FI MU

Faculty of Informatics
Masaryk University Brno

Deeper Connections between LTL and Alternating Automata

by

Radek Pelánek
Jan Strejček

FI MU Report Series **FIMU-RS-2004-08**

 September 2004

Publications in the FI MU Report Series are in general accessible via WWW:

`http://www.fi.muni.cz/veda/reports/`

Further information can obtained by contacting:

Faculty of Informatics
Masaryk University
Botanická 68a
602 00 Brno
Czech Republic

Deeper Connections between LTL and Alternating Automata*

Radek Pelánek and Jan Strejček
Faculty of Informatics, Masaryk University
Botanická 68a, 602 00 Brno, Czech Republic
{xpelanek,strejcek}@fi.muni.cz

September 7, 2004

Abstract

It is known that Linear Temporal Logic (LTL) has the same expressive power as alternating 1-weak automata (A1W automata, also called alternating linear automata or very weak alternating automata). A translation of LTL formulae into a language equivalent A1W automata has been introduced in [MSS88]. The inverse translation has been developed independently in [Roh97] and [LT00]. We improve the latter translation and present deeper connections between LTL and A1W automata. Using these translations we identify the classes of A1W automata equivalent to LTL fragments given by bounds on nesting depths of temporal operators (see, e.g., [Wil99, KS02]) and the fragments of Until-Release hierarchy [ČP03].

1 Introduction

An automata-theoretic approach to the study of temporal logics proved to be very fruitful. The best example is the well known fact that each formula of Linear Temporal Logic (LTL) [Pnu77] can be transformed into nondeterministic Büchi automaton that accepts exactly the infinite words satisfying the formula [WVS83, VW94]. This transformation is one of the cornerstones of the automata-based model checking of LTL properties [VW86]. It is also known that there are Büchi automata accepting languages that are not definable by any LTL formula [Wol83].

*Supported by the Grant Agency of Czech Republic, grant No. 201/03/1161.

Later on, alternating 1-weak Büchi automata (or A1W automata for short, also known as alternating linear automata or very weak alternating automata) have been identified as the type of automata with the same expressive power as LTL. Muller, Saoudi, and Schupp [MSS88] introduced a translation of LTL formula into equivalent A1W automata. This translation produces A1W automata with number of states linear in the lentgh of input formula contrary to the mentioned translation into Büchi automata where the number of states is exponential. The translation of LTL into A1W automata has been recently used to built new and more efficient algorithms for translation of LTL into nondeterministic Büchi automata [GO01, Tau03][1]. The translation of A1W automata into equivalent LTL formula has been presented independently by Rohde [Roh97] and Löding and Thomas [LT00]. Hence, the two formalisms are equivalent.

Soon after introduction of LTL, the study of fragments of the logic started. It is often connected with searching for more efficient verification algorithm working with a fragment of LTL. In [Lam83] Lamport argue that for reasoning about concurrent system we should use the fragment of LTL without *next* operator rather than complete LTL. Many reduction methods improving the verification algorithms have been designed for the *next*-free fragment so far (see, for example, [Val91, God96, HP95, Pel98]). Besides fragments given by restricted set of temporal operators, the fragments given by bounds on nesting depths of temporal operators or alternation of temporal operators are studied as well (for more details see the surveys [Wil99, DS98, Sch03] or some recent papers like [KS02, ČP03, TW04]).

In the light of research on LTL fragments, it is natural to ask for classes of A1W automata with the same expressive power as the studied LTL fragments. It turns out that the translation of A1W automata into LTL mentioned above is not appropriate for study of such automata classes as it wastes temporal operators. For example, the automaton corresponding to the formula $a \mathsf{U} (b \wedge (b \mathsf{U} c))$ is translated into formula $a \mathsf{U} (b \wedge \mathsf{X}(b \mathsf{U} c))$. In our paper we present an improved translation of A1W automata into equivalent LTL formulae. Our translation reduces the nesting depth of *next* operators and prefers the use of less expressive temporal operators (*eventually* or *globally* instead of *until*). We prove that for an A1W automaton produced by the standard translation of an LTL formula φ our translation provides a formula with the same (or even

[1] In fact, the paper [GO01] employs alternating 1-weak co-Büchi automata. However, Büchi and co-Büchi acceptance conditions are expressively equivalent for alternating 1-weak automata.

less) nesting depths of *until*, *next*, and *eventually* operators comparing to these nesting depths in φ.

With use of this results we identify classes of A1W automata defining the same classes of languages as LTL fragments given by bounds on nesting depths of temporal operators. Further, with use of standard translations between LTL and A1W automata we identify the classes of A1W automata corresponding to the fragments of Until-Release hierarchy [ČP03].

The paper is structured as follows. In Section 2 we recall the definition of LTL and some of its fragments, and the definition of alternating 1-weak automata. Section 3 presents the standard translation of A1W automata into LTL formulae [MSS88, Var97] and translation of LTL formulae into A1W automata [Roh97, LT00]. Section 4 provides an improved version of the latter translation, proof of its correctness, and basic observations about relation between nesting depths of temporal operators in a formula and properties of the corresponding automaton. In Section 5 we define classes of A1W automata corresponding to the mentioned LTL fragments. Section 6 sums up presented results and indicates several topics for future work.

2 Preliminaries

The syntax of LTL is given by the abstract syntax equation

$$\varphi \ ::= \ \top \ \mid \ a \ \mid \ \neg\varphi \ \mid \ \varphi_1 \wedge \varphi_2 \ \mid \ \mathsf{X}\varphi \ \mid \ \mathsf{F}\varphi \ \mid \ \varphi_1 \,\mathsf{U}\, \varphi_2,$$

where a ranges over countable set $\Lambda = \{a, b, c, \ldots\}$ of *letters*. We also use $\bot$ to abbreviate $\neg\top$, $\mathsf{G}\varphi$ to abbreviate $\neg\mathsf{F}\neg\varphi$, and $\varphi \,\mathsf{R}\, \psi$ to abbreviate $\neg(\neg\varphi \,\mathsf{U}\, \neg\psi)$. The operators $\mathsf{X}, \mathsf{F}, \mathsf{U}, \mathsf{G}, \mathsf{R}$ are called *next*, *eventually*, *until*, *globally*, and *release*, respectively. Let us note that $\mathsf{F}\varphi$ can be equivalently defined as an abbreviation for $\top \,\mathsf{U}\, \varphi$.

We define the semantics of LTL in terms of languages over infinite words. An *alphabet* is a finite set $\Sigma \subseteq \Lambda$. An *ω-word* over alphabet Σ is an infinite sequence $w = w(0)w(1)w(2)\ldots \in \Sigma^\omega$ of letters from Σ. For every $i \in \mathbb{N}_0$, by w_i we denote the suffix of w beginning with the letter $w(i)$.

The *validity* of an LTL formula φ for $w \in \Sigma^\omega$ is defined as follows:

$w \models \top$

$w \models a \qquad \text{iff} \quad a = w(0)$

$$
\begin{array}{lll}
w \models \neg\varphi & \text{iff} & w \not\models \varphi \\
w \models \varphi_1 \wedge \varphi_2 & \text{iff} & w \models \varphi_1 \wedge w \models \varphi_2 \\
w \models \mathsf{X}\varphi & \text{iff} & w_1 \models \varphi \\
w \models \mathsf{F}\varphi & \text{iff} & \exists i \in \mathbb{N}_0 : w_i \models \varphi \\
w \models \varphi_1 \,\mathsf{U}\, \varphi_2 & \text{iff} & \exists i \in \mathbb{N}_0 : w_i \models \varphi_2 \wedge \forall\, 0 \le j < i : w_j \models \varphi_1
\end{array}
$$

For every alphabet Σ, an LTL formula φ defines the language $L^{\Sigma}(\varphi) = \{w \in \Sigma^{\omega} \mid w \models \varphi\}$.

Let us (inductively) define the *nesting depths* of the X, F and the U modalities in a given LTL formula φ.

$$
\begin{array}{lll}
\mathsf{X}\text{-}\mathit{depth}(\top) & = & 0 \\
\mathsf{X}\text{-}\mathit{depth}(a) & = & 0 \\
\mathsf{X}\text{-}\mathit{depth}(\neg\varphi) & = & \mathsf{X}\text{-}\mathit{depth}(\varphi) \\
\mathsf{X}\text{-}\mathit{depth}(\varphi_1 \wedge \varphi_2) & = & \max\{\mathsf{X}\text{-}\mathit{depth}(\varphi_1), \mathsf{X}\text{-}\mathit{depth}(\varphi_2)\} \\
\mathsf{X}\text{-}\mathit{depth}(\mathsf{X}\varphi) & = & \mathsf{X}\text{-}\mathit{depth}(\varphi) + 1 \\
\mathsf{X}\text{-}\mathit{depth}(\mathsf{F}\varphi) & = & \mathsf{X}\text{-}\mathit{depth}(\varphi) \\
\mathsf{X}\text{-}\mathit{depth}(\varphi_1 \,\mathsf{U}\, \varphi_2) & = & \max\{\mathsf{X}\text{-}\mathit{depth}(\varphi_1), \mathsf{X}\text{-}\mathit{depth}(\varphi_2)\}
\end{array}
$$

$$
\begin{array}{lll}
\mathsf{F}\text{-}\mathit{depth}(\top) & = & 0 \\
\mathsf{F}\text{-}\mathit{depth}(a) & = & 0 \\
\mathsf{F}\text{-}\mathit{depth}(\neg\varphi) & = & \mathsf{F}\text{-}\mathit{depth}(\varphi) \\
\mathsf{F}\text{-}\mathit{depth}(\varphi_1 \wedge \varphi_2) & = & \max\{\mathsf{F}\text{-}\mathit{depth}(\varphi_1), \mathsf{F}\text{-}\mathit{depth}(\varphi_2)\} \\
\mathsf{F}\text{-}\mathit{depth}(\mathsf{X}\varphi) & = & \mathsf{F}\text{-}\mathit{depth}(\varphi) \\
\mathsf{F}\text{-}\mathit{depth}(\mathsf{F}\varphi) & = & \mathsf{F}\text{-}\mathit{depth}(\varphi) + 1 \\
\mathsf{F}\text{-}\mathit{depth}(\varphi_1 \,\mathsf{U}\, \varphi_2) & = & \max\{\mathsf{F}\text{-}\mathit{depth}(\varphi_1), \mathsf{F}\text{-}\mathit{depth}(\varphi_2)\}
\end{array}
$$

$$
\begin{array}{lll}
\mathsf{U}\text{-}\mathit{depth}(\top) & = & 0 \\
\mathsf{U}\text{-}\mathit{depth}(a) & = & 0 \\
\mathsf{U}\text{-}\mathit{depth}(\neg\varphi) & = & \mathsf{U}\text{-}\mathit{depth}(\varphi) \\
\mathsf{U}\text{-}\mathit{depth}(\varphi_1 \wedge \varphi_2) & = & \max\{\mathsf{U}\text{-}\mathit{depth}(\varphi_1), \mathsf{U}\text{-}\mathit{depth}(\varphi_2)\} \\
\mathsf{U}\text{-}\mathit{depth}(\mathsf{X}\varphi) & = & \mathsf{U}\text{-}\mathit{depth}(\varphi) \\
\mathsf{U}\text{-}\mathit{depth}(\mathsf{F}\varphi) & = & \mathsf{U}\text{-}\mathit{depth}(\varphi) \\
\mathsf{U}\text{-}\mathit{depth}(\varphi_1 \,\mathsf{U}\, \varphi_2) & = & \max\{\mathsf{U}\text{-}\mathit{depth}(\varphi_1), \mathsf{U}\text{-}\mathit{depth}(\varphi_2)\} + 1
\end{array}
$$

Now we define a notation of LTL fragments given by bounds on nesting depths of temporal operators. For all $m, n, k \in \mathbb{N}_0 \cup \{\infty\}$, by $\mathrm{LTL}(\mathsf{U}^m, \mathsf{X}^n, \mathsf{F}^k)$ we denote the set

$$\{\varphi \in \mathrm{LTL} \mid \mathsf{U}\text{-}\mathit{depth}(\varphi) \le m \text{ and } \mathsf{X}\text{-}\mathit{depth}(\varphi) \le n \text{ and } \mathsf{F}\text{-}\mathit{depth}(\varphi) \le k\}.$$

We omit upper indices equal to ∞ from LTL($\mathsf{U}^m, \mathsf{X}^n, \mathsf{F}^k$). Moreover, we usually omit the whole operator if its index is 0. For example, by LTL(X^n, F) we mean the fragment LTL($\mathsf{U}^0, \mathsf{X}^n, \mathsf{F}^\infty$).

Alternating automata are generalizations of nondeterministic ones. Transition function of an alternating automaton combines the nondeterministic and universal mode. More formally, transition function assigns to each state and letter a positive boolean formula over states. The set of *positive boolean formulae* over finite set of states Q (denoted $\mathcal{B}^+(Q)$) contains formulae $\top$ (true), $\perp$ (false), all elements of Q and boolean combinations over Q built with $\wedge$ and $\vee$. A subset S of Q is a *model* of $\varphi \in \mathcal{B}^+(Q)$ iff the valuation assigning true just to states in S satisfies φ. A set S is a *minimal model* of φ (denoted $S \models \varphi$) iff S is a model of φ and no proper subset of S is a model of φ.

An *alternating Büchi automaton* is a tuple $A = (\Sigma, Q, q_0, \delta, F)$, where Σ is a finite input alphabet, Q is a finite set of states, $q_0 \in Q$ is an initial state, $\delta : Q \times \Sigma \to \mathcal{B}^+(Q)$ is a transition function, and $F \subseteq Q$ is a set of accepting states. By $A(p)$ we denote the automaton A with initial state $p \in Q$ instead of q_0.

A run of an alternating automaton is a (potentially infinite) tree. A *tree* is a set $T \subset \mathbb{N}^*$ such that if $xc \in T$, where $x \in \mathbb{N}^*$ and $c \in \mathbb{N}$, then also $x \in T$ and $xc' \in T$ for all $0 \leq c' < c$. A Q-*labeled tree* is a pair (T, r) where T is a tree and $r : T \to Q$ is a labeling function. A *run* of an automaton $A = (\Sigma, Q, q_0, \delta, F)$ over ω-word $w \in \Sigma^\omega$ is a Q-labeled tree (T, r) with the following properties:

1. $r(\epsilon) = q_0$.

2. For each $x \in T$ the set $S = \{r(xc) \mid xc \in T\}$ satisfies $S \models \delta(r(x), w(|x|))$.

A run (T, r) is *accepting* iff for every infinite path π in T it holds that $\mathit{Inf}(\pi) \cap F \neq \emptyset$, where $\mathit{Inf}(\pi)$ is a set of all labels (i.e. states) appearing infinitely often on π. An automaton A *accepts* an ω-word $w \in \Sigma^\omega$ iff there exists an accepting run of A over w. A language of all ω-words accepted by an automaton A is denoted by $L(A)$.

Let $\mathit{Succ}(p)$ denotes the set $\mathit{Succ}(p) = \{q \mid \exists a \in \Sigma, S \subseteq Q : S \cup \{q\} \models \delta(p, a)\}$ of all possible successors of p, and $\mathit{Succ}'(p) = \mathit{Succ}(p) \setminus \{p\}$. An automaton is called *1-weak* if there exists an ordering $<$ on the set of states Q such that $q \in \mathit{Succ}'(p)$ implies $q < p$. In the following we use *A1W automaton* or simply *automaton* meaning "alternating 1-weak Büchi automaton".

An A1W automaton $A = (\Sigma, Q, q_0, \delta, F)$ can be drawn as a graph; nodes are the states of the automaton and every $S \subseteq Q$ satisfying $S \models \delta(p, a)$ (where $p \in Q$ and $a \in \Sigma$) is

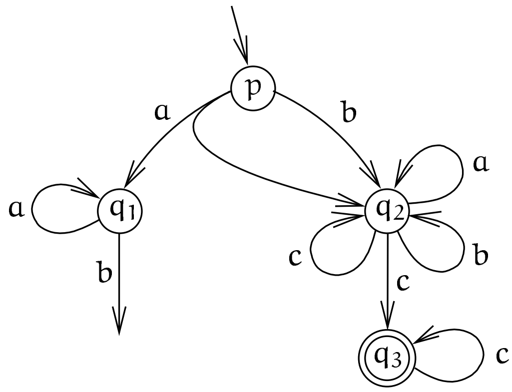

Figure 1: The automaton accepting the language $a^*b\{a, b, c\}^*c^\omega$

.

depicted as a branching edge labelled with a and leading from node p to the nodes from S. Edges that are not leading to any node correspond to the cases when S is an empty set. An initial state is indicated by a special unlabelled edge leading to the corresponding node. Nodes corresponding to the accepting states are double-circled. For example, the Figure 1 depicts an automaton accepting the language $a^*b\{a, b, c\}^*c^\omega$. Instead of $S \models \delta(a, p)$ we write $p \xrightarrow{a} S$ and say that an automaton has a transition leading from p to S under a. A state p of an automaton has a *loop* whenever $p \in \mathit{Succ}(p)$.

3 Standard translations between LTL and A1W automata

In this section we recall the standard translation of LTL formulae to A1W automata [MSS88] and (a variant of) a translation of A1W automata to LTL presented recently in [LT00]. The translation of A1W automata to LTL has been independently introduced by Rohde in [Roh97]. In this section we treat every (sub)formula of the form $\mathsf{F}\varphi$ as an abbreaviation for $\top \mathsf{U} \varphi$.

3.1 LTL→A1W translation

Let φ be an LTL formula and Σ an alphabet. The formula can be translated into an automaton A satisfying $L(A) = L^\Sigma(\varphi)$, where $A = (\Sigma, Q, q_\varphi, \delta, F)$ and

- the states $Q = \{q_\psi, q_{\neg\psi} \mid \psi \text{ is a subformula of } \varphi\}$ correspond to the subformulae of φ and their negations,

- the transition function δ is defined inductively in the following way:

$$\begin{aligned}
&\delta(q_\top, a) = \top \\
&\delta(q_a, b) = \top \text{ if } a = b, \delta(q_a, b) = \bot \text{ otherwise} \\
&\delta(q_{\neg\psi}, a) = \overline{\delta(q_\psi, a)} \\
&\delta(q_{\psi\wedge\rho}, a) = \delta(q_\psi, a) \wedge \delta(q_\rho, a) \\
&\delta(q_{\mathsf{X}\psi}, a) = q_\psi \\
&\delta(q_{\psi\mathsf{U}\rho}, a) = \delta(q_\rho, a) \vee (\delta(q_\psi, a) \wedge q_{\psi\mathsf{U}\rho})
\end{aligned}$$

  where $\overline{\alpha}$ denotes the positive boolean formula dual to α defined by induction on the structure of α as follows:

$$\begin{array}{lll}
\overline{\top} = \bot & \overline{q_{\neg\psi}} = q_\psi & \overline{\beta \wedge \gamma} = \overline{\beta} \wedge \overline{\gamma} \\
\overline{\bot} = \top & \overline{q_\psi} = q_{\neg\psi} & \overline{\beta \vee \gamma} = \overline{\beta} \vee \overline{\gamma}
\end{array}$$

- the set of accepting states $F = \{q_{\neg(\psi\mathsf{U}\rho)} \mid \psi\,\mathsf{U}\,\rho \text{ is a subformula of } \varphi\}$.

We use the notation $A^\Sigma(\varphi)$ for the automaton given by the translation of an LTL formula φ with respect to an alphabet Σ. The number of states of the automaton $A^\Sigma(\varphi)$ is clearly linear in the length of φ.

For example, the translation applied on the formula $\varphi = (a\,\mathsf{U}\,b) \wedge \mathsf{FG}c$ and the alphabet $\Sigma = \{a, b, c\}$ produces the automaton depicted on Figure 1, where p, q_1, q_2, q_3 stand for $q_\varphi, q_{a\mathsf{U}b}, q_{\mathsf{FG}c}, q_{\mathsf{G}c}$, respectively.

3.2 A1W→LTL translation

Let $A = (\Sigma, Q, q_0, \delta, F)$ be an A1W automaton. For each $p \in Q$ we define an LTL formula φ_p such that $L^\Sigma(\varphi_p) = L(A(p))$ (in particular $L^\Sigma(\varphi_{q_0}) = L(A)$). The definition proceeds by induction respecting the ordering of states; the formula φ_p employs formulae of the form φ_q where $q \in \mathit{Succ}'(p)$. This is the point where the 1-weakness of the automaton is used. To illustrate the inductive step of the translation, let us consider the situation depicted on Figure 2. The formula corresponding to state p is $\varphi_p = (a \wedge \mathsf{X}\varphi_q)\,\mathsf{U}\,(b \wedge \mathsf{X}\varphi_r)$.

Before we give a formal definition of φ_p, we introduce some auxiliary formulae. Let $a \in \Sigma$ be a letter and $S \subseteq Q$ be a set of states. The formula

$$\theta(a, S) = a \wedge \bigwedge_{q\in S} \mathsf{X}\varphi_q$$

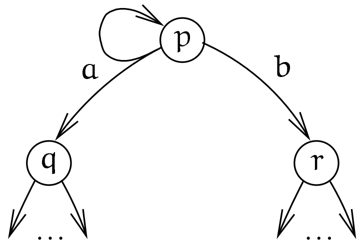

Figure 2: Part of an automaton translated into the formula $\varphi_p = (a \wedge X\varphi_q) \, U \, (b \wedge X\varphi_r)$.

intuitively corresponds to a situation when automaton makes transition under a into the set of states S. Formulae α_p and β_p defined as

$$\alpha_p = \bigvee_{\substack{p \xrightarrow{a} S \\ p \in S}} \theta(a, S \smallsetminus \{p\}) \qquad\qquad \beta_p = \bigvee_{\substack{p \xrightarrow{a} S \\ p \notin S}} \theta(a, S)$$

intuitively correspond to all transitions leading from state p; α_p covers transitions with a loop (i.e., the transitions leading to a set of states containing p) while β_p cover the others. The definition of φ_p then depends on whether p is an accepting state or not.

$$\varphi_p = \begin{cases} \alpha_p \, U \, \beta_p & \text{if } p \notin F \\ (\alpha_p \, U \, \beta_p) \vee G\alpha_p & \text{if } p \in F \end{cases}$$

The proof of the correctness of this translation can be found in [LT00]. Given an A1W automaton A with an initial state q_0, by $\varphi(A)$ we denote the formula $\varphi(A) = \varphi_{q_0}$.

4 Improved translation

The A1W→LTL translation presented in the previous section wastes temporal operators. The main problem of the translation is that for each successor $q \in \mathit{Succ}'(p)$ of a state p the formula φ_p contains a subformula $X\varphi_q$ even if the X operator is not needed. This can be illustrated by an automaton A on Figure 3.
The automaton is an automaton for a formula $a \, U \, (b \wedge (b \, U \, c))$. The A1W→LTL translation provides an equivalent formula $\varphi(A) = a \, U \, (b \wedge X(b \, U \, c))$ in spite of it.

Let $p \xrightarrow{a} S$ be a transition and $X \subseteq S$. We now formulate conditions that are sufficient to omit the X operator in front of φ_q (for every $q \in X$) in a subformula of φ_p corresponding to the transition $p \xrightarrow{a} S$. A set X satisfying these conditions is called X-*free*.

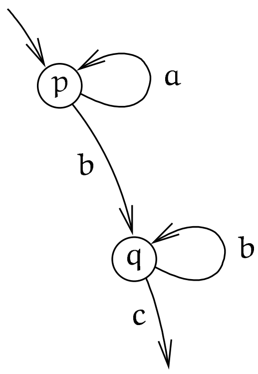

Figure 3: An automaton for the formula $a \mathsf{U} (b \wedge (b \mathsf{U} c))$.

Definition 4.1. *Let* $p \xrightarrow{a} S$ *be a transition of an automaton* $\mathcal{A}$. *A set* $X \subseteq S \setminus \{p\}$ *is said to be* X-free for $p \xrightarrow{a} S$ *if following conditions hold.*

1. *For each* $q \in X$ *there is* $S'_q \subseteq S$ *such that* $q \xrightarrow{a} S'_q$.
2. *Let* $Y \subseteq X$ *and for each* $q \in Y$ *let* $S'_q \subseteq Q$ *be a set satisfying* $q \xrightarrow{a} S'_q$ *and* $q \notin S'_q$. *Then there exists a set* $S'' \subseteq (S \setminus Y) \cup \bigcup_{q \in Y} S'_q$ *satisfying* $p \xrightarrow{a} S''$.

Figure 4 illustrates the conditions for X-freeness. Please note that it can be the case that $p \in S$. Further, in the first condition it can be the case that $q \in S'_q$.

It is easy to see that empty set is X-free for every transition. Further, every subset of a X-free set for a transition is a X-free set for the transition as well. On the other hand, Figure 5 demonstrates that the union of two X-free sets need not be X-free; in the automaton indicated on the figure, the sets $\{q_1\}, \{q_2\}$ are X-free for $p \xrightarrow{a} \{q_1, q_2\}$ while the set $\{q_1, q_2\}$ is not.

Let Xfree be an arbitrary but fixed function assigning to each transition $p \xrightarrow{a} S$ a set that is X-free for $p \xrightarrow{a} S$. We now introduce an improved A1W→LTL translation. Roughly speaking, the translation omits the X operators in front of subformulae which correspond to the states in X-free sets given by the function Xfree. Thereafter we prove that this translation remains correct.

The improved A1W→LTL translation exhibits similar structure as the original one. Instead of formulae of the form $\theta(a, S)$ representing a transition under a leading from an arbitrary state p to S, we define a specialized formula $\theta'_p(a, S)$ for each transition $p \xrightarrow{a} S$.

$$\theta'_p(a, S) = a \wedge \bigwedge_{\substack{q \in S \setminus \mathsf{Xfree}(p \xrightarrow{a} S) \\ q \neq p}} \mathsf{X}\varphi'_q \wedge \bigwedge_{q \in \mathsf{Xfree}(p \xrightarrow{a} S)} \varphi'_q$$

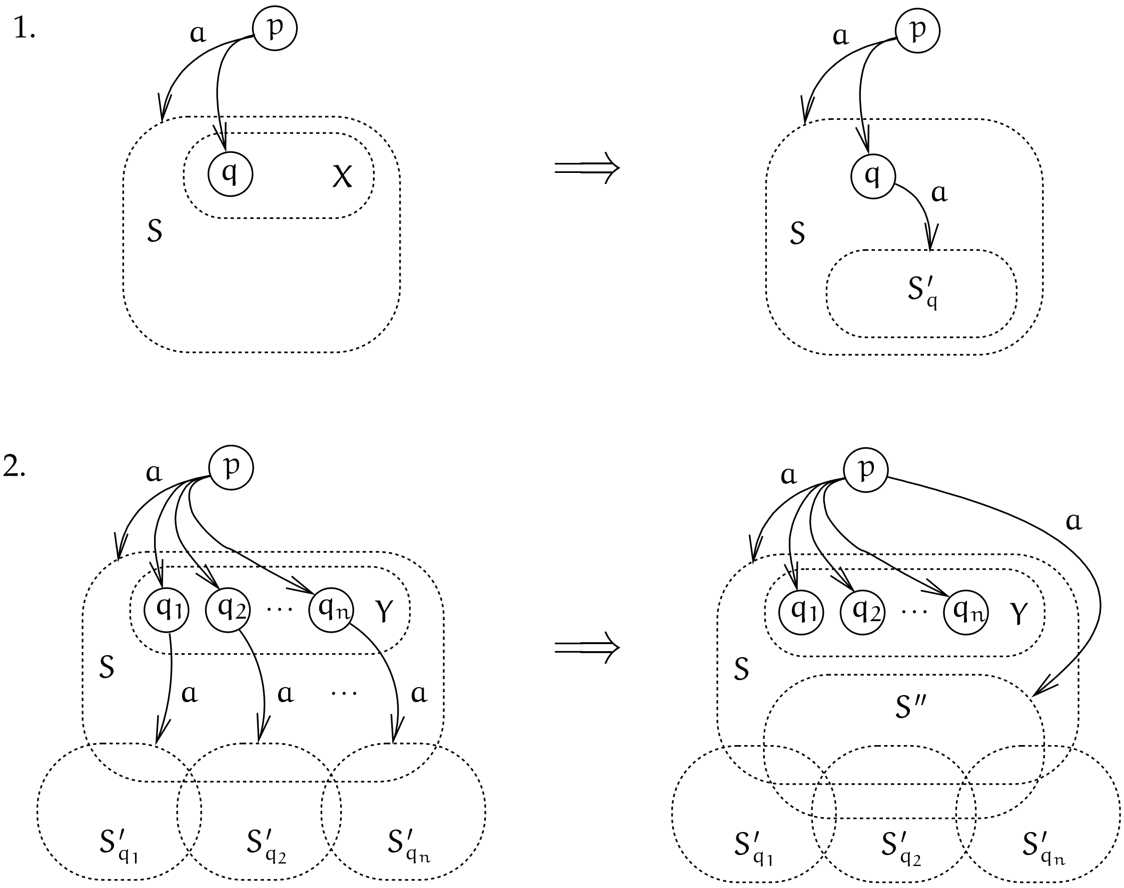

Figure 4: The conditions for X-freeness.

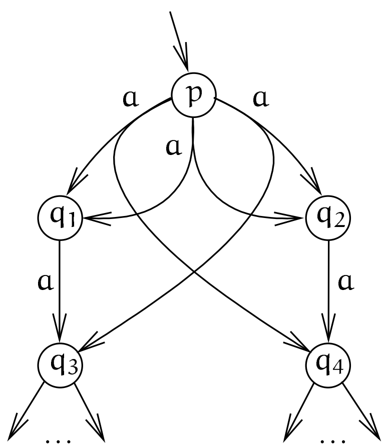

Figure 5: The sets $\{q_1\}, \{q_2\}$ are X-free for $p \xrightarrow{a} \{q_1, q_2\}$ while the set $\{q_1, q_2\}$ is not.

$$\alpha'_p = \bigvee_{\substack{p \xrightarrow{a} S \\ p \in S}} \theta'_p(a, S) \qquad\qquad \beta'_p = \bigvee_{\substack{p \xrightarrow{a} S \\ p \notin S}} \theta'_p(a, S)$$

In the following definition of a formula φ'_p we identify some cases when U can be replaced by "weaker" operators F or G. To this end we define two special types of states. A state p is of the F-*type* if there is a transition $p \xrightarrow{a} \{p\}$ for every $a \in \Sigma$. A state p is of the G-*type* if every transition of the form $p \xrightarrow{a} S$ satisfies $p \in S$.

$$\varphi'_p = \begin{cases} \beta'_p & \text{if } p \notin \mathit{Succ}(p) \\ \bot & \text{if } p \in \mathit{Succ}(p),\ p \notin F,\ p \text{ is of } \mathsf{G}\text{-type} \\ \mathsf{F}\beta'_p & \text{if } p \in \mathit{Succ}(p),\ p \notin F,\ p \text{ is of } \mathsf{F}\text{-type and not of } \mathsf{G}\text{-type} \\ \alpha'_p \,\mathsf{U}\, \beta'_p & \text{if } p \in \mathit{Succ}(p),\ p \notin F,\ p \text{ is neither of } \mathsf{F}\text{-type nor of } \mathsf{G}\text{-type} \\ \top & \text{if } p \in \mathit{Succ}(p),\ p \in F,\ p \text{ is of } \mathsf{F}\text{-type} \\ \mathsf{G}\alpha'_p & \text{if } p \in \mathit{Succ}(p),\ p \in F,\ p \text{ is of } \mathsf{G}\text{-type and not of } \mathsf{F}\text{-type} \\ (\alpha'_p \,\mathsf{U}\, \beta'_p) \vee \mathsf{G}\alpha'_p & \text{if } p \in \mathit{Succ}(p),\ p \in F,\ p \text{ is neither of } \mathsf{F}\text{-type nor of } \mathsf{G}\text{-type} \end{cases}$$

The new cases in the definition of φ'_p make only a cosmetic change comparing to the original A1W→LTL translation. First, we add a case for states without any loop. This change does not influence the correctness of the translation as the condition $p \notin \mathit{Succ}(p)$ says that $\alpha'_p = \bot$ and therefore $\varphi'_p = \beta_p$ is equivalent to $\alpha'_p \,\mathsf{U}\, \beta'_p$ as well as to $(\alpha'_p \,\mathsf{U}\, \beta'_p) \vee \mathsf{G}\alpha'_p$. Further, it is easy to check that if a state p is of F-type then $\alpha'_p \iff \top$ and if p is of G-type then $\beta'_p = \bot$. Hence, all cases for $p \in \mathit{Succ}(p)$ and $p \notin F$ are equivalent to $\alpha'_p \,\mathsf{U}\, \beta'_p$ and all cases for $p \in \mathit{Succ}(p)$ and $p \in F$ are equivalent to $(\alpha'_p \,\mathsf{U}\, \beta'_p) \vee \mathsf{G}\alpha'_p$.

Before we show that the improved translation is equivalent to the original one (and thus also correct), we prove two auxiliary lemmata.

Lemma 4.2. *Let $p \xrightarrow{a} S$ be a transition of an A1W automaton A. If $\varphi_q \iff \varphi'_q$ for each $q \in \mathit{Succ}'(p)$ then the implication $\theta(a, S \smallsetminus \{p\}) \implies \theta'_p(a, S)$ holds.*

Proof. Due to the definitions of $\theta(a, S \smallsetminus \{p\})$ and $\theta'_p(a, S)$ and the assumption of the lemma, it is sufficient to show for each ω-word $w \in \Sigma^\omega$ that

$$\text{if } q \in \mathsf{Xfree}(p \xrightarrow{a} S) \text{ and } w \models \theta(a, S \smallsetminus \{p\}) \text{ then } w \models \varphi_q.$$

The first condition for X-freeness gives us that there is $S'_q \subseteq S$ such that $q \xrightarrow{a} S'_q$. From the 1-weakness of the automaton we have that $p \notin S'_q$. Thus, $S'_q \subseteq S \smallsetminus \{p\}$ and $w \models \theta(a, S \smallsetminus \{p\})$ implies $w \models \theta(a, S'_q \smallsetminus \{q\})$. As $w \models \theta(a, S'_q \smallsetminus \{q\})$ and $q \xrightarrow{a} S'_q$ we have that either $q \in S'_q$ and then $w \models \alpha_q$, or $q \notin S'_q$ and then $w \models \beta_q$. Anyway, $w \models \alpha_q \vee \beta_q$ holds. At the same time $w \models \theta(a, S \setminus \{p\})$ implies $w \models \mathsf{X}\varphi_q$. We are done as $w \models \alpha_q \vee \beta_q$ and $w \models \mathsf{X}\varphi_q$ imply $w \models \varphi_q$. $\square$

Lemma 4.3. *Let $p \xrightarrow{a} S$ be a transition of an A1W automaton A such that for each $q \in \mathit{Succ}'(p)$ the equivalence $\varphi_q \iff \varphi'_q$ holds. Then $\theta'_p(a, S) \implies \beta_p \vee \alpha_p$. Moreover, if $p \notin S$ then $\theta'_p(a, S) \implies \beta_p$.*

Proof. Let us suppose that $w \in \Sigma^\omega$ is an ω-word such that $w \models \theta'_p(a, S)$. Due to the assumption of the lemma the formula $\theta'_p(a, S)$ is equivalent to

$$a \wedge \bigwedge_{\substack{q \in S \setminus \mathsf{Xfree}(p \xrightarrow{a} S) \\ q \neq p}} \mathsf{X}\varphi_q \wedge \bigwedge_{q \in \mathsf{Xfree}(p \xrightarrow{a} S)} \varphi_q .$$

Obviously $\varphi_q \implies \beta_q \vee \mathsf{X}\varphi_q$. Let $Y = \{q \in \mathsf{Xfree}(p \xrightarrow{a} S) \mid w \models \beta_q\}$. For each $q \in Y$, the definition of the formula β_q and the assumption $w \models a$ (due to $w \models \theta'_p(a, S)$) imply that there exists a set S'_q such that $q \xrightarrow{a} S'_q$, $q \notin S'_q$, and $w \models \theta(a, S'_q)$. The second condition for X-freeness gives us that there exists a set $S'' \subseteq (S \setminus Y) \cup \bigcup_{q \in Y} S'_q$ satisfying $p \xrightarrow{a} S''$. As $w \models \mathsf{X}\varphi_q$ for every $q \in S \setminus Y$ and $w \models \theta(a, S'_q)$ for each $q \in Y$, we get that $w \models \theta(a, S'' \setminus \{p\})$ as well. To sum up, we have a set S'' such that $p \xrightarrow{a} S''$ and $w \models \theta(a, S'' \setminus \{p\})$. If $p \in S''$ then $w \models \alpha_p$. Moreover, 1-weakness of the automaton implies that $p \notin S'_q$ and therefore $p \in S$. Finally, if $p \notin S''$ then $w \models \beta_p$. $\square$

We are now ready to prove the correctness of the improved translation.

Theorem 4.4. *Let $A = (\Sigma, Q, q_0, \delta, F)$ be an A1W automaton. For every state $p \in Q$ the equation $L(A(p)) = L^\Sigma(\varphi'_p)$ holds.*

Proof. It is sufficient to show that $\varphi_p \iff \varphi'_p$ holds for every $p \in Q$. The proof proceeds by induction with respect to the ordering on Q. If $\mathit{Succ}'(p) = \emptyset$ then an empty set is the only X-free set for any transition leading from p. Hence φ_p and φ'_p are equivalent.

Let us now assume that the equivalence holds for every $q \in \mathit{Succ}'(p)$. The Lemma 4.2 implies $\varphi_p \implies \varphi'_p$. Lemma 4.3 gives us that β'_p implies β_p, and α'_p implies $\beta_p \vee \alpha_p$. As an immediate consequence we get $\varphi'_p \implies \varphi_p$. $\square$

Let A be an A1W automaton and q_0 its initial state. By $\varphi^{\mathsf{Xfree}}(A)$ we denote the formula φ'_{q_0} given by the improved translation using the function Xfree.

After it has been proven that the improved translation remains correct, it is only natural to examine the "quality" of formulae it produces. The improved translation has been motivated by the observation that the standard one wastes X operators. Therefore, we show that the improved translation allows to translate an automaton $A^{\Sigma}(\varphi)$ derived from a formula $\varphi \in \mathrm{LTL}(\mathsf{U}^m, \mathsf{X}^n)$ back into a formula from $\mathrm{LTL}(\mathsf{U}^m, \mathsf{X}^n)$.[2] In order to do so, we define two metrics for A1W automata, namely *loop-height* and X-*height*, and show that an automaton A with loop-height m and X-height n can be translated into a formula form $\mathrm{LTL}(\mathsf{U}^m, \mathsf{X}^n)$. Then we prove that an automaton $A^{\Sigma}(\varphi)$ given by the standard LTL→A1W translation of a formula $\varphi \in \mathrm{LTL}(\mathsf{U}^m, \mathsf{X}^n)$ has loop-hieght and X-height at most m and n, respectively.

Definition 4.5. *Let $A = (\Sigma, Q, q_0, \delta, F)$ be an A1W automaton. For each state $p \in Q$ we inductively define its* loop-height *and* X-height *(denoted by $lh(p)$ and $\mathsf{X}h(p)$ respectively) as*

$$lh(p) = \begin{cases} \max\{lh(q) \mid q \in Succ'(p)\} + 1 & \textit{if } p \in Succ(p), \\ \max\{lh(q) \mid q \in Succ'(p)\} & \textit{otherwise}, \end{cases}$$

$$\mathsf{X}h(p) = \max_{p \xrightarrow{a} S} \{ \min_{X \textit{ is } X\textit{-free for } p \xrightarrow{a} S} \{need\mathsf{X}(p \xrightarrow{a} S, X)\} \},$$

where maximum over empty set is 0 *and*

$$need\mathsf{X}(p \xrightarrow{a} S, X) = \max(\{\mathsf{X}h(q) \mid q \in X\} \cup \{\mathsf{X}h(q) + 1 \mid q \in S \smallsetminus X, q \neq p\}).$$

We also define loop-height *and* X-height *of the automaton A as the* loop-height *and* X-height *of its initial state, i.e., $lh(A) = lh(q_0)$ and $\mathsf{X}h(A) = \mathsf{X}h(q_0)$.*

Intuitively, loop-height of an automaton A holds the maximal height of states of A with a loop. The X-height counts the minimal nesting depth of X operators achievable by the improved translation; the definition consider the minimum over all choices of X-free sets.

Theorem 4.6. *Let A be an A1W automaton. There exists a function* Xfree *such that* $\varphi^{\mathsf{Xfree}}(A) \in \mathrm{LTL}(\mathsf{U}^{lh(A)}, \mathsf{X}^{\mathsf{X}h(A)})$.

Proof. Please note that U-*depth*$(\varphi^{\mathsf{Xfree}}(A))$ does not depend on the choice of the function Xfree. Contrary to the U-*depth*, the X-*depth* of the resulting formula depends on the function Xfree. The function Xfree satisfying X-*depth*$(\varphi^{\mathsf{Xfree}}(A)) = \mathsf{X}h(A)$ can be derived

[2] In the rest of this section, $\mathsf{F}\psi$ is seen as an abbreviation for $\top \,\mathsf{U}\, \psi$.

directly from the definition of X-height; for every transition $p \xrightarrow{a} S$ we set $\mathsf{Xfree}(p \xrightarrow{a} S) = X$, where X is a X-free set for $p \xrightarrow{a} S$ such that the value of $need\mathsf{X}(p \xrightarrow{a} S, X)$ is minimal. It is a straightforward observation that this function satisfies $\varphi^{\mathsf{Xfree}}(A) \in \mathrm{LTL}(\mathsf{U}^{lh(A)}, \mathsf{X}^{\mathsf{X}h(A)})$. $\square$

We should note that the bound on U-$depth(\varphi^{\mathsf{Xfree}}(A))$ given by $lh(A)$ is not tight. The structure of a formula φ'_p shows that there can be states with a loop that are translated into $\top$ or $\perp$ and thus they do not bring any new temporal operators. However, if we remove these states and all transitions of the form $p \xrightarrow{a} S$ such that S contains a state translated into $\perp$, we get an automaton A' that is language equivalent to the original one and U-$depth(\varphi^{\mathsf{Xfree}}(A')) = lh(A')$.

Theorem 4.7. *Let* $\varphi \in \mathrm{LTL}(\mathsf{U}^m, \mathsf{X}^n)$ *be a formula. Then* $lh(A^\Sigma(\varphi)) \leq m$ *and* $\mathsf{X}h(A^\Sigma(\varphi)) \leq n$ *for each alphabet* Σ*.*

Proof. Before we prove the inequalities, we examine the automaton $A^\Sigma(\varphi)$. Let $q_{\varphi'}$ be a state of $A^\Sigma(\varphi)$. States in $Succ(q_{\varphi'})$ are of the form

1. $q_{\psi \mathsf{U} \rho}$ or $q_{\neg(\psi \mathsf{U} \rho)}$, where $\psi \,\mathsf{U}\, \rho$ is such a subformula of φ' that is not in a scope of any X operator, or

2. q_ψ or $q_{\neg\psi}$, where $\mathsf{X}\psi$ is such a subformula of φ' that is not in a scope of any other X operator.

Let us note that some of the states can match both cases, e.g., a state $q_{\psi \mathsf{U} \rho} \in Succ(q_{(\psi \mathsf{U} \rho) \vee \mathsf{X}(\psi \mathsf{U} \rho)})$. Moreover, the definition of a transition function δ implies that only the states of the form $q_{\psi \mathsf{U} \rho}$ or $q_{\neg(\psi \mathsf{U} \rho)}$ can have a loop.

From the above observations and the definition of loop-height it directly follows that $lh(A^\Sigma(\varphi)) \leq \mathsf{U}\text{-}depth(\varphi) \leq m$.

Let $q_{\varphi'}$ be a state of the automaton and $Y \subseteq Succ'(q_{\varphi'})$ be a set of its successors of the first form (excluding these of both forms). In order to prove the second inequality of the theorem, we show that for every transition $q_{\varphi'} \xrightarrow{a} S$ the set $X_S = S \cap Y$ is X-free. The construction of an automaton $A^\Sigma(\varphi)$ implies that if a positive boolean formula $\delta(q_{\varphi'}, a)$ contains a state $q_{\psi \mathsf{U} \rho} \in Y$ then the state always occurs in the formula

$$\delta(q_{\psi \mathsf{U} \rho}, a) \;=\; \delta(q_\rho, a) \vee (\delta(q_\psi, a) \wedge q_{\psi \mathsf{U} \rho}).$$

Similarly, if $\delta(q_{\varphi'}, a)$ contains a state $q_{\neg(\psi \mathsf{U} \rho)} \in Y$ then the state always occurs in the formula

$$\overline{\delta(q_{\psi \mathsf{U} \rho}, a)} \;=\; \overline{\delta(q_\rho, a)} \wedge (\overline{\delta(q_\psi, a)} \vee q_{\neg(\psi \mathsf{U} \rho)}).$$

We show that the each set X_S defined above satisfies the conditions for X-freeness for all transitions $q_{\varphi'} \xrightarrow{a} S$.

1. Let $q_{\psi \mathsf{U} \rho} \in X_S$. The property of $\delta(q_{\varphi'}, a)$ mentioned above gives us that there is a set $S' \subseteq S$ such that $S' \models \delta(q_\rho, a)$ or $S' \models \delta(q_\psi, a) \wedge q_{\psi \mathsf{U} \rho}$. Anyway, $q_{\psi \mathsf{U} \rho} \xrightarrow{a} S'$. The argumentation for the case $q_{\neg(\psi \mathsf{U} \rho)} \in X_S$ is similar.

2. Let $Y \subseteq X_S$ be such a set that for every $q \in Y$ there is a transition $q \xrightarrow{a} S'_q$ such that $q \notin S'_q$. Thus if q is of the form $q_{\psi \mathsf{U} \rho}$, then $S'_q \models \delta(q_\rho, a)$. Otherwise, q is of the form $q_{\neg(\psi \mathsf{U} \rho)}$ and $S'_q \models \overline{\delta(q_\rho, a)} \wedge \overline{\delta(q_\psi, a)}$. Again, the property of $\delta(q_{\varphi'}, a)$ mentioned above (together with the fact that $\delta(q_{\varphi'}, a)$ is in positive normal form) implies that there is a set $S'' \subseteq (S \smallsetminus Y) \cup \bigcup_{q \in Y} S'_q$ satisfying $S'' \models \delta(q_{\varphi'}, a)$.

Proving the X-freeness of the considered sets we have demonstrated that the improved translation allows to omit the X operators in front of the subformulae corresponding to the successors of $q_{\varphi'}$ of the first form (and not of both forms). Let us recall that these successors correspond to the subformulae of the form $\psi \,\mathsf{U}\, \rho$ of the original formula φ' that are never in a scope of any X operator in φ'. Hence, the improved translation with use of the Xfree function assigning the X-free sets defined above produces the formula $\varphi^{\mathsf{Xfree}}(\mathcal{A}^\Sigma(\varphi))$ with at most the same X-*depth* as the original formula φ. We are done as $\mathsf{X}h(\mathcal{A}^\Sigma(\varphi))$ keeps the lowest X-*depth* achievable by the improved translation and thus $\mathsf{X}h(\mathcal{A}^\Sigma(\varphi)) \leq \mathsf{X}\text{-}depth(\varphi) \leq n$. □

Combining Theorem 4.6 and Theorem 4.7 we get the following two corollaries.

Corollary 4.8. *For each A1W automaton* $\mathcal{A} = (\Sigma, Q, q_0, \delta, F)$ *there exists a function* Xfree *such that* $lh(\mathcal{A}) \geq lh(\mathcal{A}^\Sigma(\varphi^{\mathsf{Xfree}}(\mathcal{A})))$ *and* $\mathsf{X}h(\mathcal{A}) \geq \mathsf{X}h(\mathcal{A}^\Sigma(\varphi^{\mathsf{Xfree}}(\mathcal{A})))$.

Corollary 4.9. *For each formula* $\varphi \in \mathrm{LTL}(\mathsf{U}^m, \mathsf{X}^n)$ *and each alphabet* Σ *there exists a function* Xfree *such that* $\varphi^{\mathsf{Xfree}}(\mathcal{A}^\Sigma(\varphi)) \in \mathrm{LTL}(\mathsf{U}^m, \mathsf{X}^n)$.

5 Defining LTL fragments via A1W automata

In this section we define classes of A1W automata matching fregments of the form $\mathrm{LTL}(\mathsf{U}^m, \mathsf{X}^n, \mathsf{F}^k)$, where $m, n, k \in \mathbb{N}_0 \cup \{\infty\}$, and LTL fragments from so-called *Until-Release hierarchy* [ČP03]. All these fragments are given by syntactical constraints on LTL formulae. Basically, the classes of A1W automata can be defined by constraints on transition function and by constraints on the set of accepting states.

In order to improve the presentation of the following results, we overload the notation of LTL fragments; we identify an LTL fragment $\mathcal{F}$ with a set

$$\{L^{\Sigma}(\varphi) \mid \varphi \in \mathcal{F} \text{ and } \Sigma \text{ is an alphabet}\},$$

i.e., with a set of languages defined by formulae from the fragment. The interpretation of $\mathcal{F}$ is always clearly determined by the context.

5.1 Fragments given by bounds on nesting depths of modalities

In order to identify classes of A1W automata matching all LTL fragments of the form LTL($\mathsf{U}^m, \mathsf{X}^n, \mathsf{F}^k$), we need to replace the bound on U-*depth*$(\varphi^{\mathsf{Xfree}}(A))$ given by the loop-height of A by bounds on U-*depth* and F-*depth* of the formula. Therefore we define another metrics called U-*height*. The definition of U-height reflects the structure of φ'_p.

Definition 5.1. *Let* $A = (\Sigma, Q, q_0, \delta, F)$ *be an A1W automaton. For each state* $p \in Q$ *we inductively define its* U-height*, written* $\mathsf{U}h(p)$*, as*

$$\mathsf{U}h(p) = \begin{cases} \max\{\mathsf{U}h(q) \mid q \in \mathit{Succ}'(p)\} + 1 & \text{if } p \in \mathit{Succ}(p) \text{ and} \\ & p \text{ is neither of } \mathsf{F}\text{-type nor of } \mathsf{G}\text{-type}, \\ \max\{\mathsf{U}h(q) \mid q \in \mathit{Succ}'(p)\} & \text{otherwise}, \end{cases}$$

where maximum over empty set is 0*. The* U-height *of the automaton* A *is then defined as the* U-height *of its initial state, i.e.* $\mathsf{U}h(A) = \mathsf{U}h(q_0)$*.*

We are now ready to define the classes of A1W automata matching LTL fragments of the form LTL($\mathsf{U}^m, \mathsf{X}^n, \mathsf{F}^k$). To shorten our notation, a class is formally defined as a set of languages accepted by A1W automata rather than a set of A1W automata.

Definition 5.2. *Let* $m, n, k \in \mathbb{N}_0 \cup \{\infty\}$*. We define* A1W$(m, n, k)$ *to be the set* $\{L(A) \mid A$ *is an A1W automaton and* $\mathsf{U}h(A) \leq m$, $\mathsf{X}h(A) \leq n$, $lh(A) \leq m+k\}$*.*

Lemma 5.3. *For all* $m, n, k \in \mathbb{N}_0 \cup \{\infty\}$ *it holds* LTL$(\mathsf{U}^m, \mathsf{X}^n, \mathsf{F}^k) =$ A1W(m, n, k)*.*

Proof. Let $\varphi \in$ LTL$(\mathsf{U}^m, \mathsf{X}^n, \mathsf{F}^k)$ and Σ be an alphabet. Theorem 4.7 implies that $\mathsf{X}h(A^{\Sigma}(\varphi)) \leq n$ and $lh(A^{\Sigma}(\varphi)) \leq m + k$. Further, every state of the automaton $A^{\Sigma}(\varphi)$ corresponding to a subformula $\mathsf{F}\psi$ has the form $q_{\top\mathsf{U}\psi}$ or $q_{\neg(\top\mathsf{U}\psi)}$. One can readily check that each state $q_{\top\mathsf{U}\psi}$ is of F-type and each state $q_{\neg(\top\mathsf{U}\psi)}$ is of G-type. Hence, these states do not increase the U-height of the automaton. We get $\mathsf{U}h(A^{\Sigma}(\varphi)) \leq m$ and thus $L^{\Sigma}(\varphi) \in$ A1W(m, n, k).

Let A be an A1W automaton such that $\mathsf{U}h(A) \leq m$, $\mathsf{X}h(A) \leq n$, and $lh(A) \leq m + k$. Theorem 4.6 says that the automaton can be translated into formula from $\mathrm{LTL}(\mathsf{U}^{m+k}, \mathsf{X}^n)$. As the definition of U-height reflects the structure of formulae φ'_p given by the improved translation it is easy to check that $\varphi^{\mathsf{Xfree}}(A) \in \mathrm{LTL}(\mathsf{U}^{\mathsf{U}h(A)}, \mathsf{X}^n, \mathsf{F}^{lh(A)})$. Moreover, arbitrary occurrence of F operator can be replaced by U operator. Hence, the automaton can be translated into a formula from $\mathrm{LTL}(\mathsf{U}^m, \mathsf{X}^n, \mathsf{F}^{lh(A)-m})$. Hence, $L(A) \in \mathrm{LTL}(\mathsf{U}^m, \mathsf{X}^n, \mathsf{F}^k)$. □

Let us emphasize that Lemma 5.3 covers some previously studied LTL fragments. For example, languages defined by $\mathrm{LTL}(\mathsf{U}^k, \mathsf{X}, \mathsf{F})$ fragment (fragments of this form constitute so-called *Until hierarchy* [TW96, EW00]) can be defined by A1W automata with U-height at most k and vice versa. In particular, a language can be expressed by a formula from the $\mathrm{LTL}(\mathsf{X}, \mathsf{F})$ fragment (also called *Restricted LTL* [PP04]) if and only if it is recognized by an A1W automaton such that every state with a loop is of F-type or G-type.

5.2 Until-Release hierarchy

Until-Release heirarchy of LTL formulae has been introduced in [ČP03]. It is based on alternation depth of U and R operators. Therefore it is also called *alternating hierarchy*. The hierarchy has a strong connection to the hierarchy of temporal properties introduced by Manna and Pnueli [MP90, CMP92]. Moreover, the classes of alternating hierarchy reflects the complexity of their verification problem (for more information see [ČP03]).

Definition 5.4. *The classes $\Sigma_i^{LTL}, \Pi_i^{LTL}$ of the Until-Release hierarchy are defined inductively.*

- *The classes Σ_0^{LTL} and Π_0^{LTL} are both identical to* $\mathrm{LTL}(\mathsf{X})$.
- *The class Σ_{i+1}^{LTL} is the least set containing Π_i^{LTL} and closed under the application of operators $\wedge, \vee, \mathsf{X}$, and U.*
- *The class Π_{i+1}^{LTL} is the least set containing Σ_i^{LTL} and closed under the application of operators $\wedge, \vee, \mathsf{X}$, and R.*

Let us note that the hierarchy collapses on third level with respect to its expressive power. More precisely, each language is definable by LTL if and only if it is definable by a positive boolean combination of Σ_2^{LTL} and Π_2^{LTL} formulae. These formulae are contained in Σ_3^{LTL} as well as in Π_3^{LTL}. Again, we identify a fragment from the alternating hierarchy with the set of languages defined by formulae of the fragment.

The hierarchy does not care about X operators as well as the different expressiveness of F and U operators. Therefore we employ the standard translations given in Section 3.

Intuitively, we show that alternation of U and R operators in a formula corresponds to the alternation of nonaccepting and accepting states in the structure of an A1W automaton.

Definition 5.5. *Let* $A = (\Sigma, Q, q_0, \delta, F)$ *be an A1W automaton. For each* $i \in \mathbb{N}_0$ *we inductively define sets of states* σ_i *and* π_i *as follows.*

- $\sigma_0 = \pi_0 = \{p \mid lh(p) = 0\}$.
- σ_{i+1} *is the smallest set of states satisfying*
  - $\sigma_i \cup \pi_i \subseteq \sigma_{i+1}$ *and*
  - *if* $p \notin F$ *and* $Succ'(p) \subseteq \sigma_{i+1}$ *then* $p \in \sigma_{i+1}$,
- π_{i+1} *is the smallest set of states satisfying*
  - $\sigma_i \cup \pi_i \subseteq \pi_{i+1}$ *and*
  - *if* $p \in F$ *and* $Succ'(p) \subseteq \pi_{i+1}$ *then* $p \in \pi_{i+1}$.

We also define functions $\sigma_A, \pi_A : Q \longrightarrow \mathbb{N}_0$ *as*

$$\sigma_A(p) = \min\{i \mid p \in \sigma_i\} \text{ and } \pi_A(p) = \min\{i \mid p \in \pi_i\}.$$

Definition 5.6. *For each* $i \in \mathbb{N}_0$ *we define sets* Σ_i^{A1W} *and* Π_i^{A1W} *as*

$$\begin{aligned} \Sigma_i^{A1W} &= \{L(A) \mid A = (\Sigma, Q, q_0, \delta, F) \text{ is an A1W automaton and } \sigma_A(q_0) \leq i\}, \\ \Pi_i^{A1W} &= \{L(A) \mid A = (\Sigma, Q, q_0, \delta, F) \text{ is an A1W automaton and } \pi_A(q_0) \leq i\}. \end{aligned}$$

Theorem 5.7. *For each* $i \in \mathbb{N}_0$ *it holds that* $\Sigma_i^{LTL} = \Sigma_i^{A1W}$ *and* $\Pi_i^{LTL} = \Pi_i^{A1W}$.

Before we give a proof of this statement, we present some auxiliary results.

Definition 5.8. *Let* $A = (\Sigma, Q, q_0, \delta, F)$ *be an A1W automaton. For each* $i \in \mathbb{N}_0$ *we inductively define sets of states* σ'_i *and* π'_i *as follows.*

- $\sigma'_0 = \pi'_0 = \{p \mid lh(p) = 0\}$.
- σ'_{i+1} *is the smallest set of states satisfying*

- $\sigma'_i \cup \pi'_i \subseteq \sigma'_{i+1}$ *and*
- *if* p *has no loop or* $p \notin F$*, and Succ*$'(p) \subseteq \sigma'_{i+1}$ *then* $p \in \sigma'_{i+1}$.

- π_{i+1} *is the smallest set of states satisfying*
  - $\sigma'_i \cup \pi'_i \subseteq \pi'_{i+1}$ *and*
  - *if* p *has no loop or* $p \in F$*, and Succ*$'(p) \subseteq \pi'_{i+1}$ *then* $p \in \pi'_{i+1}$.

We also define functions $\sigma'_A, \pi'_A : Q \longrightarrow \mathbb{N}_0$ *as*

$$\sigma'_A(p) = \min\{i \mid p \in \sigma'_i\} \text{ and } \pi'_A(p) = \min\{i \mid p \in \pi'_i\}.$$

Lemma 5.9. *For every A1W automaton* A *with an initial state* q_0 *there exists an A1W automaton* B *with an initial state* q'_0 *such that* $L(A) = L(B)$ *and* $\sigma'_A(q_0) = \sigma_B(q'_0)$.

Proof. On intuitive level, σ_i counts the maximal alternation of nonaccepting and accepting states of the automaton, while σ'_i counts just the alternation of nonaccepting and accepting states with a loop. Hence, we need to modify an automaton A is such a way that the states without any loop do not increase the number of alternations of nonaccepting and accepting states. To do this, we make an accepting and a nonaccepting copy of each state without any loop and we modify transition function such that every state without any loop in every transition is be replaced by one of its two copies. As acceptance or nonacceptance of states without any loop has no influence on language given by the automaton, the modified automaton accepts the same language as the original one.

Let $A = (\Sigma, Q, q_0, \delta, F)$ be an A1W automaton. Let W denote the set of its states without any loop. We set $B = (\Sigma, Q', q'_0, \delta', F')$, where

- $Q' = (Q \setminus W) \cup \{q^a, q^n \mid q \in W\}$,
- $q'_0 = q_0$ if q_0 has a loop; $q'_0 = q^n_0$ otherwise, and
- $F' = (F \setminus W) \cup \{q^a \mid q \in W\}$.

For every $p \in Q'$, by $o(p)$ we denote a state of the original system corresponding to the state p:

$$o(p) = \begin{cases} q & \text{if } p = q^a \text{ or } p = q^n \\ p & \text{otherwise} \end{cases}$$

For every $p \in Q'$ and $a \in \Sigma$, the positive boolean formula $\delta'(p, a)$ arises from $\delta'(o(p), a)$ by replacement of every state $q \in W$ with q^a if $p \in F'$ and with q^n otherwise.

It remains to show that $\sigma'_A(q_0) = \sigma_B(q'_0)$. From the construction of the automaton B it follows that $\sigma'_B(p) = \sigma'_A(o(p))$ for every $p \in Q'$. As $o(q'_0) = q_0$, it is sufficient to show that $\sigma'_B(q'_0) = \sigma_B(q'_0)$. If $lh(q'_0) = 0$, then we are done as $\sigma'_B(q'_0) = 0 = \sigma_B(q'_0)$. In the rest of the proof we show that the equation holds for $lh(q'_0) > 0$ as well.

For each state $p \in Q'$ we define $Succ^+(p)$ to be a transitive closure of $Succ'$ relation, i.e., $Succ^+(p)$ is the smallest set satisfying

- $Succ'(p) \subseteq Succ^+(p)$ and
- if $q \in Succ^+(p)$ then $Succ'(q) \subseteq Succ^+(p)$.

Further, by $LSucc^+(p)$ we denote the set of all states with a loop that are in $Succ^+(p)$.

From the construction of the automaton B it follows that for every state $p \in Q'$ such that $lh(p) > 0$ the following equations hold. Again, maximum over empty set is 0.

$$\sigma_B(p) = \begin{cases} \max\{\pi_B(q) \mid q \in LSucc^+(p) \cap F'\} + 1 & \text{if } p \notin F' \\ \pi_B(p) + 1 & \text{if } p \in F' \end{cases}$$

$$\pi_B(p) = \begin{cases} \sigma_B(p) + 1 & \text{if } p \notin F' \\ \max\{\sigma_B(q) \mid q \in LSucc^+(p) \setminus F'\} + 1 & \text{if } p \in F' \end{cases}$$

If we replace in the above equations the function σ_B with σ'_B and the function π_B with π'_B, the resulting equations hold for each state p with a loop. Hence, we get that for every state $p \in Q'$ with a loop it holds that $\sigma'_B(p) = \sigma_B(p)$ and $\pi'_B(p) = \pi_B(p)$. In particular, if q'_0 has a loop then $\sigma'_B(q'_0) = \sigma_B(q'_0)$.

Let us note that if q'_0 has no loop then it is a nonaccepting state. Further, one can prove that if $lh(q'_0) > 0$, q'_0 has no loop, and $LSucc^+(q'_0) \cap F' = \emptyset$ then $\sigma'_B(q'_0) = 1 = \sigma_B(q'_0)$.

Finally, let us assume that $lh(q'_0) > 0$, q'_0 has no loop, and $LSucc^+(q'_0) \cap F' \neq \emptyset$. As $q'_0 \notin F'$, then

$$\begin{aligned} \sigma'_B(q'_0) &= \max\{\sigma'_B(q) \mid q \in LSucc^+(q'_0)\} && (1) \\ &= \max\{\sigma_B(q) \mid q \in LSucc^+(q'_0)\} && (2) \\ &\geq \max\{\sigma_B(q) \mid q \in LSucc^+(q'_0) \cap F'\} && (3) \\ &= \max\{\pi_B(q) + 1 \mid q \in LSucc^+(q'_0) \cap F'\} && (4) \\ &= \max\{\pi_B(q) \mid q \in LSucc^+(q'_0) \cap F'\} + 1 && (5) \\ &= \sigma_B(q'_0), && (6) \end{aligned}$$

where (1) follows from the definition of σ'_B, (2) is due to the fact that for states with a loop the functions σ'_B and σ_B coincide, and the equation $\sigma_B(q) = \pi_B(q) + 1$ valid for all $q \in F'$ gives us (4). To sum up, $\sigma'_B(q'_0) \geq \sigma_B(q'_0)$. We are done as it is easy to see that $\sigma'_B(p) \leq \sigma_B(p)$ holds for each state p of an arbitrary A1W automaton B. $\square$

By analogy, for an A1W automaton A with initial state q_0 one can construct an equivalent A1W automaton B with an initial state q'_0 such that $\pi'_A(q_0) = \pi_B(q'_0)$.

Now we are ready to prove Theorem 5.7.

of Theorem 5.7. We focus on the former equation as proof of the latter one is analogous. In order to prove the inclusion $\Sigma_i^{LTL} \subseteq \Sigma_i^{A1W}$, we show that for every alphabet Σ and a formula $\varphi \in \Sigma_i^{LTL}$ the automaton $A = A^\Sigma(\varphi)$ given by standard LTL$\to$A1W translation satisfies $\sigma'_A(q_\varphi) \leq i$, where q_φ is an initial state of the automaton. This is sufficient due to Lemma 5.9.

Please note that operators U and R are not in scope of any negation in φ. Hence, the states of the automaton $A^\Sigma(\varphi)$ can be divided into three kinds according to the corresponding subformula of φ.

1. A subformula of the form $\mathsf{X}(\psi \,\mathsf{U}\, \rho)$ or $\psi \,\mathsf{U}\, \rho$ is translated into a state $q_{\psi \mathsf{U} \rho}$ satisfying $q_{\psi \mathsf{U} \rho} \notin F$.

2. A formula $\psi \,\mathsf{R}\, \rho$ is seen as an abbreviation for $\neg(\neg\psi \,\mathsf{U}\, \neg\rho)$. Hence, a subformula of the form $\mathsf{X}(\psi \,\mathsf{R}\, \rho)$ or $\psi \,\mathsf{R}\, \rho$ is translated into a state $q_{\neg(\neg\psi \mathsf{U} \neg\rho)} \in F$.

3. A subformula of the form $\mathsf{X}\psi$ that is not covered by the previous cases is translated into a state q_ψ such that $q_\psi \notin F$ and q_ψ has no loop.

To sum up, states corresponding to U operator are not accepting while the states corresponding to R operator are accepting. Moreover, these states are the only states that can have a loop. Using this observation it is easy to see that each subformula ψ of φ satisfies

$$\psi \in \Sigma_j^{LTL} \Longrightarrow \sigma'_A(q_\psi) \leq j \;\text{ and }\; \psi \in \Pi_j^{LTL} \Longrightarrow \pi'_A(q_\psi) \leq j.$$

In particular, $\sigma'_A(q_\varphi) \leq i$.

To prove the inclusion $\Sigma_i^{LTL} \supseteq \Sigma_i^{A1W}$ we assume that A is an A1W automaton with an initial state q_0 satisfying $\sigma_A(q_0) \leq i$. We show that the formula $\varphi(A)$ given by the standard A1W$\to$LTL translation can be equivalently expressed by a formula from $\Sigma^{LTL}_{\sigma_A(q_0)}$.

Here we employ the fact that the formulae $(\alpha \mathsf{U} \beta) \wedge \mathsf{G}\alpha$ and $((\mathsf{X}\beta) \mathsf{R} \alpha) \vee \beta$ are equivalent for all subformulae α, β. Therefore, for each state p of the automaton A the formula φ_p can be equivalently given as follows.

$$\varphi_p = \begin{cases} \alpha_p \mathsf{U} \beta_p & \text{if } p \notin F \\ ((\mathsf{X}\beta_p) \mathsf{R} \alpha_p) \vee \beta_p & \text{if } p \in F \end{cases}$$

This modified construction produces a formula that is equivalent to $\varphi(A)$. Moreover, U operators correspond to nonaccepting states while R operators correspond to accepting ones. Hence, the alternation of nonaccepting and accepting states in the automaton correspond to the alternation of U and R operators in the resulting formula. In other words, the formula is in $\Sigma^{LTL}_{\sigma_A(q_0)}$. □

6 Summary and future work

We have improved a translation of A1W automata into LTL formulae that are language equivalent. The improvement allows us to define classes of A1W automata corresponding to some LTL fragments given by bounds on nesting depths of temporal operators. Further, we provide an automata-based definition of classes in Until-Release hierarchy [ČP03]. It is worth mentioning that using given automata-based definitions of the classes LTL$(\mathsf{U}^m, \mathsf{X}^n)$ and LTL$(\mathsf{U}, \mathsf{X}^n)$ one can prove some pumping lemmata similar to general stuttering and n-stuttering principles [KS02].

Beside the presented results our research brought several topics for future work. The most interesting topics follows.

The conditions of X-freeness. The original A1W→LTL translation has the property that a formula φ_p corresponding to the state p contains a subformula $\mathsf{X}\varphi_q$ for every successor q of p. Definition 4.1 presents conditions for X-freeness of a set of successors. Roughly speaking, if a set of successors satisfies the conditions then we can omit X operators in front φ_q for every q from the set and the translation remains correct.

The question is whether there are some more general and/or simpler conditions with the same effect. For example, we have no counterexample showing that the improved translation produces incorrect results when we set $\mathsf{Xfree}(p \xrightarrow{a} S)$ to be a set of all states $q \in S \smallsetminus \{p\}$ satisfying

1. there is $S' \subseteq S$ such taht $q \xrightarrow{a} S'$, and

2. if $q \xrightarrow{a} S'$ and $q \notin S'$ then there exists $S'' \subseteq (S \setminus \{q\}) \cup S'$ satisfying $p \xrightarrow{a} S''$.

Unfortunately, we have no proof of correctness of the translation for this Xfree function.

Succintness of A1W automata. Let us consider a formula φ with a more than one copy of a subformula ψ, e.g. $\varphi = (a \vee \mathsf{X}\psi)\,\mathsf{U}\,(b \wedge \mathsf{XX}\psi)$. All copies of the subformula correspond to one state q_ψ of the automaton produced by LTL→A1W translation. Therefore we suppose (but we have no proof yet) that an A1W automaton can be exponentially more succint than arbitrary corresponding LTL formula in some cases.

Expressiveness of AkW and connection to LTL extensions. In Section 2 we have defined alternating automata and alternating 1-weak atomata. An alternating automaton is called *weak* if the set of states Q can be partitioned into disjoint sets $Q_1, Q_2, \ldots, Q_n$ such that

- if $q \in \mathit{Succ}(p)$, $q \in Q_i$, and $p \in Q_j$ then $i \leq j$, and
- $Q_i \cap F = \emptyset$ or $Q_i \subseteq F$ for every $0 < i \leq n$,

where F is a set of accepting states. The automaton is called k-*weak* if $|Q_i| \leq k$ for every $0 < i \leq n$.

General alternating weak automata recognize all ω-regular languages, whereas alternating 1-weak automata recognize all LTL definable languages (star-free ω-regular languages). The definition of alternating k-weak automata (or *AkW automata* for short) brings several interesting questions:

- What is the expressive power of AkW automata?
- Is the hierarchy of classes of AkW automata expressively strict?
- For each k, is there any natural extension LTL_k of LTL such that AkW automata define the same languages as LTL_k?

References

[CMP92] Edward Chang, Zohar Manna, and Amir Pnueli. Characterization of temporal property classes. In Werner Kuich, editor, *Automata, Languages and Programming, 19th International Colloquium (ICALP '92)*, volume 623 of *Lecture Notes in Computer Science*, pages 474–486. Springer-Verlag, 1992.

[ČP03] Ivana Černá and Radek Pelánek. Relating hierarchy of temporal properties to model checking. In *Mathematical Foundations of Computer Science (MFCS)*, volume 2747 of *Lecture Notes in Computer Science*. Springer, 2003.

[DS98] Stéphane Demri and Philippe Schnoebelen. The complexity of propositional linear temporal logics in simple cases (extended abstract). In *Proc. 15th Ann. Symp. Theoretical Aspects of Computer Science (STACS'98)*, volume 1373, pages 61–72. Springer, 1998.

[EW00] Kousha Etessami and Thomas Wilke. An until hierarchy and other applications of an Ehrenfeucht-Fraïssé game for temporal logic. *Information and Computation*, 160:88–108, 2000.

[GO01] Paul Gastin and Denis Oddoux. Fast LTL to Büchi automata translation. In G. Berry, H. Comon, and A. Finkel, editors, *Proceedings of the 13th Conference on Computer Aided Verification (CAV'01)*, volume 2102 of *Lecture Notes in Computer Science*, pages 53–65. Springer, 2001.

[God96] Patrice Godefroid. *Partial-order methods for the verification of concurrent systems: an approach to the state-explosion problem*, volume 1032 of *Lecture Notes in Computer Science*. Springer-Verlag Inc., 1996.

[HP95] Gerard Holzmann and Doron Peled. Partial order reduction of the state space. In *First SPIN Workshop*, Montrèal, Quebec, 1995. A position paper.

[KS02] Antonín Kučera and Jan Strejček. The stuttering principle revisited: On the expressiveness of nested X and U operators in the logic LTL. In Julian Bradfield, editor, *CSL '02: 11th Annual Conference of the European Association for Computer Science Logic*, volume 2471 of *Lecture Notes in Computer Science*, pages 276–291. Springer-Verlag, 2002.

[Lam83] Leslie Lamport. What good is temporal logic? In R. E. A. Mason, editor, *Proceedings of the IFIP Congress on Information Processing*, pages 657–667, Amsterdam, 1983. North-Holland.

[LT00] Christof Löding and Wolfgang Thomas. Alternating automata and logics over infinite words (extended abstract). In J. van Leeuwen et al., editors, *Theoretical computer science: exploring new frontiers of theoretical informatics: International*

Conference IFIP TCS 2000, volume 1872 of *Lecture Notes in Computer Science*, pages 521–535. Springer-Verlag, 2000.

[MP90] Zohar Manna and Amir Pnueli. A hierarchy of temporal properties. In *Proc. ACM Symposium on Principles of Distributed Computing*, pages 377–410. ACM Press, 1990.

[MSS88] David E. Muller, Ahmed Saoudi, and Paul E. Schupp. Weak alternating automata give a simple explanation of why most temporal and dynamic logics are decidable in exponential time. In *Proceedings of the 3rd IEEE Symposium on Logic in Computer Science (LICS 1988)*, pages 422–427. IEEE Computer Society Press, 1988.

[Pel98] Doron Peled. Ten years of partial order reduction. In Alan J. Hu and Moshe Y. Vardi, editors, *Proceedings of the 10th International Conference on Computer Aided Verification*, volume 1427 of *Lecture Notes in Computer Science*, pages 17–28. Springer, 1998.

[Pnu77] Amir Pnueli. The temporal logic of programs. In *Proceedings of the 18th IEEE Symposium on the Foundations of Computer Science*, pages 46–57. IEEE Computer Society Press, 1977.

[PP04] Dominique Perrin and Jean-Eric Pin. *Infinite words*, volume 141 of *Pure and Applied Mathematics*. Elsevier, 2004.

[Roh97] Scott Rohde. *Alternating automata and the temporal logic of ordinals*. PhD thesis, University of Illinois at Urbana-Champaign, 1997.

[Sch03] Philippe Schnoebelen. The complexity of temporal logic model checking. In *Advances in Modal Logic, vol. 4, selected papers from 4th Conf. Advances in Modal Logic (AiML'2002)*, pages 437–459. King's College Publication, 2003.

[Tau03] Heikki Tauriainen. On translating linear temporal logic into alternating and nondeterministic automata. Research Report A83, Helsinki University of Technology, Laboratory for Theoretical Computer Science, 2003.

[TW96] Denis Thérien and Thomas Wilke. Temporal logic and semidirect products: An effective characterization of the until hierarchy. In *37th Annual Symposium on Foundations of Computer Science (FOCS '96)*, pages 256–263. IEEE, 1996.

[TW04] Denis Thérien and Thomas Wilke. Nesting Until and Since in linear temporal logic. *Theory of Computing Systems*, 37(1):111–131, 2004.

[Val91] Antti Valmari. A stubborn attack on state explosion. In Edmund M. Clarke and Robert P. Kurshan, editors, *Proceedings of Computer-Aided Verification (CAV '90)*, volume 531 of *Lecture Notes in Computer Science*, pages 156–165. Springer, 1991.

[Var97] Moshe Y. Vardi. Alternating automata: Unifying truth and validity checking for temporal logics. In William McCune, editor, *Proceedings of the 14th International Conference on Automated Deduction*, volume 1249 of *LNAI*, pages 191–206. Springer, 1997.

[VW86] Moshe Y. Vardi and Pierre Wolper. An automata-theoretic approach to automatic program verification. In *Proceedings of the First Symposium on Logic in Computer Science*, pages 322–331, Cambridge, June 1986.

[VW94] Moshe Y. Vardi and Pierre Wolper. Reasoning about infinite computations. *Information and Computation*, 115(1):1–37, 1994.

[Wil99] Thomas Wilke. Classifying discrete temporal properties. In Chr. Meinel and S. Tison, editors, *STACS '99: Annual Symposium on Theoretical Aspects of Computer Science*, volume 1563 of *Lecture Notes in Computer Science*, pages 32–46. Springer-Verlag, 1999.

[Wol83] Pierre Wolper. Temporal logic can be more expressive. *Information and Control*, 56:72–99, 1983.

[WVS83] Pierre Wolper, Moshe Y. Vardi, and A. Prasad Sistla. Reasoning about infinite computation paths (extended abstract). In *24th Annual Symposium on Foundations of Computer Science*, pages 185–194. IEEE, 1983.