Relationship between Alternating ω-Automata and Symbolically Represented Nondeterministic ω-Automata

Thomas Tuerk and Klaus Schneider

Internal Report
No. 340/05

November 2005

Reactive Systems Group
Department of Computer Science, University of Kaiserslautern
P.O. Box 3049, 67653 Kaiserslautern, Germany
{tuerk, klaus.schneider}@informatik.uni-kl.de
http://rsg.informatik.uni-kl.de

Abstract

There is a well known relationship between alternating automata on finite words and symbolically represented nondeterministic automata on finite words. This relationship is of practical relevance because it allows to combine the advantages of alternating and symbolically represented nondeterministic automata on finite words. However, for infinite words the situation is unclear. Therefore, this work investigates the relationship between alternating ω-automata and symbolically represented nondeterministic ω-automata. Thereby, we identify classes of alternating ω-automata that are as expressive as $\mathsf{DET}_{\mathsf{G}}$, $\mathsf{DET}_{\mathsf{F}}$ and $\mathsf{DET}_{\mathsf{Prefix}}$ automata, respectively. Moreover, some very simple symbolic nondeterminisation procedures are developed for the classes corresponding to $\mathsf{DET}_{\mathsf{G}}$ and $\mathsf{DET}_{\mathsf{F}}$.

Contents

1 Motivation

Classically, finite state automata on finite [10] and infinite [6] words allow an existential choice. Those automata are called *nondeterministic*. In addition, *universal* automata that allow an universal choice and *deterministic* automata that do not allow any choice are used, too. A natural extension of these kinds of automata are *alternating* automata that allow both universal and existential choice [8].

Alternating automata are interesting, because they are exponentially more succinct than nondeterministic automata [8]. Moreover, in contrast to nondeterministic automata it is easy to complement alternating automata [14, 22].

For finite words, good data structures and algorithms for alternating automata are known. Therefore, alternating automata on finite words are used in practice. However, for infinite words, the situation is different: usually symbolic representations of nondeterministic automata are used.

Similar to alternating automata these symbolic representations are exponentially more succinct than nondeterministic automata. Moreover, there are efficient implementations using BDDs. However, these representations have the same limitations as nondeterministic automata. For example, it is not easily possible to complement a symbolically represented automaton. On the other hand, it is not easily possible to transfer the efficient implementations for nondeterministic automata to alternating automata.

Interestingly, alternating automata on finite words are strongly connected to symbolically represented nondeterministic automata on infinite words. In fact, one can consider alternating automata on finite words as a special normal form of symbolically represented nondeterministic automata on finite words. Thus, for finite words one can combine the benefits of alternating and symbolically represented nondeterministic automata.

For infinite words, the situation is unclear. Therefore, in this paper the relationship between symbolically represented nondeterministic and alternating automata on infinite words will be investigated. Especially translations of alternating automata to symbolically represented nondeterministic automata are interesting. To avoid errors, the most important lemmata are formally proved using the interactive theorem prover HOL. However, the HOL theories consider only alternating automata on infinite words, while this paper considers finite words, too. The HOL libraries containing these proofs can be found at `http://rsg.informatik.uni-kl.de/tools`.

There are many slightly different definitions of alternating automata and of symbolically represented nondeterministic automata. This is especially true in the case of infinite words. Moreover, many definitions which are used in the context of nondeterministic automata mean something slightly different in the context of alternating automata and vice versa. Therefore, we have to define alternating, nondeterministic and symbolically represented nondeterministic automata carefully before we can start. Then the situation for finite words is presented.

2 Basics

The very basic formalism we will use is propositional logic. Although it is very well standardised it is shortly defined in the following for reasons of completeness:

2.1 Propositional Logic

Definition 2.1 (Propositional Logic)
Let $\mathcal{V}$ be a set of variables. Then, the set of propositional formulas *over $\mathcal{V}$ (short $\mathcal{B}(\mathcal{V})$) is recursively given as follows:*

- *each variable $v \in \mathcal{V}$ is a propositional formula*
- $\neg\varphi \in \mathcal{B}(\mathcal{V})$*, if* $\varphi \in \mathcal{B}(\mathcal{V})$
- $\varphi \wedge \psi \in \mathcal{B}(\mathcal{V})$*, if* $\varphi, \psi \in \mathcal{B}(\mathcal{V})$

An assignment *s over $\mathcal{V}$ is a subset of $\mathcal{V}$. The semantics of a propositional formula with respect to an assignment s is given by the relation $\models_{\mathsf{prop}}$ that is defined as follows:*

- $s \models_{\mathsf{prop}} v$ *iff* $v \in s$
- $s \models_{\mathsf{prop}} \neg\varphi$ *iff* $s \not\models_{\mathsf{prop}} \varphi$
- $s \models_{\mathsf{prop}} \varphi \wedge \psi$ *iff* $s \models_{\mathsf{prop}} \varphi$ *and* $s \models_{\mathsf{prop}} \psi$

If $s \models_{\mathsf{prop}} \varphi$ holds, then the assignment s is said to fulfil *(or to* model*) the propositional formula φ. Therefore, s is called* a model *of φ. A model s of a formula φ is called a* minimal model *(short $s \models^{\mathsf{min}}_{\mathsf{prop}} \varphi$) iff all proper subsets of s do not model φ, i. e. $s \models^{\mathsf{min}}_{\mathsf{prop}} \varphi \Longleftrightarrow s \models_{\mathsf{prop}} \varphi \wedge \forall s'.\ s' \subset s \rightarrow s' \not\models_{\mathsf{prop}} \varphi$. Two propositional formulas φ_1 and φ_2 are called* equivalent *(short $\varphi_1 \equiv \varphi_2$) iff for all assignments s the relation $s \models \varphi_1 \Leftrightarrow s \models \varphi_2$ holds.*

For reasons of simplicity, the operator $\wedge$ is often omitted. For example, x_1x_2 means $x_1 \wedge x_2$. Additionally, further propositional operators like $\vee, \rightarrow, \leftrightarrow$ etc. are added as *syntactic sugar*, i. e. they are added as shorthands for formulas not containing these operators:

- $\varphi \vee \psi := \neg(\neg\varphi \wedge \neg\psi)$
- $\varphi \rightarrow \psi := \neg\varphi \vee \psi$
- $\varphi \leftrightarrow \psi := \varphi \rightarrow \psi \wedge \psi \rightarrow \varphi$
- $\mathsf{true} := v \vee \neg v$ for an arbitrary variable $v \in \mathcal{V}$
- $\mathsf{false} := \neg\mathsf{true}$

Definition 2.2 (Positive propositional formulas)
A propositional formula $\varphi \in \mathcal{B}(\mathcal{V})$ is called positive, iff for all $s \subseteq \mathcal{V}$ the following equivalence holds:

$$s \models_{\mathsf{prop}} \varphi \Longleftrightarrow \forall s'.\, s \subseteq s' \subseteq \mathcal{V} \rightarrow s' \models_{\mathsf{prop}} \varphi$$

The set of all positive propositional formulas is denoted over $\mathcal{V}$ is denoted by $\mathcal{B}^+(\mathcal{V})$.

The last definition defines positive propositional formulas by their semantics. Often they are defined by the syntax. Obviously, for every positive propositional formula $\varphi \in \mathcal{B}^+(\mathcal{V})$ an equivalent formula φ' of the form

$$\varphi' = (\varphi_{1,1} \wedge \varphi_{1,2} \wedge \ldots \wedge \varphi_{1,n_1}) \vee \ldots \vee (\varphi_{m,1} \wedge \ldots \wedge \varphi_{m,n_m})$$

with $\varphi_{i,j} \in \mathcal{V}$ exists. On the other hand, every formula of this normal form is a positive propositional formula. Therefore, this normal form can be used to define positive propositional formulas syntactically.

2.2 Finite State Automata on Finite Words

In this section, we introduce alternating automata on finite words. Therefore, words, semiautomata, runs and paths are introduced first. Then these definitions are used to define automata on finite words. In the following section these concepts are used to define alternating automata on infinite words.

Definition 2.3 (Words)
A finite word v over a set Σ of length $|v| = n + 1$ is a function $v : \{0, \ldots n\} \rightarrow \Sigma$. An infinite word v over Σ is a function $v : \mathbb{N} \rightarrow \Sigma$. Its length is denoted by $|v| = \infty$. The set Σ is called alphabet. *The elements of Σ are called* letters. *The finite word of length 0 is called the* empty word *(denoted by ε). For reasons of simplicity, $v(i)$ is often denoted by v^i for $i \in \mathbb{N}$. Using this notation, words are often given in the form $v^0v^1v^2 \ldots v^n$ or $v^0v^1 \ldots$. The set of all finite words over Σ is denoted by Σ^*, and the set of all infinite words over Σ is denoted by Σ^ω.*

Counting of letters starts with zero, i. e. v^{i-1} refers to the i-th letter of v. Furthermore, $v^{i..}$ denotes the suffix of v starting at position i, i. e. $v^{i..} = v^iv^{i+1} \ldots$ for all $i < |v|$. The finite word $v^iv^{i+1} \ldots v^j$ is denoted by $v^{i..j}$. Notice that in case $j < i$ the expression $v^{i..j}$ evaluates to the empty word ε. For two words v_1, v_2, we use v_1v_2 for the concatenation of v_1 and v_2. Finally, we use l^ω for the infinite word v with $v^j = l$ for all j.

Definition 2.4 (Semiautomata)
A semiautomaton *$\mathfrak{A} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R})$ is a tuple where $\mathcal{Q}$ is the finite set of* states, *Σ a finite alphabet, $\mathcal{I} \in \mathcal{B}^+(\mathcal{Q})$ is the* initial condition *and $\mathcal{R} : \mathcal{Q} \times \Sigma \rightarrow \mathcal{B}^+(\mathcal{Q})$ is the* transition function *of $\mathfrak{A}$.*

Definition 2.5 ((Minimal) Run of a Word)
Given a semiautomaton $\mathfrak{A} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R})$ and a finite or infinite word α over Σ. Then, each directed graph $G = (V, E)$ with

- $V \subseteq \mathcal{Q} \times \{i \mid i \in \mathbb{N} \wedge i \leq |\alpha|\}$
- $E \subseteq \bigcup_{l>0} (\mathcal{Q} \times \{l\}) \times (\mathcal{Q} \times \{l+1\})$
- *for every* $(q, l+1) \in V$ *exists a* $q' \in \mathcal{Q}$*, such that* $\big((q', l), (q, l+1)\big) \in E$*.*
- $\{q \mid (q, 0) \in V\}$ *is a (minimal) model of* $\mathcal{I}$
- *for all* $(q, l) \in V$ *with* $l < |\alpha|$ *the set* $\{q' \mid \big((q, l), (q', l+1)\big) \in E\}$ *is a (minimal) model of* $\mathcal{R}(q, \alpha^l)$

is called a (minimal) run of α through $\mathfrak{A}$. *The set of all runs of a word α through a semiautomaton $\mathfrak{A}$ is denoted by $\mathsf{RUN}_{\mathfrak{A}}(\alpha)$, the set of all minimal runs is denoted by $\mathsf{RUN}^{\mathsf{MIN}}_{\mathfrak{A}}(\alpha)$. Obviously, every minimal run is a run.*

Definition 2.6 (Path trough a Run)
Let $r = (V, E)$ be a run through a semiautomaton $\mathfrak{A} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R})$. If $V = \emptyset$ holds, then ε is the unique path *through r. Otherwise, each finite or infinite word p over $\mathcal{Q}$ with*

- $|p| > 0$
- $(p^0, 0) \in V$
- $\big((p^i, i), (p^{i+1}, i+1)\big) \in E$ *for all* $i < |p| - 1$
- $\forall q'.\ \big((p^{|p|-1}, |p| - 1), (q', |p|)\big) \notin E$ *if p is finite*

is called a path through r*.*

Definition 2.7 (Subrun)
A run r is called a subrun of a run r' iff every path through r is also a path through r'.

Definition 2.8 (Finite Automata on Finite Words)
A finite automaton on finite words is a tuple $\mathfrak{A} = (\mathfrak{B}, \mathcal{F})$ where $\mathfrak{B} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R})$ is a semiautomaton and $\mathcal{F} \subseteq \mathcal{Q}$ a set of final states. $\mathfrak{A}$ accepts a finite word $\alpha \in \Sigma^*$ *(denoted by $\alpha \models \mathfrak{A}$) iff a run $\beta \in \mathsf{RUN}_{\mathfrak{A}}(\alpha)$ exists such that every path γ of length $|\alpha| + 1$ through β ends in a final state, i. e. $\gamma^{|\alpha|} \in \mathcal{F}$ holds. The set of all words that are accepted by $\mathfrak{A}$ – i. e. the language recognised by $\mathfrak{A}$ – is denoted by $\mathcal{L}(\mathfrak{A})$. A finite automaton over finite words $\mathfrak{A}$ is often denoted by $(\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R}, \mathcal{F})$.*

Lemma 2.9 *For every run r of a word α through a semi-automaton $\mathfrak{A}$, a minimal run r' of α through $\mathfrak{A}$ exists, such that r' is a subrun of r*[1].

The semantics of finite automata are defined with respect to arbitrary runs. However, it is sufficient to consider only minimal runs.

Lemma 2.10 *Let $\mathfrak{A} = (\mathfrak{B}, \mathcal{F})$ be a finite automaton with $\mathfrak{B} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R})$. Then $\mathfrak{A}$* accepts a finite word $\alpha \in \Sigma^*$ *iff* a minimal *run $\beta \in \mathsf{RUN}_{\mathfrak{A}}(\alpha)$ exists such that every path γ of length $|\alpha| + 1$ through β ends in a final state.*

Proof In case such a minimal run β exists, obviously $\alpha \models \mathfrak{A}$ holds, because β is a run. On the other hand, assume $\alpha \models \mathfrak{A}$. Then, a run β of α through $\mathfrak{B}$ exists such that every path of length $|\alpha| + 1$ through β ends in a final state. According to Lemma 2.9 a subrun β' of β exists such that β' is a minimal run of α through $\mathfrak{B}$. Every path through this minimal run β' of length $|\alpha| + 1$ is also a path through β. Thus, it ends in a final state. □

Example 2.11 (Finite Alternating Automata on Finite Words) *Consider the finite automaton on finite words $\mathfrak{A} = (\mathfrak{B}, \mathcal{F})$ with $\mathfrak{B} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R})$, $\Sigma = \{a, b\}$, $\mathcal{Q} = \{q_0, q_1, q_2\}$, $\mathcal{I} = q_0 \vee q_2$, $\mathcal{F} = \{q_1\}$ and $\mathcal{R}$ given by*

	a	b
q_0	$q_0 \vee q_1 \vee q_2$	$q_1 q_2$
q_1	false	$q_0 q_2$
q_2	$q_0 \vee q_1$	true

Then runs r_1, r_2 and r_3 shown in Figure 1 are runs of the word bbaa through $\mathfrak{B}$. The runs r_1 and r_3 are minimal runs, r_2 is not minimal. Moreover, r_1 is a proper subrun of r_2. There are 3 paths through r_1: $q_0 q_2$, $q_0 q_1 q_2 q_0 q_1$ and $q_0 q_1 q_0 q_0 q_1$. As all paths through r_1 of length 5 end in q_1, the input bbaa is accepted by $\mathfrak{A}$. However, there are paths of length 5 through r_3 that do not end in q_1. For example, $q_0 q_1 q_2 q_0 q_0$ is such a path. Notice that for inputs that start with ba there are no runs through $\mathfrak{B}$.

Definition 2.12 (Classes of automata)
A general semiautomaton $\mathfrak{A} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R})$ as defined above is called alternating. *For some input an alternating semiautomaton may have arbitrary many runs and every run may contain arbitrary many paths. According to Lemma 2.10 the semantics of finite automata check that there exists a minimal run such that for all paths through this run some property holds. In this sense, alternating semiautomata provide an existential and an universal choice.*

Iff $\mathcal{I}$ and *$\mathcal{R}(q, i)$ are of the form $q_1 \vee \ldots \vee q_n$ the semiautomaton $\mathfrak{A}$ is called* nondeterministic. *Notice, that the case $n = 0$, i. e.* false *is allowed. For some input a nondeterministic*

[1]theorem `ALTERNATING_RUN___ALTERNATING_MIN_RUN_EXISTS` in theory `Alternating_Omega_Automata`

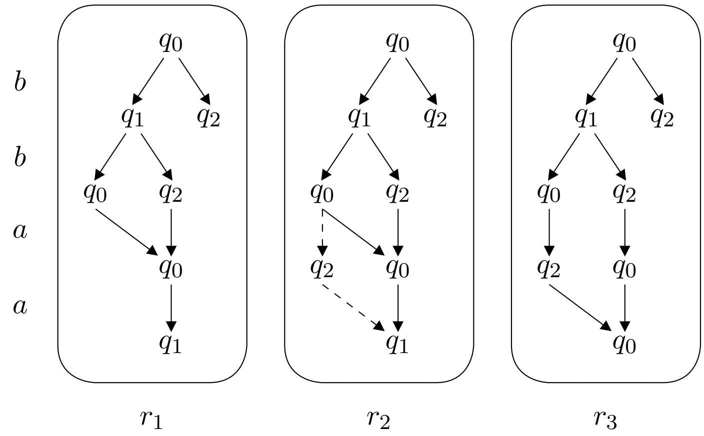

Figure 1: Example 2.11

semiautomaton may have arbitrary many minimal runs, but every minimal run contains exactly one path. Therefore, nondeterministic semiautomata provide only existential choice.

On the other hand, $\mathfrak{A}$ *is called* universal *iff* $\mathcal{I}$ *and* $\mathcal{R}(q,i)$ *are of the form* $q_1 \wedge \ldots \wedge q_n$*. Notice, that the case* $n = 0$*, i.e.* true *is allowed. For any input, an universal semiautomaton has exactly one minimal run, but this run may contain arbitrary many paths. Thus, universal semiautomata provide universal choice.*

Iff $\mathcal{I} \not\equiv \mathsf{false}$ *and* $\mathcal{R}(q,i) \not\equiv \mathsf{false}$ *hold for all* $q \in \mathcal{Q}$ *and all* $i \in \Sigma$*,* $\mathfrak{A}$ *is called* existentially total. *Iff* $\mathcal{I} \not\equiv \mathsf{true}$ *and* $\mathcal{R}(q,i) \not\equiv \mathsf{true}$ *hold for all* $q \in \mathcal{Q}$ *and all* $i \in \Sigma$*,* $\mathfrak{A}$ *is called* universally total. *If* $\mathfrak{A}$ *is existentially and universally total it is called* total. *For any input, an existentially total automaton has at least one run and therefore at least one minimal run. Every path through a run of* α *through an universally total automaton is of length* $|\alpha|+1$ *if* α *is finite or of infinite length if* α *is infinite. Notice, that nondeterministic automata are universally total while universal automata are existentially total.*

Iff $\mathfrak{A}$ *is nondeterministic and universal, it is called* deterministic. *For every input a deterministic semiautomaton has exactly one minimal run and this run contains exactly one path. Therefore, deterministic semiautomata provide no choice. For all* $q \in \mathcal{Q}$ *and all* $i \in \Sigma$ *the transition function* $\mathcal{R}(q,i)$ *of an deterministic semiautomaton is of the form* q*. The initial condition* $\mathcal{I}$ *is of the same form. Thus, following this definition all deterministic semiautomata are total. (Notice, that this is a difference to some definitions in literature [19]. There, nondeterministic semiautomata, whose transition function* $\mathcal{R}(q,i)$ *is for all* $q \in \mathcal{Q}$ *and all* $i \in \Sigma$ *of the form* q *or* false *are called deterministic. However, in most cases only total, deterministic automata are considered, which are deterministic automata in the sense of the definitions above.)*

An automaton is called alternating, nondeterministic, universal, total or deterministic iff the corresponding semiautomaton is alternating, nondeterministic, universal, total or deterministic, respectively.

2.3 ω-Automata

ω-automata are finite automata on infinite words. Similarly to the case of finite words, a set of accepting states is often used to define the acceptance condition. However, in the case of infinite words, there are several reasonable definitions of acceptance conditions. For example, an infinite word α could be accepted by an automaton $\mathfrak{A}$ iff there exists a run of α through $\mathfrak{A}$ that

- never leaves the set of accepting states
- visits the set of accepting states at least once
- from some point of time never leaves the set of accepting states
- visits the set of accepting states infinitely often.

The last acceptance condition is the one used by Büchi. Therefore, the resulting ω-automata are called Büchi automata. However, the other acceptance conditions and a lot of similar ones are used in practice, too. They lead to different classes of ω-automata with different expressive power.

Definition 2.13 (Finite Automata on Infinite Words)
A finite automaton on infinite words is a tuple $\mathfrak{A} = (\mathfrak{B}, \mathcal{AC})$ *where* $\mathfrak{B} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R})$ *is a semiautomaton and* $\mathcal{AC}$ *an acceptance component. There are different kinds of acceptance components, which lead to automata with different expressive power.* $\mathfrak{A}$ accepts an infinite word $\alpha \in \Sigma^\omega$ *(short* $\alpha \models \mathfrak{A}$*) iff a run* $\beta \in \mathsf{RUN}_{\mathfrak{A}}(\alpha)$ *exists such that every infinite path* γ *through* β *is accepted by the acceptance component (short* $\gamma \models_{\mathsf{AC}} \mathcal{AC}$*). The set of all words that are accepted by* $\mathfrak{A}$ *– i. e. the language recognised by* $\mathfrak{A}$ *– is denoted by* $\mathcal{L}(\mathfrak{A})$.

Similar to the finite case, it is sufficient to consider only minimal runs:

Lemma 2.14 *Let* $\mathfrak{A} = (\mathfrak{B}, \mathcal{AC})$ *be an alternating automaton on infinite words with* $\mathfrak{B} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R})$*. Then* $\mathfrak{A}$ accepts a infinite word $\alpha \in \Sigma^\omega$ *iff* a minimal *run* $\beta \in \mathsf{RUN}_{\mathfrak{A}}(\alpha)$ *exists such that every infinite path* γ *through* β *is accepted by* $\mathcal{AC}$[2].

Proof In case such a minimal run β exists, obviously $\alpha \models \mathfrak{A}$ holds, because β is a run. On the other hand, assume $\alpha \models \mathfrak{A}$ hold. Then a run β of α through $\mathfrak{B}$ exists such that every infinite path through β is accepted by $\mathcal{AC}$. According to Lemma 2.9 a subrun β' of β exists such that β' is a minimal run of α through $\mathfrak{B}$. Every path p through this minimal run β' is also a path through β. Thus, $p \models_{\mathsf{AC}} \mathcal{AC}$ holds. $\square$

[2]theorem `ALT_SEM___ALT_SEM_MIN___EQUIV` in theory `Alternating_Omega_Automata`

Definition 2.15 (Acceptance Components)
There are a lot different acceptance components. Therefore, a logic for acceptance components is defined in the following that is able to express the most common acceptance components:

Let $\mathcal{Q}$ be a set of states. Then, the set $\mathsf{ac}_{\mathcal{Q}}$ of all acceptance components over $\mathcal{Q}$ is given by:

- $\mathcal{F} \in \mathsf{ac}_{\mathcal{Q}}$ *if* $\mathcal{F} \subseteq \mathcal{Q}$
- $\mathsf{B}(\mathcal{F}) \in \mathsf{ac}_{\mathcal{Q}}$ *if* $\mathcal{F} \subseteq \mathcal{Q}$
- $\mathsf{WB}(\mathcal{F}) \in \mathsf{ac}_{\mathcal{Q}}$ *if* $\mathcal{F} \subseteq \mathcal{Q}$
- $\mathsf{P}(p) \in \mathsf{ac}_{\mathcal{Q}}$ *if* $p : \mathcal{Q} \to \mathbb{N}$
- $\mathsf{WP}(p) \in \mathsf{ac}_{\mathcal{Q}}$ *if* $p : \mathcal{Q} \to \mathbb{N}$
- $\neg a \in \mathsf{ac}_{\mathcal{Q}}$ *if* $a \in \mathsf{ac}_{\mathcal{Q}}$
- $a_1 \wedge a_2 \in \mathsf{ac}_{\mathcal{Q}}$ *if* $a_1, a_2 \in \mathsf{ac}_{\mathcal{Q}}$
- $\mathsf{G}\,\mathsf{a} \in \mathsf{ac}_{\mathcal{Q}}$ *if* $a \in \mathsf{ac}_{\mathcal{Q}}$

To define the semantics of acceptance components, two definitions are needed: For some path γ over an alphabet $\mathcal{V}$ let occ(γ) denote the set $\{\gamma^i \mid i \in \mathbb{N}\}$, i. e. the set of all states that occur on γ. Further, let inf(γ) denote the set of all states that occur infinitely often in γ, i. e. the set $\{s \mid \{i \mid \gamma^i = s\}$ is infinite$\}$. Using these definitions, the semantics of acceptance components are for a path γ given by:

- $\gamma \models_{\mathsf{AC}} \mathcal{F}$ *iff* $\gamma^0 \in \mathcal{F}$
- $\gamma \models_{\mathsf{AC}} \mathsf{B}(\mathcal{F})$ *iff* $\mathit{inf}(\gamma) \cap \mathcal{F} \neq \emptyset$
- $\gamma \models_{\mathsf{AC}} \mathsf{WB}(\mathcal{F})$ *iff* $\mathit{occ}(\gamma) \cap \mathcal{F} \neq \emptyset$
- $\gamma \models_{\mathsf{AC}} \mathsf{P}(p)$ *iff* $\min\{p(q) \mid q \in \mathit{inf}(\gamma)\}$ *is even*
- $\gamma \models_{\mathsf{AC}} \mathsf{WP}(p)$ *iff* $\min\{p(q) \mid q \in \mathit{occ}(\gamma)\}$ *is even*
- $\gamma \models_{\mathsf{AC}} \neg a$ *iff not* $\gamma \models_{\mathsf{AC}} a$
- $\gamma \models_{\mathsf{AC}} a_1 \wedge a_2$ *iff* $\gamma \models_{\mathsf{AC}} a_1$ and $\gamma \models_{\mathsf{AC}} a_2$
- $\gamma \models_{\mathsf{AC}} \mathsf{G}\,\mathsf{a}$ *iff* $\gamma^{i..} \models_{\mathsf{AC}} a$ *holds for all* $i \in \mathbb{N}$

Two acceptance components $a_1, a_2 \in \mathsf{ac}_{\mathcal{Q}}$ are said to be equivalent (short $a_1 \equiv a_2$) iff for all paths γ over $\mathcal{Q}$ the proposition $\gamma \models_{\mathsf{AC}} a_1 \Leftrightarrow \gamma \models_{\mathsf{AC}} a_2$ holds.

Additionally, some syntactic sugar is used:

- $\mathsf{false} := \emptyset$

- $\mathsf{true} := \neg\mathsf{false}$
- $a_1 \vee a_2 := \neg((\neg a_1) \wedge (\neg a_2))$
- $\mathsf{F}\, a := \neg \mathsf{G} \neg a$
- $\mathsf{co} - \mathsf{B}(\mathcal{F}) := \neg \mathsf{B}(\mathcal{F})$
- $\mathsf{co} - \mathsf{WB}(\mathcal{F}) := \neg \mathsf{WB}(\mathcal{F})$

There are two groups of acceptance components. The acceptance components $\mathsf{B}(\mathcal{F})$ (Büchi condition), $\mathsf{co} - \mathsf{B}(\mathcal{F})$ (co-Büchi condition), $\mathsf{WB}(\mathcal{F})$ (weak Büchi condition), $\mathsf{co} - \mathsf{WB}(\mathcal{F})$ (weak co-Büchi condition), $\mathsf{P}(p)$ (parity condition) and $\mathsf{WP}(p)$ (weak parity condition) are usually used in the context of alternating automata. When talking about symbolic representations of nondeterministic ω-automata, the remaining acceptance conditions are more common. Notice, that some of the defined acceptance conditions are just introduced to pay respect to these two worlds. For example, the following equivalences hold:

- $\neg\neg a \equiv_{\mathcal{Q}} a$
- $\neg \mathcal{F} \equiv_{\mathcal{Q}} \mathcal{Q} \setminus \mathcal{F}$
- $\mathsf{B}(\mathcal{F}) \equiv_{\mathcal{Q}} \mathsf{GF}\, \mathcal{F}$
- $\mathsf{WB}(\mathcal{F}) \equiv_{\mathcal{Q}} \mathsf{F}\, \mathcal{F}$
- $\mathsf{co} - \mathsf{B}(\mathcal{F}) \equiv_{\mathcal{Q}} \mathsf{FG} \neg \mathcal{F}$
- $\mathsf{FG}\, \mathcal{F} \equiv_{\mathcal{Q}} \mathsf{co} - \mathsf{B}(\neg \mathcal{F})$
- $\mathsf{co} - \mathsf{WB}(\mathcal{F}) \equiv_{\mathcal{Q}} \mathsf{G} \neg \mathcal{F}$
- $\mathsf{G}\, \mathcal{F} \equiv_{\mathcal{Q}} \mathsf{co} - \mathsf{WB}(\neg \mathcal{F})$

2.4 Classes of ω-Automata

Definition 2.16 (Classes of Acceptance Components)
Let Φ_i, Ψ_i be subsets of $\mathcal{Q}$ and $p : \mathcal{Q} \to \mathbb{N}$. Then, the following classes of acceptance components over $\mathcal{Q}$ are defined [13]:

True condition:	true
False condition:	false
Initial condition:	Φ_0
Safety / weak co-Büchi condition:	$\mathsf{G}\Phi_0$
Liveness / weak Büchi condition:	$\mathsf{F}\Phi_0$
Büchi condition [6, 7]:	$\mathsf{GF}\Phi_0$
Persistence / co-Büchi condition [13]:	$\mathsf{FG}\Phi_0$
Rabin condition [16]:	$\bigvee_{j=0}^{f} \left(\mathsf{GF}\Phi_0 \ \wedge\ \mathsf{FG}\Psi_0\right)$
Streett condition [20]:	$\bigwedge_{j=0}^{f} \left(\mathsf{FG}\Phi_0 \ \vee\ \mathsf{GF}\Psi_0\right)$
Prefix condition (1. kind) [13, 17]:	$\bigwedge_{j=0}^{f} \left(\mathsf{G}\Phi_0 \ \vee\ \mathsf{F}\Psi_0\right)$
Prefix condition (2. kind) [13, 17]:	$\bigvee_{j=0}^{f} \left(\mathsf{F}\Phi_0 \ \vee\ \mathsf{G}\Psi_0\right)$
Parity condition:	$\mathsf{P}(\mathsf{p})$
Weak parity condition:	$\mathsf{WP}(\mathsf{p})$

Definition 2.17 (Classes of ω-Automata)
*The classes of ω-automata are defined by the transition relation of the corresponding semiautomaton, i. e. alternating, nondeterministic, etc., and by the class of the acceptance component. They are denoted by expressions of the form X_Y^Z. X denotes the class of transition relation: alternating (*A*), nondeterministic (*NDET*), universal (*U*) or deterministic (*DET*). Y is used to denote the class of the acceptance component: true (*True*), false (*False*), initial (*Initial*), safety (*G*), liveness (*F*), Büchi (*GF*), persistence (*FG*), Rabin (*Rabin*), Streett (*Streett*), prefix (1st kind) (*Prefix1*), prefix (2nd kind) (*Prefix2*), prefix (1st and 2nd kind) (*Prefix*), parity (*P*) or weak parity (*WP*). Finally, Z is used to denote special restrictions of the class. For example, $\mathsf{U}_{\mathsf{GF}}^{\mathsf{total}}$ denotes the class of total, universal, Büchi-automata.*

These classes form a hierarchy in terms of expressiveness. To explain this hierarchy some definitions are needed before. For a class $\mathcal{C}$ let $\mathcal{L}_\mathcal{C}$ denote the set of all languages that are recognisable by automata in $\mathcal{C}$, i. e. $\mathcal{L}_\mathcal{C} := \{\mathcal{L}(\mathfrak{A}) \mid \mathfrak{A} \in \mathcal{C}\}$. A class $\mathcal{C}_1$ is strictly more expressive than a class $\mathcal{C}_2$ (denoted by $\mathcal{C}_2 \precnapprox \mathcal{C}_1$) iff $\mathcal{L}_{\mathcal{C}_2} \subset \mathcal{L}_{\mathcal{C}_1}$ holds. Further, $\mathcal{C}_1$ is as expressive as $\mathcal{C}_2$ (denoted by $\mathcal{C}_1 \approx \mathcal{C}_2$) iff $\mathcal{L}_{\mathcal{C}_1} = \mathcal{L}_{\mathcal{C}_2}$ holds. $\mathcal{C}_1$ and $\mathcal{C}_2$ are dual (denoted by $\mathcal{C}_1 \parallel \mathcal{C}_2$) iff $\mathcal{L} \in \mathcal{L}_{\mathcal{C}_1} \Leftrightarrow (\Sigma^\omega \setminus \mathcal{L}) \in \mathcal{L}_{\mathcal{C}_2}$ holds for all input sets Σ and all languages $\mathcal{L} \subseteq \Sigma^\omega$. Notice, that $\mathcal{C}_1 \parallel \mathcal{C}_2$ and $\mathcal{C}_2 \parallel \mathcal{C}_3$ imply $\mathcal{C}_1 \approx \mathcal{C}_3$. Using these notations an important part of the hierarchy of ω-automata is shown in Figure 2 [12, 19].

2.5 Boolean Operations on Alternating Automata

One advantage of alternating automata is that boolean operations are very easy to perform. For nondeterministic and universal automata it is also easy to compute dis-

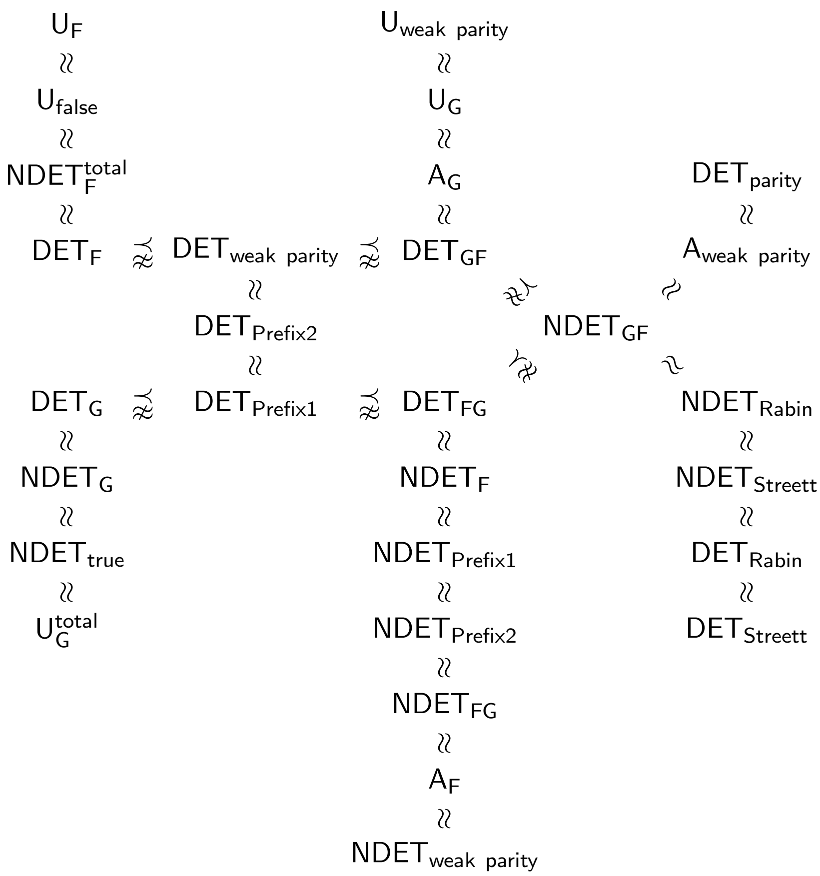

Figure 2: Hierarchy of ω-Automata [12, 19]

junctions and conjunctions of automata. However, the negation of nondeterministic and universal automata is difficult in general. Although it is simple to negate alternating automata, some definitions are needed to present the negation:

$$\begin{cases} \text{negVars}(v) & := & \neg v \\ \text{negVars}(\neg\varphi) & := & \neg\text{negVars}(\varphi) \\ \text{negVars}(\varphi_1 \wedge \varphi_2) & := & \text{negVars}(\varphi_1) \wedge \text{negVars}(\varphi_2) \end{cases}$$

$$\begin{array}{rcl} \widetilde{\varphi} & := & \neg\text{negVars}(\varphi) \\ \widetilde{\mathcal{R}}(q,s) & := & \widetilde{\mathcal{R}(q,s)} \\ \widetilde{p}(q) & := & p(q)+1 \end{array}$$

Using these definitions one can state the following lemmata [15]:

Lemma 2.18 *Let $\mathfrak{A} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R}, \mathcal{F})$ be an alternating automaton on finite words that accepts a language $\mathcal{L}$. Then $\widetilde{\mathfrak{A}} := (\Sigma, \mathcal{Q}, \widetilde{\mathcal{I}}, \widetilde{\mathcal{R}}, \mathcal{Q} \setminus \mathcal{F})$ accepts $\Sigma^* \setminus \mathcal{L}$.*

Lemma 2.19 *Let $\mathfrak{A} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R}, \mathsf{P}(\mathsf{p}))$ be an alternating automaton that accepts a language $\mathcal{L}$. Then $\widetilde{\mathfrak{A}} := (\Sigma, \mathcal{Q}, \widetilde{\mathcal{I}}, \widetilde{\mathcal{R}}, \mathsf{P}(\widetilde{p}))$ accepts $\Sigma^\omega \setminus \mathcal{L}$.*

$\widetilde{\mathfrak{A}}$ is called the *dual of* $\mathfrak{A}$. Therefore, the construction described in Lemma 2.18 and Lemma 2.19 is called *dualisation*. The proof of these lemmata is quite complicated and needs some concepts such as logical games and strategies. Therefore, it is omitted here.

Disjunction and conjunctions of alternating automata are also very easy to compute. Essentially the initial conditions have to be combined:

Lemma 2.20 *Let $\mathfrak{A}_1 = (\Sigma, \mathcal{Q}_1, \mathcal{I}_1, \mathcal{R}_1, \mathcal{F}_1)$ and $\mathfrak{A}_2 = (\Sigma, \mathcal{Q}_2, \mathcal{I}_2, \mathcal{R}_2, \mathcal{F}_2)$ be two alternating automata on finite words with $\mathcal{Q}_1 \cap \mathcal{Q}_2 = \emptyset$. Further, let $\mathfrak{A}$ for $\odot \in \{\vee, \wedge\}$ be defined by*

$$\mathfrak{A} := (\Sigma, \mathcal{Q}_1 \cup \mathcal{Q}_\in, \mathcal{I}_1 \odot \mathcal{I}_2, \mathcal{R}, \mathcal{F}_1 \cup \mathcal{F}_2)$$

with $\mathcal{R}(q,i) := \mathcal{R}_1(q,i)$ if $q \in \mathcal{Q}_1$ and $\mathcal{R}(q,i) := \mathcal{R}_2(q,i)$ otherwise. Then for all inputs $w \in \Sigma^*$ the relation $w \models \mathfrak{A} \Leftrightarrow \big(w \models \mathfrak{A}_1 \;\odot\; w \models \mathfrak{A}_2\big)$ holds.

Lemma 2.21 *Let $\mathfrak{A}_1 = (\Sigma, \mathcal{Q}_1, \mathcal{I}_1, \mathcal{R}_1, \mathcal{AC}_1)$ and $\mathfrak{A}_2 = (\Sigma, \mathcal{Q}_2, \mathcal{I}_2, \mathcal{R}_2, \mathcal{AC}_2)$ be two alternating automata on infinite words with $\mathcal{Q}_1 \cap \mathcal{Q}_2 = \emptyset$. Further, let $\mathfrak{A}$ for $\odot \in \{\vee, \wedge\}$ be defined by*

$$\mathfrak{A} := (\Sigma, \mathcal{Q}_1 \cup \mathcal{Q}_\in, \mathcal{I}_1 \odot \mathcal{I}_2, \mathcal{R}, \mathcal{AC})$$

with the transition relation

$$\mathcal{R}(q,i) := \begin{cases} \mathcal{R}_1(q,i) & \textit{if} q \in \mathcal{Q}_1 \\ \mathcal{R}_2(q,i) & \textit{otherwise} \end{cases}$$

and some acceptance condition $\mathcal{AC}$ such that

$$p \in \mathcal{Q}_1^\omega \implies p \models_{\mathsf{AC}} \mathcal{AC} \Leftrightarrow p \models_{\mathsf{AC}} \mathcal{AC}_1$$
$$p \in \mathcal{Q}_2^\omega \implies p \models_{\mathsf{AC}} \mathcal{AC} \Leftrightarrow p \models_{\mathsf{AC}} \mathcal{AC}_2$$

holds. For

$$(p_1 \cup p_2)(q) := \begin{cases} p_1(q) & \text{if } q \in \mathcal{Q}_1 \\ p_2(q) & \text{otherwise} \end{cases},$$

some possible constructions of such an combined acceptance condition are shown in the following table:

$\mathcal{AC}_1$	$\mathcal{AC}_2$	$\mathcal{AC}$
true	true	true
false	false	false
Φ_1	Φ_2	$\Phi_1 \cup \Phi_2$
$\mathsf{G}\Phi_1$	$\mathsf{G}\Phi_2$	$\mathsf{G}(\Phi_1 \cup \Phi_2)$
$\mathsf{F}\Phi_1$	$\mathsf{F}\Phi_2$	$\mathsf{F}(\Phi_1 \cup \Phi_2)$
$\mathsf{GF}\Phi_1$	$\mathsf{GF}\Phi_2$	$\mathsf{GF}(\Phi_1 \cup \Phi_2)$
$\mathsf{FG}\Phi_1$	$\mathsf{FG}\Phi_2$	$\mathsf{FG}(\Phi_1 \cup \Phi_2)$
$\mathsf{P}(\mathsf{p}_1)$	$\mathsf{P}(\mathsf{p}_2)$	$\mathsf{P}(\mathsf{p}_1 \cup \mathsf{p}_2)$
$\mathsf{WP}(\mathsf{p}_1)$	$\mathsf{WP}(\mathsf{p}_2)$	$\mathsf{WP}(\mathsf{p}_1 \cup \mathsf{p}_2)$

Then for all inputs $w \in \Sigma^\omega$, the relation $w \models \mathfrak{A} \Leftrightarrow \big(w \models \mathfrak{A}_1 \;\odot\; w \models \mathfrak{A}_2\big)$ holds[3].

2.6 Symbolic Representation

Although good model checking procedures for CTL were known [9], first implementations of these procedures were not able to verify large systems, because no efficient data structures were used. Verification tools were only able to handle systems with a thousand states. A breakthrough was achieved by representing the systems with Boolean functions, which are stored as *binary decision diagrams (BDDs)* [2]. The resulting *symbolic model checking* procedures [1, 3, 4, 5] allow checking of systems with more than 10^{20} states. In general, the symbolic representation of an nondeterministic automaton is exponentially more succinct than the corresponding automaton.

To explain the symbolic representation of nondeterministic automata, nondeterministic semiautomata are considered first: Let $\mathfrak{A} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R})$ be a nondeterministic semiautomaton. As the set Σ and the set of states $\mathcal{Q}$ are finite, they can be encoded by a finite set of propositional variables. So, let $\Sigma = \mathcal{P}(\mathcal{V}_\Sigma)$ with $\mathcal{V}_\Sigma = \{i_0, \ldots, i_n\}$ and $\mathcal{Q} = \mathcal{P}(\mathcal{V}_\mathcal{Q})$ with $\mathcal{V}_\mathcal{Q} = \{q_0, \ldots, q_m\}$ hold. With these settings, a state of $\mathfrak{A}$ is a subset of $\mathcal{V}_\mathcal{Q}$, and a letter of the input alphabet Σ is a subset of $\mathcal{V}_\Sigma$. Those subsets of a set of propositional variables can be interpreted as assignments. Therefore, it is possible to encode a set of those subsets S by a propositional formula Φ_S that has the following property: $s \in S \Leftrightarrow s \models_{\mathsf{prop}} \Phi_S$ (see Example 2.24).

[3] theorem `ALTERNATING_AUTOMATA_CONJUNCTION` and theorem `ALTERNATING_AUTOMATA_DISJUNCTION` in theory `Alternating_Omega_Automata`

Because we will translate between symbolic and explicit representations of automata, some denotations are introduced: For some finite set S let $\mathcal{V}_S$ denote a sufficient large set of variables to encode S. Further let $\mathsf{enc}_S : S \to \mathcal{P}(\mathcal{V}_S)$ denote such an encoding. The definition of enc_S is extended to words by the following definition: for all words w over S let $\mathsf{enc}_S(w)$ denote the word v with $|v| = |w|$ and $v^i = \mathsf{enc}_S(w^i)$ for all $i < |w|$. Additionally, let $\mathsf{Enc}_S : \mathcal{P}(S) \to \mathcal{B}(\mathcal{V}_S)$ denote an encoding of subsets of S by propositional formulas, i.e. for all $T \subseteq S$ and all $m \subseteq \mathcal{V}_S$ the following holds: $m \models_{\mathsf{prop}} \mathsf{Enc}_S(T) \iff \exists\, t \in T.\ \mathsf{enc}_S(t) = m$. For reasons of simplicity, let $\mathsf{Enc}_S(s_1 \vee \ldots \vee s_n)$ be defined as $\mathsf{Enc}_S(\{s_1, \ldots, s_n\})$.

For nondeterministic semiautomata the initial condition $\mathcal{I}$ and all parts of the transition $\mathcal{R}(q, i)$ are of the form $q_1 \vee \ldots \vee q_n$. Therefore, it is sufficient to encode these sets $\{q_1, \ldots, q_n\}$ of states by a propositional formula over $\mathcal{V}_\mathcal{Q}$. Let for example $\mathcal{I} = q_1 \vee \ldots \vee q_n$ hold. Then $\Phi_\mathcal{I} := \mathsf{Enc}_\mathcal{Q}(\{q_1, \ldots q_n\})$ is a symbolic representation of $\mathcal{I}$. Using this settings, $Q \models^{\mathsf{min}}_{\mathsf{prop}} \mathcal{I} \iff \exists q.\,(Q = \{q\}) \wedge \mathsf{enc}_\mathcal{Q}(q) \models_{\mathsf{prop}} \Phi_\mathcal{I}$ holds. Thus, models of the symbolic representation correspond to minimal models of the original initial condition. According to Lemma 2.10 this is sufficient.

Thus, it is easily possible, to encode the initial condition $\mathcal{I}$. Notice, that with n propositional variables 2^n states can be encoded. Moreover, small propositional formulas can encode large sets. For example, if every state of the semiautomaton is an initial state, the set of initial states $\mathcal{I}$ can be encoded by the propositional formula true.

It is much harder to encode the transition function $\mathcal{R} : \mathcal{Q} \times \Sigma \to \mathcal{B}^+(\mathcal{Q})$ of alternating automata. However, for this work it is sufficient to consider only nondeterministic automata. For nondeterministic automata, we are already able to encode $\mathcal{R}(q, i)$ for all q, i. To encode the entire function, we have to be able to distinguish between the states occuring as the input of the transition function and the states occuring in the result. Therefore, for every state variable $q \in \mathcal{V}_\mathcal{Q}$, a new state variable is introduced. These new state variables are used to describe the states occuring in the result of $\mathcal{Q}$. Thus, the new state variable corresponding to a variable $q \in \mathcal{V}_\mathcal{Q}$ represents the value of the variable q at the next point of time. Therefore, it is denoted by $\mathtt{X}q$ following usual temporal logic notations. Further let $\mathbf{X}\varphi$ denote the formula that results from replacing every variable q in φ with $\mathtt{X}q$.

Using this denotations a transition function can for example be encoded by the following propositional formula :

$$\Phi_\mathcal{R} := \bigvee_{q \in \mathcal{Q}, s \in \Sigma} \mathsf{Enc}_\mathcal{Q}(\{q\}) \wedge \mathsf{Enc}_\Sigma(\{s\}) \wedge \mathbf{X}\mathsf{Enc}_\mathcal{Q}(\mathcal{R}(q, s))$$

Again, the models of the symbolic representation correspond to minimal models of the original transition function:

$$Q \models^{\mathsf{min}}_{\mathsf{prop}} \mathcal{R}(q, s) \iff$$
$$\exists q'.\,(Q = \{q'\}) \ \wedge\ \{\mathtt{X}q'' \mid q'' \in \mathsf{enc}_\mathcal{Q}(q')\} \ \cup\ \mathsf{enc}_\mathcal{Q}(q) \ \cup\ \mathsf{enc}_\Sigma(s) \models_{\mathsf{prop}} \Phi_\mathcal{R}$$

Altogether, a nondeterministic semiautomaton $\mathfrak{A} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R})$ can be symbolically represented by $\mathfrak{A}_{\mathsf{sym}} = (\mathcal{V}_\Sigma, \mathcal{V}_\mathcal{Q}, \Phi_\mathcal{I}, \Phi_\mathcal{R})$. To use this symbolic representation, it remains to translate the concept of runs to the symbolic representation.

Definition 2.22 (Symbolic Runs)
Let $\mathfrak{A}_{\mathsf{sym}} = (\mathcal{V}_\Sigma, \mathcal{V}_\mathcal{Q}, \Phi_\mathcal{I}, \Phi_\mathcal{R})$ be a symbolically represented nondeterministic semiautomaton and let w be a finite or infinite word over $\mathcal{P}(\mathcal{V}_\Sigma)$. Then, each word r over $\mathcal{P}(\mathcal{V}_\mathcal{Q})$ with

- $|r| = |w| + 1$ *if w is finite and r infinite otherwise*
- $r^0 \models_{\mathsf{prop}} \Phi_\mathcal{I}$
- $\forall n < |w|.\ r^n \cup w^n \cup \{\mathtt{X}q \mid q \in r^{n+1}\} \models_{\mathsf{prop}} \Phi_\mathcal{R}$

is called a symbolic run of w through $\mathfrak{A}_{\mathsf{sym}}$. *The set of all symbolic runs of a word w through a symbolically represented nondeterministic semiautomaton $\mathfrak{A}_{\mathsf{sym}}$ is denoted by* $\mathsf{SymRUN}_{\mathfrak{A}_{\mathsf{sym}}}(w)$.

Lemma 2.23 *Let $\mathfrak{A} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R})$ be a nondeterministic semiautomaton and let $\mathfrak{A}_{\mathsf{sym}} = (\mathcal{V}_\Sigma, \mathcal{V}_\mathcal{Q}, \Phi_\mathcal{I}, \Phi_\mathcal{R})$ be its symbolic representation. Then for all inputs $\alpha \in \Sigma^* \cup \Sigma^\omega$ and for all words $r \in \mathcal{Q}^* \cup \mathcal{Q}^\omega$, the symbolic representation of r denoted by $\mathsf{enc}_\mathcal{Q}(r)$ is a symbolic run of $\mathsf{enc}_\Sigma(\alpha)$ through $\mathfrak{A}_{\mathsf{sym}}$ if and only if $R = (V, E)$ with $V = \{(r^i, i) \mid i < |r|\}$ and $E = \{((q_1, i), (q_2, i+1)) \mid (q_1, i), (q_2, i+1) \in V\}$ is a minimal run of $\mathfrak{A}$.*

Notice that in Lemma 2.23 r is the unique path through R. Moreover, as $\mathfrak{A}$ is nondeterministic, every minimal run through $\mathfrak{A}$ has the form of R. Therefore, the symbolic runs of $\mathfrak{A}_{\mathsf{sym}}$ correspond to minimal runs of $\mathfrak{A}$. They are the symbolic representation of the unique paths through these minimal runs.

Together with the concept of symbolic runs the symbolic representation of semiautomata directly leads to a symbolic representation of nondeterministic automata over finite words, since the set of final states can be encoded in the same way. Let $\mathcal{A}_\exists^{\mathrm{fin}}(\mathcal{V}_\Sigma, \mathcal{V}_\mathcal{Q}, \Phi_\mathcal{I}, \Phi_\mathcal{R}, \Phi_\mathcal{F})$ denote the symbolic representation of the nondeterministic automaton $\mathfrak{A} := (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R}, \mathcal{F})$. Using Lemma 2.10 and Lemma 2.23 one can easily show that the following relation holds between the nondeterministic automaton and its symbolic representation:

$$\forall w \in \Sigma^*.\, w \models \mathfrak{A} \iff \exists r \in \mathsf{SymRUN}_{(\mathcal{V}_\Sigma, \mathcal{V}_\mathcal{Q}, \Phi_\mathcal{I}, \Phi_\mathcal{R})}(\mathsf{enc}_\Sigma(w)).\ r^{|r|-1} \models_{\mathsf{prop}} \Phi_\mathcal{F}$$

For ω-automata, it is also possible to get a symbolic representation quite easily, because the acceptance component can be described symbolically. This leads to *automaton formulas* [18, 19].

Example 2.24 *Let $\mathfrak{A} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R}, \mathcal{F})$ be a automaton (see Figure 3) with:*

- $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

- $\mathcal{Q} = \{Q_0, Q_1, Q_2\}$
- $\mathcal{I} = Q_0$
- $\mathcal{R}(Q_i, s) = Q_{(i+s \mod 3)}$
- $\mathcal{F} = \{Q_0\}$

Obviously $\mathfrak{A}$ is nondeterministic and even deterministic.

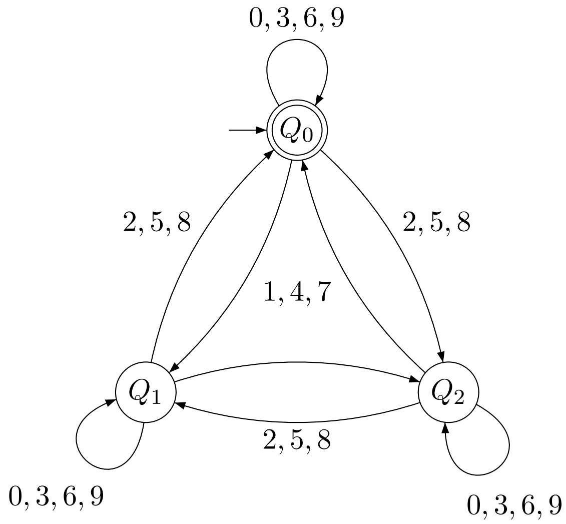

Figure 3: Example 2.24

The ten letters of Σ can be encoded by four propositional variables. For the state set $\mathcal{Q}$, two variables are sufficient. Thus let $\mathcal{V}_\Sigma$, $\mathcal{V}_\mathcal{Q}$, enc_Σ and $\mathsf{enc}_\mathcal{Q}$ be defined by:

$$\mathcal{V}_\Sigma = \{i_0, i_1, i_2, i_3\} \qquad \mathcal{V}_\mathcal{Q} = \{q_0, q_1\}$$

	0	1	2	3	4	5	6	7	8	9
enc_Σ	$\emptyset$	$\{i_0\}$	$\{i_1\}$	$\{i_1, i_0\}$	$\{i_2\}$	$\{i_2, i_0\}$	$\{i_2, i_1\}$	$\{i_2, i_1, i_0\}$	$\{i_3\}$	$\{i_3, i_0\}$

	Q_0	Q_1	Q_2
$enc_\mathcal{Q}$	$\emptyset$	$\{q_0\}$	$\{q_1\}$

With these settings, the initial condition of $\mathfrak{A}$ can be encoded by the formula $\neg q_0 \neg q_1$.

By encoding Σ and $\mathcal{S}$ by propositional variables, additional states and inputs are introduced. For example, the state $\{q_0, q_1\}$ was not present before. However, these additional states and inputs will not influence the semantics of an automaton, provided that no additional accepted runs are introduced. This is obviously guaranteed by the representation of the transition function.

The transition function can be encoded by

$$\Phi_\mathcal{R} := \bigvee_{q \in \mathcal{Q}, s \in \Sigma} \mathsf{Enc}_\mathcal{Q}(\{q\}) \wedge \mathsf{Enc}_\Sigma(\{s\}) \wedge \mathbf{X}\mathsf{Enc}_\mathcal{Q}(\mathcal{R}(q, s))$$

As for all M_1, M_2, S the two propositional formulas $\mathsf{Enc}_S(M_1) \vee \mathsf{Enc}_S(M_2)$ and $\mathsf{Enc}_S(M_1 \cup M_2)$ are equivalent, the inputs can be grouped according to the labels of the transitions in Figure 3. This leads to the following encodings:

- $\mathsf{Enc}_\Sigma(\{0,3,6,9\}) = \neg i_0 \neg i_1 \neg i_2 \neg i_3 \vee i_0 i_1 \neg i_2 \neg i_3 \vee \neg i_0 i_1 i_2 \neg i_3 \vee i_0 \neg i_1 \neg i_2 i_3$
- $\mathsf{Enc}_\Sigma(\{1,4,7\}) = i_0 \neg i_1 \neg i_2 \neg i_3 \vee \neg i_0 \neg i_1 i_2 \neg i_3 \vee i_0 i_1 i_2 \neg i_3$
- $\mathsf{Enc}_\Sigma(\{2,5,8\}) = \neg i_0 i_1 \neg i_2 \neg i_3 \vee i_0 \neg i_1 i_2 \neg i_3 \vee \neg i_0 \neg i_1 \neg i_2 i_3$

Using these encodings, the transition relation can be encoded by:

$$\begin{array}{lrcrcrl}
Q_0 \xrightarrow{1,4,7} Q_1: & \neg q_0 \neg q_1 & \wedge & \mathsf{Enc}_\Sigma(\{1,4,7\}) & \wedge & \mathtt{X}q_0 \neg \mathtt{X}q_1 & \vee \\
Q_0 \xrightarrow{2,5,8} Q_2: & \neg q_0 \neg q_1 & \wedge & \mathsf{Enc}_\Sigma(\{2,5,8\}) & \wedge & \neg \mathtt{X}q_0 \ \mathtt{X}q_1 & \vee \\
Q_1 \xrightarrow{0,3,6,9} Q_1: & q_0 \neg q_1 & \wedge & \mathsf{Enc}_\Sigma(\{0,3,6,9\}) & \wedge & \mathtt{X}q_0 \neg \mathtt{X}q_1 & \vee \\
Q_1 \xrightarrow{1,4,7} Q_2: & q_0 \neg q_1 & \wedge & \mathsf{Enc}_\Sigma(\{1,4,7\}) & \wedge & \neg \mathtt{X}q_0 \ \mathtt{X}q_1 & \vee \\
Q_1 \xrightarrow{2,5,8} Q_0: & q_0 \neg q_1 & \wedge & \mathsf{Enc}_\Sigma(\{2,5,8\}) & \wedge & \neg \mathtt{X}q_0 \neg \mathtt{X}q_1 & \vee \\
Q_2 \xrightarrow{0,3,6,9} Q_2: & \neg q_0 \ q_1 & \wedge & \mathsf{Enc}_\Sigma(\{0,3,6,9\}) & \wedge & \neg \mathtt{X}q_0 \ \mathtt{X}q_1 & \vee \\
Q_2 \xrightarrow{1,4,7} Q_0: & \neg q_0 \ q_1 & \wedge & \mathsf{Enc}_\Sigma(\{1,4,7\}) & \wedge & \neg \mathtt{X}q_0 \neg \mathtt{X}q_1 & \vee \\
Q_2 \xrightarrow{2,5,8} Q_1: & \neg q_0 \ q_1 & \wedge & \mathsf{Enc}_\Sigma(\{2,5,8\}) & \wedge & \neg \mathtt{X}q_0 \ \mathtt{X}q_1 &
\end{array}$$

This formula has not been simplified to show the relation between the original translation relation and its encoding. Finally the set of final states can be encoded by $\Phi_\mathcal{F} = \mathsf{Enc}_\Sigma(\mathcal{F}) = \neg q_0 \neg q_1$. Thus, $\mathfrak{A}_{\mathsf{sym}} := \mathcal{A}^{\mathrm{fin}}_{\exists}(\mathcal{V}_\Sigma, \mathcal{V}_\mathcal{Q}, \Phi_\mathcal{I}, \Phi_\mathcal{R}, \Phi_\mathcal{F})$ is the symbolic representation of $\mathfrak{A}$.

Consider the input $w = 825$. The unique minimal run of w through $\mathfrak{A}$ is $R = (V, E)$ with

$$V = \big\{(Q_0,0),(Q_2,1),(Q_1,2),(Q_0,3)\big\} \qquad \text{and}$$
$$E = \big\{\big((Q_0,0),(Q_2,1)\big),\big((Q_2,1),(Q_1,2)\big),\big((Q_1,2),(Q_0,3)\big)\big\}.$$

The unique path through R is $r = Q_0Q_2Q_1Q_0$. It ends in the set $\mathcal{F} = \{Q_0\}$. Thus, w is accepted by $\mathfrak{A}$. The symbolic representation of w is $\mathsf{enc}_\Sigma(w) = \{i_3\}\{i_1\}\{i_0, i_2\}$. The unique symbolic run of $\mathsf{enc}_\Sigma(w)$ through $\mathfrak{A}_{\mathsf{sym}}$ is $R_{\mathsf{sym}} = \emptyset\{q_1\}\{q_0\}\emptyset$. Notice, that $R_{\mathsf{sym}} = \mathsf{enc}_\mathcal{Q}(r)$ holds. The last state of R_{sym} models $\Phi_\mathcal{F}$. Thus $\mathsf{enc}_\Sigma(w)$ is accepted by $\mathfrak{A}_{\mathsf{sym}}$.

2.6.1 Automaton Formulas

Definition 2.25 (Syntax of Flat Acceptance Conditions)
The following mutually recursive definitions introduce the set of flat acceptance conditions $\mathsf{symac}_\mathcal{V}$ *over a set of variables* $\mathcal{V}$*:*

- *every propositional formula $p \in \mathcal{B}(\mathcal{V})$ is an acceptance condition over $\mathcal{V}$*
- $\neg\varphi \in \mathsf{symac}_\mathcal{V}$*, if* $\varphi \in \mathsf{symac}_\mathcal{V}$

- $\varphi \wedge \psi \in \mathsf{symac}_{\mathcal{V}}$, *if* $\varphi, \psi \in \mathsf{symac}_{\mathcal{V}}$
- $\mathsf{G}\varphi \in \mathsf{symac}_{\mathcal{V}}$, *if* $\varphi \in \mathsf{symac}_{\mathcal{V}}$

Definition 2.26 (Syntax of Automaton Formulas)
The following mutually recursive definitions introduce the set of automaton formulas *$\mathcal{L}_\omega(\mathcal{V})$ over a set of variables $\mathcal{V}$:*

- *every flat acceptance condition $\Phi_{\mathcal{F}} \in \mathsf{symac}_{\mathcal{V}}$ is an automaton formula over $\mathcal{V}$*
- $\neg\varphi \in \mathcal{L}_\omega(\mathcal{V})$, *if* $\varphi \in \mathcal{L}_\omega(\mathcal{V})$
- $\varphi \wedge \psi \in \mathcal{L}_\omega(\mathcal{V})$, *if* $\varphi, \psi \in \mathcal{L}_\omega(\mathcal{V})$
- *$\mathcal{A}_\exists(Q, \Phi_{\mathcal{I}}, \Phi_{\mathcal{R}}, \Phi_{\mathcal{F}}) \in \mathcal{L}_\omega$, if $\Phi_{\mathcal{F}} \in \mathcal{L}_\omega(Q)$ and $Q, \Phi_{\mathcal{I}}, \Phi_{\mathcal{R}}$ are the symbolic representations of the set of states, the set of initial states and the transition relation of a semiautomaton, i. e. Q is a set of variables with $Q \cap \mathcal{V} = \emptyset$, $\Phi_{\mathcal{I}} \in \mathsf{prop}_Q$ and $\Phi_{\mathcal{R}} \in \mathsf{prop}_{Q \cup \mathcal{V} \cup \{\mathtt{X}q \mid q \in Q\}}$. As automaton formulas can be nested, the set of input variables is omitted for reasons of simplicity.*

Flat acceptance conditions are used to distinguish between the parts of an automaton formula that may contain automaton operators and the parts that may not contain these operators. This could as well be achieved without explicitly introducing flat acceptance conditions.

The semantics of alternating automata is defined with respect to runs and paths. However, for nondeterministic automata, these concepts do not need to be distinguished, because every run through a nondeterministic automaton contains exactly on path. Therefore, the semantics of symbolically represented nondeterministic automata is just defined with respect to paths. However, for reasons of simplicity, the paths are often called runs. Moreover, in contrast to alternating automata the semantics of symbolically represented nondeterministic automata are defined with respect to minimal runs. According to Lemma 2.10 this does not make a difference.

Definition 2.27 (Semantics of Flat Acceptance Conditions)
Flat acceptance conditions are a symbolic representation of a subset of acceptance components. Therefore, the semantics of a flat acceptance condition is given by the semantics of acceptance components. The semantics of a flat automaton formula $\varphi \in \mathsf{symac}_{\mathcal{V}}$ is for an infinite word $v \in \mathcal{P}(\mathcal{V})^\omega$ and a point of time $t \in \mathbb{N}$ given by

- $v \models_{\mathsf{AC}} p$ *iff* $v^0 \models_{\mathsf{prop}} p$
- $v \models_{\mathsf{AC}} \neg\varphi$ *iff* $v \not\models_{\mathsf{AC}} \varphi$
- $v \models_{\mathsf{AC}} \varphi \wedge \psi$ *iff* $v \models_{\mathsf{AC}} \varphi$ *and* $v \models_{\mathsf{AC}} \psi$
- $v \models_{\mathsf{AC}} \mathsf{G}\varphi$ *iff* $\forall k.\, v^{k..} \models_{\mathsf{AC}} \varphi$

If $v \models_{\mathsf{AC}} \varphi$ holds for a word $v \in \mathcal{P}(\mathcal{V})^\omega$ and a flat acceptance condition φ, then v is said to model φ.

Definition 2.28 (Semantics of Automaton Formulas)
The semantics of an automaton formulas $\varphi \in \mathcal{L}_\omega(\mathcal{V})$ is for an infinite word $v \in \mathcal{P}(\mathcal{V})^\omega$ given by:

- $v \models_{\mathsf{omega}} \Phi_{\mathcal{F}}$ *iff* $v \models_{\mathsf{AC}} \Phi_{\mathcal{F}}$
- $v \models_{\mathsf{omega}} \neg\varphi$ *iff* $v \not\models_{\mathsf{omega}} \varphi$
- $v \models_{\mathsf{omega}} \varphi \wedge \psi$ *iff* $v \models_{\mathsf{omega}} \varphi$ *and* $v \models_{\mathsf{omega}} \psi$
- $v \models_{\mathsf{omega}} \mathcal{A}_{\exists}(Q, \Phi_{\mathcal{I}}, \Phi_{\mathcal{R}}, \Phi_{\mathcal{F}})$ *iff an infinite word* $\beta \in Q^\omega$ *exists with*
  - $\beta^0 \models_{\mathsf{prop}} \Phi_{\mathcal{I}}$
  - $\left(\beta^i \cup v^i \cup \{\mathtt{X}q \mid q \in \beta^{i+1}\}\right) \models_{\mathsf{prop}} \Phi_{\mathcal{R}}$ *for all* $i \in \mathbb{N}$
  - $\beta \models_{\mathsf{omega}} \Phi_{\mathcal{F}}$

  Notice, that this implies, that β is a symbolic run through the described semiautomaton.

A word $v \in \mathcal{P}(\mathcal{V})^\omega$ is said to satisfy an automaton formula φ iff $v \models_{\mathsf{omega}} \varphi$ holds. An automaton formula φ is equivalent *to an automaton formula ψ (denoted by $\varphi \equiv_{\mathsf{omega}} \psi$) iff for all v the relation $v \models_{\mathsf{omega}} \varphi$ holds iff $v \models_{\mathsf{omega}} \psi$ holds.*

Definition 2.29 (Syntactic Sugar)
Automaton formulas are able to express most of the nondeterministic and universal automata classes mentioned before. However, to be able to express these classes in a convenient way, some syntactic sugar for automaton formulas and flat acceptance conditions is needed:

- $\varphi \vee \psi := \neg(\neg\varphi \wedge \neg\psi)$
- $\varphi \rightarrow \psi := \neg\varphi \vee \psi$
- $\varphi \leftrightarrow \psi := \varphi \rightarrow \psi \wedge \psi \rightarrow \varphi$
- $\mathsf{F}\varphi := \neg\mathsf{G}\neg\varphi$
- $\mathcal{A}_{\forall}(Q, \Phi_{\mathcal{I}}, \Phi_{\mathcal{R}}, \Phi_{\mathcal{F}}) := \neg\mathcal{A}_{\exists}(Q, \Phi_{\mathcal{I}}, \Phi_{\mathcal{R}}, \neg\Phi_{\mathcal{F}})$

Especially, the operator $\mathcal{A}_{\forall}$ is interesting. It describes universal automata!

2.6.2 Flat Automaton Formulas

Automaton formulas are a convenient way to represent nondeterministic and universal ω-automata. However, the connection between automaton formulas and ω-automata is not obvious in general, because some automaton formulas like

$$\mathcal{A}_{\exists}(Q_1, \Phi_{\mathcal{I}_1}, \Phi_{\mathcal{R}_1}, \neg\mathcal{A}_{\exists}(Q_2, \Phi_{\mathcal{I}_2}, \Phi_{\mathcal{R}_2}, \Phi_{\mathcal{F}})) \qquad \text{or}$$
$$\mathcal{A}_{\exists}(Q_1, \Phi_{\mathcal{I}_1}, \Phi_{\mathcal{R}_1}, \Phi_{\mathcal{F}_1}) \wedge \mathcal{A}_{\exists}(Q_2, \Phi_{\mathcal{I}_2}, \Phi_{\mathcal{R}_2}, \Phi_{\mathcal{F}_2})$$

contain more than one automaton operator and other automaton formulas like $\mathsf{G}q$ contain no automaton operators.

On the other hand, automaton formulas of the form $\mathcal{A}_{\exists}(Q, \Phi_{\mathcal{I}}, \Phi_{\mathcal{R}}, \Phi_{\mathcal{F}})$ and $\mathcal{A}_{\forall}(Q, \Phi_{\mathcal{I}}, \Phi_{\mathcal{R}}, \Phi_{\mathcal{F}})$ where $\Phi_{\mathcal{F}}$ is a flat acceptance condition are obviously related to nondeterministic or universal ω-automata, respectively. Those automaton formulas are called *flat*. Since a flat automaton formula φ directly corresponds to an ω-automaton $\mathfrak{A}_{\varphi}$, it is said that φ is *total* or *deterministic* iff $\mathfrak{A}_{\varphi}$ is total or deterministic.

For every automaton formula φ, there is a flat automaton formula φ_{flat} that is equivalent to φ [19]. However, the flattening of automaton formulas has nonelementary complexity [19, 21]. In the following, only flat automaton formulas will be considered.

2.6.3 Classes of Flat Automaton Formulas

Flat automaton formulas are symbolic representations of nondeterministic or universal ω-automata. Therefore, their classes are defined according to the classes of the corresponding ω-automata. To find a suitable syntactic characterisation is quite easy, because flat acceptance conditions are a symbolic representation of a subset of acceptance components:

Definition 2.30 (Classes of Acceptance Conditions)
Let Φ_i, Ψ_i be propositional formulas for all $i \in \{0, \ldots, f\}$. Then, the following classes of acceptance conditions are defined [13]:

True condition:	true
False condition:	false
Initial condition:	Φ_0
Safety condition:	$\mathsf{G}\Phi_0$
Liveness condition:	$\mathsf{F}\Phi_0$
Büchi condition [6, 7]:	$\mathsf{GF}\Phi_0$
Persistence condition [13]:	$\mathsf{FG}\Phi_0$
Rabin condition [16]:	$\bigvee_{j=0}^{f} \left(\mathsf{GF}\Phi_0 \ \wedge \ \mathsf{FG}\Psi_0\right)$
Streett condition [20]:	$\bigwedge_{j=0}^{f} \left(\mathsf{FG}\Phi_0 \ \vee \ \mathsf{GF}\Psi_0\right)$
Prefix condition (1. kind) [13, 17]:	$\bigwedge_{j=0}^{f} \left(\mathsf{G}\Phi_0 \ \vee \ \mathsf{F}\Psi_0\right)$
Prefix condition (2. kind) [13, 17]:	$\bigvee_{j=0}^{f} \left(\mathsf{F}\Phi_0 \ \vee \ \mathsf{G}\Psi_0\right)$

3 Comparison of Alternating ω-Automata and Automaton Formulas

In the previous chapter the basic definitions and lemmata have been explained. This enables us to compare alternating automata with symbolically represented nondeterministic automata. For finite words the situation is well known. Alternating, nondeterministic, universal and deterministic automata on finite words share the same expressiveness. All these classes can express the set of regular languages. However, there are alternating automata with n states such that the smallest deterministic automaton needs $2^n - 1$ states [22]. Interestingly, symbolic represented nondeterministic automata need only n state variables to express these $2^n - 1$ states. Therefore, symbolic represented nondeterministic automata need at most as many state variables as a corresponding alternating automaton needs states. Moreover, it is very easy to translate an alternating automaton to a symbolically represented nondeterministic automaton:

Lemma 3.1 (Nondeterminisation) *Let $\mathfrak{A} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R}, \mathcal{F})$ be an alternating automaton on finite words. Further, let $\mathfrak{B}$ be defined by*

$$\mathfrak{B} := \mathcal{A}_{\exists}^{fin}(\mathcal{V}_{\Sigma}, \mathcal{Q}, \mathcal{I}, \bigwedge_{q \in \mathcal{Q}} \big(q \leftrightarrow \big(\bigvee_{s \in \Sigma} \mathsf{Enc}_{\Sigma}(\{s\}) \ \wedge \ \mathbf{X}\mathcal{R}(q,s)\big)\big), \bigwedge_{q \in \mathcal{Q} \setminus \mathcal{F}} \neg q)$$

Then any word $\alpha \in \Sigma^*$ is accepted by $\mathfrak{A}$, iff $\mathsf{enc}_{\Sigma}(\alpha)$ is accepted by $\mathfrak{B}$.

As this translation is very easy and as it preserves the structure of the alternating automaton, one could even consider alternating automata as symbolically represented nondeterministic automata in a special normal form. Thus, the benefits of alternating and symbolically represented nondeterministic automata can be combined in the case of finite words.

In contrast, the situation is unclear for infinite words. Therefore, it will be investigated in this work. As explained in Section 2.4 there are many different classes of ω-automata. Therefore, we will first investigate semiautomata and runs through semiautomata. In particular, constructions similar to the nondeterminisation of alternating automata on finite words are interesting. Therefore, the following definition is of special interest:

Definition 3.2 ($\mathsf{SymNDET}_{\leftrightarrow}$)
Let $\mathfrak{A} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R})$ be an alternating semiautomaton. Then let $\mathsf{SymNDET}_{\leftrightarrow}(\mathfrak{A})$ be defined by

$$\mathsf{SymNDET}_{\leftrightarrow}(\mathfrak{A}) := (\mathcal{V}_{\Sigma}, \mathcal{Q}, \mathcal{I}, \bigwedge_{q \in \mathcal{Q}} \Big(q \leftrightarrow \big(\bigvee_{s \in \Sigma} \mathsf{Enc}_{\Sigma}(\{s\}) \ \wedge \ \mathbf{X}\mathcal{R}(q,s)\big)\Big))$$

However, to show the connections between an alternating semiautomaton $\mathfrak{A}$ and $\mathsf{SymNDET}_{\leftrightarrow}(\mathfrak{A})$, the following is interesting as an intermediate step:

Definition 3.3 ($\mathsf{SymNDET}_{\rightarrow}$)
Let $\mathfrak{A} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R})$ be an alternating semiautomaton. Then let $\mathsf{SymNDET}_{\rightarrow}(\mathfrak{A})$ be defined by

$$\mathsf{SymNDET}_{\rightarrow}(\mathfrak{A}) := (\mathcal{V}_\Sigma, \mathcal{Q}, \mathcal{I}, \bigwedge_{q\in\mathcal{Q}} \Big(q \rightarrow (\bigvee_{s\in\Sigma} \mathsf{Enc}_\Sigma(\{s\}) \ \wedge \ \mathbf{X}\mathcal{R}(q,s))\Big))$$

The idea of the construction of $\mathsf{SymNDET}_{\rightarrow}$ is that if $R = (V, E)$ is a run of some word w through a semiautomaton $\mathfrak{A}$, then r defined by $|r| := |w| + 1$ and $r^i := \{q \mid (q, i) \in V\}$ is a symbolic run of $\mathsf{enc}_\Sigma(w)$ through $\mathsf{SymNDET}_{\rightarrow}(\mathfrak{A})$. This means that the levels of a run of the original semiautomaton are collapsed to one state of the symbolic run of the nondeterministic symbolically described semiautomaton. On the other hand, a symbolic run can be used to construct a run of the original automaton. More formally, this means:

Definition 3.4 (Collapsed Run)
Let $R = (V, E)$ be a run of a word w through an alternating semiautomaton $\mathfrak{A}$. Then the collapsed run *of R (denoted by $\mathsf{coll}(R)$) is defined by:*

- $|\mathsf{coll}(R)| = |w| + 1$
- $\mathsf{coll}(R)^i = \{q \mid (q, i) \in V\}$

Definition 3.5 (Decollapsed Run)
Let r be a symbolic run. Then the decollapsed run *of r (denoted by $\mathsf{decoll}(r)$) is defined by $\mathsf{decoll}(r) := (V, E)$ with*

- $V = \{(q, i) \mid \big(\forall j. j < i \Rightarrow r^j \neq \emptyset\big) \ \wedge \ q \in r^i\}$
- $E = \{\big((q, i), (q', i + 1)\big) \mid (q, i), (q', i + 1) \in V\}$

Lemma 3.6 *Let $\mathfrak{A} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R})$ be an alternating semiautomaton. Further let $R = (V, E)$ be a run of some word w through $\mathfrak{A}$. Then $\mathsf{coll}(R)$ is a symbolic run of $\mathsf{enc}_\Sigma(w)$ through $\mathsf{SymNDET}_{\rightarrow}(\mathfrak{A})$[4].*

Proof $|\mathsf{coll}(R)| = |\mathsf{enc}_\Sigma(w)| + 1 = |w| + 1$ and $\mathsf{coll}(R)^0 \models_{\mathsf{prop}} \mathcal{I}$ hold obviously. Thus, is remains to show that for all $n < |w|$ the following holds:

$$\mathsf{coll}(R)^n \cup \mathsf{enc}_\Sigma(w)^n \cup \{\mathtt{X}q \mid q \in \mathsf{coll}(R)^{n+1}\} \models_{\mathsf{prop}} \bigwedge_{q\in\mathcal{Q}} \Big(q \rightarrow (\bigvee_{s\in\Sigma} \mathsf{Enc}_\Sigma(\{s\}) \wedge \mathbf{X}\mathcal{R}(q,s))\Big)$$

[4] theorem `COLLAPSED_ALTERNATING_RUN___IMPL_COLLAPSED___THM` in theory `Alternating_Omega_Automata_Lemmata`

This can be simplified to:

$$\forall q \in \mathsf{coll}(R)^n.\ \mathsf{coll}(R)^{n+1} \models_{\mathsf{prop}} \mathcal{R}(q, w^n)$$

$q \in \mathsf{coll}(R)^n$ is equivalent to $(q, n) \in V$. As $R = (V, E)$ is a run of w through $\mathfrak{A}$, $\{q' \mid ((q, n), (q', n+1)) \in E\}$ is a model of $\mathcal{R}(q, w^n)$. By the definition of collapsed runs, $\{q' \mid ((q, n), (q', n+1)) \in E\} \subseteq \mathsf{coll}(R)^{n+1}$ holds. Moreover, we know that $\mathcal{R}(q, w^n)$ is a positive boolean formula. Thus, $\mathsf{coll}(R)^{n+1} \models_{\mathsf{prop}} \mathcal{R}(q, w^n)$ holds. □

Lemma 3.7 *Let $\mathfrak{A} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R})$ be an alternating semiautomaton. Further let r be a symbolic run of some encoded word $\mathsf{enc}_\Sigma(w)$ through $\mathsf{SymNDET}_{\rightarrow}(\mathfrak{A})$. Then $R = (V, E) := \mathsf{decoll}(R)$ is a run of word w through $\mathfrak{A}$*[5]*.*

Proof To show that $\mathsf{decoll}(r)$ is really a run of w through $\mathfrak{A}$, we have to show:

- for every $(q, l+1) \in V$ exists a $q' \in \mathcal{Q}$, such that $\big((q', l), (q, l+1)\big) \in E$.
- $\{q \mid (q, 0) \in V\}$ is a model of $\mathcal{I}$
- for all $(q, l) \in V$ with $l < |w|$ the set $\{q' \mid \big((q, l), (q', l+1)\big) \in E\}$ is a model of $\mathcal{R}(q, w^l)$

If $(q, l+1) \in V$ holds, $r^j \neq \emptyset$ holds for all $j < l+1$. This implies that there exists some q' with $(q', l) \in V$. By definition of E we know $((q', l), (q, l+1)) \in E$. Next, $\{q \mid (q, 0) \in V\} = r^0 \models_{\mathsf{prop}} \mathcal{I}$ holds by the definition of symbolic runs. Finally, let $(q, l) \in V$ hold for some $l < |w|$. This implies $q \in r^l$. Therefore the transitions relation of $\mathsf{SymNDET}_{\rightarrow}(\mathfrak{A})$ implies, $r^{l+1} = \{q' \mid \big((q, l), (q', l+1)\big) \in E\} \models_{\mathsf{prop}} \mathcal{R}(q, w^l)$. □

Thus, runs of an alternating semiautomaton $\mathfrak{A}$ are strongly related to symbolic runs of $\mathsf{SymNDET}_{\rightarrow}(\mathfrak{A})$. However, when collapsing a run, information is lost. Thus, it is not possible to reconstruct the original run from the collapsed run. Moreover, in general R does not has to be a run through $\mathfrak{A}$ if $\mathsf{coll}(\mathfrak{A})$ is a symbolic run through $\mathsf{SymNDET}_{\rightarrow}(\mathfrak{A})$. However, a symbolic run through $\mathsf{SymNDET}_{\rightarrow}(\mathfrak{A})$ can be used to construct a run through $\mathfrak{A}$.

Example 3.8 *Consider the finite semiautomaton $\mathfrak{A} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R})$ with $\Sigma = \{a, b\}$, $\mathcal{Q} = \{q_0, q_1, q_2\}$, $\mathcal{I} = q_0 \vee q_2$ and $\mathcal{R}$ given by*

	a	b
q_0	$q_0 \vee q_1 \vee q_2$	$q_1 q_2$
q_1	false	$q_0 q_2$
q_2	$q_0 \vee q_1$	true

[5] theorem `DECOLLAPSED_RUN___IMPL_COLLAPSED___THM` in theory `Alternating_Omega_Automata_Lemmata`

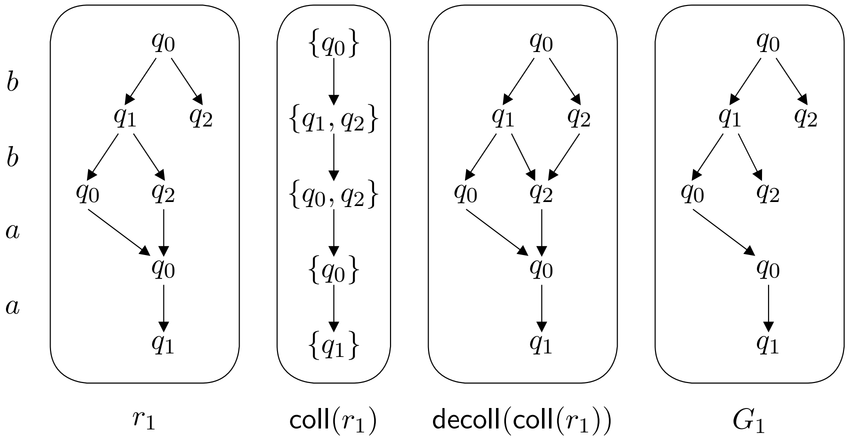

Figure 4: Example 3.8

from Example 2.11. Further let $\mathcal{V}_\Sigma := \{i_0\}$ *and* $\mathsf{enc}_\Sigma(a) := \emptyset$, $\mathsf{enc}_\Sigma(b) := \{i_0\}$ *be the encoding used for* $\mathsf{SymNDET}_\rightarrow(\mathfrak{A})$*. Then* $\mathsf{SymNDET}_\rightarrow(\mathfrak{A})$ *is given by*

$$\begin{array}{rcl} \mathsf{SymNDET}_\rightarrow(\mathfrak{A}) & = & (\{i_0\}, \{q_0, q_1, q_2\}, q_0 \vee q_2, \\ & & \quad (q_0 \rightarrow \Big((\neg i_0 \wedge (q_0 \vee q_1 \vee q_2)) \vee i_0 q_1 q_2\Big)) \quad \wedge \\ & & \quad (q_1 \rightarrow i_0 q_0 q_2) \quad \wedge \\ & & \quad (q_2 \rightarrow \Big((\neg i_0 \wedge (q_0 \vee q_1)) \vee i_0\Big)) \\ & &) \end{array}$$

Reconsider the run r_1 *shown in Figure 1 and Figure 4. It is a run of the word bbaa through* $\mathfrak{A}$*. The encoding* $\mathsf{enc}_\Sigma(bbaa)$ *equals to* $\{i_0\}\{i_0\}\emptyset\emptyset$*. Then* $\mathsf{coll}(r_1) = \{q_0\}\{q_1 q_2\}\{q_0 q_2\}\{q_0\}$ $\{q_1\}$ *is a symbolic run of* $\mathsf{enc}_\Sigma(bbaa)$ *through* $\mathsf{SymNDET}_\rightarrow(\mathfrak{A})$*. Consider the graph* G_1 *given in Figure 4.* $\mathsf{coll}(G_1) = \mathsf{coll}(r_1)$ *holds, but* G_1 *is not a run of bbaa through* $\mathfrak{A}$*. In contrast* $\mathsf{decoll}(\mathsf{coll}(r_1))$ *is a run of bbaa through* $\mathfrak{A}$*. Notice that* $\mathsf{decoll}(\mathsf{coll}(r_1)) \neq r_1$ *holds.*

After we have seen the connection between an alternating semiautomaton $\mathfrak{A}$ and $\mathsf{SymNDET}_\rightarrow(\mathfrak{A})$, we can have a look at $\mathsf{SymNDET}_\leftrightarrow(\mathfrak{A})$ now. Obviously every symbolic run through $\mathsf{SymNDET}_\leftrightarrow(\mathfrak{A})$ is also a symbolic run through $\mathsf{SymNDET}_\rightarrow(\mathfrak{A})$:

Lemma 3.9 *Let* $\mathfrak{A}$ *be a semiautomaton and* r *be a symbolic run of some word* w *through* $\mathsf{SymNDET}_\leftrightarrow(\mathfrak{A})$*. Then* r *is also a symbolic run of* w *through* $\mathsf{SymNDET}_\rightarrow(\mathfrak{A})$[6]*.*

Moreover, every symbolic run through $\mathsf{SymNDET}_\rightarrow(\mathfrak{A})$ can be extended to a symbolic run through $\mathsf{SymNDET}_\leftrightarrow(\mathfrak{A})$.

[6] theorem `EQ_COLLAPSED_RUN___IMPLIES___IMPL_COLLAPSED_RUN` in theory `Alternating_Omega_Automata_Lemmata`

Lemma 3.10 *Let $\mathfrak{A} = (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R})$ be a semiautomaton and r be a symbolic run of some word w through* $\mathsf{SymNDET}_{\rightarrow}(\mathfrak{A})$. *Then there is a symbolic run s of w through* $\mathsf{SymNDET}_{\leftrightarrow}(\mathfrak{A})$ *with $|r| = |s|$ and $r^i \subseteq s^i$ for all $i < |s|$. Additionally, $s^{|r|-1} = r^{|r|-1}$ if r is finite*[7].

Proof Let $f : (\mathcal{V_Q}^* \cup \mathcal{V_Q}^\omega) \rightarrow (\mathcal{V_Q}^* \cup \mathcal{V_Q}^\omega)$ and $s \in \mathcal{V_Q}^* \cup \mathcal{V_Q}^\omega$ be defined by

$$\begin{aligned} f(v)^n &:= \begin{cases} v^n \cup \{q \mid v^{n+1} \models_{\mathsf{prop}} \mathcal{R}(q, \mathsf{enc}_\Sigma^{-1}(w^n))\} & \text{if } n < |v| - 1 \\ v^n & \text{otherwise} \end{cases} \\ |s| &:= |r| \\ s^n &:= \bigcup_{j \in \mathbb{N}} \left(f^j(r)\right)^n \end{aligned}$$

Using these definitions s is a symbolic run of w through $\mathsf{SymNDET}_{\leftrightarrow}(\mathcal{A})$ with the demanded properties. Obviously $\forall i. \forall v.\, v^i \subseteq f(v)^i$ holds. Thus $\forall i.\, r^i \subseteq s^i$ holds. Moreover, $s^{|r|-1} = r^{|r|-1}$ holds if r is finite.

$r^0 \models_{\mathsf{prop}} \mathcal{I}$ holds, because r is a symbolic run of w through $\mathsf{SymNDET}_{\rightarrow}(\mathfrak{A})$. As $\mathcal{I}$ is a positive propositional formula, $s^0 \models_{\mathsf{prop}} \mathcal{I}$ holds. It remains to show that for all $i < |s| - 1$ and all $q \in \mathcal{Q}$ the proposition $q \in s^i \Leftrightarrow s^{i+1} \models_{\mathsf{prop}} \mathcal{R}(q, \mathsf{enc}_\Sigma^{-1}(w^i))$ holds. Thus let $i < |s| - 1$ and $q \in \mathcal{Q}$ be arbitrary but constant values.

The definition of f implies $q \in f(s)^i \Leftarrow s^{i+1} \models_{\mathsf{prop}} \mathcal{R}(q, \mathsf{enc}_\Sigma^{-1}(w^i))$ for all q and all $i < |s| - 1$. As $f(s) = s$ holds, one direction of the proposition is proved. It remains to show that $q \in s^i \Rightarrow s^{i+1} \models_{\mathsf{prop}} \mathcal{R}(q, \mathsf{enc}_\Sigma^{-1}(w^i))$ holds.

As $\mathcal{Q}$ is finite there is for all n a m_0 such that for all $m > m_0$ the proposition $(f^m(r))^n = (f^{m_0}(r))^n$ holds. Obviously $s^n = f^{m_0}(r)^n$ holds for this m_0. Thus, an m_0 exists with $s^i = f^{m_0}(r)^i$ and $s^{i+1} = f^{m_0}(r)^{i+1}$.

$q \in r^i \Rightarrow r^{i+1} \models_{\mathsf{prop}} \mathcal{R}(q, \mathsf{enc}_\Sigma^{-1}(w^i))$ holds, because r is a symbolic run of w through $\mathsf{SymNDET}_{\rightarrow}(\mathfrak{A})$. This property is preserved by the application of f to a path, i. e. for all j the proposition $q \in f^j(r)^i \Rightarrow f^j(r)^{i+1} \models_{\mathsf{prop}} \mathcal{R}(q, \mathsf{enc}_\Sigma^{-1}(w^i))$ holds. This implies $q \in f^{m_0}(r)^i \Rightarrow f^{m_0}(r)^{i+1} \models_{\mathsf{prop}} \mathcal{R}(q, \mathsf{enc}_\Sigma^{-1}(w^i))$. Thus $q \in s^i \Rightarrow s^{i+1} \models_{\mathsf{prop}} \mathcal{R}(q, \mathsf{enc}_\Sigma^{-1}(w^i))$ holds. □

Thus also runs through an alternating semiautomaton $\mathfrak{A}$ are connected to symbolic runs through $\mathsf{SymNDET}_{\leftrightarrow}(\mathfrak{A})$. These result can be used to prove a lot of interesting result about alternating automata. For example, it is easily possible to prove Lemma 3.1 using Lemma 3.6, Lemma 3.7, Lemma 3.9 and Lemma 3.10.

Lemma 3.11 (Lemma 3.1) *Let $\mathfrak{A} = (\mathfrak{A}', \mathcal{F})$ be an alternating automaton on finite words. Further, let $\mathfrak{B}$ be defined by*

$$\mathfrak{B} := \mathcal{A}_\exists^{\mathit{fin}}(\mathsf{SymNDET}_{\leftrightarrow}(\mathfrak{A}'), \bigwedge_{q \in \mathcal{Q} \setminus \mathcal{F}} \neg q)$$

[7] theorem `IMPL_COLLAPSED_RUN___EQ_COLLAPSED_RUN___ENRICHMENT` in theory `Alternating_Omega_Automata_Lemmata`

Then any word $\alpha \in \Sigma^*$ is accepted by $\mathfrak{A}$, iff $\mathsf{enc}_\Sigma(\alpha)$ is accepted by $\mathfrak{B}$.

Proof Let α be accepted by $\mathfrak{A}$. Then a run $R = (V, E)$ of α through $\mathfrak{A}'$ exists such that all paths through R of length $|\alpha|+1$ end in $\mathcal{F}$. This implies $\{q \mid (q, |\alpha|) \in V\} \subseteq \mathcal{F}$. Then $\mathsf{coll}(R)$ is a symbolic run of $\mathsf{enc}_\Sigma(\alpha)$ through $\mathsf{SymNDET}_\rightarrow(\mathfrak{A}')$ according to Lemma 3.6. According to Lemma 3.10 an extension r' of $\mathsf{coll}(R)$ exists such that r' is a symbolic run of $\mathsf{enc}_\Sigma(\alpha)$ through $\mathsf{SymNDET}_\leftrightarrow(\mathfrak{A}')$ and $r'^{|\alpha|} = \mathsf{coll}(R)^{|\alpha|}$ holds. Then $r'^{|\alpha|} = \{q \mid (q, |\alpha|) \in V\} \subseteq \mathcal{F}$ holds, because for all $(q, |\alpha|) \in V$ a path p of length $|\alpha|+1$ through R with $p^{|\alpha|} = q$ exists. Thus $r'^{|\alpha|} \models_{\mathsf{prop}} \bigwedge_{q \in \mathcal{Q} \setminus \mathcal{F}} \neg q$. Therefore, $\mathsf{enc}_\Sigma(\alpha)$ is accepted by $\mathfrak{B}$.

Thus, let $\mathsf{enc}_\Sigma(\alpha)$ be accepted by $\mathfrak{B}$. Then a symbolic run r' of $\mathsf{enc}_\Sigma(\alpha)$ through $\mathsf{SymNDET}_\leftrightarrow(\mathfrak{A}')$ exists such that $r'^{|\alpha|} \subseteq \mathcal{F}$ holds. Then according to Lemma 3.9 r' is a run of $\mathsf{enc}_\Sigma(\alpha)$ through $\mathsf{SymNDET}_\rightarrow(\mathfrak{A}')$. Therefore $\mathsf{decoll}(r')$ is a run of α through $\mathfrak{B}$. Every path through $\mathsf{decoll}(r')$ of length $|\alpha|+1$ ends in $r'^{|\alpha|}$ and therefore in $\mathcal{F}$. Thus α is accepted by $\mathfrak{A}$. □

4 Consequences

In the previous section, the relationship between alternating semiautomata and the symbolically described nondeterministic automata $\mathsf{SymNDET}_\rightarrow$ and $\mathsf{SymNDET}_\leftrightarrow$ have been presented. This relationship has been used to give an alternative proof of the well known symbolic nondeterminisation algorithm for alternating automata on finite words. In this section, the presented relationship will be used to prove new results for alternating automata on infinite words. Let us start with the very basic case, where only the existence of runs is of interest:

Lemma 4.1 (Nondeterminisation of $\mathsf{A}_{\mathsf{true}}$) *Let $\mathfrak{A} = (\mathfrak{A}', \mathsf{true})$ be an alternating automaton on infinite words over an alphabet Σ. Further, let $\mathfrak{B}$ and $\mathfrak{C}$ be defined by*

$$\begin{aligned} \mathfrak{B} &:= \mathcal{A}_\exists(\mathsf{SymNDET}_\rightarrow(\mathfrak{A}'), \mathsf{true}) \\ \mathfrak{C} &:= \mathcal{A}_\exists(\mathsf{SymNDET}_\leftrightarrow(\mathfrak{A}'), \mathsf{true}) \end{aligned}$$

Then any infinite word $\alpha \in \Sigma^\omega$ is accepted by $\mathfrak{A}$, iff $\mathsf{enc}_\Sigma(\alpha)$ is accepted by $\mathfrak{B}$. Furthermore, $\mathsf{enc}_\Sigma(\alpha)$ is accepted by $\mathfrak{B}$ iff it is accepted by $\mathfrak{C}$[8].

Proof That $\mathsf{enc}_\Sigma(\alpha)$ is accepted by $\mathfrak{B}$ iff it is accepted by $\mathfrak{C}$, is a direct consequence of Lemma 3.9 and Lemma 3.10. That α is accepted by $\mathfrak{A}$ iff $\mathsf{enc}_\Sigma(\alpha)$ is accepted by $\mathfrak{B}$ is a consequence of Lemma 3.6 and Lemma 3.7. □

Remark 4.2
One could guess that similar to Lemma 4.1 the automaton $\mathfrak{A} = (\mathfrak{A}', \mathsf{false})$ corresponds to $\mathfrak{B} := \mathcal{A}_\exists(\mathsf{SymNDET}_\rightarrow(\mathfrak{A}'), \mathsf{false})$ and $\mathfrak{C} := \mathcal{A}_\exists(\mathsf{SymNDET}_\leftrightarrow(\mathfrak{A}'), \mathsf{false})$. However, this

[8]theorem `NDET_TRUE___A_TRUE___IMPL` and theorem `NDET_TRUE___A_TRUE___EQ` in theory `Alternating_Omega_Automata`

is not true. In fact, an nondeterministic false automaton does not accept any word, because its semantics demands that a symbolic run exists that fulfils the acceptance condition false*.*

A little bit more complicated than the nondeterminisation algorithm in this simple case is the nondeterminisation of $\mathsf{A}_\mathsf{G}^{\mathsf{universally\ total}}$ automata.

Lemma 4.3 (Nondeterminisation of $\mathsf{A}_\mathsf{G}^{\mathsf{universally\ total}}$) *Let $\mathfrak{A} = (\mathfrak{A}', \mathsf{G}\mathcal{F})$ be an alternating automaton on infinite words such that $\mathfrak{A}'$ is universally total. Further, let $\mathfrak{B}$ be defined by*

$$\mathfrak{B} := \mathcal{A}_\exists(\mathsf{SymNDET}_\rightarrow(\mathfrak{A}'), \mathsf{G} \bigwedge_{q \in \mathcal{Q} \setminus \mathcal{F}} \neg q)$$

Then any word $\alpha \in \Sigma^*$ is accepted by $\mathfrak{A}$, iff $\mathsf{enc}_\Sigma(\alpha)$ is accepted by $\mathfrak{B}$[9]*.*

Proof Let α be an infinite word that is accepted by $\mathfrak{A}$. Then, a run $R = (V, E)$ of α through $\mathfrak{A}$ exists such that every infinite path p through R contains only states from $\mathcal{F}$. As $\mathfrak{A}'$ is universally total, every run through R is infinite. Thus, $(q, i) \in V$ implies $q \in \mathcal{F}$. Thus $\mathsf{coll}(R)^i \subseteq \mathcal{F}$ holds for all i, i. e. $\mathsf{coll}(R)$ fulfils the acceptance condition of $\mathcal{B}$. Moreover, Lemma 3.6 implies that $\mathsf{coll}(R)$ is a symbolic run of $\mathsf{enc}_\Sigma(\alpha)$ through $\mathfrak{B}$. Thus, $\mathsf{enc}_\Sigma(\alpha)$ is accepted by $\mathfrak{B}$.

On the other hand, let $\mathsf{enc}_\Sigma(\alpha)$ be accepted by $\mathfrak{B}$. Then a symbolic run r of $\mathsf{enc}_\Sigma(\alpha)$ through $\mathfrak{B}$ exists with $r^i \subseteq \mathcal{F}$ for all i. Then, $\mathsf{decoll}(r)$ is a run of α through $\mathfrak{A}$ according to Lemma 3.7. As $\mathfrak{A}'$ is universally total, there is a infinite path through $\mathsf{decoll}(r)$. This path contains only states from $\mathcal{F}$. Thus, α is accepted by $\mathfrak{A}$.

In contrast to the nondeterminisation of alternating automata on finite words and of A_true automata the $\mathsf{A}_\mathsf{G}^{\mathsf{universally\ total}}$ automata cannot be extended to an equational transition function, as the following counterexample shows:

Example 4.4 *Let $\mathfrak{A} := (\mathfrak{A}', \mathsf{G}\mathcal{F})$ be an alternating automaton with $\mathfrak{A}' := (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R})$, $\Sigma := \{a\}$, $\mathcal{Q} := \{q_0, q_1\}$, $\mathcal{I} := q_0$, $\mathcal{R}(q, s) := q_0$ for all q, s and $\mathcal{F} := \{q_0\}$. Then $R = (V, E)$ with $V = \{(q_0, i) \mid i \in \mathbb{N}\}$ and $E = \{((q_0, i), (q_0, i + 1)) \mid i \in \mathbb{N}\}$ is a run of a^ω through $\mathfrak{A}'$. The unique path through R is q_0^ω. Therefore, a^ω is accepted by $\mathfrak{A}$.*

Let $\mathfrak{B}$ be defined by $\mathfrak{B} := \mathcal{A}_\exists(\mathsf{SymNDET}_\leftrightarrow(\mathfrak{A}'), \mathsf{G} \bigwedge_{q \in \mathcal{Q} \setminus \mathcal{F}} \neg q)$. Then $\{q_0, q_1\}^\omega$ is the unique symbolic run of $\mathsf{enc}_\Sigma(a^\omega)$ through $\mathfrak{B}$. Therefore, $\mathsf{enc}_\Sigma(a^\omega)$ is not accepted by $\mathfrak{B}$.

However, for every $\mathsf{A}_\mathsf{G}^{\mathsf{universally\ total}}$ automaton an equivalent symbolically represented NDET_G automaton with an equational transition relation exists, because a $\mathsf{A}_\mathsf{G}^{\mathsf{universally\ total}}$ automaton can be translated to an A_true automaton. This automaton can be translated

[9] theorem `NDET_G___A_UNIVERSALLY_TOTAL_WEAK_CO_BUECHI___IMPL` in theory `Alternating_Omega_Automata`

to a symbolically represented $\mathsf{NDET}_{\mathsf{true}}$ automaton with an equational transition function according to Lemma 4.6. However, a $\mathsf{NDET}_{\mathsf{true}}$ automaton is a special $\mathsf{NDET}_{\mathsf{G}}$ automaton.

Lemma 4.5 *Let $\mathfrak{A} := (\mathfrak{A}', \mathsf{G}\mathcal{F})$ be an alternating automaton such that $\mathfrak{A}' := (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R})$ is universally total. Further let $\mathfrak{B} := (\mathfrak{B}', \mathsf{true})$ with $\mathfrak{B}' := (\Sigma, \mathcal{Q}, \mathcal{I}, \mathcal{R}')$ such that*

$$\mathcal{R}'(q, s) := \begin{cases} \mathcal{R}'(q, s) & \text{if } s \in \mathcal{F} \\ \mathsf{false} & \text{otherwise} \end{cases}$$

Then a word $w \in \Sigma^\omega$ is accepted by $\mathfrak{A}$ iff w is accepted by $\mathfrak{B}$[10].

Proof Let $w \in \Sigma^\omega$ be accepted by $\mathfrak{A}$. Then a run $R = (V, E)$ of w through $\mathfrak{A}'$ exists such that $p^n \in \mathcal{F}$ holds for all infinite paths p through R and all $n \in \mathbb{N}$. As $\mathfrak{A}'$ is universally total, for all $(q, n) \in V$ an infinite path p through R exists with $p^n = q$. Therefore, $(q, n) \in V$ implies $q \in \mathcal{F}$. Thus, R is a run through $\mathfrak{B}'$ and w is accepted by $\mathfrak{B}$.

Thus, let w be accepted by $\mathfrak{B}$. Then a run $R = (V, E)$ of w through $\mathfrak{B}'$ exists. The definition of $\mathfrak{B}'$ implies that R is also a run of w through $\mathfrak{A}'$. Moreover, it implies that only states from $\mathcal{F}$ occur in R. Thus, every infinite path through R contains only states from $\mathcal{F}$ and w is accepted by $\mathfrak{A}$. □

Similar to the translation of of $\mathsf{A}_{\mathsf{G}}^{\text{universally total}}$ automata into $\mathsf{NDET}_{\mathsf{true}}$ automata with an equational transition, the negation of alternating automata by dualisation [12, 15] can be used as an intermediate step to translate $\mathsf{A}_{\mathsf{F}}^{\text{existentially total}}$ and $\mathsf{A}_{\mathsf{false}}$ automata to $\mathsf{U}_{\mathsf{false}}$ automata efficiently. Although the negation of alternating automata can be computed easily, the correctness proof of this negation is quite complicated and needs some concepts such as logical games and strategies. Therefore, neither the negation of alternating automata nor the translations to $\mathsf{U}_{\mathsf{false}}$ automata will be explained in detail here.

The presented translations can be used to extend the hierarchy of ω-automata shown in Figure 2.

Lemma 4.6 $\mathsf{A}_{\mathsf{true}} \approx \mathsf{NDET}_{\mathsf{true}}$

Proof The direction $\mathsf{NDET}_{\mathsf{true}} \preceq \mathsf{A}_{\mathsf{true}}$ is clear by definition. The direction $\mathsf{A}_{\mathsf{true}} \preceq \mathsf{NDET}_{\mathsf{true}}$ is a consequence of Lemma 4.1. □

Lemma 4.7 $\mathsf{A}_{\mathsf{false}} \approx \mathsf{U}_{\mathsf{false}}$

Proof The classes $\mathsf{A}_{\mathsf{false}}$ and $\mathsf{A}_{\mathsf{true}}$ are dual. This can be easily shown using the negation of alternating automata described in [15]. As $\mathsf{U}_{\mathsf{false}}$ and $\mathsf{NDET}_{\mathsf{true}}$ are dual too, Lemma 4.6 implies $\mathsf{A}_{\mathsf{false}} \approx \mathsf{U}_{\mathsf{false}}$. □

[10] theorem `A_TRUE___A_UNIVERSALLY_TOTAL_WEAK_CO_BUECHI` in theory `Alternating_Omega_Automata`

Lemma 4.8 $\mathsf{A}_{\mathsf{G}}^{\text{universally total}} \approx \mathsf{NDET}_{\mathsf{G}}$

Proof The direction $\mathsf{NDET}_{\mathsf{G}} \preceq \mathsf{A}_{\mathsf{G}}^{\text{universally total}}$ is clear by definition, because every nondeterministic automaton is universally total. The direction $\mathsf{A}_{\mathsf{G}}^{\text{universally total}} \preceq \mathsf{NDET}_{\mathsf{G}}$ is a consequence of Lemma 4.3. □

Lemma 4.9 $\mathsf{A}_{\mathsf{F}}^{\text{existentially total}} \approx \mathsf{DET}_{\mathsf{F}}$

Proof The classes $\mathsf{A}_{\mathsf{F}}^{\text{existentially total}}$ and $\mathsf{A}_{\mathsf{G}}^{\text{universally total}}$ are dual. This can be easily shown using the negation of alternating automata described in [15]. As $\mathsf{DET}_{\mathsf{F}}$ and $\mathsf{NDET}_{\mathsf{G}}$ are also dual, Lemma 4.8 implies $\mathsf{A}_{\mathsf{F}}^{\text{existentially total}} \approx \mathsf{NDET}_{\mathsf{F}}$. □

For most of the classes of ω-automata shown in Figure 2 we have presented an equivalent class of alternating ω-automata. The only exception are classes that are equivalent to $\mathsf{DET}_{\mathsf{Prefix}}$. However, the results $\mathsf{DET}_{\mathsf{G}} \approx \mathsf{A}_{\mathsf{true}}$ and $\mathsf{DET}_{\mathsf{F}} \approx \mathsf{A}_{\mathsf{false}}$ can be used to identify such a class:

Theorem 4.10
$\mathsf{DET}_{\mathsf{Prefix}} \approx \mathsf{A}_{\mathsf{Initial}}$

Proof Let $\mathfrak{A}$ be a $\mathsf{DET}_{\mathsf{Prefix}}$ automaton. Then there exists $\mathfrak{A}_{1,1}, \ldots\ \mathfrak{A}_{1,n_1}$, $\mathfrak{A}_{2,1} \ldots$, $\mathfrak{A}_{m,n_m} \in \mathsf{DET}_{\mathsf{G}} \cup \mathsf{DET}_{\mathsf{F}}$ such that $\bigvee_{i=1..m} \bigwedge_{j=1..n_m} \mathfrak{A}_{i,j}$ is equivalent to $\mathcal{A}$, because $\mathsf{DET}_{\mathsf{Prefix}}$ is the boolean closure of $\mathsf{DET}_{\mathsf{G}}$ and $\mathsf{DET}_{\mathsf{F}}$ [19]. As $\mathsf{DET}_{\mathsf{G}} \approx \mathsf{A}_{\mathsf{TRUE}}$ and $\mathsf{DET}_{\mathsf{F}} \approx \mathsf{A}_{\mathsf{FALSE}}$ hold, there are $\mathfrak{B}_{i,j} \in \mathsf{A}_{\mathsf{TRUE}} \cup \mathsf{A}_{\mathsf{FALSE}}$ of the form $\mathfrak{B}_{i,j} = (\Sigma, \mathcal{Q}_{i,j}, \mathcal{I}_{i,j}, \mathcal{R}_{i,j}, \mathcal{AC}_{i,j})$ such that $\mathcal{L}(\mathcal{B}_{i,j}) = \mathcal{L}(\mathcal{A}_{i,j})$ holds for all i, j and the sets $\mathcal{Q}_{i,j}$ are pairwise distinct. Let $\mathcal{R} : \bigcup_{i,j} \mathcal{Q}_{i,j} \times \Sigma \to \mathcal{B}^+(\bigcup_{i,j} \mathcal{Q}_{i,j})$ be defined as $\mathcal{R}(q, s) = \mathcal{R}_{i,j}(q, s)$ for all $q \in \mathcal{Q}_{i,j}$. As all $\mathcal{Q}_{i,j}$ are pairwise distinct, this definition is sound. Then $\mathfrak{A}$ is equivalent to

$$\mathfrak{B} := (\Sigma, \bigcup_{i,j} \mathcal{Q}_{i,j}, \bigvee_{i=1..m} \bigwedge_{j=1..n_m} \mathcal{I}_{i,j}, \mathcal{R}, \bigcup_{\mathfrak{B}_{i,j} \in \mathsf{A}_{\mathsf{true}}} Q_{i,j})$$

This equivalence can be easily shown using Lemma 2.21. Obviously, $\mathfrak{B}$ is a $\mathsf{A}_{\mathsf{Initial}}$ automaton. Thus, $\mathsf{DET}_{\mathsf{Prefix}} \preceq \mathsf{A}_{\mathsf{Initial}}$ holds.

On the other hand, let $\mathfrak{B} = (\Sigma, \mathcal{Q}, \mathcal{R}, \mathcal{F})$ be a $\mathsf{A}_{\mathsf{Initial}}$ automaton. As $\mathcal{I}$ is a positive propositional formula there are $q_{i,j} \in Q$ such that $\bigvee_{i=1..m} \bigwedge_{j=1..n_m} q_{i,j}$ is equivalent to $\mathcal{I}$. Let $\mathfrak{B}_{i,j}$ be defined by $\mathfrak{B}_{i,j} := (\Sigma, \mathcal{Q}, q_{i,j}, \mathcal{R}, \mathcal{AC}_{i,j})$ with $\mathcal{AC}_{i,j} := \mathsf{true}$ if $q_{i,j} \in \mathcal{F}$ and $\mathcal{AC}_{i,j} = \mathsf{false}$ otherwise. Using this definition, $\mathfrak{B}$ is equivalent to $\bigvee_{i=1..m} \bigwedge_{j=1..n_m} \mathfrak{B}_{i,j}$. Such a decomposition is not possible for arbitrary alternating ω-automata. However, it is possible for $\mathsf{A}_{\mathsf{Initial}}$ automata. The proof is omitted here, because it is long but mainly technical. It can be found in the HOL theories[11].

[11] theorem `ALT_SEM_S0_TRUE`, theorem `ALT_SEM_S0_FALSE`, theorem `ALT_SEM_S0_OR_SPLIT`, theorem `ALT_SEM_S0_AND_SPLIT___INITIAL` and theorem `ALT_SEM_INITIAL_S0_P_PROP` in theory `Alternating_Omega_Automata`

As $\mathcal{B}_{i,j} \in \mathsf{A}_{\mathsf{TRUE}} \cup \mathsf{A}_{\mathsf{FALSE}}$ for all i, j there are $\mathcal{A}_{i,j} \in \mathsf{DET}_{\mathsf{G}} \cup \mathsf{DET}_{\mathsf{F}}$ such that $\mathcal{L}(\mathcal{A}_{i,j}) = \mathcal{L}(\mathcal{B}_{i,j})$ holds for all i, j. Thus, $\mathcal{B}$ is equivalent to $\bigvee_{i=1..m} \bigwedge_{j=1..n_m} \mathcal{A}_{i,j}$. Since $\mathsf{DET}_{\mathsf{Prefix}}$ is the boolean closure of $\mathsf{DET}_{\mathsf{G}}$ and $\mathsf{DET}_{\mathsf{F}}$, there is also a $\mathsf{DET}_{\mathsf{Prefix}}$ automaton that is equivalent to $\mathcal{B}$. Thus, $\mathsf{A}_{\mathsf{Initial}} \preceq \mathsf{DET}_{\mathsf{Prefix}}$ holds.

Alltogether, these results lead to the extended hierarchy of alternating ω-automata shown in Figure 5.

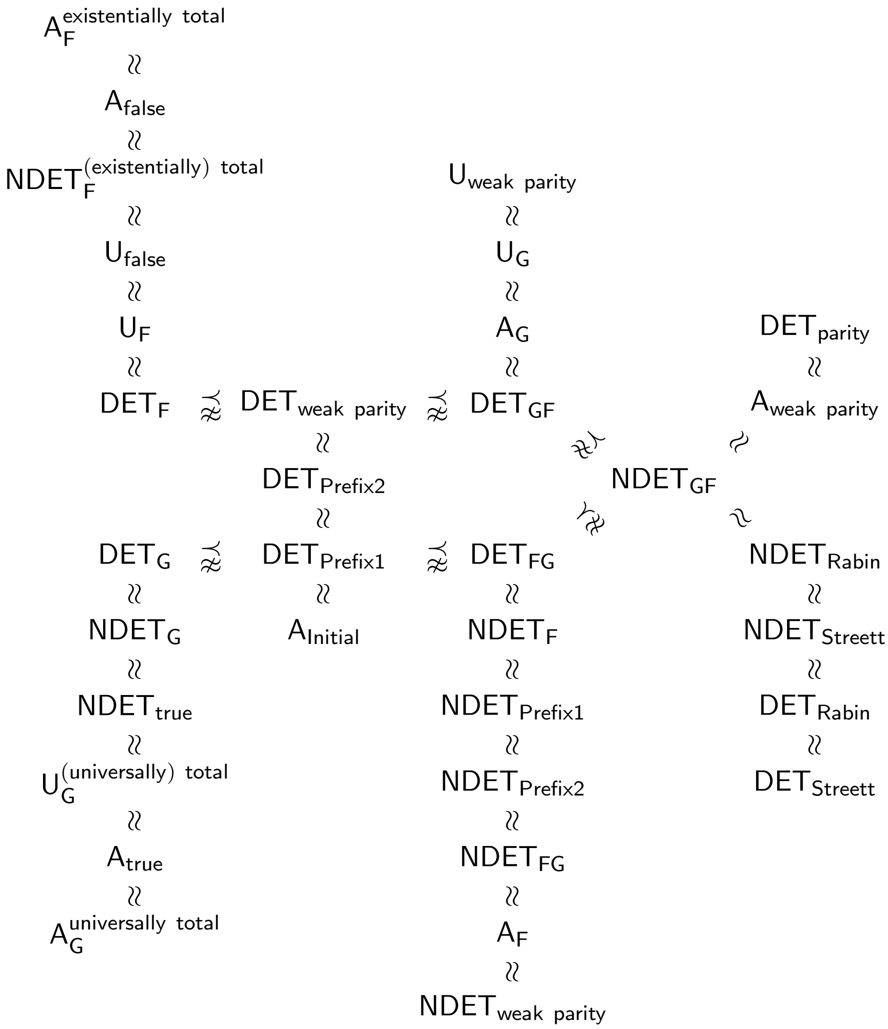

Figure 5: Extended Hierarchy of ω-Automata

5 Conclusions and Future Work

In this paper, we investigated the relationship between alternating and symbolically represented nondeterministic automata. There is a well known strong relationship between alternating automata and symbolically represented nondeterministic automata for finite words. One can even regard alternating automata on finite words as a special normal form of symbolically represented nondeterministic automata on finite words.

In this work, we have been able to lift this strong relationship to alternating automata on infinite words, that represent safety properties. Although this class is one of the weakest classes in the presented hierarchy, the results are nevertheless of practical relevance, because safety properties are widely used. Moreover, the results can be used to handle liveness properties, too. Additionally, we could use the results to extend the hierarchy of ω-automata. We identified classes of alternating ω-automata that are as expressive as DET_F, DET_G and $\mathsf{DET}_\mathsf{Prefix}$.

However, our results can not be used for other classes like for example A_G. During the nondeterminisation, similar problems as during determinisation occur. For simple classes of automata, the Rabin-Scott subset construction is sufficient for determinisation. This subset construction is similar to the concept of $\mathsf{SymNDET}_\rightarrow$ automata. For more expressive classes of ω-automata, more information about the individual runs is needed. Therefore, additional variables are needed to determinise for example NDET_F automata. However, is is sufficient to store information about the past of a path in these additional variables. For example, it is stored, whether a path has visited some set of accepted states yet or which states have been visited in the past. For alternating automata the problem is even more complicated. Also the future of a path through a run has to be considered, because paths may be accepted by the occurrence of true. A practical consequence is that NDet_G automata can be determinised using the Rabin-Scott subset construction, but $\mathsf{SymNDET}_\rightarrow$-automata can not be used to nondeterminise A_G automata. However, there is a well known nondeterminisation procedure for alternating Büchi automata [11, 14]. This nondeterminisation procedure can be used to translate Büchi automata with n states directly to a symbolically represented nondeterministic automaton with $2n$ state variables. This is sufficient for practice. However, in a future work, it would be interesting to investigate how much additional state variables are really needed and if this nondeterminisation procedure can be simplified. Thereby, one should try to preserve the structure of the original automaton as much as possible.

References

[1] C. Berthet, O. Coudert, and J.C. Madre. New ideas on symbolic manipulations of finite state machines. In *International Conference on Computer Design (ICCD)*, pages 224–227. IEEE, 1990.

[2] R.E. Bryant. Graph-based algorithms for Boolean function manipulation. *IEEE Transactions on Computers*, C-35(8):677–691, August 1986.

[3] J.R. Burch, E.M. Clarke, K.L. McMillan, and D.L. Dill. Sequential circuit verification using symbolic model checking. In *International Design Automation Conference (DAC)*, pages 46–51, Orlando, Florida, USA, 1990. IEEE.

[4] J.R. Burch, E.M. Clarke, K.L. McMillan, D.L. Dill, and L.J. Hwang. Symbolic model checking: 10^{20} states and beyond. In *Symposium on Logic in Computer Science (LICS)*, pages 1–33, Washington, D.C., June 1990. IEEE Computer Society.

[5] J.R. Burch, E.M. Clarke, K.L. McMillan, D.L. Dill, and L.J. Hwang. Symbolic model checking: 10^{20} states and beyond. *Information and Computation*, 98(2):142–170, June 1992.

[6] J.R. Büchi. On a decision method in restricted second order arithmetic. In E. Nagel, editor, *International Congress on Logic, Methodology and Philosophy of Science*, pages 1–12, Stanford, CA, 1960. Stanford University Press.

[7] J.R. Büchi. Weak second order arithmetic and finite automata. *Z. Math. Logik Grundlagen Math.*, 6:66–92, 1960.

[8] A.K. Chandra, D. Kozen, and L.J. Stockmeyer. Alternation. *Journal of the ACM*, 28(1):114–133, 1981.

[9] E.M. Clarke, E.A. Emerson, and A.P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 8(2):244–263, April 1986.

[10] S.C. Kleene. Representation of events in nerve nets and finite automata. In C. Shannon and J. McCarthy, editors, *Automata Studies*, pages 3–41. Princeton University Press, Princeton, NJ, 1956.

[11] O. Kupferman and M.Y. Vardi. Weak alternating automata are not that weak. In *Israeli Symposium on Theory of Computing and Systems*, pages 147–158. IEEE Computer Society, 1997.

[12] C. Löding and W. Thomas. Alternating automata and logics over infinite words. In *Conference on Theoretical Computer Science (TCS)*, volume 1872 of *LNCS*, pages 521–535. Springer, 2000.

[13] Z. Manna and A. Pnueli. *The temporal Logic of Reactive and Concurrent Systems.* Springer, 1992.

[14] S. Miyano and T. Hayashi. Alternating automata on ω-words. *Theoretical Computer Science*, 32:321–330, 1984.

[15] D.E. Muller and P.E. Schupp. Alternating automata on infinite trees. *Theoretical Computer Science*, 54:267–276, 1987.

[16] M.O. Rabin. Automata on infinite objects and Church's problem. In *Regional Conference Series in Mathematics*, volume 13. American Mathematical Society (AMS), 1972.

[17] K. Schneider. *Ein einheitlicher Ansatz zur Unterstützung von Abstraktionsmechanismen der Hardwareverifikation*, volume 116 of *DISKI (Dissertationen zur Künstlichen Intelligenz)*. Infix, Sankt Augustin, 1996. ISBN 3-89601-116-2.

[18] K. Schneider. Improving automata generation for linear temporal logic by considering the automata hierarchy. In *International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR)*, volume 2250 of *LNAI*, pages 39–54, Havanna, Cuba, 2001. Springer.

[19] K. Schneider. *Verification of Reactive Systems – Formal Methods and Algorithms.* Texts in Theoretical Computer Science (EATCS Series). Springer, 2003.

[20] R.S. Streett. Propositional dynamic logic of looping and converse is elementarily decidable. *Information and Control*, 54(1-2):121–141, 1982.

[21] T. Tuerk. A hierarchy for Accellera's property specification language. Master's thesis, University of Kaiserslautern, Department of Computer Science, 2005.

[22] S. Yu. *Regular Languages*, chapter 2, pages 41–110. Springer, 1997.