. *The non-emptiness problem for Strong Büchi Automata over the alphabet frame ($\mathcal{M}$, MSO) is decidable*

Proof. We will begin with the proof for Strong Automata. The Strong Büchi Automata only require some minor modifications. Let $\mathfrak{A} = (Q, M, q_1, \Delta, F)$ be a Strong Automaton over the relational structure $\mathcal{M} = (M, A_1, .., A_n)$. W.l.o.g. let $Q = \{q_1, .., q_m\}$, and therefore $|Q| = m$. In order to show the decidability of the non-emptiness problem of $\mathfrak{A}$ we will construct and $\mathcal{N}$-MSO-formula φ of the structure $\mathcal{N}$, which will be the m-extension of $\mathcal{M}$. We will construct φ in a way, such that

$$\mathcal{N} \models \varphi \Leftrightarrow L(\mathfrak{A}) \neq \emptyset$$

The general idea will be a reachability analysis from the initial state q_1. For Strong Automata another condition we must consider the possibility of q_1 being the only final state without having any incoming transitions. Then we can't build a run of the form

$$q_1 \xrightarrow{\psi_{p_1}(a_1)} (a_1, p_1) \xrightarrow{\psi_{p_1,p_2}(a_1,a_2)} (a_2, p_2) \ldots (a_{n-1}, p_{n-1}) \xrightarrow{\psi_{p_{(n-1)},p_n}(a_{n-1},a_n)} (a_n, p_n)$$

As the depicted run shows, each state consists of an actual state of the automaton and the previously read letter. We will use the term state as synonym for such a tuple as well as actual states of the automaton. The first component of a state will be the previously read letter while the second component is the actual state of the automaton. The context decides whether we refer to an actual state of the automaton or such a tuple.

So, we need to express that there is some final state, a state where the second component is a final state, reachable from the initial state or that the initial state itself is a final state. We will define the general formula as

$$\varphi = \exists S \exists q(\varphi_i(q) \wedge \varphi_e^+(q,S) \wedge \varphi_f(S)) \vee \varphi_s$$

The variable q will become the first state after one transition from q_1. $\varphi_i(q)$ will assert that condition. $\varphi_r(q,S)$ will assert that S contains all from q reachable states including q itself. For Strong Automata $\varphi_f(S)$ will assert that there is some final state in S. At last there is φ_s which will take care of the case where the initial state is a final state.

As already explained the special case of q_1 being a final state is static for each Strong Automaton $\mathfrak{A}$. Therefore the formula φ_s will be a constant true or false, depending on whether $q_1 \in F$ or not. It is defined as:

$$\varphi_s = \begin{cases} true & \text{, if } q_1 \in F \\ false & \text{, otherwise} \end{cases}$$

In order to define φ_r we first define another formula $\varphi_e(q,p)$ which will express that p is a direct successor of q. More specifically

$$\begin{aligned} &\mathcal{N} \models \varphi_e[q,p] \\ \Leftrightarrow\, &q = (a,i) \wedge p = (b,j) \wedge \mathcal{M} \models \psi_{q_i,q_j}[a,b] \wedge \psi_{q_i,q_j}(x,y) \in \Delta \\ \Leftrightarrow\, &\mathcal{N} \models \psi'_{q_i,q_j}[q,p] \wedge \psi_{q_i,q_j}(x,y) \in \Delta \end{aligned}$$

where $\psi'_{q_i,q_j}(x,y)$ is constructed as described in the previous lemma. Furthermore, the lemma also proves the equivalence between the second and third equivalence of the above. Now let us define $\varphi_e(x,y)$.

$$\varphi_e(x,y) = \bigvee_{\psi_{q_i,q_j}(x,y) \in \Delta} (\psi'_{q_i,q_j}(x,y) \wedge K_i(q) \wedge K_j(p))$$

With that formula we will define another subformula $\varphi_e^+(q,S)$ to express that q is contained in S and that for all states $p \in S$ all reachable successors of p must also be contained in S. That can be expressed as:

$$\varphi_e^+(q,S) = S(q) \wedge \forall x \forall y(S(x) \wedge \varphi_e(x,y) \rightarrow S(y))$$

$\varphi_e^+(q,S)$ does not express the same as $\varphi_r(q,S)$, however. By choosing $S = M \times \underline{m}$, $\varphi_e^+(q,S)$ will always be fulfilled for any q. Instead we search the smallest set which fulfills that condition. That will be done by $\varphi_r(q,S)$.

$$\varphi_r(q,S) = \varphi_e^+(q,S) \wedge \neg \exists S'(S' \subset S \wedge \varphi_e^+(q,S'))$$

With that S will only contain from q reachable states. Now we need a possible second state of the run. It has to assure that q is a state, which is reached via the first transition of a run. More specifically

$$\begin{aligned} &\mathcal{N} \models \varphi_i[q] \\ \Leftrightarrow\, &q = (a,j) \wedge \mathcal{M} \models \psi_{q_1,q_j}[a] \wedge \psi_{q_1,q_j}(x) \in \Delta \\ \Leftrightarrow\, &\mathcal{N} \models \psi'_{q_1,q_j}[q] \wedge \psi_{q_1,q_j}(x) \in \Delta \end{aligned}$$

Once again we use the previous lemma to construct the $\psi' q_1, q_i(x)$ in the same manner to assure the equivalence between the second and third line. With that in mind, we can construct $\varphi_i(q)$ as

$$\varphi_i(q) = \bigvee_{\psi_{q_0,q_j}(x) \in \Delta} (\psi'_{q_0,q_j}(q) \wedge K_j(q))$$

Finally we need to express that a final state is reachable from q. Since S contains all from q reachable states, we just need to check whether there is one final state in S. That can be done by

$$\varphi_f(S) = \exists q(S(q) \wedge \bigvee_{q_i \in F} K_i(q))$$

With that φ is completely defined. Now we need to show that the equivalence between the non-emptiness of $L(\mathfrak{A})$ and the satisfiability of φ holds.

Let us assume $L(\mathfrak{A})$ is not empty, then there exists some word $w \in L(\mathfrak{A})$. If $w = \varepsilon$, then φ_s is true since $q_1 \in F$. And then φ is also true. Otherwise $\varepsilon \notin L(\mathfrak{A})$, which contradicts the assumption. If $w \neq \varepsilon$, then there exists an accepting run in $\mathfrak{A}$. However then the last letter of w combined with a final state must be in S by construction of φ. And then $\mathcal{N} \models \varphi$.

Assuming $L(\mathfrak{A}) = \emptyset$, then either no state is a final state or all final states are unreachable. If no state is a final state φ_s and φ_f can never be satisfied. Therefore φ can't satisfied either. If there are only unreachable final states, no final state can be in S, since S is the smallest set containing all states reachable from q_1. Therefore $\mathcal{N} \not\models \varphi$.

Since φ is decidable and $L(\mathfrak{A}) \neq \emptyset \Leftrightarrow \mathcal{N} \models \varphi$, the decidability of the non-emptiness problem for Strong Automata is decidable as well.

For Strong Büchi Automata two modifications need to be made. The first is setting $\varphi_s = false$, since the special case does not hold for Strong Büchi Automata. Even if the first state is a final state, a run needs to visit final states infinitely often. Therefore ignoring the first of those infinitely many visits will not change the acceptance of the word.

The second modification is the modification of $\varphi_f(S)$, since Strong Büchi Automata need to visit infinitely many final states on a single path. Let $R \subseteq M \times \underline{m}$ contain the reachable states. First we will try to find a non-empty subset F' of R such that $q = (a, i) \in F'$ iff $q_i \in F$. In other words all elements of F' have a final state as second component. Furthermore F' needs to satisfy the property

$$q \in F' \Leftrightarrow \exists p \in F', \text{ s.t. } p \text{ is reachable from } q.$$

If F' has those two properties, then either there exist a finite non-empty subset of F' with the same property. That implies the existence of a circle which can be traversed in a run and therefore implies infinitely many visits of final states for some word. Or, if there is no finite non-empty subset with the same property, then F' has an infinite subset $\{q_0, q_1, ...\}$ with the property q_{i+1} is reachable from q_i. That also implies, however, that there is a run which visits infinitely many final states.

Now we need to express that in the formula $\varphi_f(S)$:

$$\varphi_f(S) = \exists F'(F' \subseteq S \wedge \forall x(F'(x) \rightarrow \bigvee_{q_i \in F} K_i(x)) \wedge$$
$$\forall x(F'(x) \rightarrow \exists y(F'(y) \wedge \exists z(\varphi_e(x, z) \wedge \exists S'(\varphi_r(z, S') \wedge S'(y))))))$$

The formula $\varphi_f(S)$ catches that condition and, as already explained, shows that an accepting run of Strong Büchi Automata has to exist. Therefore, due to the decidability of φ, the non-emptiness problem for Strong Büchi Automata is also decidable.

□

With that we will close the section on Strong (Büchi) Automata. As with $\mathcal{M}$-$\mathcal{L}$-Automata, the remaining closure properties will be shown for the more general model. Since Strong (Büchi) Automata will be a special case of that model, the closure properties will also hold for the model of Strong (Büchi) Automata.

4 Extended Strong Automata

Strong (Büchi) Automata can only look back one letter and use it in the transition formula. In this section we will extend Strong (Büchi) Automata as defined in [Spe13] by allowing the automaton to look back up to s letters for some $s \in \mathbb{N}$. That way the automaton will ultimately have formulas at transitions which have up to $s+1$ times the dimension of the word free variables. First we will define the model. Then we will show that for a look-back greater than 1 the non-emptiness problem is undecidable.

4.1 Defintion of extended Strong Automata

Definition 4.1. Let $(\mathcal{M}, \mathcal{L})$ be an alphabet frame. A Strong Automaton over $(\mathcal{M}, \mathcal{L})$ with look-back s is defined as

$$\mathfrak{A} = (Q, M^n, q_1, \Delta, F)$$

where

- Q is a finite set of states
- the input alphabet is M^n, where M is the domain of $\mathcal{M}$
- $q_1 \in Q$ is the initial state
- $\Delta \subseteq Q \times (\bigcup_{i=1}^{s+1} \Psi_{i*n}) \times Q$ is the finite transition relation where Ψ_{i*n} are sets of $\mathcal{M}$-$\mathcal{L}$-formulas with $i * n$ free variables.
- and lastly $F \subseteq Q$ is the set of accepting states.

For simplicity's sake we will call them extended Strong Automata. As with Strong Automata, extended Strong Automata can be used to recognize languages containing finite or infinite words. The acceptance conditions are similar to the acceptance conditions of Strong Automata, so we will focus on the differences. For finite words $w = a_1, .., a_m$ this means:

- $\rho(i) = q$, if $\rho(i-1) = p$, $(p, \psi_{p,q}(x_0^1, .., x_0^n, x_1^1, .., x_{i-1}^n), q) \in \Delta$ and $\mathcal{M} \models \psi_{p,q}[a_1(1), .., a_1(n), a_2(1), ..., a_i(n)]$ for all $i \leq s \wedge i \leq m$
- $\rho(i) = q$, if $\rho(i-1) = p$, $(p, \psi_{p,q}(x_0^1, .., x_0^n, x_1^1, .., x_s^n), q) \in \Delta$ and $\mathcal{M} \models \psi_{p,q}[a_{i-s}(1), .., a_{i-s}(n)..., a_i(n)]$ for all $s < i \leq m$

When using an extended Strong Automaton to handle infinite words, the second requirement has to hold for all $i > s$ and final states need to be visited infinitely many times during a run. As before, we will distinguish between them by calling them extended Strong Automata and extended Strong Büchi Automata. Note that $\mathcal{M}$-$\mathcal{L}$-Automata are a special case for a

look-back of 0 and Strong Automata from the previous chapter are the special case for a look-back of 1. Therefore we will focus on the look-backs larger than 1. With a look-back of two, for example, it is possible to recognize the language $L = \{a_1a_2a_3 \mid a_1 + a_2 = a_3\}$ for the alphabet frame $((\mathbb{N}, +, <, 0), \text{FO logic})$. This can be recognized by

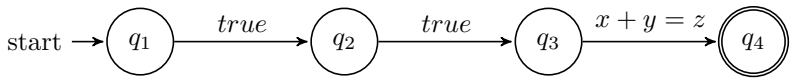

Figure 4.1: extended Strong Automaton recognizing the + relation on $\mathbb{N}$

, where x is the predecessor letter of y and y is the predecessor letter of z.

In the following we will show that the non-emptiness problem for extended Strong Automata is undecidable for FO- and MSO-logic once the automaton has a look-back of at least 2. Once again we will only cover the case for words of dimension 1, since that will also show the undecidability for all look-backs and word dimensions larger than 1. We also need to show the non-emptiness problem for extended Strong Automata with look-back of 1 and a word dimension $n > 1$. We will show that a word dimension of 2 already causes the non-emptiness problem to become undecidable.

4.2 Non-emptiness problem for extended Strong Automata

As mentioned earlier, we will focus on the non-emptiness problem of extended Strong Automata for a look-back of 2 or greater.

Theorem 4.2. *The non-emptiness problem for an extended Strong (Büchi) Automaton with a look-back of 2 and a word dimension of 1 is undecidable for FO-and MSO-logic.*

Proof. We will show this by reduction from the halting problem for 2-register machines. Such a machine R has two registers X_1 and X_2, which can store any integer. The machine R is defined by a finite sequence

$$1 \text{ instr}_1; \ldots; k - 1 \text{ instr}_{k-1}; k \text{ stop}$$

where each instruction$_j$ is of the form

- Inc(X_1), Inc(X_2), which increments the respective register by 1, or
- Dec(X_1), Dec(X_2), which decrements the respective register by 1, unless the register is 0, or
- If $X_i = 0$ goto l_1 else goto l_2, where $i = 1, 2$ and $1 \leq l_1, l_2 \leq k$, with the natural interpretation.

A configuration of R is given as a triple (l, m, n). The value l indicates that the l-th instruction is to be executed, whereas m and n are the current values of X_1 and X_2 respectively. A computation of R is a sequence of such triplets. We write $(l, m, n) \vdash (l', m', n')$ if (l', m', n') is the direct successor of (l, m, n), respecting the instruction l. If it terminates, we obtain a sequence $(l_1, m_1, n_1) \vdash \cdots \vdash (l_r, m_r, n_r)$ of R-configurations with $l_0 = 1$ and $l_r = k$. The halting problem for 2-register machines is to decide, whether an R-computation starting with

$(1,0,0)$ terminates or not. The undecidability of the halting problem has been proven by Minsky in 1967[Min67].

We will use the undecidability of that problem to show the undecidability of the non-emptiness for extended Strong (Büchi) Automata with a look-back of 2. In order to do so we need a computable transformation from R to $\mathfrak{A}_R$, s.t.

$$\text{R has a terminating computation from } (1,0,0) \Leftrightarrow L(\mathfrak{A}_R) \neq \emptyset$$

where $\mathfrak{A}_R$ is an extended Strong (Büchi) Automaton with a look-back of 2.

In order to do that we use $((\mathbb{N}, +1, 0), \text{FO-logic})$ as alphabet frame. Since MSO-logic is a superset of FO-logic, this will show the undecidability for MSO-logic as well.

The general idea behind the construction of the automaton will be the simulation of the sequence of R configurations. One configuration needs two letters to be represented correctly however, since it is not possible to encode and decode two integers into one integer in the alphabet frame $((\mathbb{N}, +1, 0), \text{FO logic})$. Therefore even steps will take care of simulation for the first register while the odd steps take care of the simulation for the second register. Since it is possible to look back 2 letters, the letter of the previous configuration and the current configuration can always be compared and modified.

We define $\mathfrak{A}_R = (Q, \mathbb{N}, q_0, \Delta, F)$ as

- $Q = \{q_i \mid 0 \leq i \leq k\} \cup$
  $\{q'_i \mid 0 < i < k, \text{instr}_i \neq \text{If } X_1 = 0 \text{ goto } l_1 \text{ else } l_2\} \cup$
  $\{q_i^1, q_i^2 \mid 0 < i < k, \text{instr}_i = \text{If } X_1 = 0 \text{ goto } l_1 \text{ else } l_2\}$
- $F = \{q_k\}$
- $\Delta = \{(q_0, x_0 = 0, q_0), (q_0, x_1 = 0, q_1)\} \cup \bigcup_{i=1}^{k-1} \Delta_i$
  where Δ_i depends on instr_i:
  - $\text{instr}_i = \text{Inc}(X_j)$: $\Delta_i = \{(q_i, \varphi_1(x_0, x_1, x_2), q'_i), (q'_i, \varphi_2(x_0, x_1, x_2), q_{i+1})\}$
    where
    - $\varphi_j := x_0 + 1 = x_2$
    - $\varphi_{3-j} := x_0 = x_2$
  - $\text{instr}_i = \text{Dec}(X_j)$: $\Delta_i = \{(q_i, \varphi_1(x_0, x_1, x_2), q'_i), (q'_i, \varphi_2(x_0, x_1, x_2), q_{i+1})\}$
    where
    - $\varphi_j := x_0 = x_2 + 1 \vee (x_0 = 0 \wedge x_2 = 0)$
    - $\varphi_{3-j} := x_0 = x_2$
  - $\text{instr}_i = \text{If } X_1 = 0 \text{ goto } l_1 \text{ else } l_2$:
    $\Delta_i = \{(q_i, \varphi_t(x_0, x_1, x_2), q_i^1), (q_i, \varphi_f(x_0, x_1, x_2), q_i^2)\} \cup$
    $\{(q_i^1, \varphi_e(x_0, x_1, x_2), q_{l_1}), (q_i^2, \varphi_e(x_0, x_1, x_2), q_{l_2})\}$
    where
    - $\varphi_e(x_0, x_1, x_2) := x_0 = x_2$
    - $\varphi_t(x_0, x_1, x_2) := \varphi_e(x_0, x_2) \wedge x_2 = 0$
    - $\varphi_f(x_0, x_1, x_2) := \varphi_e(x_0, x_2) \wedge x_2 \neq 0$

– and finally instr_i = If $X_2 = 0$ goto l_1 else l_2:
$\Delta_i = \{(q_i, \varphi_e(x_0, x_1, x_2), q_i'), (q_i', \varphi_f(x_0, x_1, x_2), q_{l_1}), (q_i', \varphi_e(x_0, x_1, x_2), q_{l_2})\}$
where the formulas are the same as before.

The transitions $\{(q_0, x_0 = 0, q_0), (q_0, x_1 = 0, q_1\} \subset \Delta$ enforce that the first two letters are 0. It remains to show the equivalence between the halting problem for the machine R and the non-emptiness problem for $\mathfrak{A}_R$. In order to show that, we will show the following Lemma first.

Lemma 4.3. *If R is in the configuration (l, m, n) with the direct successor configuration (l', m', n') and the current state of the automaton $\mathfrak{A}_R$ is q_l while the previously read letter is n and the one before that is m, then the automaton can only accept the letters $m'n'$ and end in state $q_{l'}$*

Proof. In order to show the lemma, we need to distinguish between all instructions. Let's assume the premises hold as described. Then it depends on the actual instruction, since the automaton depends on it.

- Case $\text{Inc}(X_1)$: Then we know $l' = l + 1$, $n' = n + 1$ and $m' = m$. The constructed automaton requires the next letter from q_l to be $n + 1 = n'$ to advance to q_l'. Then the last two read letters are mn'. The next transition needs the next letter to be m to enter state q_{l+1}. Then the last two read letters are $n'm = n'm'$ and the state is $q_{l'}$, which is exactly the claim.

- Case $\text{Inc}(X_2)$, $\text{Dec}(X_1)$, $\text{Dec}(X_2)$: are proven with similar or the same arguments as the $\text{Inc}(X_1)$ case. Therefore we will omit them here.

- Case If $X_1 = 0$ goto l_1 else goto l_2: Here we need to distinguish between the case $n = 0$ and $n \neq 0$. If $n = 0$, then $l' = l_1$, $n' = n$, $m' = m$. As $n = 0$ the transition φ_t will only be fulfilled if $n' = 0$. Furthermore, ψ_t and ψ_f are mutually exclusive, which is why the result is deterministic. The next transition requires to read m, which is why the last two read letters are nm, which are equal to $n'm'$. Furthermore this will end in the state q_{l_1}, so the claim holds. The case for $n \neq 0$ works analogously.

- Case If $X_2 = 0$ goto l_1 else goto l_2: Uses the same arguments as the previous case, just the order of the decision is switched. Therefore we will omit this.

- Case stop: Stop has no successor configuration, therefore nothing needs to be shown.

□

Since the initialization sequence of the automaton enforces that the last two read letters during the first visit of q_1 need to be 00, and the previous lemma, the construction only ends in q_k iff the computation of R terminates. Since q_k is the only final state, the language $L(\mathfrak{A}_R) = \{0^2 s n_2 m_2 n_3 m_3 \dots n_r m_r\}$ iff R stops with the sequence $(1, 0, 0) \vdash (l_2, n_2, m_2) \vdash \dots \vdash (k, n_r, m_r)$. If R does not stop $L(\mathfrak{A}_R) = \emptyset$. If the non-emptiness problem for extended Strong Automata with a look-back larger than 1 were decidable, the halting problem for R would also be decidable. Since that is not possible, however, the non-emptiness problem for extended Strong Automata must also be undecidable.

For extended Strong Büchi Automata, we will add the self-loop $(q_k, x_2 = 0, q_k)$ to Δ. If the computation ever reaches q_k, only 0 can be read. Then $L(\mathfrak{A}_R) = \{0^s n_2 m_2 n_3 m_3 \dots n_r m_r \cdot 0^\omega\}$. As the same argumentation holds for extended Strong Büchi Automata, their non-emptiness problem is undecidable as well.

Furthermore, since the MSO-logic is a superset of the FO-logic, the same holds for the MSO-logic as well. □

Proposition 4.4. Since the non-emptiness problem for Strong (Büchi) Automata with a look-back of 2 and a word dimension of 1 is undecidable, the non-emptiness problem for Strong (Büchi) Automata with a look-back greater than 1 and any word dimension is undecidable as well.

We used FO-logic as logic for the alphabet frame, during the construction of the automaton, however, we did not use a single quantifier. Therefore restricting all possible formulas to the quantifier-free subset of FO-logic suffices for the proof.

And now we need to consider the remaining non-emptiness problem for extended Strong (Büchi) Automata with a look-back of 1 and a word dimension greater than 1. This is a result from [Spe13], which we wanted to add for completeness sake.

Theorem 4.5. *The non-emptiness problem for an extended Strong (Büchi) Automaton with a look-back of 1 and a word dimension of 2 is undecidable for FO- and MSO-logic.*

Proof. The proof is similar to the one for look-back of 2 and word dimension 1. We will use $((\mathbb{N}, +1_1, 0)$,FO-logic) as alphabet frame as well. Assuming the 2-register machine consists out of k instructions again, we construct $\mathfrak{A}_R = (Q, \mathbb{N}^2, q_0, \Delta, \{q_k\})$ as

- $Q = \{q_i \mid 0 \leq i \leq k\}$
- $\Delta = \{(q_0, x_0^0 = 0 \wedge x_0^1 = 0, q_1)\} \cup \bigcup_{i=1}^{k-1} \Delta_i$ where Δ_i depends on the instruction
  - instr$_i$ = Inc(X_j): $\Delta_i = \{(q_i, x_0^{j-1} + 1 = x_1^{j-1} \wedge x_0^{2-j} = x_1^{2-j}, q_{i+1})\}$
  - instr$_i$ = Dec(X_j): $\Delta_i = \{(q_i, ((x_0^{j-1} = x_1^{j-1} \wedge x_0^{j-1} = 0) \vee x_0^{j-1} = x_1^{j-1} + 1) \wedge x_0^{2-j} = x_1^{2-j}, q_{i+1})\}$
  - instr$_i$ = If $X_j = 0$ goto l_1 else l_2:
    $\Delta_i = \{(q_i, x_0^0 = x_1^0 \wedge x_0^1 = x_1^1 \wedge x_0^{j-1} = 0, q_{l_1}),$
    $(q_i, x_0^0 = x_1^0 \wedge x_0^1 = x_1^1 \wedge x_0^{j-1} \neq 0, q_{l_2})\}$

The basic idea is the same as before. This time however, since we have a tuple of natural numbers, the automaton can simulate one step of the 2-register machine in one transition instead of two. The first transition ensures that the first letter has to be $\binom{0}{0}$. The emulation of the configuration is then taken care of by the Δ_i. If the 2-register machine R stops with the sequence of configurations $(1, 0, 0) \vdash (l_2, m_2, n_2) \vdash \dots \vdash (l_r, m_r, n_r)$, then $L(\mathfrak{A}_R) = \{\binom{0}{0}\binom{m_2}{n_2} \cdots \binom{m_r}{n_r}\}$. If R does not stop, then $R = \emptyset$. Since the complete proof for the run is similar to the one before we will omit it here. For infinite words, we simply add the transition $(q_k, x_1^0 = 0 \wedge x_1^1 = 0, q_k)$ to Δ. Then the language accepts the same word with infinitely many $\binom{0}{0}$ appended to it. □

Proposition 4.6. Since the non-emptiness problem for Strong (Büchi) Automata with a look-back of 1 and a word dimension of 2 is undecidable, the non-emptiness problem for Strong (Büchi) Automata with a look-back greater than 1 and a word dimension greater than 1 is undecidable as well.

That completes the proofs for the non-emptiness problem of extended Strong Automata. Next we will cover the closure properties of extended Strong (Büchi) Automata, starting with the union and the intersection, since they can be easily adapted from Nondeterministic Finite Automata and Büchi Automata. The Complementation, however is not as easily adopted and will be handled separately.

4.3 Union and Intersection of extended Strong (Büchi) Automata

Let us start with the union of extended Strong (Büchi) Automata.

Theorem 4.7. *Given two extended Strong (Büchi) Automata with a look-back of s $\mathfrak{A}_1 = (Q_1, M^n, q_1^1, \Delta_1, F_1)$ and $\mathfrak{A}_2 = (Q_2, M^n, q_1^2, \Delta_2, F_2)$ for the alphabet frame $(\mathcal{M}, \mathcal{L})$. Then the language $L(\mathfrak{A}_1) \cup L(\mathfrak{A}_2)$ can be recognized by an extended Strong (Büchi) Automata with a look-back of s as well.*

Proof. Let $\mathfrak{A}_1$ and $\mathfrak{A}_2$ be as described above. Then we can w.l.o.g. assume that $Q_1 \cap Q_2 = \emptyset$ and $q_0 \notin Q_1 \cup Q_2$. Define $\mathfrak{B} = \{Q, M^n, q_0, \Delta, F\}$ where

- $Q = Q_1 \cup Q_2 \cup \{q_0\}$
- $\Delta = \Delta_1 \cup \Delta_2 \cup \{(q_0, \psi, p) \mid (q_1^1, \psi, p) \in \Delta_1\} \cup \{(q_0, \psi, p) \mid (q_1^2, \psi, p) \in \Delta_2\}$
- $F = F_1 \cup F_2 \cup \{q_0 \mid q_1^1 \in F_1 \text{ or } q_1^2 \in F_2\}$

Observe that $\mathfrak{B}$ has a maximum look-back of s, since it only features formulas from $\mathfrak{A}_1$ and $\mathfrak{A}_2$.

To finish the proof we need to show that $L(\mathfrak{B}) = L(\mathfrak{A}_1) \cup L(\mathfrak{A}_2)$. Suppose $w \in L(\mathfrak{B})$. If $w = \varepsilon$, then $q_0 \in F$. Due to the construction that means $q_1^1 \in F_1$ or $q_1^2 \in F_2$. Therefore $\varepsilon \in L(\mathfrak{A}_1)$ or $L(\mathfrak{A}_2)$. If $w \neq \varepsilon$ then there exists a run ρ with $\rho(|w|) \in F_1 \cup F_2$, since q_0 only has outgoing transitions. W.l.o.g. assume $\rho(|w|) \in F_1$, since there is no transition from a state of Q_1 to a state from Q_2, $\rho(i) \in Q_1 \forall 0 < i \leq |w|$. Let the run take as first transition $(q_0, \psi, \rho(1)) \in \mathit{Delta}$. By construction $(q_1^1, \psi, \rho(1)) \in \Delta_1$. Therefore the run ρ' of $\mathfrak{A}_1$ on the word w exists with $\rho'(0) = q_1^1$ and $\rho'(i) = \rho(i) \forall 0 < i \leq |w|$, and therefore $w \in L(\mathfrak{A}_1) \subseteq L(\mathfrak{A}_1) \cup L(\mathfrak{A}_2)$. Therefore $L(\mathfrak{B}) \subseteq L(\mathfrak{A}_1) \cup L(\mathfrak{A}_2)$. The direction for $L(\mathfrak{A}_1) \cup L(\mathfrak{A}_2) \subseteq L(\mathfrak{B})$ uses very similar arguments, which is why we will omit it here. Furthermore, the proof for the inclusions of extended Strong Büchi Automata is analogous to the proof for extended Strong Automata, which is why we will omit it here as well. From the two inclusion we can infer that $L(\mathfrak{B}) = L(\mathfrak{A}_1) \cup L(\mathfrak{A}_2)$ for both, extended Strong Automata and extended Strong Büchi Automata. □

Now let's look at the intersection. Both, extended Strong Automata and extended Strong Büchi Automata will be closed under intersection. The construction of the respective automaton is different though, which is why we will cover them as two separate problems. We will begin with extended Strong Automata.

Theorem 4.8. *Given two extended Strong Automata with a look-back of s* $\mathfrak{A}_1 = (Q_1, M^n, q_1^1, \Delta_1, F_1)$ *and* $\mathfrak{A}_2 = (Q_2, M^n, q_1^2, \Delta_2, F_2)$ *for the alphabet frame* $(\mathcal{M}, \mathcal{L})$*. Then the language* $L(\mathfrak{A}_1) \cap L(\mathfrak{A}_2)$ *can be recognized by an extended Strong Automata with a look-back of s as well.*

Proof. We will construct $\mathfrak{B} = (Q_1 \times Q_2, M^n, (q_1^1, q_1^2), \Delta, F_1 \times F_2)$ with Δ as

$$\Delta = \{((q_1, q_2), \varphi_1(x_0^1, ..., x_i^n) \wedge \varphi_2(x_0^1, ..., x_i^n), (p_1, p_2)) \mid (q_1, \varphi_1(x_0^1, ..., x_i^n), p_1) \in \Delta_1 \text{ and } (q_2, \varphi_1(x_0^1, ..., x_i^n), p_2) \in \Delta_2\}$$

Note that $\mathfrak{B}$ is of look-back s, since $\mathfrak{A}_1$ and $\mathfrak{A}_2$ are of look-back s, and $\mathfrak{B}$ uses the same formulas. Now we need to show that $L(\mathfrak{B}) = L(\mathfrak{A}_1) \cap L(\mathfrak{A}_2)$. If $w \in L(\mathfrak{B})$, then there exists a run ρ of $\mathfrak{B}$ on w with $\rho(|w|) \in F_1 \times F_2$. Due to the definition of $\mathfrak{B}$, there exists the run ρ_1 of $\mathfrak{A}_1$ on w and the run ρ_2 of $\mathfrak{A}_2$ on w s.t. $\rho(i) = (\rho_1(i), \rho_2(i))$. Since $\rho(|w|) \in F_1 \times F_2$, that means $\rho_1(|w|) \in F_1$ and $\rho_2(|w|) \in F_2$, which in turn means $w \in L(\mathfrak{A}_1)$ and $w \in L(\mathfrak{A}_2)$. Similarly, if $w \in L(\mathfrak{A}_1) \cap L(\mathfrak{A}_2)$, then there exist the runs ρ_1 and ρ_2 for $\mathfrak{A}_1$, respective $\mathfrak{A}_2$ on w, s.t. $\rho_1(|w|) \in F_1$ and $\rho_2(|w|) \in F_2$. Then, by definition of $\mathfrak{B}$, there exists the run ρ of $\mathfrak{B}$ on w with $\rho(i) = (\rho_1(i), \rho_2(i))$. Furthermore that means $\rho(|w|) \in F_1 \times F_2$ and with that $w \in L(\mathfrak{B})$. Finally this means that $L(\mathfrak{B}) = L(\mathfrak{A}_1) \cap L(\mathfrak{A}_2)$. □

Theorem 4.9. *Given two extended Strong Büchi Automata with a look-back of s* $\mathfrak{A}_1 = (Q_1, M^n, q_1^1, \Delta_1, F_1)$ *and* $\mathfrak{A}_2 = (Q_2, M^n, q_1^2, \Delta_2, F_2)$ *for the alphabet frame* $(\mathcal{M}, \mathcal{L})$*. Then the language* $L(\mathfrak{A}_1) \cap L(\mathfrak{A}_2)$ *can be recognized by an extended Strong Büchi Automata with a look-back of s as well.*

Proof. We will construct $\mathfrak{B} = (Q_1 \times Q_2 \times \{1, 2\}, M^n, (q_1^1, q_1^2, 1), \Delta, Q \times F_2 \times \{2\})$ with Δ as

$$\begin{aligned}\Delta = \{&((q_1, q_2, 1), \varphi_1(x_0^1, ..., x_i^n) \wedge \varphi_2(x_0^1, ..., x_i^n), (p_1, p_2, 1)) \mid (q_1, \varphi_1(x_0^1, ..., x_i^n), p_1) \in \Delta_1, \\ &(q_2, \varphi_2(x_0^1, ..., x_i^n), p_2) \in \Delta_2 \text{ and } (q_1, q_2) \in (Q_1 \backslash F_1) \times Q_2\} \cup \\ \{&((q_1, q_2, 1), \varphi_1(x_0^1, ..., x_i^n) \wedge \varphi_2(x_0^1, ..., x_i^n), (p_1, p_2, 2)) \mid (q_1, \varphi_1(x_0^1, ..., x_i^n), p_1) \in \Delta_1, \\ &(q_2, \varphi_2(x_0^1, ..., x_i^n), p_2) \in \Delta_2 \text{ and } (q_1, q_2) \in F_1 \times Q_2\} \cup \\ \{&((q_1, q_2, 2), \varphi_1(x_0^1, ..., x_i^n) \wedge \varphi_2(x_0^1, ..., x_i^n), (p_1, p_2, 2)) \mid (q_1, \varphi_1(x_0^1, ..., x_i^n), p_1) \in \Delta_1, \\ &(q_2, \varphi_2(x_0^1, ..., x_i^n), p_2) \in \Delta_2 \text{ and } (q_1, q_2) \in Q_1 \times (Q_2 \backslash F_2)\} \cup \\ \{&((q_1, q_2, 2), \varphi_1(x_0^1, ..., x_i^n) \wedge \varphi_2(x_0^1, ..., x_i^n), (p_1, p_2, 1)) \mid (q_1, \varphi_1(x_0^1, ..., x_i^n), p_1) \in \Delta_1, \\ &(q_2, \varphi_2(x_0^1, ..., x_i^n), p_2) \in \Delta_2 \text{ and } (q_1, q_2) \in Q_1 \times F_2\}\end{aligned}$$

Note that $\mathfrak{B}$ is of look-back s, since $\mathfrak{A}_1$ and $\mathfrak{A}_2$ are of look-back s, and $\mathfrak{B}$ uses the same formulas. Now we need to show that $L(\mathfrak{B}) = L(\mathfrak{A}_1) \cap L(\mathfrak{A}_2)$. If $\alpha \in L(\mathfrak{B})$, then there exists an accepting run. That in turn means there is an infinite sequence $i_1 < i_2 < i_3 < ...$ with $\rho(i_j) \in Q_1 \times F_2 \times \{2\}$. Furthermore, due to the construction of $\mathfrak{B}$, we know there also exists an infinite sequence $i'_1 < i'_2 < ...$ with $i_j < i'_{j+1}$, $i'_j < i_j$ and $\rho(i'_j) \in F_1 \times Q_2 \times \{1\}$. We also know, again due to the construction of $\mathfrak{B}$, that there exists a run ρ_1 of $\mathfrak{A}_1$ on α and a run ρ_2 of $\mathfrak{A}_2$ on α such that $\rho(i) = (\rho_1(i), \rho_2(i), 1)$ or $\rho(i) = (\rho_1(i), \rho_2(i), 2)$. Since $\rho(i'_j) \in F_1 \times Q_2 \times \{1\}$, $\rho_1(i'_j) \in F_1$ must also hold, and with that $\alpha \in L(\mathfrak{A}_1)$. The same holds for $\mathfrak{A}_2$ with i_j respectively. Hence $\alpha \in L(\mathfrak{A}_1) \cap L(\mathfrak{A}_2)$.

Similarly if $\alpha \in L(\mathfrak{A}_1) \cap L(\mathfrak{A}_2)$ there exist the runs ρ_1 of $\mathfrak{A}_1$ on α and ρ_2 if $\mathfrak{A}_2$ on α. By definition of $\mathfrak{B}$ we know there is a run ρ of $\mathfrak{B}$ on α with $\rho(i) = (\rho_1(i), \rho_2(i), 1)$ or $\rho(i) = (\rho_1(i), \rho_2(i), 2)$. Let $M_j = \{i \mid \rho_j(i) \in F_j\}$ be the set of all states where ρ_j visits a final state. We know that M_j is infinite, since $\mathfrak{A}_j$ accepts α. Now we need an infinite sequence $n_1 < n_2 < ...$ such that $\rho(n_i) \in Q_1 \times F_2 \times \{2\}$. For that we will need another infinite

sequence $m_1 < m_2 < m_3$ such that $\rho(m_i) \in F_1 \times Q_2 \times \{1\}$ and $m_1 < n_1 < m_2 < n_2 < ....$ We will define the m_i and n_i as follows:

- Let $m_1 = min(M_1)$.
- Given m_i define M_2^i as $M_2^i = \{n \in M_2 \mid n > m_i\}$ and with that $n_i = min(M_2^i)$.
- Given n_i define M_1^i as $M_1^i = \{m \in M_1 \mid m > n_i\}$ and with that $m_{i+1} = min(M_1^i)$.

By defining m_i and n_i like that, we know, due to the definition of $\mathfrak{B}$ and that $\alpha \in L(\mathfrak{A}_1) \cap L(\mathfrak{A}_2)$, that

- $\rho(i) = (\rho_1(i), \rho_2(i), 1)$ for all $i \leq m_1$
- $\rho(i) = (\rho_1(i), \rho_2(i), 2)$ for all $m_i < i \leq n_i$
- and $\rho(i) = (\rho_1(i), \rho_2(i), 1)$ for all $n_i < i \leq m_{i+1}$.

And with that we know that $(\rho_1(n_i), \rho_2(n_i), 2) \in Q_1 \times F_2 \times \{2\}$, and therefore $\mathfrak{B}$ accepts the word α. And with that $\alpha \in L(\mathfrak{B})$. From those two inclusions relations we can infer that $L(\mathfrak{B}) = L(\mathfrak{A}_1) \cap L(\mathfrak{A}_2)$. □

And with that we can conclude this chapter with the results that extended Strong Automata and extended Strong Büchi Automata with any fixed look-back are closed under union and intersection. With that, only the complementation of extended Strong (Büchi) Automata is left.

4.4 Complementation of extended Strong Automata

In this section we are going to cover the complementation of extended Strong Automata for any look-back s. This will also cover the complementation for $\mathcal{M}$-$\mathcal{L}$-Automata and Strong Automata.

Theorem 4.10. *If $L \subseteq (M^n)^*$ is recognizable by an extended strong automaton $\mathfrak{A} = (Q, M^n, q_1, \Delta, F)$ with a look-back of s, then $(M^n)^* \backslash L$ is recognizable by an extended Strong automaton with a look-back of s as well. In other words, extended Strong Automata with look-back s are closed under complementation.*

Proof. We will show this by constructing an automaton $\mathfrak{B}$ from the automaton $\mathfrak{A}$ which recognizes the language $(M^n)^* \backslash L$. In order to do that, we will first construct an equivalent deterministic extended strong automaton $\mathfrak{A}'$, which we will use to construct $\mathfrak{B}$.

Let $\mathfrak{A} = (Q, M^n, q_1, \Delta, F)$ be an extended Strong Automaton with look-back s, then a corresponding deterministic extended strong automaton $\mathfrak{A}'$ is

$$\mathfrak{A}' = (2^Q, M^n, \{q_1\}, \Delta', F')$$

where

- $(A, \psi'(x_0^1, ..., x_i^n), B) \in \Delta'$ for $0 \leq i \leq s$ iff

$$\psi'(x_0^1, ..., x_i^n) := (\bigwedge_{q \in B} \bigvee_{p \in A} \bigvee_{\psi_{p,q}(x_0^1,...,x_i^n) \in \Delta} \psi_{p,q}(x_0^1, ..., x_i^n)) \wedge \bigwedge_{q \in (Q \backslash B)} \bigwedge_{p \in A} \bigwedge_{\psi_{q,p}(x_0^1,...,x_i^n) \in \Delta} \neg \psi_{p,q}(x_0^1, ..., x_i^n)$$

- $F' = \{A \subseteq Q \mid A \cap F \neq \emptyset\}$

Note that if there is no incoming transition to a $q \in B$ from some $p \in A$, then all disjunctions for that q will be on the empty set, which is equivalent to $false$, which will force the whole formula to be equivalent to $false$. If the same holds for some $q \in (Q \backslash B)$, then the conjunction has the empty set as argument which will evaluate to $true$. That will therefore not affect the evaluation from the formula.

Now we need to show two claims.

1. For every $A \in 2^Q$ and every sequence of letters $a_0, .., a_i \in M^n$ for $0 \leq i \leq s$ exists exactly one formula $\psi_{q,p}(x_0^1, ..., x_i^n) \in \Delta'$, s.t. $\mathcal{M} \models \psi_{q,p}[a_0^1, ..., a_i^n]$

2. $\mathfrak{A}$ and $\mathfrak{A}'$ recognize the same language.

The first claim consists of two parts. Firstly that for each state and each sequence of letters there do not exist two transition which could be taken at the same time. Secondly that for each state and each sequence of letters there has to exist a transition which can be taken.

We will show the first part by contradiction. For that we will assume there exists some $A \in 2^Q$ and some sequence of letters $a_0, .., a_i \in M^n$ for $0 \leq i \leq s$, s.t. there exist $\psi_{A,P_1}(x_0^1, ..., x_i^n), \psi_{A,P_2}(x_0^1, ..., x_i^n) \in \Delta'$ where $\mathcal{M} \models \psi_{A,P_1}[a_0^1, ..., a_i^n]$, $\mathcal{M} \models \psi_{A,P_2}[a_0^1, ..., a_i^n]$ and $P_1 \neq P_2$.

Since $P_1 \neq P_2$, there exists some $p \in ((P_1 \cup P_2) \backslash (P_1 \cap P_2))$. W.l.o.g. let p be in P_1. By definition of Δ', $\psi_{A,P_1}(x_0^1, ..., x_i^n)$ there has to exist some $\psi_{q,p}(x_0^1, ..., x_i^n) \in \Delta$ for some $q \in A$, s.t. $\mathcal{M} \models \psi_{q,p}[a_0(1), ..., a_i(n)]$, otherwise $\mathcal{M} \not\models \psi_{A,P_1}[a_0(1), ..., a_i(n)]$. But then $\mathcal{M} \not\models \neg\psi_{q,p}[a_0(1), ..., a_i(n)]$. By Definition of $\psi_{A,P_2}(x_0^1, ..., x_i^n)$ it follows that $\mathcal{M} \not\models \psi_{A,P_2}[a_0(1), ..., a_i(n)]$, since that formula needs $\mathcal{M} \models \neg\psi_{q,p}[a_0(1), ..., a_i(n)]$ to hold. That contradicts the assumption. And due to that it follows that there can not exist two transitions with the same source to different target states that can be fulfilled by the same sequence of letters.

The second part of the first claim can be shown by giving the state in which we will inevitably end when being in state $A \in 2^Q$ and the sequence of letters $a_0, .., a_i \in M^n$ for $0 \leq i \leq$. Let $P = \{p \in Q \mid q \in A \wedge \psi_{q,p}(x_0^1, ..., x_i^n) \in \Delta \wedge \mathcal{M} \models \psi_{q,p}[a_0(1), ..., a_i(n)]\}$. By definition, $\psi_{A,P}(x_0^1, ..., x_i^n) \in \Delta'$ and by construction of $\psi_{A,P}(x_0^1, ..., x_i^n)$, $\mathcal{M} \models \psi_{A,P}[a_0(1), ..., a_i(n)]$ must also hold. This, together with the first part shows the first claim.

In order to show the second claim we will show that for every complete run ρ of word w in the automaton $\mathfrak{A}$ and the deterministic run ρ' for the word w in the automaton $\mathfrak{A}'$

$$\rho(i) \in \rho'(i) \text{ for } 0 \leq i \leq |w|$$

has to hold.

We will show this by induction on the length of the word w. If $|w| = 0$, then $w = \varepsilon$. The only existing run ρ of $\mathfrak{A}$ on w is $\rho = q_1$. The only possible run of $\mathfrak{A}'$ on w is $\rho = \{q_1\}$. Therefore the claim holds for $|w| = 0$.

As induction hypothesis we will assume that the induction holds for all words $|w|$ of length m. It remains to show that the case holds for $m + 1$ as well. Let $w' = wa$ for $a \in M^n$ be such a word.

Due to the induction hypothesis we can assume that $\rho(m) \in \rho'(m)$. Since ρ is a complete run there has to be a formula $\psi_{\rho(m),\rho(m+1)}$ s.t. the formula is satisfied in $\mathcal{M}$ by the last s letters and the current letter a. By construction and determinism of AA', $\rho(m+1) \in \rho'(m+1)$. Otherwise the $\neg\psi_{\rho(m),\rho(m+1)}$ has to be true with the same assignment, which would contradict the assumption. Therefore the claim follows.

With that lemma we can show that $L(\mathfrak{A}) \subseteq L(\mathfrak{A}')$. If $w \in L(\mathfrak{A})$, there exists a run ρ in $\mathfrak{A}$, s.t. $\rho(|w|) \in F$. Due to the previous lemma we know $\rho(|w|) \in \rho'(|w|)$. Therefore $\rho'(|w|) \cap F \neq \emptyset$ and by construction $\rho'(|w|) \in F'$. Therefore $w \in L(\mathfrak{A}')$.

Now it remains to show that $L(\mathfrak{A}') \subseteq L(\mathfrak{A})$. In order to show that, we will show that for the unique run ρ of $\mathfrak{A}'$ on the word w the following has to hold.

$$q \in \rho(|w|) \text{ then there exits a run } \rho' \text{ of } w \text{ in } \mathfrak{A} \text{ s.t. } \rho'(|w|) = q.$$

We will show this by induction on the length of the word w. If $|w| = 0$, then $w = \varepsilon$. And by definition $\rho'(0) = q_1 \in \{q_1\} = \rho(0)$.

As induction hypothesis we will assume the assumption holds for a word w where $|w| = m$. Now we need to show that it also holds for the word wa for $a \in M^n$.

By construction of $\mathfrak{A}'$ we know that for each $q \in \rho(n + 1)$ some $p \in \rho(n)$ exists such that $\psi_{p,q}(x_1^1, ..., x_i^n) \in \Delta$ and the last $i - 1$ letters and a fulfill that formula. Otherwise $\psi_{\rho(m),\rho(m+1)}(x_1^1, ..., x_i^n)$ could not be fulfilled by the same sequence of letters. Then, by induction hypothesis we know there exists a run ρ' of w in $\mathfrak{A}$, s.t. $\rho'(m) = p$. Furthermore we know, since $\psi_{p,q}(x_1^1, ..., x_i^n)$ holds for the sequence of letters wa, that we can go from $\rho'(m)$ to q with the letter a and the previous word w. Therefore there exists a run ρ'' of wa in $\mathfrak{A}$ s.t. $\rho''(m + 1) = q$.

With this we can show that $L(\mathfrak{A}') \subseteq L(\mathfrak{A})$. Let $w \in L(\mathfrak{A}')$. Then the unique run ρ on w of $\mathfrak{A}'$ exists, s.t. $q \in \rho(|w|)$ with $q \in F$, since it would not be accepted otherwise. By the previous lemma we know that there exists some run ρ' of $\mathfrak{A}$ on the word w, s.t. $\rho'(|w|) \in F$. Therefore $w \in L(\mathfrak{A})$.

With that we have finally shown that it is possible to determinize any extended Strong Automaton $\mathfrak{A}$. Since $\mathfrak{A}'$ is deterministic, we can define $\mathfrak{B}$ as $(2^Q, M^n, \{q_1\}, \Delta', 2^Q \backslash F')$. It is clear that $\mathfrak{B}$ accepts the language $(M^n)^* \backslash L(\mathfrak{A})$.

□

Proposition 4.11. For the alphabet frame ($\mathcal{M}$,MSO-logic), where the MSO-logic of $\mathcal{M}$ is decidable, the non-universality and equivalence problem for extended Strong Automata are decidable for a look-back of 0 and for a look-back of 1 where the word dimension is also 1, and otherwise undecidable.

Proof. The non-universality problem can be reduced to the complement and the non-emptiness problem.

$$\begin{aligned} &L(\mathfrak{A}) \neq (M^n)^* \\ \Leftrightarrow &((M^n)^* \backslash L(\mathfrak{A})) \neq \emptyset \end{aligned}$$

Since all extended Strong Automata, regardless of look-back and word dimension, are closed under complementation only the non-emptiness problem remains. Since that problem is decidable for $\mathcal{M}$-$\mathcal{L}$-Automata regardless of word dimension and for Strong Automata for words of dimension 1, the universality problem is decidable as well.

For the other cases we know the non-emptiness problem is undecidable. If the universality problem were decidable we would be able to solve the non-emptiness problem by checking whether the complement of the language is universal or not. Therefore those cases are undecidable.

The equivalence problem can be reduced to the complement, intersection and the non-emptiness problem.

$$\begin{aligned} &L(\mathfrak{A}) = L(\mathfrak{B}) \\ \Leftrightarrow &L(\mathfrak{A}) \cap ((M^n)^* \backslash L(\mathfrak{B}) = \emptyset \end{aligned}$$

As before, we use the decidability for both closure properties and the non-emptiness problem to decide this problem. And, as before, we use that only the non-emptiness problem is undecidable for the other types, s.t. the decidability of the equivalence problem would contradict the undecidability for the non-emptiness problem, since all extended Strong Automata are closed under complementation and intersection.

□

4.5 Complementation of extended Strong Büchi Automata

The same results hold for the extended Strong Büchi Automata. For the equivalence and the universality problem we will once again use that extended Strong Büchi Automata are closed under complementation, union and intersection. For that we will first show that extended Strong Büchi Automata for any fixed look-back are closed under complementation.

Theorem 4.12. *If $L \subseteq (M^n)^\omega$ is recognizable by an extended Strong Büchi Automaton with a look-back of s $\mathfrak{A} = (Q, M^n, q_1, \Delta, F)$ over the alphabet frame $(\mathcal{M}, \mathcal{L})$, then $(M^n)^\omega \backslash L$ is recognizable by an extended Strong Büchi Automaton with look-back s as well. In other words, extended Strong Büchi Automata with look-back s are closed under complementation.*

Proof. We will focus on the differences to normal Büchi Automata during this proof and explain the approach with a simple example. We will once again use the language $L = \{012\cdots\}$ as the example language. This time however, we will use $\mathfrak{A}_c$ as in Figure 4.2 to recognize the language L. Since the transitions $x_0 = 0$ only considers the current letter, the transition can only be used for the first step. The transition $x_0 + 1 = x_1$ considers the current and the last letter, however, which is why this transition will be used after the first letter has been read. By the definition of a run it follows that $L(\mathfrak{A}_c) = L$.

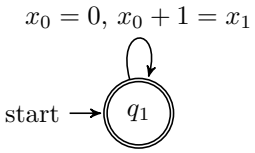

Figure 4.2: extended Strong Büchi Automaton $\mathfrak{A}_c$ recognizing L

In his original proof Büchi used the characterization of Büchi recognizable languages as $\bigcup_{i=1}^{n} U_i \cdot V_i^\omega$, where $U \cdot V = \{uv \mid u \in U, v \in V\}$ and $V^\omega = \{v_1 v_2 ... \mid v_i \in V\}$, and the given automaton to build blocks in such a way that the complement language could also be built as $\bigcup_{i=1}^{n} U_i \cdot V_i^\omega$ for some regular U_i and V_i, cf. [Bü62]. We will choose the same approach and show how the case of extended Strong Büchi Automata differs from the original proof. For that we first need introduce a transition relation $\sim_{\mathfrak{A}}$.

Definition 4.13. Let $\mathfrak{A} = (Q, M^n, q_0, \Delta, F)$. For $q, p \in Q$ let

- $p \xrightarrow{w} q \Leftrightarrow$ there is a run from p to q via w in $\mathfrak{A}$
- $p \overset{w}{\Rightarrow} q \Leftrightarrow$ there is a run from p to q via w in $\mathfrak{A}$ visiting a state of F.

$$\text{We define } w \sim_{\mathfrak{A}} w' \forall p, q \in Q: \qquad p \xrightarrow{w} q \Leftrightarrow p \xrightarrow{w'} q$$
$$p \overset{w}{\Rightarrow} q \Leftrightarrow p \overset{w'}{\Rightarrow} q$$

Note that

- $\sim_{\mathfrak{A}}$ is an equivalence relation with finitely many equivalence classes, since there are only finitely many states.
- For normal Büchi Automata $\sim_{\mathfrak{A}}$ is a congruence relation with respect to concatenation. This is not the case for extended Strong Büchi Automata with a look-back greater than 0.
- Each equivalence class can be expressed by a extended Strong Automaton.

Given an automaton $\mathfrak{A}$, a procedure to calculate the $\sim_{\mathfrak{A}}$-classes uses so called transition profiles. For normal Büchi Automata it is possible to calculate them by listing all (p, q) with $p \xrightarrow{w} q$ and $p \overset{w}{\Rightarrow} q$ for each finite word w. The equivalence class of a word w is described by its transition profile $\tau(w)$. Since $\sim_{\mathfrak{A}}$ is not a congruence relation in respect to concatenation, we are, in general, not able to calculate the transition profile automaton $\mathrm{TP}(\mathfrak{A})$ efficiently. Instead, we will construct $\mathrm{TP}(\mathfrak{A})$ with the full possible state set. In general, however, $\mathrm{TP}(\mathfrak{A})$ will have a state set, which is far larger than all reachable states from the initial state $\tau(\varepsilon)$.

Definition 4.14. Given an extended Strong Büchi Automaton $\mathfrak{A} = (Q, M^n, q_1, \Delta, F)$ with look-back s over the alphabet frame $(\mathcal{M}, \mathcal{L})$, the respective transition profile automaton is defined as $\mathrm{TP}(\mathfrak{A}) = (Q_t, M^n, \tau(\epsilon), \Delta_t, F_t)$ with

- $Q_t = \{(E, E_f) \mid E, E_f \subseteq Q \times Q \wedge E_f \subseteq E\}$ being the possible transition profiles of $\mathfrak{A}$ with the following semantics:
  - If $(q, p) \in E$, then we can reach p from q via some word.
  - If $(q, p) \in E_f$, then we can reach p from q via some word and some final state of $\mathfrak{A}$ is visited during the run.
- $\tau(\varepsilon) = (E, E_f)$ where $E = \{(q, q) \mid q \in Q\}$ and $E_f = \{(q, q) \mid q \in F\}$
- $\Delta = \{((E, E_f), \psi(x_1^1, ..x_i^n), (E', E'_f)) \mid (E, E_f), (E', E'_f) \in Q_t, i \leq s\}$, where $\psi(x_0^1, .., x_i^n)$ is defined as

$$\begin{aligned}
\psi(x_0^1, .., x_i^n) = & \bigwedge_{(q,p)\in E'_f, p\notin F} \bigvee_{(q,p')\in E_f} \bigvee_{(p', \varphi(x_0^1,..,x_i^n), p)\in\Delta} \varphi(x_0^1, ..x_i^n) \\
\wedge & \bigwedge_{(q,p)\in E'_f, p\in F} \bigvee_{(q,p')\in E} \bigvee_{(p', \varphi(x_0^1,..,x_i^n), p)\in\Delta} \varphi(x_0^1, ..x_i^n) \\
\wedge & \bigwedge_{(q,p)\in E', (q,p)\notin E'_f, p\notin F} \bigvee_{(q,p')\in E} \bigvee_{(p', \varphi(x_0^1,..,x_i^n), p)\in\Delta} \varphi(x_0^1, ..x_i^n) \\
\wedge & \bigwedge_{(q,p)\in E', (q,p)\notin E'_f, p\notin F} \neg \bigvee_{(q,p')\in E_f} \bigvee_{(p', \varphi(x_0^1,..,x_i^n), p)\in\Delta} \varphi(x_0^1, ..x_i^n) \\
\wedge & \bigwedge_{(q,p)\notin E'} \neg \bigvee_{(q,p')\in E} \bigvee_{(p', \varphi(x_0^1,..,x_i^n), p)\in\Delta} \varphi(x_0^1, ..x_i^n)
\end{aligned}$$

The first line takes care of transitions from q to p which visit a final state somewhere along the path, without p being that final state, by coming via a transition which has already visited a final state previously. The second line takes care of the case where p is a final state. Then it does not matter whether we visited a final state previously or not. The third and fourth part checks that we visit p via a transition which has not seen a final state yet. The last part ensures that p is not reachable from q, by prohibiting the existence of a path from q to p via some state p'.

We will not define the final states now, since this automaton will be used with different sets of final states to create the previously mentioned building blocks.

Note that the transition profile automaton of an extended Strong Büchi Automaton is completely deterministic by construction. That means for all states and a sequence of up to $s + 1$ letters exists exactly one transition which is fulfilled by that sequence.

The transition profile automaton of $\mathfrak{A}_c$ is illustrated in Figure 4.3.

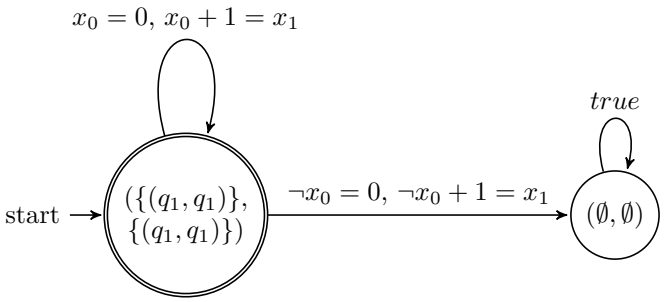

Figure 4.3: Transition profile automaton $\mathrm{TP}(\mathfrak{A}_c)$ of $\mathfrak{A}_c$, reduced to reachable states

In the original proof the Transition-Profile Automaton was used to create the regular languages for the characterization of Büchi languages, by using each state as a potential final state. This means we will use $F_1 = \{(\{(1,1)\},\{(1,1)\})\}$ as the first building block and $F_2 = \{(\emptyset,\emptyset)\}$ as the second building block. For F_1 the recognized language is $L_1 = \{0 \cdots n \mid n \in \mathbb{N}\} \cup \{\varepsilon\}$. For F_2 it is $L_2 = (M^n)^* \backslash L_1$. By using those building blocks, the word recognized by $\mathfrak{A}_c$ is in $L_1 \cdot L_2^\omega$. That language contains a lot of words which are not recognized by $\mathfrak{A}_c$ however. Since we are unable model a finite look-back for an infinite alphabet with states, we need to adjust the concatenation in order to match the transitions.

Furthermore you can see that the equivalence relation $\sim_{\mathfrak{A}_c}$ is not a congruence relation with respect to concatenation. For $u = u' = 0 \in L_1$ and $v = 1, v' = 2 \in L_2$. If $\sim_{\mathfrak{A}_c}$ were a congruence relation $uv \sim_{\mathfrak{A}_c} u'v'$ would hold, but $uv \in L_1$ and $u'v' \in L_2$. Therefore $\sim_{\mathfrak{A}}$ is not a congruence relation with respect to concatenation.

Since extended Strong Büchi Automata need the context of the previously read letters, we will transfer the last s letters which were processed to the new run of the automaton. Then all latter runs start as if they have already read the transferred letters. For that we will use the function

$$\text{last}_s(w) = \begin{cases} w & |w| < s \\ a_1 \dots a_s & va_1a_2 \dots a_s = w, v \in (M^n)^* \end{cases}$$

Now we can define $\cdot_{\mathfrak{A}}$ as a partial function

$$u \cdot_{\mathfrak{A}} v = \begin{cases} uv & \text{, if } \mathfrak{A} \text{ accepts } v \text{ and starts with } \text{last}_s(u) \\ undefined & \text{, otherwise} \end{cases}$$

And with that, we define the concatenation on the language level as.

$$L \cdot_{\mathfrak{A}} L' = \{w \mid w = uv, u \in L, v \in L' = L(\mathfrak{A}), u \cdot_{\mathfrak{A}} v = uv\}$$

If $\mathfrak{A}$ does not accept the language L' the result is undefined. Furthermore, $L \cdot_{\mathfrak{A}} L' \subseteq L \cdot L'$. Similarly we define

$$L^{\omega_{\mathfrak{A}}} = \{w_1 w_2 \cdots \mid L(\mathfrak{A}) = L, \mathfrak{A} \text{ starts with } \text{last}_s(w_1 \cdots w_i) \text{ for the } (i+1)\text{-th run and accepts } w_{i+1}, w_i \in L\}$$

Those definitions require an adjusted definition of a run.

Definition 4.15. A run of an extended Strong Automaton $\mathfrak{A} = (Q, M^n, q_1, \Delta, F)$ with look-back s starting with a word $u = u_1 \cdots u_j$ with $j \leq s$ and $u_i \in M^n$ on a word $w = a_1 ... a_m$ is defined as

- $\rho(0) = q_1$
- $\rho(i) = q$, where $(p, \psi_{p,q}(x_0^1, ...x_{j+i-1}^n), q) \in \Delta$ with $p = \rho(i-1)$ and $\mathcal{M} \models \psi_{q_1,q}[u_1(1), .., u_j(n), a_1(1), ..., a_i(n)]$ $\forall i$ with $0 < i \leq min(m, s-j+1)$
- $\rho(i) = q$ if $\rho(i-1) = p$, $(p, \psi_{p,q}(x_1^1, .., x_s^n), q) \in \Delta$ and $\mathcal{M} \models \psi_{q,p}(u_{i-s+j}(1)..., u_j(n), a_1(1), .., a_i(n))$ for $s - j + 1 < i \leq min(s, m)$
- $\rho(i) = q$ if $\rho(i-1) = p$, $(p, \psi_{p,q}(x_1^1, .., x_s^n), q) \in \Delta$ and $\mathcal{M} \models \psi_{q,p}(a_{i-s}(1), .., a_i(n))$ for $s < i \leq m$

In other words, the automaton starts a run in the first state as if it has read the word u and reentered the first state. A run is successful if the usual conditions hold. For $u = \varepsilon$ the normal definition of a run is used. The definition does not cover any u with $|u| > s$, since it would just consider the last s letters. That can be achieved by using last_s on u.

Note that with this definition it can happen that $L \cdot_{\mathfrak{A}} L' = \emptyset$ although $L, L' \neq \emptyset$. This definition, however, can not be used with the current Transition Profile Automaton yet. The word recognized by $\mathfrak{A}_c$ should be in the concatenation of $L_1 \cdot_{\text{TP}(\mathfrak{A}_c)} L_2^{\omega_{\text{TP}(\mathfrak{A}_c)}}$. That language does not contain that word however. The original word would now be recognized by $L_1 \cdot_{\text{TP}(\mathfrak{A}_c)} L_1^{\omega_{\text{TP}(\mathfrak{A}_c)}}$, if restricted to the transitions and ignoring that partial words $i \cdots j$ for $i \leq j$ are not in L_1. But since L_1 does not contain those words, we can not use the current Transition Profile Automaton to create building blocks for the complementation.

In order to fix that, we will first transform the original automaton $\mathfrak{A}$ to an automaton $\mathfrak{A}'$, such that each state in $\mathfrak{A}'$ has only incoming transition of a fixed look-back level $i < s$ and all outgoing transitions have the look-back size $i + 1$, except for states where i equals the maximum look-back size s. For those states the in- and outgoing transitions have to have the look-back size s. We will call such an automaton normalized. Then we will use that automaton to create a second automaton where all transitions with a look-back level greater than 0 will be quantified to all lower look-back levels.

Definition 4.16. An extended Strong Automata $\mathfrak{A} = (Q, M^n, q_0, \Delta, F)$ with look-back s over the alphabet frame $(\mathcal{M}, \mathcal{L})$ is normalized, iff $\forall q$

- either $\exists i < s$ s.t. $\forall (q, \varphi, p) \in \Delta \Leftrightarrow \varphi(x_0^1, ..., x_i^n)$ and $\forall (p, \varphi, q) \in \Delta \Leftrightarrow \varphi(x_0^1, ..., x_{i-1}^n)$
- or $\forall (q, \varphi, p) \in \Delta \Leftrightarrow \varphi(x_0^1, ..., x_s^n)$ and $\forall (p, \varphi, q) \in \Delta \Leftrightarrow \varphi(x_0^1, ..., x_s^n)$

That way, the set of states is separated in such that each state can only be reached when the current look-back level of the run is necessary to take their outgoing transition.

Lemma 4.17. *Every extended Strong Automaton $\mathfrak{A}$ with a look-back s can be transformed into a normalized extended Strong Automaton $\mathfrak{A}'$ with look-back s, s.t. $L(\mathfrak{A}) = L(\mathfrak{A}')$*

Proof. Given an extended Strong Automaton $\mathfrak{A} = (Q, M^n, q_0, \Delta, F)$ with a look-back of s over the alphabet frame $(\mathcal{M}, \mathcal{L})$. We can build the normalized automaton $\mathfrak{A}'$ as $\mathfrak{A}' = (Q', M^n, q_0^0, \Delta', F')$ with

- $Q' = \{q^i \mid q \in Q, i \leq s\}$
- $\Delta' = \{(q^i, \varphi(x_0^1, ..., x_i^n), p^{i+1}) \mid (q, \varphi(x_0^1, ..., x_i^n), p) \in \Delta \wedge i < s\} \cup$ $\{(q^s, \varphi(x_0^1, ..., x_s^n), p^s) \mid (q, \varphi(x_0^1, ..., x_s^n), p) \in \Delta\}$
- $F' = \{q^i \mid q \in F, i \leq s\}$

It remains to show that $L(\mathfrak{A}) = L(\mathfrak{A}')$.

$$
\begin{aligned}
& w \in L(\mathfrak{A}) \\
\Rightarrow & \exists \text{ run } \rho \text{ of } \mathfrak{A} \text{ on } w, \text{ s.t. } \rho(|w|) \in F \\
\Rightarrow & \exists \text{ run } \rho' \text{ of } \mathfrak{A}' \text{ on } w, \text{ s.t. } \rho'(|w|) \in F' \text{ and } \rho'(i) = \rho(i)^{\min(i,s)}, \forall i \leq |w| \\
\Rightarrow & w \in L(\mathfrak{A}')
\end{aligned}
$$

The other direction is completely analogous to this direction, which is why we will omit it here.

For extended Strong Büchi Automata it suffices to replace $\rho(|w|) \in F$ through $\forall i \exists j > i$ s.t $\rho(j) \in F$ and $\rho'(|w|) \in F'$ through $\forall i \exists j > i$ s.t $\rho'(j) \in F'$ □

Simply normalizing, however, does not solve the problem. Normalizing $\mathfrak{A}_c$ will lead to $\mathfrak{A}'_c$ as

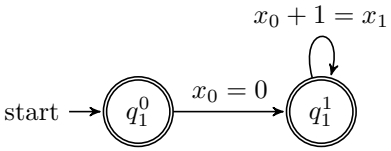

Figure 4.4: normalized extended Strong Büchi Automaton $\mathfrak{A}'_c$ recognizing L

The respective transition profile automaton for $\mathfrak{A}'_c$, reduced to necessary states, has four reachable states

- $p_1 = \{\{(q_1^0, q_1^0), (q_1^1, q_1^1)\}, \{(q_1^0, q_1^0), (q_1^1, q_1^1)\}\}$
- $p_2 = \{\{(q_1^0, q_1^1)\}, \{(q_1^0, q_1^1)\}\}$
- $p_3 = \{\{(q_1^1, q_1^1)\}, \{(q_1^1, q_1^1)\}\}$
- $p_4 = \{\emptyset, \emptyset\}$

with the transition profile automaton being shown in Figure 4.5.

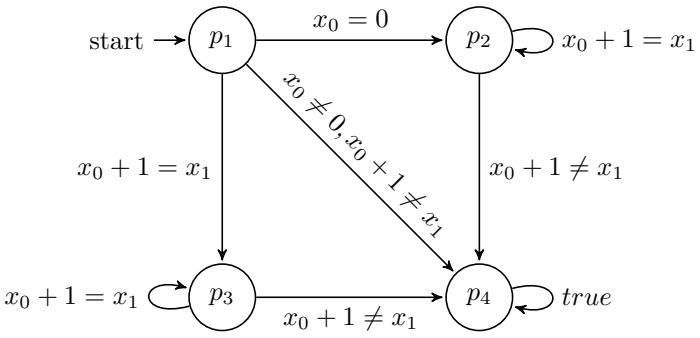

Figure 4.5: Transition Profile Automaton TP($\mathfrak{A}'_c$) of the normalized extended Strong Büchi Automaton $\mathfrak{A}'_c$

Note that the transition from p_1 to p_4 has two formulas, one for look-back level 0 and one for look-back level 1.

As you can see, the languages recognized depending on the final state are $L'_1 = \{\varepsilon\}$, $L'_2 = \{0 \cdots n \mid n \in \mathbb{N}\}$, $L'_3 = \emptyset$ and $L'_4 = ((M^n)^* \backslash L'_1) \backslash L'_2$. L'_3 is empty, since the only incoming transition to p_3 requires at least one letter to be read. That is not possible, however, since the p_1 is the initial state and it can never be reached again after reading a letter. But

this poses a new problem. The original language L would now be $L = L_2' \cdot_{\mathrm{TP}(\mathfrak{A}_c')} L_3'^{\omega_{\mathrm{TP}(\mathfrak{A}_c')}}$. Since L_3' is empty, this is also not possible. Therefore we need to adjust the automaton one more time. We need to add quantified transitions to the normalized automaton. If a transition is for look-back level i, then we will add transitions for all $j < i$ by quantifying the transitions respectively.

Definition 4.18. Quantifying the transitions results, in general, in an automaton which is not normalized anymore. For the sake of simplicity we will call that automaton denormalized.

A denormalized extended Strong Automaton $\mathfrak{A}' = (Q, M^n, q, \Delta', F)$ for the normalized Strong Automaton $\mathfrak{A} = (Q, M^n, q, \Delta, F)$ is defined as

$$\Delta' = \Delta \cup \{(q, \varphi_j(x_0^1, ..., x_j^n), p) \mid (q, \varphi(x_0^1, ..., x_i^n), p) \in \Delta, \forall j \text{ with } j < i\}$$

where $\varphi_j(x_0^1, ..., x_j^n) = \exists x_{j+1}^1 ... x_i^n \varphi(x_{j+1}^1, ..., x_i^n, x_0^1, ..., x_j^n)$

The formulas φ_j simply swaps around the variables in order to have the same variable order as usual.

Note that we are using a normalized automaton in the definition. By normalizing, the automaton gained the property that a state will only be reached once its look-back level transition can just be used. The added transitions with a lower look-back level can never be used, since a run visiting that state needs to take the highest look-back level transition available. Therefore the denormalized Automaton also recognizes the same language as the normalized automaton. Denormalizing a non-normalized automaton will lead to an automaton which recognizes a language that is in general a superset of the language of the original automaton.

For our example automaton that means $\mathfrak{A}_c'$ is transformed to

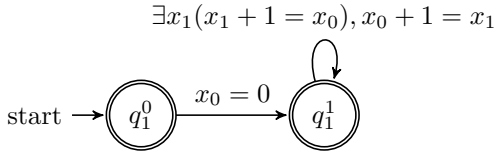

Figure 4.6: Denormalized extended Strong Büchi Automaton $\mathfrak{A}_c''$ recognizing L

As you can see, the new transition can never be taken, since the automaton needs to take a transition with a look-back of 1 once it enters the state q_1^1. The corresponding transition profile automaton $\mathrm{TP}(\mathfrak{A}_c'')$ is illustrated in Figure 4.7.

The languages for p_1 and p_2 do not change. Therefore $L_1'' = L_1'$ and $L_2'' = L_2'$. The language for p_3 did change though. $L_3'' = \{i \cdots j \mid i, j \in \mathbb{N}, 0 < i <= j\}$. And since the automaton is deterministic, $L_4'' = ((((M^n)^*) \backslash L_1'') \backslash L_2'') \backslash L_3''$. Once again $L = L_2'' \cdot_{\mathrm{TP}(\mathfrak{A}_c'')} L_3''^{\omega_{\mathrm{TP}(\mathfrak{A}_c'')}}$. This time, however, both languages contain all the necessary partial words needed to build L.

Before we start the final parts of the proof, we will first show that for denormalized extended Strong Automata the equivalence relation $\sim_{\mathfrak{A}}$ is a congruence relation w.r.t. the parametrized concatenation $\cdot_{\mathrm{TP}(\mathfrak{A})}$.

Lemma 4.19. *Given a denormalized extended Strong Automaton $\mathfrak{A} = (Q, M^n, q_1, \Delta, F)$ with look-back s for the alphabet frame $(\mathcal{M}, \mathcal{L})$. If TP($\mathfrak{A}$) with some final state $\tau(w)$ for some word $w \in (M^n)^*$ accepts some word $w' = a_1 ... a_m$ starting with a memory filled with some*

word $u \in (M^n)^*$ *where* $0 < |u| \leq s$*, then for* $\tau(w) = (E, E_f)$ *and* $\tau(w') = (E', E'_f)$*,* $E \subseteq E'$ *and* $E_f \subseteq E'_f$*.*

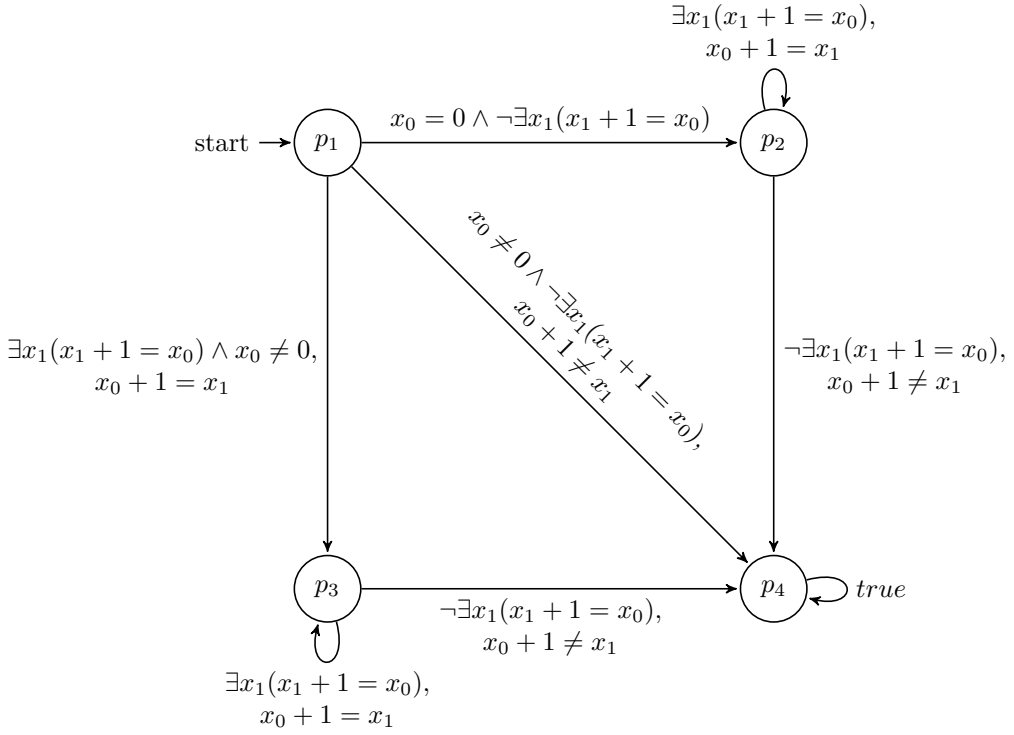

Figure 4.7: Transition Profile Automaton $\mathrm{TP}(\mathfrak{A}''_c)$ of the denormalized extended Strong Büchi Automaton $\mathfrak{A}''_c$

Proof. If $p \to q \in \tau(w)$, and $\rho(|w'|) = \tau(w)$, then that means $\xrightarrow{u} p \xrightarrow{w'} q$ in $\mathfrak{A}$. Assuming that run took some transition $(s, \varphi(x_1^1, ..., x_i^n), t) \in \Delta$ along the way, which was fulfilled as $\mathcal{M} \models \varphi[u_j^1, ..., u_{|u|}^n, a_1^1, ..., a_{i-|u|+j}^n]$ in accordance with the definition of a run. Since $\mathfrak{A}$ is denormalized, there exists a transition $(s, \psi(x_1^1, ..., x_{i-|u|+j}^n), t) \in \Delta$ with $\psi(x_1^1, ..., x_{i-|u|+j}^n), t) = \exists x_{i-|u|+j+1}^1, ..., x_i^n \varphi(x_{i-|u|+j+1}^1, ..., x_i^n, x_1^1, ..., x_{i-|u|+j}^n)$. By assigning $x_{i-|u|+j+1}^1, ..., x_i^n$ the values of $u_j^1, ..., u_{|u|}^n$, it follows that $\mathcal{M} \models \psi[a_1^1, ..., a_{i-|u|+j}^n]$. Therefore the run $p \xrightarrow{w'} q$ exists in $\mathfrak{A}$ as well. From that we can infer that $p \to q \in \tau(w')$ and therefore $E \subseteq E'$. The same arguments can be used to show this for $E_f \subseteq E'_f$, which is why we will omit it here. □

Lemma 4.20. *Let* $\mathfrak{A} = (Q, M^n, q_1, \Delta, F)$ *be a denormalized extended Strong Automata with look-back* s*, then, if* $u \sim_{\mathfrak{A}} v$*,* $u' \sim_{\mathfrak{A}} v'$*,* $u \cdot_{TP(\mathfrak{A})} u' = uu'$ *and* $v \cdot_{TP(\mathfrak{A})} v' = vv'$*, where* $TP(\mathfrak{A})$ *has the only final state* $\tau(u')$*, then* $uu' \sim_{\mathfrak{A}} vv'$*.*

Proof. Since $u \sim_{\mathfrak{A}} v$, $\tau(u) = \tau(v)$ and for the analog reason $\tau(u') = \tau(v')$. If $p \to q \in \tau(uu')$, then there exists some $t \in Q$ s.t. $p \xrightarrow{u} t \xrightarrow{u'} q$. Since $t \xrightarrow{u'} q$ starting as if having read $\mathrm{last}_s(u)$,

there also must be $t \to q \in \tau(u')$, since the previous lemma can be applied due to $\mathfrak{A}$ being denormalized. Therefore $t \to q \in \tau(v')$, since $u' \sim_{\mathfrak{A}} v'$. But then, since $p \xrightarrow{v} t$, $t \to q \in \tau(v')$ and $v \cdot_{\mathrm{TP}(\mathfrak{A})} v' = vv'$, $p \xrightarrow{v} t \xrightarrow{v'} q$ is also a possible run of $\mathfrak{A}$. Therefore $p \to q \in \tau(vv')$. The case of $p \Rightarrow q \in \tau(uu')$ is similar to the case for $p \to q$, where one or both partial paths needs to visit a final state. □

Now, to complete the proof, we need to show the following statements.

1. Given a language of finite words L and an Automaton $\mathfrak{A}$ with $L(\mathfrak{A}) = L$, it is possible to construct an automaton $\mathfrak{A}'$ such that $L(\mathfrak{A}') = L^{\omega_{\mathfrak{A}}}$
2. Given a language of finite words L_1 and a language of infinite words L_2, as well as 2 extended Strong (Büchi) Automata $\mathfrak{A}_1$ and $\mathfrak{A}_2$ with $L(\mathfrak{A}_1) = L_1$ and $L(\mathfrak{A}_2) = L_2$, it is possible to construct an extended Strong Büchi Automaton $\mathfrak{A}'$ with $\mathfrak{A}'$ with $L(\mathfrak{A}') = L_1 \cdot_{\mathfrak{A}_2} L_2$
3. Given the transition profile automaton $\mathfrak{A}'$ of the denormalized extended Strong Büchi Automaton $\mathfrak{A}$ recognizing L and $L(\mathfrak{A}'_q) \cdot_{\mathfrak{A}'} L(\mathfrak{A}'_p)^{\omega_{\mathfrak{A}'}} \cap L \neq \emptyset$, then $L(\mathfrak{A}'_q) \cdot_{\mathfrak{A}'} L(\mathfrak{A}'_p)^{\omega_{\mathfrak{A}'}} \subseteq L$ where $\mathfrak{A}'_q$ and $\mathfrak{A}'_p$ are the transition profile automaton $\mathfrak{A}'$ with q and p as final state, where q and p are some transition profile of $\mathfrak{A}$
4. And lastly, each $\alpha \in (M^n)^\omega$ can be decomposed into $\alpha = w_0 \cdot w_1 \cdot w_2 \cdots$ such that w_i in some fixed $\sim_{\mathfrak{A}}$-class, where $\mathfrak{A}$ is a denormalized extended Strong Automaton recognizing some language L, for all $i > 0$ and $v_0 \cdots v_{i-1} \cdot_{\mathrm{TP}(\mathfrak{A})} v_i$ is defined for all $i > 0$.

The first two points are exactly the same for normal Büchi Automata. The different semantic of the parametrized concatenation and the parametrized $^\omega$ operator are taken care of by the transitions of the extended Strong Büchi Automata. We don't need to show the union, since it has already been shown earlier.

To show the fourth point, suppose $\alpha \in L(\mathfrak{A}'_q) \cdot_{\mathfrak{A}'} L(\mathfrak{A}'_p)^{\omega_{\mathfrak{A}'}} \cap L \neq \emptyset$. That means $\mathfrak{A}$ has a run ρ visiting a final state infinitely often. That means α has the form $\alpha = uv_1v_2v_3 \cdots$ with $u \in L(\mathfrak{A}'_q)$ and $v_i \in L(\mathfrak{A}'_p)$. The accepting run has the form $q_0 \xrightarrow{u} p_1 \xrightarrow{v_0} p_2 \xrightarrow{v_1} p_3 \cdots$ where a final state is visited for infinitely many segment runs $p_i \overset{v_i}{\Rightarrow} p_{i+1}$. Now we need to show that any $\beta \in L(\mathfrak{A}'_q) \cdot_{\mathfrak{A}'} L(\mathfrak{A}'_p)^{\omega_{\mathfrak{A}'}}$ is also accepted by $\mathfrak{A}$. Since $\beta \in L(\mathfrak{A}'_q) \cdot_{\mathfrak{A}'} L(\mathfrak{A}'_p)^{\omega_{\mathfrak{A}'}}$, it must also be of the form $\beta = u'v'_1v'_2v'_3 \cdots$ with $u \in L(\mathfrak{A}'_q)$ and $v_i \in L(\mathfrak{A}'_p)$. Since u' and v'_i lead to the same set of states in $\mathfrak{A}$ as u and v_i do, we obtain a run $q_0 \xrightarrow{u'} p_1 \xrightarrow{v'_0} p_2 \xrightarrow{v'_1} p_3 \cdots$ passing a final state of v'_i for exactly those indices i for which this was the case for v_i, since the transition profiles are closed under the parametrized concatenation for the Transition Profile Automaton for denormalized extended Strong Büchi Automata. Therefore $\mathfrak{A}$ accepts the word β.

The proof for the fifth theorem uses a combinatorial result due to Ramsey. For that, let $\alpha \in (M^n)^\omega$ and $\mathfrak{A}$ be the denormalized Strong Büchi Automata for some language L. Define sets $M_i \subseteq \mathbb{N}$ for $i = 0, 1, 2, ...$

- $M_0 = \mathbb{N}$
- Given M_i, define M_{i+1} as follows. Let $m_i := min(M_i)$, and consider the transition profiles of $\alpha[m_i, n)$ for all $n \in M_i, n > m_i$. Choose some transition profile τ_i, s.t. there

are infinitely many $n \in M_i$ with $\alpha[0, m_i) \cdot_{\mathrm{TP}(\mathfrak{A})} \alpha[m_i, n)$ is defined for $\mathrm{TP}(\mathfrak{A})$ with τ_i as final state.

$$M_{i+1} := \{n \in M_i : n > m_i, \text{ s.t. } \alpha[0, m_i) \cdot_{\mathrm{TP}(\mathfrak{A})} \alpha[m_i, n) \text{ is defined for } \mathrm{TP}(\mathfrak{A}) \text{ with } \tau_i \text{ as final state}\}$$

In other words, $\alpha[m_i, n)$ starting with with $\alpha[0, m_i)$ has τ_i as final state. Note that M_{i+1} is an infinite subset of M_i.

By doing this, we obtain a sequence $0 = m_0 < m_1 < m_2 ....$ Observe that the transition profile of $\alpha[m_i, m_j)$ when starting with $\mathrm{last}_s(\alpha[0, m_i))$ equals τ_i for all $i < j$.

Since there are only finitely many transition profiles, some profile τ has to occur infinitely often among the profiles τ_i. Now take a subsequence $n_o < n_1 < n_2 < ...$ of $m_0 < m_1 < m_2 < ...$ s.t. the profile of $\alpha[n_i, m_j)$ when starting with $\mathrm{last}_s(\alpha[0, n_i))$ equals τ for $n_i < m_j$. Due to the subsequence property is $n_i < n_k$ for all $i < k$, which implies that the transition profile of $\alpha[n_i, n_k)$ when starting with $\mathrm{last}_s(\alpha[0, n_i))$ equals τ.

That way we can decompose α into $\alpha = \alpha[0, n_0)\alpha[n_0, n_1]\alpha[n_1, n_2)....$ Let $\tau = \tau(\alpha[0, n_0))$ and $\tau' = \tau(\alpha[n_0, n_1))$ when starting with $\mathrm{last}_s(\alpha[0, n_0))$. With those we define $\mathrm{TP}(\mathfrak{A}_\tau)$ as $\mathrm{TP}(\mathfrak{A})$ with τ as final state and $\mathrm{TP}(\mathfrak{A}_{\tau'})$ as $\mathrm{TP}(\mathfrak{A})$ with τ' as final state. Then $\alpha \in L(\mathrm{TP}(\mathfrak{A}_\tau)) \cdot_{\mathrm{TP}(\mathfrak{A}_{\tau'})} L(\mathrm{TP}(\mathfrak{A}_{\tau'}))^{\omega_{\mathrm{TP}(\mathfrak{A}_{\tau'})}}$.

With this we know that each $\alpha \in (M^n)^\omega$ can be decomposed into such a way. We also know that such a decomposition is either completely contained in L or in $(M^n)^\omega \backslash L$. Therefore we can define $\overline{L}$ as

$$\overline{L} = \{w \in (M^n)^\omega \mid w \in L(\mathrm{TP}(\mathfrak{A}')_\tau) \cdot_{\mathrm{TP}(\mathfrak{A}')_{\tau'}} L(\mathrm{TP}(\mathfrak{A}')_{\tau'})^{\omega_{\mathrm{TP}(\mathfrak{A}')_{\tau'}}}, \\ L(\mathrm{TP}(\mathfrak{A}')_\tau) \cdot_{\mathrm{TP}(\mathfrak{A}')_{\tau'}} L(\mathrm{TP}(\mathfrak{A}')_{\tau'})^{\omega_{\mathrm{TP}(\mathfrak{A}')_{\tau'}}} \cup L = \emptyset, \\ \text{with } \mathrm{TP}(\mathfrak{A}')_\tau, \text{ resp. } \mathrm{TP}(\mathfrak{A}')_{\tau'}, \text{ being the} \\ \text{Automaton } \mathrm{TP}(\mathfrak{A}') \text{ with } \tau, \text{ resp. } \tau', \text{ as final state and} \\ \mathfrak{A}' \text{ being the denormalized automaton for } \mathfrak{A}\}$$

And $\overline{L}$ can be recognized by an extended Strong Automaton with look-back s, since all partial automatons have look-back s and the other three points. Note however that the construction uses the decidability of the non-emptiness problem. While the proof appears to be a construction in general, we have only shown that such an automaton has to exist, not that it can be constructed this way for every language. □

Now, with the complementation covered, we can focus on the remaining problems.

Proposition 4.21. For the alphabet frame ($\mathcal{M}$,MSO-logic), where the MSO-theory of $\mathcal{M}$ is decidable, the non-universality and equivalence problem for extended Strong Büchi Automata are decidable for

- extended Strong Büchi Automata with look-back $s = 0$, if the transition logic of the structure is decidable,
- and extended Strong Büchi Automata with look-back 1 and a word dimension of 1, if the MSO-theory of the structure is decidable.

Proof. The proof is analogous to the proof for extended Strong Automata. However, since the complementation is not effective for the other cases, we are not able to reduce the equivalence and universality problem for those automata to the non-emptiness problem. □

5 Conclusion and Further Challenges

We have presented decidability results and covered the major closure properties of extended Strong Automata. There are still several points which could be refined however. Currently, we only considered the Büchi-acceptance condition when recognizing infinite words. When using a finite alphabet the languages recognized by that condition and the Muller-acceptance condition are the same, cf. [Tho90]. Therefore, we assume that modifying Muller Automata appropriately may yield a model which can recognize the same languages the extended Strong Büchi Automata recognize.

If they do, the complementation of extended Strong Büchi Automata may be reduced to the complementation of the modified Muller Automata model, which might yield an effective construction instead of just an existential confirmation. And with an effective method for complementation, we could immediately draw further conclusions about the universality, equality and inclusion problems of extended Strong Büchi Automata.

Furthermore, for finite alphabets, Nondeterministic Finite Automata and Regular Expressions denote the same set of languages, cf. [Kle56]. This inspires the search for a formal description of the languages which are recognized by the extended Strong Automaton model without using that automaton.

Bibliography

[Bès08] Alexis Bès. An application of the feferman-vaught theorem to automata and logics for words over an infinite alphabet. *Logical Methods in Computer Science*, 4(1), 2008.

[Bü62] J. R. Büchi. On a decision method in restricted second-order arithmetic. *Proc. International Congress on Logic, Method, and Philosophy of Science*, pages 1–12, 1962.

[Kle56] S.C. Kleene. Representation of events in nerve nets and finite automata. In *Automata studies*, pages 3–40. Princeton University Press, 1956.

[Min67] Marvin L. Minsky. *Computation: finite and infinite machines*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1967.

[Sem84] A.L. Semenov. Decidability of monadic theories. In Mathematical Foundations of Computer Science 1984, editor, *Chytil, M.P. and Koubek, V.*, volume 176 of *Lecture Notes in Computer Science*, pages 162–175. Springer Berlin Heidelberg, 1984.

[Spe13] Alexandra Spelten. *Paths in Infinite Trees: Logics and Automata*. PhD thesis, RWTH-Aachen University, 2013.

[Tho90] Wolfgang Thomas. Automata on infinite objects. In Handbook of theoretical computer science (vol. B), editor, *van Leeuwen, Jan*, pages 133–191. MIT Press, Cambridge, MA, USA, 1990.

[Tho97] Wolfgang Thomas. Languages, automata, and logic. In Handbook of Formal Languages, editor, *Rozenberg, Grzegorz and Salomaa, Arto*, pages 389–455. Springer Berlin Heidelberg, 1997.

[Wal96] Igor Walukiewicz. Monadic second order logic on tree-like structures. In STACS 96, editor, *Puech, Claude and Reischuk, Rüdiger*, volume 1046 of *Lecture Notes in Computer Science*, pages 399–413. Springer Berlin Heidelberg, 1996.