# Efficient Regular Linear Temporal Logic using Dualization and Stratification

César Sánchez
*IMDEA Software Institute, Madrid, Spain*
*Institute for Applied Physics, CSIC, Spain*
*Email: Cesar.Sanchez@imdea.org*

Julián Samborski-Forlese
*IMDEA Software Institute, Madrid, Spain*
*Email: Julian.SF@imdea.org*

*Abstract*—**We study efficient translations of Regular Linear Temporal Logic (RLTL) into automata on infinite words. RLTL is a temporal logic that fuses Linear Temporal Logic (LTL) with regular expressions, extending its expressive power to all $\omega$-regular languages.**

**The first contribution of this paper is a novel bottom up translation from RLTL into alternating parity automata of linear size that requires only colors $0$, $1$ and $2$. Moreover, the resulting automata enjoy the stratified internal structure of hesitant automata. Our translation is defined inductively for every operator, and does not require an upfront transformation of the expression into a normal form. Our construction builds at every step two automata: one equivalent to the formula and another to its complement. Inspired by this construction, our second contribution is to extend RLTL with new operators, including universal sequential composition, that enrich the logic with duality laws and negation normal forms.**

**The third contribution is a ranking translation of the resulting alternating automata into non-deterministic Büchi automata. To provide this efficient translation we introduce the notion of stratified rankings, and show how the translation is optimal for the LTL fragment of the logic.**

*Keywords*-**temporal logic; formal verification; formal methods;**

## I. Introduction

We study the problem of formal temporal verification of reactive systems, which starts from a specification of the intended behavior in some temporal logic. In this paper we study the logic RLTL [1], [2] that extends LTL [3], [4] with regular expressions.

The automata-theoretic approach to model checking reduces this verification problem to automata constructions and automata decision problems. The verification process begins by translating the negation of the formula into an equivalent automaton on infinite words. This automaton accepts all the traces that violate the specification. Then, the automaton is composed synchronously with the system description. Finally, a non-emptiness check reveals whether the system admits some counterexample trace.

Modernly, specifications are translated into alternating automata because their richer structure enables a direct translation from temporal logics, postponing a potentially exponential blow-up. Another advantage of alternation is the easy dualization (see Muller and Schupp [5]) provided by the availability of both conjunctive and disjunctive transition relations. However, to obtain an automaton accepting the complement language of a given automaton, one also needs to complement the acceptance condition (see for example [6]). For LTL one can first translate a formula (e.g., the negation of the specification) into negation normal form (NNF) by pushing negation to the propositional level, and then use automata with weak acceptance conditions [7], [8], in which the structured of the automaton consists of strongly connected components (SCC) all of which are either accepting or rejecting. Extensions of LTL with regular expression, like RLTL, do not have negation normal forms. Hence, a translation of the logical negation operator must be given, precluding the use of weak acceptance conditions.

In this paper we show how to translate RLTL into strong parity automata on words (APW) with a particular internal structure, and study the complementation construction for the resulting APW. The classical complementation for the parity condition increments in one unit the color assigned to every state, turning an arbitrary sequence of states from accepting into rejecting (and viceversa). However, if this construction is used to translate the logical negation operator, the total number of colors in the resulting automaton can grow linearly in the size of the formula. The best known algorithm [9] for translating an APW with $n$ states and $k$ colors into a non-deterministic Büchi automaton requires $2^{O(nk \log nk)}$. In this work, we use a faster complementation construction based on the following intuition. Traces of runs of the automaton get trapped in an SCC, meaning that all states in a suffix of a given trace belong to some SCC of the automaton. Hence, it is sufficient for a complementation construction to consider SCCs independently. This idea enables a translation of RLTL (including the negation operator) into APW using only colors $0$, $1$ and $2$. These automata are equivalent to alternating Streett automata on words (ASW) with one accepting pair (denoted ASW$\{1\}$). The translation proceeds inductively, building at each step a pair of complement automata. Then, inspired by this translation we enrich RLTL with new constructs, including universal sequential composition. The enriched logic admits

a negation normal form.

Finally, we study the translation into non-deterministic Büchi automata (NBW). Streett{1} rankings (see [10]) directly allow to translate an ASW{1} into an NBW of size $2^{O(n \log n)}$. Here, we use again the particular stratified structure of the ASW{1} automata obtained from RLTL expressions. Each stratum in the generated ASW{1} is either Büchi (only colors 1 and 2) or coBüchi (colors 0 and 1), making these automata equivalent to hesitant automata AHW (see [11]). We introduce a notion of stratified ranking and show that for all RLTL operators (except one), the ranking of each state can be statically predetermined. This result produces NBW with size $2^{O(n \log m)}$ where $m$ is the size of the largest stratum that cannot be predetermined. In particular, all LTL operators generate strata of size 1, which result into NBW of size $2^{O(n)}$ when using our method to translate LTL into NBW.

The rest of the of paper is structured as follows. Section II presents the preliminaries, and Section III introduces RLTL. Section IV shows the translation from RLTL into stratified ASW{1}, and Section V the translation into NBW, including stratified rankings. Section VI shows our empirical study. Finally, Section VII concludes.

## II. PRELIMINARIES

We use $\mathcal{B}^+(\mathcal{X})$ for the *positive Boolean formulas* over a set of propositions $\mathcal{X}$. These formulas are built from **true**, **false** and elements of $\mathcal{X}$, using $\wedge$ and $\vee$. A minimal model $M$ of a positive Boolean formula $\theta$ is a subset of $\mathcal{X}$ such that $M$ satisfies $\theta$ but no strict subset of $M$ satisfies $\theta$. For example, given the set $Q = \{q_0, q_1, q_2, q_3\}$, the formula $\theta_1 = (q_1 \wedge q_2) \vee q_3$ is a $\mathcal{B}^+(Q)$ formula. The minimal models of $\theta_1$ are $\{q_1, q_2\}$ and $\{q_3\}$. Given a positive Boolean formula $\theta$ there is a *dual formula* $\widetilde{\theta}$ obtained by switching $\wedge$ and $\vee$, and switching **true** and **false**. For example, the dual of $\theta_1$ above is $\widetilde{\theta_1} = (q_1 \vee q_2) \wedge q_3$, or equivalently in disjunctive normal form $\widetilde{\theta_1} = (q_1 \wedge q_3) \vee (q_2 \wedge q_3)$. The minimal models of $\widetilde{\theta_1}$ are $\{q_1, q_3\}$ and $\{q_2, q_3\}$.

An *alternating automaton* is a tuple $\mathcal{A} : \langle \Sigma, Q, \delta, I, F \rangle$ where $\Sigma$ is a propositional alphabet, $Q$ is a finite set of states, $\delta : Q \times \Sigma \to \mathcal{B}^+(Q)$ is the transition function, $I \in \mathcal{B}^+(Q)$ is the initial condition, and $F$ is the acceptance condition. A *frame* of an automaton $\mathcal{A}$ is the tuple $\langle \Sigma, Q, \delta, I \rangle$. A frame is called *non-deterministic* whenever $I$, and $\delta(q, a)$ for all states $q$ and symbols $a$, have singleton sets as minimal models. In other words, $I$ and $\delta(q, a)$ are equivalent to disjunctive formulas. A frame is called *universal* if $I$, and $\delta(q, a)$ for all states $q$ and symbols $a$, have a unique minimal model. In other words, $I$ and $\delta(q, a)$ are equivalent to conjunctive formulas. A frame is *deterministic* if it is both non-deterministic and universal, that is if both the initial condition and transition functions correspond to **true**, **false** or a single successor state. In general a frame is neither universal nor non-deterministic, but fully alternating.

A transition function $\delta$ can be extended to positive Boolean formulas $\delta : \mathcal{B}^+(Q) \times \Sigma \to \mathcal{B}^+(Q)$ in the standard way, taking $\delta(q, a)$ as the base case and letting $\delta(\mathbf{true}, a) = \mathbf{true}$, $\delta(\mathbf{false}, a) = \mathbf{false}$, $\delta(A \vee B, a) = \delta(A, a) \vee \delta(B, a)$ and $\delta(A \wedge B, a) = \delta(A, a) \wedge \delta(B, a)$.

Given a word $w \in \Sigma^\omega$, a *run* of $w$ on a frame $\mathcal{F} : \langle \Sigma, Q, \delta, I \rangle$ is a DAG $(V, E)$ with nodes $V \subseteq Q \times \mathbb{N}$, s.t.:
1) The set $\{m \mid (m, 0) \in V\}$ is a minimal model for $I$.
2) $\forall (q, k) \in V$, $\{q' \mid (q', k + 1) \in V \wedge ((q, k), (q', k + 1)) \in E\}$ is a minimal model for $\delta(q, w[k])$.

A *trace* of a run is an infinite path in the run, or a finite path finishing in an state with no successor in the run. A non-deterministic frame may admit multiple different runs for a given word, but each run contains a unique trace. A universal frame admits just one run for each word, but this run may contain multiple traces. In general a frame admits multiple runs each with multiple traces.

Given a frame $\mathcal{F} : \langle \Sigma, Q, \delta, I \rangle$, the *specular frame* is $\widetilde{\mathcal{F}} : \langle \Sigma, Q, \widetilde{\delta}, \widetilde{I} \rangle$, where $\widetilde{I}$ is the dual of $I$ and $\widetilde{\delta}$ is the dual transition function: $\widetilde{\delta}(q, a)$ is the dual formula of $\delta(q, a)$ for all states $q$ and symbols $a$. The *graph* of a frame has $Q$ as a set of nodes and contains an edge $p \to q$ whenever $q$ is in some minimal model of $\delta(p, a)$ for some symbol $a$. The graphs of a frame and its specular frame are identical, because if $q$ is in some minimal model of $\delta(p, a)$ then $q$ is also in some minimal model of $\widetilde{\delta}(p, a)$. Therefore, a frame admits a trace iff its specular frame also admits the trace.

An automaton equips a frame with an acceptance condition, which determines whether an infinite sequence of states is accepting. A finite trace finishing in a state $(q, i)$ with no successor is accepting. Given an infinite sequence of states $\pi : q_0, q_1, q_2 \ldots$ we let $\inf(\pi)$ be those states from $Q$ that occur infinitely many times in $\pi$. In this paper we consider the following acceptance conditions:

- *Büchi*: $F \subseteq Q$. $\pi$ is accepting when $\inf(\pi) \cap F \neq \emptyset$.
- *coBüchi*: $F \subseteq Q$. $\pi$ is accepting when $\inf(\pi) \cap F = \emptyset$.
- *parity*: $F : Q \to \{0 \ldots d\}$. $\pi$ is accepting when $\max\{F(q) \mid q \in \inf(\pi)\}$ is even. The elements of $\{0 \ldots d\}$ are called colors.
- *Streett*: $F = \{\langle B_1, G_1 \rangle, \langle B_2, G_2 \rangle, \ldots, \langle B_k, G_k \rangle\}$. $\pi$ is accepting when for all $1 \leq i \leq k$, if $\inf(\pi) \cap B_i \neq \emptyset$ then $\inf(\pi) \cap G_i \neq \emptyset$.
- *Streett{1}*: $F = (B, G)$. $\pi$ is accepting when if $\inf(\pi) \cap B \neq \emptyset$ then $\inf(\pi) \cap G \neq \emptyset$.
- *hesitant*: $F \subseteq Q$, and $H = \langle (S_0 \ldots, S_k), <, \alpha \rangle$ is a partition of $Q$ induced by the SCCs, ordered by $<$ according to reachability in the automaton graph, and $\alpha$ marks each partition as either Büchi or coBüchi. A trace $\pi$ is accepting when
  - $\inf(\pi) \subseteq S_i$, $S_i$ is Büchi and $\inf(\pi) \cap F \neq \emptyset$, or
  - $\inf(\pi) \subseteq S_j$, $S_j$ is coBüchi and $\inf(\pi) \cap F = \emptyset$.

*Observation:* A parity acceptance condition with colors $\{0, 1, 2\}$ corresponds to the Streett condition $(B, G)$ with $B = \{q \mid F(q) = 1\}$ and $G = \{q \mid F(q) = 2\}$. The Streett

pair $(B, G)$ forces (for a trace to be accepting) that if some state marked 1 is visited infinitely often, then some state marked 2 is also visited infinitely often. The other possible case is that only states that are not marked either $B$ or $G$ states are visited infinitely often. In this case, the trace is also good for the parity automaton.

We use *stratum* to refer to an SCC of an automaton graph. The stratification of hesitant automata given by the partition implies that every infinite trace gets trapped in a stratum $S_i$. Then, the Büchi or coBüchi condition on the stratum determines whether the trace is accepting. We use ABW (resp. AcBW, APW, ASW and AHW) to represent Büchi (resp. coBüchi, parity, Streett and hesitant) alternating automata on words. We use APW$\{0, 1, 2\}$ for APW that only use colors 0, 1 and 2 and ASW$\{1\}$ for ASW with only one pair.

When a trace $\pi$ is accepted according to an acceptance condition $F$, we write $\pi \in acc(F)$. A run of an alternating automaton is called accepting whenever all its traces are accepting. We say that a word $w$ is in the language of automaton $\mathcal{A}$, and we write $w \in \mathcal{L}(\mathcal{A})$, whenever there is an accepting run for $w$ on $\mathcal{A}$.

The following definition and theorem relate the notions of specular pairs and complement languages.

**Definition 1** *Two automata $\mathcal{A}$ and $\mathcal{B}$ over the same alphabet are a specular pair whenever their frames are specular and for all paths $\pi$ in the frame graph $\pi \in acc(F_A)$ if and only if $\pi \notin acc(F_B)$.*

**Theorem 2 (Specular Automata and Complement)**
*Let $\mathcal{A}$ and $\mathcal{B}$ be a specular pair of automata. Then $\mathcal{L}(\mathcal{A}) = \Sigma^\omega \setminus \mathcal{L}(\mathcal{B})$.*

Theorem 2 reduces the proof that two automata with dual frames are complements to checking that the traces that can happen have opposite acceptance. In the next section, we use this result to build an incremental translation, in which we only need to check the new traces added at each step.

## III. The Logic RLTL

Regular Linear Temporal Logic [1], [2] is a formalism for specifying $\omega$-regular languages. The logic is defined in two stages, similarly to PSL [12] or ForSpec [13]. In the first stage we build regular expressions that define finite non-empty segments of infinite words. In the second stage, we build RLTL expressions to define sets of infinite words. The syntax of each of these two formalisms contain only a finite collection of constructor symbols. In particular, the language of RLTL contains no fix-point binders or automata constructors.

*Regular Expressions:* The basic elements of regular expressions are *basic expressions*, which are Boolean combinations of a finite set of atomic propositions, whose truth value is interpreted in a single state. The syntax of regular expressions is given by the following grammar:

$$\alpha ::= \alpha + \alpha \quad | \quad \alpha \, ; \alpha \quad | \quad \alpha * \alpha \quad | \quad p$$

where $p$ ranges over basic expressions. The intended interpretation of the operators $+$, $;$ and $*$ are the standard union, concatenation and binary Kleene-star.

Our version of regular expressions describe *segments* of infinite words. Given an infinite word $w \in \Sigma^\omega$, a *position* is a natural number. We use $w[i]$ for the symbol at position $i$ in word $w$. Given an infinite word $w$ and two positions $i$ and $j$, the tuple $(w, i, j)$ is called the *segment* of the word $w$ between positions $i$ and $j$. Note that a segment consists of the whole word $w$ with two tags, not just the sequence of symbols that occur between two positions. This allows the extension of regular expressions to past expressions [2], but in this paper we only study future expressions. A *pointed word* is a pair $(w, i)$ formed by a word $w$ and a position $i$. The formal semantics of regular expressions is defined as a binary relation $\vDash_{\text{RE}}$ between segments and regular expressions, as follows. Given a basic expression $p$, regular expressions $r$ and $s$, and a word $w$:

- $(w, i, j) \vDash_{\text{RE}} p$, whenever $w[i]$ satisfies $p$ and $j = i+1$.
- $(w, i, j) \vDash_{\text{RE}} r + s$ whenever either $(w, i, j) \vDash_{\text{RE}} r$ or $(w, i, j) \vDash_{\text{RE}} s$, or both.
- $(w, i, j) \vDash_{\text{RE}} r \, ; s$ whenever for some $k$, $(w, i, k) \vDash_{\text{RE}} r$ and $(w, k, j) \vDash_{\text{RE}} s$.
- $(w, i, j) \vDash_{\text{RE}} r * s$ whenever either $(w, i, j) \vDash_{\text{RE}} s$, or for some sequence $(i_0 = i, i_1, \ldots i_m)$ and all $k \in \{0, .., m-1\}$, $(w, i_k, i_{k+1}) \vDash_{\text{RE}} r$ and $(w, i_m, j) \vDash_{\text{RE}} s$.

*RLTL:* Expressions in Regular Linear Temporal Logic define languages over infinite words. The key elements of RLTL are the *power* operators that generalize many constructs from different linear-time logics and calculi. The syntax of RLTL expressions is defined by the grammar:

$$\varphi ::= \varnothing \quad | \quad \varphi \vee \varphi \quad | \quad \neg \varphi \quad | \quad \alpha \, ; \varphi \quad | \quad \varphi | \alpha \rangle\!\rangle \varphi \quad | \quad \varphi | \alpha \rangle \varphi$$

where $\alpha$ ranges over regular expressions. The symbol $\vee$ stands for union of languages (logical disjunction), and $\neg$ represents language complement (logical negation). The symbol $;$ stands for the concatenation of an expression over finite words followed by an expression over infinite words. The operator $\varnothing$ defines the empty language (logical *false*).

The operators $\varphi | \alpha \rangle\!\rangle \varphi$ and its weak version $\varphi | \alpha \rangle \varphi$ are called the power operators. The expressions $x | r \rangle\!\rangle y$ and $x | r \rangle y$ (read *x at r until y*, and, respectively, *x at r weak-until y*) are built from three elements: $y$ (the *attempt*), $x$ (the *obligation*) and $r$ (the *delay*). For $x | r \rangle\!\rangle y$ to hold, either the attempt holds, or the obligation is met and the whole expression evaluates successfully after the delay. For $x | r \rangle y$ to hold, the obligation must be met after a finite number of delays. On the contrary, $x | z \rangle y$ does not require the attempt to be met after a finite

number of delays, allowing the obligation and delay to be repeated ad infinitum.

These two simple operators allow the definition of many other temporal operators. For example, the strong until operator $x \, \mathcal{U} \, y$ of LTL can be seen as an attempt for $y$ to hold, or otherwise an obligation for $x$ to be met and a delay of a single step. Similarly, the $\omega$-regular expression $x^\omega$ can be interpreted as a weak power operator having no possible escape and a trivially fulfilled obligation, with a delay indicated by $x$. Then, conventional $\omega$-regular expressions can describe sophisticated delays with trivial attempts and obligations, while conventional LTL constructs allow complex attempts and obligations but trivial one-step delays. Power operators generalize both types of constructs. The completeness of RLTL with respect to $\omega$-regular languages follows immediately from the expressibility of $\omega$-regular expressions. In particular, Wolper's example [14] of an $\omega$-regular language not definable in LTL ($p$ happening at every even state) can be defined as $p|\mathbf{true}\,;\mathbf{true}\rangle\varnothing$. The *size* of an RLTL formula is defined as the total number of its symbols.

The semantics of RLTL relates expressions and pointed words. Given two RLTL expressions $x$ and $y$, a regular expression $r$, and a word $w$:

- $(w,i) \vDash \varnothing$ never holds.
- $(w,i) \vDash x \vee y$ iff either $(w,i) \vDash x$ or $(w,i) \vDash y$.
- $(w,i) \vDash \neg x$ iff $(w,i) \nvDash x$.
- $(w,i) \vDash r\,;y$ iff for some $k$, $(w,i,k)\vDash_{\mathrm{RE}} r$ and $(w,k)\vDash y$.
- $(w,i) \vDash x|r\rangle\!\rangle y$ iff $(w,i) \vDash y$ or for some sequence $(i_0 = i, i_1, \ldots i_m)$, for all $k < m$: $(w, i_k, i_{k+1}) \vDash_{\mathrm{RE}} r$ and $(w, i_k) \vDash x$, and $(w, i_m) \vDash y$.
- $(w,i) \vDash x|r\rangle y$ whenever either $x|r\rangle\!\rangle y$ or for some infinite seq $(i_0 = i, i_1, \ldots)$, for all $k > 0$, $(w, i_k, i_{k+1})\vDash_{\mathrm{RE}} r$ and $(w, i_k) \vDash x$.

The semantics of $x|r\rangle\!\rangle y$ establishes that either the obligation $y$ is satisfied at the point $i$ of the evaluation, or there is a sequence of delays—each determined by $r$—after which $y$ holds, and $x$ holds before each individual delay. The semantics of $x|r\rangle y$ also allow the case where $y$ never holds, but $x$ always holds before any number of evaluations of $r$. Languages are associated with RLTL expressions as usual: a word $w \in \Sigma^\omega$ is in the language of an expression $x$, denoted by $w \in \mathcal{L}(x)$, whenever $(w, 0) \vDash x$.

## IV. RLTL INTO APW USING SPECULAR PAIRS

We present here a translation of RLTL expressions into APW$\{0, 1, 2\}$ based on Theorem 2. The main idea is to generate, at each step, a specular automata pair with the first automaton accepting the same language as the expression. By duality, the specular automaton accepts the complement language. Handling logical negation becomes trivial: one simply needs to switch the elements of the pair.

A previous translation of RLTL presented in [2] needed $n$ colors ($n$ being the size of the formula) instead of 3.

Using [10], [9] to translate APW into NBW would produce NBW with $2^{O(n^2 \log n)}$ states for the old translation and $2^{O(n \log n)}$ states for the one presented here. In Section V below we show how to reduce it further to $2^{O(n \log m)}$ (where $m$ is the size of the largest stratum), and $2^{O(n)}$ for the LTL fragment of RLTL.

The translation is described inductively. For every operator, we show how to compute the specular automata pair, starting from the automata pairs for the sub-expressions. In particular, assume that $(\mathcal{A}_x, \overline{\mathcal{A}_x})$ and $(\mathcal{A}_y, \overline{\mathcal{A}_y})$ are specular pairs for RLTL expressions $x$ and $y$ and that $N_r$ is an NFA for regular expression $r$. We use $q \rightarrow_a F_r$ for "$q \in Q_r$ and $\delta(q,a) \cap F_r \neq \emptyset$," and we use $q \nrightarrow_a F_r$ for "$q \in Q_r$ and $\delta(q,a) \cap F_r = \emptyset$."

***Empty:*** The pair $(\mathcal{A}_\varnothing, \overline{\mathcal{A}_\varnothing})$ has state set $Q = \{q_0\}$, and initial conditions $I = q_0$ and $\overline{I} = q_0$. The acceptance conditions are $F(q_0) = 0$ and $\overline{F}(q_0) = 0$. The transition relations are $\delta(q_0, \_) = \mathbf{false}$ and $\overline{\delta}(q_0, \_) = \mathbf{true}$. This choice of $\delta$ and $\overline{\delta}$ allow all traces to be accepting for $\overline{\mathcal{A}_\varnothing}$ and no trace to be accepting for $\mathcal{A}_\varnothing$, so $\mathcal{A}_\varnothing$ accepts all words and $\overline{\mathcal{A}_\varnothing}$ accepts no word, as desired.

***Disjunction:*** The state space of $\mathcal{A}_{x \vee y} : \langle \Sigma, Q, \delta, I, F \rangle$ and $\overline{\mathcal{A}_{x \vee y}} : \langle \Sigma, Q, \overline{\delta}, \overline{I}, \overline{F} \rangle$ are $Q = Q_x \cup Q_y$. The initial conditions are $I = I_x \vee I_y$ and $\overline{I} = \overline{I_x} \wedge \overline{I_y}$. The transition functions and acceptance condition are:

| if | $\delta(q,a)$ | $\overline{\delta}(q,a)$ | $F(q)$ | $\overline{F}(q)$ |
|---|---|---|---|---|
| $q \in Q_x$ | $\delta_x(q,a)$ | $\overline{\delta_x}(q,a)$ | $F_x(q)$ | $\overline{F_x}(q)$ |
| $q \in Q_y$ | $\delta_y(q,a)$ | $\overline{\delta_y}(q,a)$ | $F_y(q)$ | $\overline{F_y}(q)$ |

***Sequential:*** The state space of both $\mathcal{A}_{r;x} : \langle \Sigma, Q, \delta, I, F \rangle$ and $\overline{\mathcal{A}_{r;x}} : \langle \Sigma, Q, \overline{\delta}, \overline{I}, \overline{F} \rangle$ are $Q_r \cup Q_x$. The initial conditions are $I = I_r$ and $\overline{I} = \widetilde{I_r}$. The transition function is:

| if | $\delta(q,a)$ | $\overline{\delta}(q,a)$ |
|---|---|---|
| $q \nrightarrow_a F_r$ | $\delta_r(q,a)$ | $\widetilde{\delta_r}(q,a)$ |
| $q \rightarrow_a F_r$ | $\delta_r(q,a) \vee I_x$ | $\widetilde{\delta_r}(q,a) \wedge \overline{I_x}$ |
| $q \in Q_x$ | $\delta_x(q,a)$ | $\overline{\delta_x}(q,a)$ |

The acceptance condition is: for $q \in Q_x$ then $F(q) = F_x(q)$ and $\overline{F}(q) = \overline{F_x}(q)$. For $q \in Q_r$ then $F(q) = 1$ and $\overline{F}(q) = 0$.

***Complementation:*** Consider now an RLTL sub-expression $x$, with specular pair $(\mathcal{A}_x, \overline{\mathcal{A}_x})$. Since $(w,i) \vDash \mathcal{A}_x$ if and only if $(w,i) \nvDash \overline{\mathcal{A}_x}$, it follows that $(\overline{\mathcal{A}_x}, \mathcal{A}_x)$ is a specular pair for $\neg x$.

***Power:*** Let $q_0$ be a fresh state, not present in $Q_x$ or $Q_y$. The state spaces of $\mathcal{A}_{x|r\rangle\!\rangle y} : \langle \Sigma, Q, \delta, I, F \rangle$ and $\overline{\mathcal{A}_{x|r\rangle\!\rangle y}} : \langle \Sigma, Q, \overline{\delta}, \overline{I}, \overline{F} \rangle$ are $Q_r \cup Q_x \cup Q_y \cup \{q_0\}$. The initial conds. are $I = q_0$ and $\overline{I} = q_0$. For the transition relation:

| if | $\delta(q,a)$ | $\overline{\delta}(q,a)$ |
|---|---|---|
| $q = q_0$ | $\delta(I_y \vee (I_x \wedge I_r), a)$ | $\overline{\delta}(\overline{I_y} \wedge (\overline{I_x} \vee \widetilde{I_r}), a)$ |
| $q \nrightarrow_a F_r$ | $\delta_r(q,a)$ | $\widetilde{\delta_r}(q,a)$ |
| $q \rightarrow_a F_r$ | $\delta_r(q,a) \vee q_0$ | $\widetilde{\delta_r}(q,a) \wedge q_0$ |
| $q \in Q_x$ | $\delta_x(q,a)$ | $\overline{\delta_x}(q,a)$ |
| $q \in Q_y$ | $\delta_y(q,a)$ | $\overline{\delta_y}(q,a)$ |

For the acceptance condition:

| if | $F(q)$ | $\overline{F}(q)$ |
|---|---|---|
| $q \in Q_x$ | $F_x(q)$ | $\overline{F_x}(q)$ |
| $q \in Q_y$ | $F_y(q)$ | $\overline{F_y}(q)$ |
| $q \in Q_r$ or $q = q_0$ | $1$ | $0$ |

Even though the frame of these automata could have been defined without introducing $q_0$ (by cleverly choosing $I$ and $\delta$), the introduction of $q_0$ is justified by the necessity to distinguish in the acceptance condition traces that visit $q_0$ infinitely often versus traces that get trapped in $Q_r$.

***Weak Power:*** Again, the state spaces of both $\mathcal{A}_{x|r\rangle y}$ : $\langle \Sigma, Q, \delta, I, F \rangle$ and $\overline{\mathcal{A}_{x|r\rangle y}} : \langle \Sigma, Q, \overline{\delta}, \overline{I}, \overline{F} \rangle$ is $Q_r \cup Q_x \cup Q_y \cup \{q_0\}$ for a fresh state $q_0$. For the initial condition $I = q_0$ and $\overline{I} = q_0$. The transition relation and acceptance condition are exactly the same as for the Power operator except for the following cases:

$$F(q) = \begin{cases} 2 & \text{if } q = q_0 \\ 1 & \text{if } q \in Q_r \end{cases} \qquad \overline{F}(q) = \begin{cases} 1 & \text{if } q = q_0 \\ 0 & \text{if } q \in Q_r \end{cases}$$

**Theorem 3** *Let $\varphi$ be an RLTL expression and $\mathcal{A}_\varphi$ be the automaton obtained using the construction described in this section. Then, $\mathcal{L}(\varphi) = \mathcal{L}(\mathcal{A}_\varphi)$.*

The proof of Theorem 3 is greatly simplified by Theorem 2 because at every stage the construction builds automata with specular frame, so one only needs to reason about the acceptance of traces that get trapped in SCCs formed by the freshly added states. The construction also satisfies two important properties:

1) each stage introduces a new stratum (SCC) that cannot be reached from strata added in previous stages. That is, traces that move to the automaton of a sub-expression do not visit the stratum added for the containing expression.

2) The stratum at each stage is decorated only with color 0 (an accepting stratum), only with color 1 (a rejecting stratum), only with colors 0 and 1 (a coBüchi stratum) or only with colors 1 and 2 (a Büchi stratum).

These two observations imply that the automaton has the particular structure of a *stratified* ASW$\{1\}$ or equivalently of hesitant automaton AHW. We show in Section V how to efficiently translate these automata into NBW using a refined version of Streett rankings.

***A Universal Sequential Operator:*** In the previous construction, we observe that the specular automaton for the sequential operator $r \,;\, x$ describes the set of traces in which "*all occurrences of $r$ (if any) are followed by failing occurrences of $x$*". This observation inspires the introduction of the universal sequential operator $r \cdot x$, whose semantics is:

- $(w, i) \vDash r \cdot x$ iff forall $k$ s.t. $(w, i, k) \vDash_{\text{RE}} r$, $(w, k) \vDash x$.

The translation of $r \cdot x$ is precisely $\overline{\mathcal{A}_{r;\neg x}}$ above, and the specular automaton is exactly $\mathcal{A}_{r;\neg x}$. Note that the stratum

corresponding to $r$ in $\mathcal{A}_{r \cdot x}$ has a universal frame, obtained by dualizing the non-deterministic transition relation of $N_r$. The duality laws $\neg(r \,;\, x) \equiv r \cdot \neg x$ and $\neg(r \cdot x) \equiv r \,;\, \neg x$ hold immediately.

***Universal Power Operators:*** Similarly, we define new operators $x\|r\rangle\!\rangle y$ and $x\|r\rangle y$, duals of $x|r\rangle y$ and $x|r\rangle\!\rangle y$, respectively. These new operators force repetitions to hold at *all* possible delays, instead of at *some* possible delay. The semantics are:

- $(w, i) \vDash x\|r\rangle\!\rangle y$ iff $(w, i) \vDash y$ and for all seq $(i = i_0, \ldots, i_m)$ with $(w, i_k, i_{k+1}) \vDash_{\text{RE}} r$, either $(w, i_j) \vDash x$ for some $j \le k$ or $(w, i_{k+1}) \vDash y$, and for all infinite seq $(i = i_0, i_1 \ldots)$ with $(w, i_k, i_{k+1}) \vDash_{\text{RE}} r$ and $(w, i_k) \vDash y$, there is an $m$ with $(w, i_m) \vDash x$.
- $(w, i) \vDash x\|r\rangle y$ iff $(w, i) \vDash x\|r\rangle\!\rangle y$, or $(w, i) \vDash y$ and for all $k$ and $j$ with $(w, i, j) \vDash_{\text{RE}} r^k$ then $(w, j) \vDash y$.

The translation of $x\|r\rangle\!\rangle y$ is $\overline{\mathcal{A}_{\overline{x}|r\rangle \overline{y}}}$ (the dual being $\overline{\mathcal{A}_{x\|r\rangle\!\rangle y}} = \mathcal{A}_{\overline{x}|r\rangle \overline{y}}$), and the translation of $x\|r\rangle y$ is the pair $(\overline{\mathcal{A}_{x|r\rangle y}}, \mathcal{A}_{x|r\rangle y})$. The following duality laws hold:

$$\neg(x|r\rangle y) \equiv \neg x\|r\rangle \neg y \qquad \neg(x\|r\rangle y) \equiv \neg x|r\rangle\!\rangle \neg y \quad (1)$$

$$\neg(x|r\rangle y) \equiv \neg x\|r\rangle\!\rangle \neg y \qquad \neg(x\|r\rangle\!\rangle y) \equiv \neg x|r\rangle \neg y \quad (2)$$

Finally, $x \wedge y$ is defined with translation $(\overline{\mathcal{A}_{\overline{x}\vee\overline{y}}}, \mathcal{A}_{\overline{x}\vee\overline{y}})$. The deMorgan laws hold: $(\neg\neg x \equiv x)$, $(\neg(x \vee y) \equiv \neg x \wedge \neg y)$ and $(\neg(x \wedge y) \equiv \neg x \vee \neg y)$.

Orienting these duality laws from left to right allows to push logical negation $\neg$ to the propositional level, so RLTL extended with these operators admits a negation normal form. Note that this negation normal form is obtained after the translation by specular pairs. It does not follow immediately that the existence of such a normal form enables a translation into automata with weak acceptance condition, because one has to show translations for the new operators, including essentially all elements of pairs in the translation of RLTL presented above.

## V. FROM STRATIFIED ASW$\{1\}$ INTO NBW

This section shows how to translate the alternating automata obtained in Section IV into NBW. We first revisit the notion of Streett ranking from [10], which in turn is based on the notion of coBüchi ranking [15]. Then, we refine rankings to exploit the stratification of the automata obtained as a result of the translation from RLTL. We first show a general translation of ASW$\{1\}$ into NBW.

***Rankings for*** ASW$\{1\}$: We use $[k]$ an abbreviation for the set $\{0 \ldots k\}$. The following definitions assume a given ASW$\{1\}$ automaton $\mathcal{A}$ with $n$ states, acceptance condition $(B, G)$, a word $w \in \Sigma^\omega$ and a run $\mathcal{G} : (V, E)$ of $\mathcal{A}$ on $w$.

**Definition 4** *An S$\{1\}$-ranking is a function $f : V \times \mathbb{N} \to [2n]$ that satisfies: (i) if $q \in B$ then $f(\langle q, l\rangle)$ is even, (ii) for all $\langle q, l\rangle \to \langle q', l'\rangle$ in $E$, either $q \in G$ or $f(\langle q, l\rangle) \ge f(\langle q', l'\rangle)$.*

It follows that for every path $\pi$ on a run DAG $\mathcal{G}$, either $\pi$ visits infinitely often $G$ states or, after some prefix, condition $(ii)$ applies continuously. Hence, since the image of $f$ is bounded, the value of $f$ converges to a value: there is a number $l$, such that, for every $l' > l$, $f(\pi(l')) = f(\pi(l))$. The following definition of odd S$\{1\}$-ranking relates the convergence to an odd value with the fact that $B$ states are visited only finitely often. Then, the construction of the NBW below is justified by Lemma 6.

**Definition 5 (odd S$\{1\}$-ranking)** *An* S$\{1\}$*-ranking is odd whenever, for every path $\pi$ of $\mathcal{G}$, either* (*i*) *$\pi$ visits infinitely often $G$ states, or* (*ii*) *$f$ converges to an odd value on $\pi$.*

**Lemma 6** *$\mathcal{G}$ is an accepting run iff there is an odd* S$\{1\}$*-ranking for $\mathcal{G}$.*

*An equivalent NBW:* We describe here the translation from ASW$\{1\}$ into NBW. The main idea is to encode in the states of the NBW cuts of a run DAG of the ASW$\{1\}$, decorated with enough information to check whether an odd-ranking exists. In particular, each state of the alternating automaton present in a given state of the NBW is labeled with a ranking value. This annotation must respect the definition of ranking (Def. 4). Additionally, the set of states of the ASW$\{1\}$ that form a state of the NBW are partitioned into those that owe an improvement in the ranking (either a visit to a $G$ state or a decrease in the ranking), and those that already showed improvement. Membership to the owe set is propagated, so an accepting state is one in which all constituent states have seen some progress since the last accepting state. After an accepting state, the owe set is reset.

Formally, we start from an ASW$\{1\}$ automaton $\mathcal{A}$ : $\langle \Sigma, Q_\mathcal{A}, I_\mathcal{A}, \delta_\mathcal{A}, \{(B, G)\} \rangle$ and we build an NBW $N$ : $\langle \Sigma, Q_N, I_N, \delta_N, F_N \rangle$ as follows:

- $Q_N$ contains elements of the form $(S, O, f)$ where $S \subseteq Q_\mathcal{A}$ is a subset of states of $\mathcal{A}$, $O \subseteq S$, and $f : S \to [2n]$ is a function that satisfies:
  **Q1**. if $q \in B$ then $f(q)$ is even.
- $I_N$ contains all those $(M, O, f) \in Q_N$ where
  **I1**. $M$ is a minimal model of $I_\mathcal{A}$ and $O = \{q \in M \mid q \notin G$ and $f(q)$ is even$\}$.
- $F_N = \{(S, O, f) \in Q_N \mid O = \emptyset\}$.
- $\delta_N : Q_N \times \Sigma \to 2^{Q_N}$, such that $(S', O', f') \in \delta_N((S, O, f), a)$ whenever there is one minimal model $M_q$ of $\delta_\mathcal{A}(q, a)$ for each $q \in S$ satisfying:
  **D1**. $S' = \cup_{q \in S} M_q$,
  **D2**. For all $p \in S'$, the rank annotation $f'(p) \leq min\{f(q) \mid q \in pred(p) \setminus G\}$ where $pred(p) = \{q \in S \mid p \in M_q\}$ denotes the set of predecessors of $p$.
  **D3**. $O'$ is given as follows. Let $p \in S' \setminus G$, we have
    - If $O = \emptyset$ then $p \in O'$ iff $f'(p)$ is even.
    - If $O \neq \emptyset$ then $p \in O'$ iff $f'(p) = f(q)$ for some $q \in (pred(p) \cap O)$.

The states of $N$ consist of a set $S$ representing elements of a cut of a run DAG of $\mathcal{A}$. The function $f$ represents an S$\{1\}$-ranking, where **Q1** guarantees that no $B$ node receives an odd value, and **D2** guarantees the non-increasing condition of rankings. Condition **D1** ensures that successor states of $N$ correspond to legal successor cuts of a run of $\mathcal{A}$. Finally, condition **D3** ensures that $O$ contains those vertices of the run DAG that have not seen progress for some path leading to them, where progress is defined as visiting a $G$ state, or experiencing a decrease in $f$. A reset of this check is represented by a final state, which can happen only when all paths to all states contain some progress, as captured by $F_N$. Finally, **I1** captures that the initial states of $N$ correspond to initial cuts of runs of $\mathcal{A}$. All these facts imply that a successful run DAG of $\mathcal{A}$ is matched by a successful run of $N$.

**Theorem 7** *Let $\mathcal{A}$ be an* ASW$\{1\}$ *and $N$ the corresponding NBW. Then $w \in \mathcal{L}(\mathcal{A})$ if and only if $w \in \mathcal{L}(N)$.*

The automaton obtained can be easily pruned with one simple observation: if there is an odd S$\{1\}$-ranking, then there is an odd S$\{1\}$-ranking where all decreases (according to **D2**) only drop to the highest legal value. That is:

$$f'(p) = \begin{cases} M \text{ or } M - 1 & \text{if } p \notin B \\ M \text{ or } M - 2 & \text{if } p \in B \end{cases}$$

where $M = min\{f(q) \mid q \in pred(p) \setminus G\}$.

This observation reduces the guessing in $f$ to only two possibilities, providing a more efficient translation. The next paragraphs exploit the internal structure of stratified ASW$\{1\}$ automata to introduce a faster solution, specific for the particular case of AHW.

*Rankings for Stratified* **ASW**$\{1\}$*:* Consider a stratified ASW$\{1\}$. This is an automaton for which $Q$ is divided into strata $(S_1, \ldots, S_k)$ ordered according to $<$, and each stratum is labeled by a function $\alpha$ as either Büchi (all states are either $B$ or $G$) or coBüchi (no state is $G$). The stratification structure implies that for every $q \in S_i$ and successor $p$ with $p \in S_j$, either $S_j = S_i$ or $S_j < S_i$.

*Remark:* This automaton is equivalent to an AHW with $H = \langle (S_1, \ldots, S_k), <, \alpha \rangle$ and

$$F = \bigcup_i \{S_i \cap G \mid \text{if } S_i \text{ is Büchi}\} \cup \bigcup_i \{S_i \cap B \mid \text{if } S_i \text{ is coBüchi}\}$$

We use $m_j = |S_j|$ to refer to the number of states in stratum $S_j$. We first define the notion of stratified S$\{1\}$-ranking:

**Definition 8** *A stratified* S$\{1\}$*-ranking is a family of functions $f_j : S_j \times \mathbb{N} \to [2m_j]$ that satisfies:* (*i*) *if $q \in S_j \cap B$ then $f_j(\langle q, l \rangle)$ is even,* (*ii*) *for every $\langle q, l \rangle \to \langle q', l' \rangle$ in $E$ with $q, q' \in S_j$, then $f_j(\langle q, l \rangle) \geq f_j(\langle q', l' \rangle)$, unless $q \in G$.*

Intuitively, a stratified ASW$\{1\}$ ranking is like an ASW$\{1\}$ ranking except values need not decrease when

moving across strata. Due to the stratification, every trace of a run gets trapped in a stratum of the automaton. Once the trace converges to a stratum, either the trace visits infinitely many good nodes, or the ranking converges to a single value. Again, the notion of odd ranking captures whether the suffix traces are accepting.

**Definition 9** *A stratified* S{1}*-ranking is odd whenever, for every infinite path $\pi$ of $\mathcal{G}$, either* (i) $\pi$ *visits infinitely often $G$ states, or* (ii) $\pi$ *gets trapped in stratum $S_j$ and $f_j$ converges to an odd value on $\pi$.*

The following lemma justifies the construction of NBW using stratified rankings.

**Lemma 10** $\mathcal{G}$ *is an accepting run iff there is a stratified odd* S{1}*-ranking for $\mathcal{G}$.*

Stratified rankings drastically limit the guessing that is necessary in the construction of the states of the NBW, because each ranking is local to the stratum under consideration. The following choices produce a good ranking for stratum $S_j$, if there is one such a good ranking
**G1**. If $S_j$ is an accepting stratum and $q_j \in S_j$, $f_j(q_j) = 1$.
**G2**. If $S_j$ is a rejecting stratum and $q_j \in S_j$, $f_j(q_j) = 2$.
**G3**. If $S_j$ is Büchi, then assign $f_j(q_j) = 2$ to $q_j \in B$, and $f_j(q_j) = 1$ to $q_j \in G$.
**G4**. If $S_j$ is coBüchi then $f_j(q_j) \in [2m_j]$.

Note that this restriction eliminates the guessing except for coBüchi strata, and consequently ranking guessing only happens to the states of $N_r$ in expressions $x\|r\rangle\!\rangle y$. In terms of the LTL fragment, all delays are one step so the size of $|N_r| = 1$ and hence the maximum size of the coBüchi strata is 1. In fact, for LTL sub-expressions of the form $x\|r\rangle\!\rangle y$, $N_r$ consists of a single $B$ state, which can be assigned value 2. Consequently, following the steps in this paper LTL expressions get translated into NBW of size $2^{O(n)}$.

*An equivalent NBW using Stratified Rankings*: We refine the construction for general ASW{1} rankings, limiting the guesses using **G1**-**G4**. Also, only predecessors within the same stratum are considered when computing $f$:
**Q1s**. if $q \in S_j \cap B$ then $f_j(q)$ is even.
**D2s**. $f'_j(p) \leq min\{f_j(q) \mid q \in pred(p) \setminus G\}$ where $pred(p) = \{q \in S_j \mid p \in M_q\}$ now only considers predecessors from the same stratum.
**D3s**. $O'$ is given as follows. Let $p \in S'_j \setminus G$, we have
  - If $O = \emptyset$ then $p \in O'$ iff $f'_j(p)$ is even.
  - If $O \neq \emptyset$ then $p \in O'$ iff $f'_j(p) = f_j(q)$ for some $q \in (pred(p) \cap O \cap S_j)$.

**Theorem 11** *Let $\mathcal{A}$ be a stratified* ASW{1} *and $N$ the corresponding NBW using stratified rankings. Then $w \in \mathcal{L}(\mathcal{A})$ if and only if $w \in \mathcal{L}(N)$.*

## VI. EMPIRICAL EVALUATION

This section reports the result of an empirical evaluation of the translation algorithms presented above. The evaluation was performed using a sequential implementation written in OCaml, available online at [16]. The running times reported in Fig. 1 were obtained using an Intel Core2 @ 2.83GHz with 8GB of RAM running a 64 bit Linux kernel. Fig. 1 compares the number of states and the running time used to compute explicit NBW representations of two families of formulas (and their negation), for $i = 8, 11, 17, 20$. These choices are inspired by [8]:
  - $A_i = (p_1 \ \mathcal{U} \ (p_2 \ \mathcal{U} \ (\ldots \ \mathcal{U} \ p_i)\ldots))$. The expression $A_i$ is equivalent to the RLTL expression $p_1|\mathbf{true}\rangle\!\rangle(p_2|\mathbf{true}\rangle\!\rangle\ldots)$.
  - $B_i = p_1|\mathbf{true}^5\rangle\!\rangle(p_2|\mathbf{true}^5\rangle\!\rangle\ldots)$, where $\mathbf{true}^5$ stands for a five instant delay $\mathbf{true};\mathbf{true};\mathbf{true};\mathbf{true};\mathbf{true}$. These are not expressible in LTL.

The table illustrates that the general ASW{1} ranking is only practical for the smallest cases. Limiting the guessing to the highest ranks allows to handle slightly larger formulas. The stratified ranking translation results in a dramatic improvement, comparable to state of the art LTL translators, particularly considering that our prototype does not use simulations or handle propositional alphabets (only discrete alphabets). Simulation reductions have been reported [8] to be a very effective method to reduce the size of the NBW generated, but this optimization is currently ongoing work.

## VII. CONCLUSIONS AND FUTURE WORK

This paper has presented a novel translation from the logic RLTL into alternating parity automata using only colors 0, 1 and 2, based on a bottom-up construction of specular pairs accepting complement languages. Inspired by the duality in the translation we introduce universal sequential operators that enrich the logic with negation normal forms. We also show that the resulting automata enjoy some stratified structure in their transition relation that makes all their strata purely Büchi or coBüchi. These automata are equivalent to hesitant automata. Then, we study translations of the resulting automata into NBW. The main result is the specialization of Street rankings to stratified automata to obtain a more efficient ranking translation. Unlike [11] our construction preserves the alphabet between the alternating automaton an the NBW. We are currently investigating alternative algorithms for model-checking RLTL specifications based on bounded model checking [17], antichains [18] and IC3 [19].

## REFERENCES

[1] M. Leucker and C. Sánchez, "Regular linear temporal logic," in *Proc. of ICTAC'07*, ser. LNCS, vol. 4711. Springer, 2007, pp. 291–305.

[2] C. Sánchez and M. Leucker, "Regular linear temporal logic with past," in *Proc. of VMCAI'10*, ser. LNCS, vol. 5944. Springer, 2010, pp. 295–311.

| | APW | | NBW (direct) | | NBW (max2) | | NBW (strat) | |
|---|---|---|---|---|---|---|---|---|
| | size | time(s) | size | time(s) | size | time(s) | size | time(s) |
| $A_8$ | 7 | 0.048 | 93 | 0.128 | 93 | 0.100 | 9 | 0.052 |
| $A_{11}$ | 10 | 0.172 | 192 | 0.504 | 192 | 0.336 | 12 | 0.176 |
| $A_{17}$ | 16 | 0.936 | 498 | 3.680 | 498 | 1.700 | 18 | 0.952 |
| $A_{20}$ | 19 | 1.796 | 705 | 8.149 | 705 | 3.184 | 21 | 1.816 |
| $\neg A_8$ | 7 | 0.048 | 142 | 0.252 | 100 | 0.100 | 9 | 0.052 |
| $\neg A_{11}$ | 10 | 0.172 | 292 | 1.140 | 202 | 0.336 | 12 | 0.172 |
| $\neg A_{17}$ | 16 | 0.940 | 754 | 10.545 | 514 | 1.688 | 18 | 0.948 |
| $\neg A_{20}$ | 19 | 1.792 | 1066 | 25.718 | 724 | 3.160 | 21 | 1.884 |
| $B_8$ | 35 | 0.268 | 2417 | 50.103 | 2417 | 5.936 | 37 | 0.296 |
| $B_{11}$ | 50 | 1.084 | 4952 | 360.571 | 4952 | 26.846 | 52 | 0.952 |
| $B_{17}$ | 80 | 5.064 | 12722 | 1h56m | 12722 | 217.698 | 82 | 5.200 |
| $B_{20}$ | 95 | 10.041 | 17957 | 8h32m | 17957 | 598.129 | 97 | 9.897 |
| $\neg B_8$ | 35 | 0.268 | 3642 | 209.157 | 2452 | 5.704 | 37 | 0.300 |
| $\neg B_{11}$ | 50 | 0.908 | 7452 | 26m56s | 5002 | 25.106 | 52 | 0.956 |
| $\neg B_{17}$ | 80 | 5.072 | 19122 | 18h15m | 12802 | 220.762 | 82 | 5.192 |
| $\neg B_{20}$ | 95 | 9.753 | 26982 | 58h8m | 18052 | 674.070 | 97 | 9.905 |

Figure 1.    Number of states and running time to compute an APW$\{0, 1, 2\}$ and an NBW.

[3] A. Pnueli, "The temporal logic of programs," in *Proc. of FOCS'77*.   IEEE CS Press, 1977, pp. 46–67.

[4] Z. Manna and A. Pnueli, *Temporal Verification of Reactive Systems*.   Springer-Verlag, 1995.

[5] D. E. Muller and P. E. Schupp, "Altenating automata on infinite trees," *Theoretical Computer Science*, vol. 54, pp. 267–276, 1987.

[6] W. Thomas, "Complementation of Büchi automata revisited," in *Jewels are Forever, Contributions on TCS in Honor of Arto Salomaa*.   Springer, 1999, pp. 109–120.

[7] O. Kupferman and M. Y. Vardi, "Weak alternating automata are not that weak," *ACM Transactions on Computational Logic*, vol. 2, no. 3, pp. 408–429, 2001.

[8] P. Gastin and D. Oddoux, "Fast LTL to Büchi automata translation," in *Proc. of CAV'01*, ser. LNCS, vol. 2102. Springer, 2001, pp. 53–65.

[9] C. Dax and F. Klaedtke, "Alternation elimination by complementation," in *Proc. of LPAR'08*, ser. LNCS, vol. 5530. Springer, 2008, pp. 214–229.

[10] O. Kupferman and M. Y. Vardi, "Complementation constructions for nondeterministic automata on infinite words," in *Proc. of TACAS'05*, ser. LNCS, vol. 3440.   Springer, 2005, pp. 206–221.

[11] O. Kupferman, N. Piterman, and M. Y. Vardi, "Extended temporal logic revisited," in *Proc. of CONCUR'01*, ser. LNCS, vol. 2154.   Springer, 2001, pp. 519–535.

[12] D. Fisman, C. Eisner, and J. Havlicek, *Formal syntax and Semantics of PSL: App. B of Accellera Property Language Ref. Manual, Ver. 1.1*, March 2004.

[13] A. Armando, S. Ranise, and M. Rusinowitch, "A rewriting approach to satisfiability procedures," *Information and Computation*, vol. 183, no. 2, pp. 140–164, 2003.

[14] P. Wolper, "Temporal logic can be more expressive," *Information and Control*, vol. 56, pp. 72–99, 1983.

[15] O. Kupferman and M. Y. Vardi, "From complementation to certification," in *Proc. of TACAS'04*, ser. LNCS, vol. 2988. Springer, 2004, pp. 591–606.

[16] http://software.imdea.org/rounded/.

[17] A. Biere, A. Cimatti, E. M. Clarke, and Y. Zhu, "Symbolic model checking without BDDs," in *Proc. of TACAS'99*, ser. LNCS.   Springer, 1999, pp. 193–207.

[18] M. D. Wulf, L. Doyen, N. Maquet, and J.-F. Raskin, "Antichains: Alternative algorithms for LTL satisfiability and model-checking," in *Proc. of TACAS'08*, ser. LNCS, vol. 4693.   Springer, 2008, pp. 63–77.

[19] A. R. Bradley, "SAT-based model checking without unrolling," in *Proc. of VMCAI'11*, ser. LNCS, vol. 6538. Springer, 2011, pp. 70–87.

This section shows a self-contained proof of Theorem 2.

*Positive Boolean Formulas::* Every positive boolean formula can be expressed in disjunctive normal form, as disjunction of conjunctions of propositions. Given a positive boolean formula $\theta$ there is a dual formula $\widetilde{\theta}$ obtained by switching $\wedge$ and $\vee$, and switching **true** and **false**. Some easy properties of dual formulas are:

**Proposition 12 (Duals)** *For every $\theta$ and $\widetilde{\theta}$, and for every $M \in Mod(\theta)$:*

*1) For every $M' \in Mod(\widetilde{\theta})$, $M \cap M' \neq \varnothing$.*
*2) Let $q \in M$. There is an $M'$ in $Mod(\widetilde{\theta})$ with $q \in M'$.*

For example, the dual of $\theta_1$ above is $\widetilde{\theta_1} = (q_1 \vee q_2) \wedge q_3$, or equivalently in disjunctive normal form $\widetilde{\theta_1} = (q_1 \wedge q_3) \vee (q_2 \wedge q_3)$. The minimal models of $\widetilde{\theta_1}$ are $\{q_1, q_3\}$ and $\{q_2, q_3\}$.

A *choice function* is a map $f$ that chooses, for a model $M$ of $\theta$ an element of $M$, i.e., $f : Mod(\theta) \rightarrow \mathcal{X}$ such $f(M) \in M$. Some interesting properties of choice functions follow:

**Proposition 13 (Choice Functions)** *Let $\theta$ be a formula and $\widetilde{\theta}$ its dual. Then*

*1) If $f$ is a choice function for $\theta$, then $Img\ f \in Mod(\widetilde{\theta})$.*
*2) If $M \in mod(\theta)$ then there is a choice function $f$ of $\widetilde{\theta}$ such that $Img\ f = M$.*

*Proof:* We prove 13.1 first. Consider $\theta$ in disjunctive normal form. Each child subexpression of the root expression corresponds to a conjunction of states that form a model. The choice function $f$ chooses one state from each model of $\theta$. Expressing $\widetilde{\theta}$ dualy, each child subexpression of $\theta$ is a disjunction of the corresponding set of states. Hence, the element that $f$ chooses in each child satisfies the corresponding disjunction, and $\left( Img\ f = \bigcup_{M \in Mod(\theta)} f(M) \right)$ is a model of $\widetilde{\theta}$.

We now show 13.2. Let $M$ be a minimal model of $\theta$. Consider an arbitrary choice function $f$ for $\widetilde{\theta}$ with:

$$f(M') = q \ \text{ for some } \ q \in M \cap M'.$$

By Prop. 12.1 for any $M'$ there is one such $q$, so $f$ is well defined, and by construction $Img\ f \subseteq M$. By Prop. 13.1 $Img\ f$ is a model of $\theta$, and since $M$ is a minimal model it has no proper sub-model, so $Img\ f = M$. $\square$ ∎

*Automata and Games:* We show now that specular automata accept complement languages, using game theory. From a given automaton $\mathcal{A}$ and a word $w$, we create a parity game called a *word game* as a tuple $\mathbf{G}(\mathcal{A}, w)$ :

$\langle V_A, V_P, E_A, E_P, f \rangle$ where:

$$V_A = Q \times \omega$$
$$V_P = \{(M, q, i) \mid M \in Mod(\delta(q, w[i]))\}$$
$$\cup \{(M, \cdot, 0) \mid M \in Mod(I)\}$$
$$E_A = (q, i) \rightarrow (M, q, i) \ \text{ for each } M \in Mod(\delta(q, w[i]))$$
$$E_P = (M, q, i) \rightarrow (q', i+1) \ \text{ for } q' \in M$$

The game is played by two players: *Automaton* $(A)$ and *Pathfinder* $(P)$. The set of positions $V = V_A \cup V_P$ is partitioned into positions in which $A$ plays and those in which $P$ plays. The game begins by $A$ choosing a model of $I$, which determines the initial position $(M, \cdot, 0)$ (here $\cdot$ represent an irrelevant state). The legal moves of the game are captured by the relation $E = E_A \cup E_P$ which correspond to $A$ choosing a model from a $V_A$ position, and $P$ choosing the next successor from a given model from a $V_P$ position. A *play* is an infinite sequence of positions $\pi : V_0 v_0 V_1 v_1 \ldots$ with $V_0$ being an initial position, $v_i$ obtained from $V_i$ by a $P$ move, and $V_{i+1}$ obtained from $v_i$ by an $A$ move. The map $f : V \rightarrow \{0 \ldots d\}$ determines the outcome of a play. We define the *trace* of a play $\pi : V_0 v_0 V_1 v_1 \ldots$ as the sequence of states $trace(\pi) : p_0 p_1 \ldots$ obtained by projecting the first component of the $V_P$ positions of the play (i.e., $v_i = (p_i, i)$). The following follows directly from the definition:

**Proposition 14** *Every trace of a play of $\mathbf{G}(\mathcal{A}, w)$ is also a trace of some run of $\mathcal{A}$ on $w$.*

As for parity automata the outcome of a play is determined by the highest color that is seen infinitely often in the play. Player $A$ wins play $\pi$ whenever:

$$\max\{f(q) \mid q \in \inf(trace(\pi))\} \text{ is even}$$

Otherwise, $P$ wins play $\pi$. A strategy for player $A$ is a map $\rho_A : (V^* V_A \cup \epsilon) \rightarrow V$, that maps histories of positions into moves. Here, $\epsilon$ denotes the empty sequence of positions, to let player $A$ choose an initial state in the game. A memoryless strategy simply takes into account the last position: $\rho_A : V_A \cup \epsilon \rightarrow V$. Since parity games are memoryless determined it is enough to consider memoryless strategies. Similarly, a strategy for player $P$ is a map $\rho_P : V_P \rightarrow V$. A play $\pi : V_0 v_0 V_1 v_1 \ldots$ is played according to strategy $\rho_A$ whenever the initial position is $V_0 = \rho_A(\epsilon)$ and all moves of $A$ are played according to it $V_i = \rho_A(v_i)$. A strategy $\rho_A$ is winning for player $A$ whenever all plays played according to $\rho_A$ are winning for $A$. Memoryless determinacy of parity games guarantees that either player $A$ has a memoryless winning strategy or player $P$ has a memoryless winning strategy. We say that $\pi$ is a $G \cdot \rho_A$ play whenever $\pi$ is played in $G$ according to $\rho_A$.

We restrict our attention to strategies for $A$ that choose minimal models, and strategies for $P$ that are proper choice functions. This is not a drastic restriction. Clearly, if there

is a winning strategy for $A$ that does not choose a minimal model, then any strategy that chooses a smaller minimal model is also winning. This is because the set of plays is reduced, and all plays in the unrestricted set are winning for $A$. Similarly, if $\rho_P$ is a winning strategy for $P$, then restricting its moves to be a proper choice functions (by restricting the image) also gives a winning strategy. In both cases, the set of successor moves is restricted but still confined within winning regions. This lemma is essentially Prop. 2 from [6], where complementation of weak alternation automata by dualization is studied.

**Lemma 15** $w \in \mathcal{L}(\mathcal{A})$ *if and only if $A$ has a winning strategy in $\mathbf{G}(\mathcal{A}, w)$.*

*Proof:* Assume $w \in \mathcal{L}(\mathcal{A})$ and let $\sigma : (V_\sigma, E_\sigma)$ be a successful run of $w$ on $\mathcal{A}$. We first build a strategy $\rho_A$ for $A$ on $\mathbf{G}(\mathcal{A}, w)$ and then show that $\rho_A$ is winning:

$$\rho_A(\epsilon) = (M_0, \cdot, 0)$$
$$\text{with } M_0 = \{q \mid (q, 0) \in V_\sigma\}$$
$$\rho_A(q, i) = (M, q, i+1)$$
$$\text{with } M = \{q' \mid (q, i) \to (q', i+1) \in E_\sigma\}$$

The set $M$ in $(M, q, i+1)$ is a model of $\delta(q, i)$ because $\sigma$ is a run. For positions $(q, i)$ that do not appear in the run $\sigma$, the strategy $\rho_A(q, i) = (M, q, i+1)$ can assign any model $M$ in $Mod(\delta(q, w[i]))$. This model is not relevant because no play played according to $\rho_A$ will visit these states. Consider now an arbitrary play $\pi : V_0 v_0 V_1 v_1 \ldots$ of $\mathbf{G}(\mathcal{A}, w)$ played according to $\rho_A$. We show by induction that $trace(\pi) : p_0 p_1 \ldots$ is a trace of $\sigma$.

- *base*: By construction $M_0$ is the set of initial positions of $\sigma$. Since $p_0$, chosen by player $P$, is $v_0 \in M_0$, then $v_0$ is a prefix of a trace of run $\sigma$.
- *induction step*: assume $p_0 \ldots p_i$ is a prefix of some trace in $\sigma$, so $(p_i, i)$ is in $V_\sigma$. Hence, $\rho_A(p_i, i) = (M, p_i, i+1)$ for $M$ being the set of successors of $(p_i, i)$ in $E_\sigma$. Consequently $p_{i+1} = (q, i+1)$ for some $(p_i, i) \to (p_{i+1}, i+1)$ in $E_\sigma$, so $v_0 \ldots v_i v_{i+1}$ is a longer prefix of a trace of run in $\sigma$.

This shows that $trace(\pi)$ is a trace of the run $\sigma$. Now, since $\sigma$ is a successful run all its traces must be accepting, and then:

$$\max\{F(q) \mid q \in \inf(trace(\pi))\} \text{ is even,}$$

which shows that $\rho_A$ is a winning strategy for $\mathbf{G}(\mathcal{A}, w)$.

We now show the other direction: we start from a winning strategy $\rho_A$ for $A$ in $\mathbf{G}(\mathcal{A}, w)$ and show that there is a successful run $\sigma$ of $w$ on $\mathcal{A}$. Let $(M, \cdot, 0) = \rho_A(\epsilon)$. Then we let $V_\sigma$ contain $(q, 0)$ for all $q \in M$. Note that $M$ is a minimal model of $I$. Now, consider an arbitrary position $(q, i)$ and let $(M, q, i+1)$ be $\rho_A(q, i)$. We add to $E_\sigma$ all pairs

of the form $(q, i) \to (q', i+1)$ for all $q' \in M$. We have to show that $\sigma$ is successful run. We show by induction that all traces of $\sigma$ correspond to plays in $\mathbf{G}(\mathcal{A}, w)$ played according to $\rho_A$. For the base case $(q, 0)$ is the initial state of the trace. By construction $(q, 0) \in \rho_A(\epsilon)$ so $(q, 0)$ is a possible choice of player $P$, and consequently a play prefix. For the inductive case, assume that trace prefix $(q_0, 0) \ldots (q_i, i)$ is a play prefix, and let $(q_i, i) \to (q_{i+1}, i+1)$ be in $E_\sigma$. By construction $\rho_A(q_i, i)$ contains position $(q_{i+1}, i+1)$ so player $P$ can again move to it. This shows that the arbitrary trace of $\sigma$ correspond to a play played according to $\rho_A$. $\square$ ∎

*Specular Pairs and Complementation*: We show now that specular automata accept complement languages. In the rest of the section we let $\mathcal{A}$ and $\widetilde{\mathcal{A}}$ be a specular automata pair, $w$ be a word and $G : \mathbf{G}(\mathcal{A}, w)$ and $\widetilde{G} : \mathbf{G}(\widetilde{\mathcal{A}}, w)$ be the corresponding word games. First we need some preliminary definitions.

**Definition 16** *We say that strategies $\rho_A$ (for $A$ in $G$) and $\widetilde{\rho}_P$ (for $P$ in $\widetilde{G}$) are duals whenever both:*
- *for every $G \cdot \rho_A$ play $\pi$ there is a $\widetilde{G} \cdot \rho_P$ play $\widetilde{\pi}$ s.t. $trace(\widetilde{\pi}) = trace(\pi)$.*
- *for every $\widetilde{G} \cdot \rho_P$ play $\widetilde{\pi}$ there is a $G \cdot \rho_A$ play $\pi$ s.t. $trace(\widetilde{\pi}) = trace(\pi)$.*

**Theorem 17 (Dual Strategies)** *The following holds:*
*(1) For every strategy $\rho_A$ for $A$ in $G$, there is a dual strategy $\widetilde{\rho}_P$ for $P$ in $\widetilde{G}$.*
*(2) For every $\rho_P$ for $P$ in $G$, there is a dual strategy $\widetilde{\rho}_A$ for $A$ in $\widetilde{G}$.*

*Proof:* We prove the two statements separately:
*(1):* Let $\rho_A$ be a strategy for $A$ in $G$. This strategy $\rho_A$ is characterized by

$$\rho_A(\epsilon) = (M_0, \cdot, 0) \quad \text{where } M_0 \in mod(I)$$
$$\rho_A((q, i)) = (M, q, i+1) \quad \text{where } M \in mod(\delta(q, w[i]))$$

By Prop. 13.1 there are choice functions satisfying

$$f_{M_0} : Mod(\widetilde{I}) \to Q \qquad Img\ f_{M_0} = M_0$$
$$f_{\langle M, q, a\rangle} : Mod(\widetilde{\delta}(q, a)) \to Q \qquad Img\ f_{\langle M, q, a\rangle} = M$$

Moreover, these functions are proper choice functions. We now define the dual strategy $\widetilde{\rho}_P$ for $P$ in $\widetilde{G}$ as follows:

$$\widetilde{\rho}_P((N_0, \cdot, 0)) = (f_{M_0}(N_0), 0)$$
$$\widetilde{\rho}_P((N, q, i+1)) = (f_{\langle M, q, a\rangle}(N), q, i+1)$$

where $M$ is the move of $A$ in $G$ from $(q, i)$: $\rho_A(q, i) = (M, q, i+1)$, and $a = w[i]$. Our choice of choice functions $f_{\langle M, q, a\rangle}$ guarantees that for every move of player $P$ from $M$, there is a move for player $A$ in $\widetilde{G}$ that, when followed by $f_{\langle M, q, a\rangle}$ results in the same state. The properties of $f_{M_0}$ and $f_{\langle M, q, a\rangle}$ ensure that the strategy $\widetilde{\rho}_P$ is proper.

We are ready to show that for every $G \cdot \rho_A$ play there is a $\widetilde{G} \cdot \rho_P$ play with the same trace, and vice-versa.

"→" Consider an arbitrary $G \cdot \rho_A$ play $\pi : V_0 v_0 V_1 v_1 \ldots$, and let $\rho_A(\epsilon) = (M_0, \cdot, 0)$ and $\rho_A(v_i) = (M_{i+1}, q_i, i+1)$. We use $q_i$ for $v_i = (q_i, i)$. Note that $q_{i+1} \in M_{i+1}$ because all moves of player $P$ in $\pi$ are legal moves. We create the $\widetilde{G} \cdot \rho_P$ play $\widetilde{\pi} : \widetilde{V_0}, \widetilde{v_0}, \widetilde{V_1}, \widetilde{v_1} \ldots$ as follows:

– $\widetilde{V_0} = (N_0, \cdot, 0)$ where $N_0$ is such that $f_{M_0}(N_0) = q_0$. One such $N_0$ exists since $Img\ f_{M_0} = M_0$ and $q_0 \in M_0$ (recall that $(q_0, 0)$ is the result of a move of $P$ in $G$ from $(M_0, \cdot, 0)$).

– From $(q_i, i)$, player $A$ chooses in $\widetilde{G}$ the position $(N_{i+1}, q_i, i+1)$, where $N_{i+1}$ is chosen such that $f_{\langle M_{i+1}, q, w[i] \rangle} = q_{i+1}$.

By induction, we show that $v_i = \widetilde{v}_i$. First, $\widetilde{v}_0 = \widetilde{\rho}_P((N_0, \cdot, 0)) = (f_{M_0}(N_0), 0) = (q_0, 0) = v_0$. Now, assume that for some $i$, $v_i = \widetilde{v}_i$. Then, $\widetilde{V}_i = (N_{i+1}, q_i, i+1)$, and $V_i = \rho_A(q_i, i) = (M_{i+1}, q_i, i+1)$. Now,

$$\widetilde{v}_{i+1} = \widetilde{\rho}_P(\widetilde{V}_i) = \widetilde{\rho}_P((N_{i+1}, q_i, i+1)) =$$
$$= (f_{\langle M_{i+1}, q_i, w[i] \rangle}(N_{i+1}), i+1) =$$
$$= (q_{i+1}, i+1) =$$
$$= v_{i+1}.$$

Hence, $trace(\pi) = trace(\widetilde{\pi})$.

"←" Consider an arbitrary $\widetilde{G} \cdot \rho_P$ play $\widetilde{\pi} : \widetilde{V_0} \widetilde{v_0} \widetilde{V_1} \widetilde{v_1} \ldots$, and let $q_i$ and $N_i$ be such that:

$$\widetilde{v}_i = (q_i, i) \quad \widetilde{V}_0 = (N_0, \cdot, 0) \quad \widetilde{V}_{i+1} = (N_{i+1}, q_i, i+1)$$

Since $\widetilde{\pi}$ is a $\widetilde{G} \cdot \rho_P$ play, it satisfies that

$$\widetilde{v}_{i+1} = \widetilde{\rho}_P(\widetilde{V}_{i+1}) = (f_{\langle M_{i+1}, q_i, w[i] \rangle}(N_{i+1}), i+1)$$

where $M_i$ is obtained from $\rho_A(q_i, i) = (M_{i+1}, i+1)$. Now, we define the play $\pi : V_0 v_0 V_1 v_1 \ldots$ as follows. First the move for $A$ is played according to $\rho_A$:

$$V_0 = \rho_A(\epsilon) = (M_0, \cdot, 0) \qquad V_{i+1} = \rho_A(v_i)$$

Then, we let the moves of $P$ to be:

$$v_0 = (q_0, 0) \qquad v_{i+1} = (q_{i+1}, i+1)$$

We only need to show that these moves for $P$ are legal. First, $q_0 = f_{M_0}(N_0)$, and since $Img\ f_{M_0} = M_0$ it follows that $q_0 \in M_0$, so moving from $V_0$ into $v_0$ is a legal move.
Moreover, $(q_{i+1} = f_{\langle M_{i+1}, q_i, w[i] \rangle}(N_i))$. Since $Img\ f_{\langle M_{i+1}, q_i, w[i] \rangle} = M_{i+1}$ it follows that $q_{i+1} \in M_{i+1}$, so again moving from $V_{i+1}$ into $v_{i+1}$ is a legal move. By construction, $trace(\pi) = trace(\widetilde{\pi})$ again.
*(2):* Assume now that $\rho_P$ is a (proper) strategy for $P$ in $G$. The strategy $\rho_P$ is characterized by

$$\rho_P((M_0, \cdot, 0)) = (q_0, 0) \qquad \rho_P((M, q, i)) = (q_i, i)$$

Since the strategy is proper there are proper choice functions:

$$g_0 : Mod(I) \to Q \qquad g_{q,i} : Mod(\delta(q, w[i])) \to Q$$

with

$$g_0 : Mod(I) \to Q$$
$$Img\ g_0 \in mod(\widetilde{I})$$
$$g_{q,i} : Mod(\delta(q, w[i])) \to Q \qquad (3)$$
$$Img\ g_{q,i} \in mod(\widetilde{\delta}(q, w[i]))$$

We define the strategy $\widetilde{\rho}_A$ for $A$ in $\widetilde{G}$ as follows:

$$\widetilde{\rho}_A(\epsilon) = Img\ g_0 \qquad \widetilde{\rho}_A((q, i)) = Img\ g_{q,i}$$

By (3), $\widetilde{\rho}_A$ is well defined. We show now that $\widetilde{\rho}_A$ and $\rho_P$ are dual strategies. First, consider $(q, i)$ an arbitrary state and $(M, q, i)$ a legal move for player $A$ in $G$. Player $P$ will move to $(q', i+1) = \rho_P((M, q, i))$ with $q' = g_{q,i}((M, q, i))$. In $\widetilde{G}$, player $A$ will move from $(q, i)$ into $(Img\ g_{q,i}, q, i)$. We let player $P$ move in $\widetilde{G}$ to $(q', i+1)$, which is legal, since $q' \in Img\ g_{q,i}$. Consider now an arbitrary state $(p, i)$ and the move of $A$ in $\widetilde{G}$: $\widetilde{\rho}_A((p, i)) = (Img\ g_{p,i}, p, i)$, and consider an arbitrary legal move for $P$, $(p', i+1)$, hence $p' \in Img\ g_{p,i}$. Consequently, there is an $M \in Mod(\delta(p, w[i]))$ such that $g_{p,i}((M, p, i)) = p'$. Let $A$ choose $(M, p, i)$ as the move from $(p, i)$, which is a legal move. Then, playing from $(M, p, i)$ in $G$ according to $\rho_P$, the resulting state is $(p', i+1)$. This shows that $\rho_A$ and $\widetilde{\rho}_P$ are dual strategies.

It is important to note that the moves of the players playing against the strategies are not restricted to follow proper strategies (give minimal models or be proper choice functions). Still, $\rho_A$ is winning precisely whenever $\rho_P$ is. □ ∎

The following theorem follows directly from Lemma 15 and Theorem 17. This theorem allows to reason about complementation simply by reasoning about traces of two automata with dual frames.

**Theorem 18** *Let $\mathcal{A}$ and $\widetilde{\mathcal{A}}$ be specular automata. Then $\mathcal{L}(\mathcal{A}) = \Sigma^\omega \setminus \mathcal{L}(\widetilde{\mathcal{A}})$.*

We show the correctness of Streett ranking algorithm.

**Lemma 19** *Let $\mathcal{G}$ be an accepting run, and let $\mathcal{G}'$ be a non-empty sub-graph of $\mathcal{G}$ with no G vertices and only infinite paths. Then, there is some node in $\mathcal{G}'$ that cannot access any B node.*

*Proof:* Consider, by contradiction that there is no one such a node in $\mathcal{G}' = (V', E')$, or equivalently, that all vertices in $\mathcal{G}'$ can access a $B$ node:

for all $\langle q, l \rangle \in V'$, there is some $\langle q', l' \rangle \in V'$
$\qquad$ with $q' \in B$ and $\langle q, l \rangle \to_{E'}^* \langle q', l' \rangle$.

Then, every node can be associated with a $B$ node by a map $next(\langle q, l \rangle)$ that returns one path to a $B$ reachable

node (for example, the shortest non-empty path to a $B$ state, and picking the smallest according to some lexicographic order among the shortest ones.) Then using induction define, starting from an arbitrary node $\langle q, l \rangle \in V'$, an infinite path in $\mathcal{G}'$ that visits infinitely $B$ nodes infinitely often by concatenating the paths returned by *next*. Let us call $\pi$ one such path.

Since $\pi(0) = \langle q, l \rangle$ is a node of $\mathcal{G}'$, and consequently a node of $\mathcal{G}$, $\pi(0)$ is reachable from some initial node by point 3 in the definition of a run. Let $\pi_{\text{pre}}$ be a finite path in $\mathcal{G}$ from a node $\langle q_0, 0 \rangle \in V$. The path $\pi_{\text{pre}}\pi$ is a trace in $\mathcal{G}$ that visits $G$ nodes finitely often (only nodes in $\pi_{\text{pre}}$ can possibly be $G$ nodes) and $B$ nodes infinitely often in $\pi$. This trace contradicts that $\mathcal{G}$ is an accepting run. $\quad\square\qquad\qquad\blacksquare$

We will use the following notation, for a given sub-graph $\mathcal{G}'$ of a run:

$$access(\mathcal{G}', \langle q, l \rangle) \stackrel{\text{def}}{=} \{\langle q', l' \rangle \mid \langle q, l \rangle \rightarrow^* \langle q', l' \rangle\}$$

$$finite(\mathcal{G}') \stackrel{\text{def}}{=} \{\langle q, l \rangle \mid access(\mathcal{G}', \langle q, l \rangle) \text{ is finite}\}$$

$$nobad(\mathcal{G}') \stackrel{\text{def}}{=} \{\langle q, l \rangle \mid access(\mathcal{G}', \langle q, l \rangle) \cap B = \emptyset\}$$

$$width(\mathcal{G}', l) \stackrel{\text{def}}{=} |\{\langle q, l \rangle \in \mathcal{G}'\}|$$

**Lemma 6** $\mathcal{G}$ *is an accepting run iff there is an odd* S$\{1\}$*-ranking for* $\mathcal{G}$.

*Proof:* We prove the two directions separately:

"$\Leftarrow$" Assume there is an odd S$\{1\}$-ranking $f$ for $\mathcal{G}$ and let $\pi$ be an arbitrary trace of $\mathcal{G}$. Since, $f$ is odd, either $\pi$ visits infinitely many $G$ states, in which case $\pi$ is accepting, or $f(\pi)$ converges to an odd value. In this second case, there is $l$ such that for all $l' > l$, $f(\pi(l')) = f(\pi(l))$ and $f(\pi(l))$ is odd. By definition of S$\{1\}$-ranking (point $(i)$), $\pi(l')$ cannot be a $B$ state, and consequently $\pi$ visits only finitely many $B$ states. Hence, $\pi$ is an accepting trace.

"$\Rightarrow$" Assume $\mathcal{G}$ is an accepting run for $\mathcal{A}$.

 – **Initial Stage** The construction of $f$ starts by removing from $\mathcal{G}$ all $G$ vertices. Let $V_G$ be $\{\langle q, l \rangle \mid q \in G\}$, then $f(\langle q, l \rangle) = 0$ for all $\langle q, l \rangle \in V_G$. Also, let $V_0 = finite(\mathcal{G} \setminus V_G)$, we let $f(\langle q, l \rangle = 0$ for all $\langle q, l \rangle \in V_0$. Also $\mathcal{G}_0 = \mathcal{G} \setminus (V_G \cup V_0)$, which contains the original graph except the $G$ nodes, and every node that reach $G$ nodes in all its outgoing paths.

 – **Incremental Stage** The algorithm proceeds in at most $n$ rounds, performing the following two operations in each round $k$. The round begins with subgraph $\mathcal{G}_{2k}$ of $\mathcal{G}$.

  – *Phase I*: Let $V_{2k+1} = nobad(\mathcal{G}_{2k})$. Then,

$$f(\langle q, l \rangle) = 2k + 1 \text{ for all } \langle q, l \rangle \in V_{2k+1}.$$
$$\mathcal{G}_{2k+1} = \mathcal{G}_{2k} \setminus V_{2k+1}.$$

 – *Phase II*: Let $V_{2k+2} = finite(\mathcal{G}_{2k+1})$. Then,

$$f(\langle q, l \rangle) = 2k + 2 \text{ for all } \langle q, l \rangle \in V_{2k+2}.$$
$$\mathcal{G}_{2k+2} = \mathcal{G}_{2k} \setminus V_{2k+2}.$$

The graphs $\mathcal{G}_0$ as well all graphs $\mathcal{G}_{2k+2}$ are either empty, or guaranteed to have only infinite paths, since all nodes that can only access finitely many nodes are removed. (a finite path ends in a node with no successor).

Hence, if $V_{2k}$ is non-empty Lemma 19 guarantees that $V_{2k+1}$ is non-empty as well: there is a node in $V_{2k+1}$ that accesses infinitely many vertices, but no $B$ node. In particular there is an infinite path that is removed in *Phase I*. Hence, for some level $l$, all $l' > l$ satisfy that

$$width(\mathcal{G}_{2k+1}, l') + 1 \leq width(\mathcal{G}_{2k}, l')$$

*Phase II* only removes nodes, so

$$width(\mathcal{G}_{2k+2}, l') \leq width(\mathcal{G}_{2k+1}, l')$$

Since, initially $width(\mathcal{G}, l) \leq n$ for all levels $l$, it follows that, at the end of round $k$, for a sufficiently large $l'$:

$$width(\mathcal{G}_{2k+1}, l') \leq n - (k+1)$$

Consequently, at the end of *Phase II* of round $n - 1$: $width(\mathcal{G}_{2n-1}, l') \leq 0$. All remaining vertices in $\mathcal{G}_{2n-1}$ can access only finitely many vertices. Hence $\mathcal{G}_{2n} = \emptyset$, and the algorithm terminates. Note that it is possible that $\mathcal{G}_{2k} = \emptyset$ in an earlier round, but guaranteed that after round $n$, $\mathcal{G}_{2n} = \emptyset$.

It remains to be seen that $f$ is indeed an odd S$\{1\}$-ranking.

**The function $f$ is a S$\{1\}$-ranking**: By construction, all $B$ vertices are marked in *Phase II* of some round because $B \cap nobad(\mathcal{G}_i) = \emptyset$, and hence receive an even value. Therefore, all $B$ nodes satisfy condition $(i)$ of the definition of S$\{1\}$-ranking. Now consider an arbitrary node $\langle q, l \rangle$. We consider three cases:

1) If $\langle q, l \rangle$ is removed in the Initial Stage then $q$ is either a $G$ node, in which case $(ii)$ holds trivially, or it is in $V_0$. In the latter case, all its outgoing paths hit a $G$ node in a finite number of steps, and all the intermediate nodes are mapped to 0. Hence, if $\langle q, l \rangle \rightarrow \langle q', l' \rangle$, then $f(\langle q', l' \rangle) = 0 = f(\langle q', l' \rangle)$ and $f(\langle q, l \rangle) \geq f(\langle q', l' \rangle)$, and condition $(ii)$ holds.
2) If $\langle q, l \rangle$ is removed in *Phase I* of round $k$, so $\langle q, l \rangle \in V_{2k+1}$. Then all its outgoing paths either hit a node removed in a previous round or are in $V_{2k+1}$. In both cases $\langle q, l \rangle \rightarrow \langle q', l' \rangle$ implies $f(\langle q, l \rangle) \geq f(\langle q', l' \rangle)$.
3) If $\langle q, l \rangle$ is removed in *Phase II* of round $k$, so $\langle q, l \rangle \in V_{2k+2}$. Then all its outgoing paths either hit a node removed in a previous round, or are in $V_{2k+1}$

or in $V_{2k+2}$. In all cases $\langle q, l \rangle \to \langle q', l' \rangle$ implies $f(\langle q, l \rangle) \geq f(\langle q', l' \rangle)$.

**The function $f$ is an odd S$\{1\}$-ranking:** Consider an arbitrary path $\pi$. If $\pi$ visits infinitely many $G$ nodes, then the condition for $f$ being odd on $\pi$ holds. If $\pi$ does not visit infinitely many $G$ nodes, then $f$ converges on $\pi$ to some value. This value cannot be even, because that would imply that all these infinitely many vertices are in some $V_{2k+2}$, but there are not infinite paths containing these kind of node: by construction all nodes label in *Phase II* have finite outgoing paths before changing ranking.

This finishes the proof. $\square$ ∎

**Theorem 7** *Let $\mathcal{A}$ be an* ASW$\{1\}$ *and $N$ the corresponding NBW. Then $w \in \mathcal{L}(\mathcal{A})$ if and only of $w \in \mathcal{L}(N)$.*

*Proof:* We prove the two directions separately:

"$\Rightarrow$" We assume $w \in \mathcal{L}(\mathcal{A})$ and show that $w \in \mathcal{L}(N)$. Let $\mathcal{G}$ be a run dag for $w$ on $\mathcal{A}$, and $f$ an odd S$\{1\}$-ranking. Consider the sequence $Q_0 Q_1 \ldots$ of states of $N$ induced by $\mathcal{G}$ and $f$ as $Q_i = (S_i, O_i, f_i)$ with the set of states in $S_i$:

$$S_i \quad = \quad \{q \mid \langle q, i \rangle \in \mathcal{G}\}$$

and the pending states $O_i$:

$$O_0 = \{p \mid \langle p, 0 \rangle \in \mathcal{G} \text{ with } p \notin G$$
$$\text{and } f(\langle p, 0 \rangle) \text{ is even}\}$$
$$O_{i+1} = \{p \mid \langle p, i+1 \rangle \in \mathcal{G} \text{ with } p \notin G$$
$$\text{and } f(\langle p, i+1 \rangle) \text{ even}\} \quad \text{if } O_i = \emptyset$$
$$O_{i+1} = \{p \mid \langle p, i+1 \rangle \in \mathcal{G}$$
$$\text{and } p \notin G, \text{ and for some } q \in O_i,$$
$$f(\langle p, i+1 \rangle) = f(\langle q, i \rangle)$$
$$\text{and } \langle q, i \rangle \to \langle p, i+1 \rangle\} \quad \text{if } O_i \neq \emptyset$$

and

$$f_i(q) \quad = \quad f(\langle q, i \rangle)$$

It is routine to check that $Q_0 Q_1 \ldots$ is a run. We show that this run is accepting for $N$. By contradiction, if $Q_0 Q_1 \ldots$ is non accepting, there exists $i$ such that, for all $j \geq i$, $Q_j \notin F_N$, hence $O_j \neq \emptyset$. By **D3** every $q_{j+1} \in O_{j+1}$ has a predecessor $q_j \in O_j$ with $f_j(q_j) = f_{j+1}(q_{j+1})$ being an even value by definition of $O_i$ above. Since, as shown above, every $O_j \neq \emptyset$, it follows that there is an infinite sub-dag of nodes in $\mathcal{G}$ of the form $\langle q_j, j \rangle$ with $f(\langle q_j, j \rangle)$ being even, and with infinitely many nodes having an incident edge. By König's lemma, since this dag is finitely branching, it has an infinite path, all whose nodes are assigned the same even value by $f$. This is a contradiction with $f$ being an odd S$\{1\}$-ranking for $\mathcal{G}$. Hence we find

that $Q_0 Q_1 \ldots$ is a run that accepts $w$ which shows $w \in \mathcal{L}(N)$.

"$\Leftarrow$" We assume now $w \in \mathcal{L}(N)$ and show that $w \in \mathcal{L}(\mathcal{A})$. Let $Q_0 Q_1 Q_2 \ldots$ be an accepting run for $w$ on $N$ and let $\mathcal{G} = (V, E)$ and $f$ be an induced run and function $V \to [2n]$. We conclude from **Q1** and **D2**, respectively, that properties $(i)$ and $(ii)$ of S$\{1\}$-ranking holds on $f$. Therefore $f$ is a S$\{1\}$-ranking. Now let us show that $f$ is an odd S$\{1\}$-ranking. To this end, consider an arbitrary path $\pi$ in $\mathcal{G}$ for which we will show that either condition $(i)$ or $(ii)$ of the definition of odd S$\{1\}$-ranking holds. If $\pi$ visits $G$ nodes infinitely often, then condition $(i)$ holds. Otherwise, there is an $i$ after which no more $G$ nodes are visited in $\pi$. Hence, since every node $\pi(i')$ with $i' > i + 1$ has a predecessor not in $G$, **D2** shows that $f$ converges on $\pi$ to some value. Let $j \geq i'$ be such that $f$ has converged already (i.e., $f(\pi(j')) = f(\pi(j))$ for all $j' \geq j$). Let $Q_k, Q_l \in F_N$ with $j \leq k < l$ be two accepting states in the run $Q_0 Q_1$ of $w$ on $N$. It must be the case that $\pi(k+1) \notin O_{k+1}$. Assume the contrary (i.e. $\pi(k+1) \in O_{k+1}$), since $\pi$ visits no $G$ node after $k$ and $f$ has converged, then we conclude by **D3** that $\pi(k') \in O_{k'}$ for all $k' > k$, hence that $O_l \neq \emptyset$, and finally that $Q_l \notin F_N$ by definition of $F_N$ which is a contradiction.
Also since $\pi(k+1) \notin O_{k+1}$ and $Q_k \in F_N$, **D3** shows that $f(\pi(k+1))$ is odd. Hence $f$ converges on $\pi$ to an odd value showing that condition $(ii)$ of the definition of S$\{1\}$-ranking holds.
This concludes that $f$ is an odd S$\{1\}$-ranking for $\mathcal{G}$. Finally Lem. 6 shows that $\mathcal{G}$ is an accepting run, hence that $w \in \mathcal{L}(\mathcal{A})$. $\square$

∎

**Lemma 20** *$\mathcal{G}$ is an accepting run iff there is a stratified odd* S$\{1\}$*-ranking for $\mathcal{G}$.*

*Proof:* We prove the two directions separately:

"$\Leftarrow$" Assume there is a stratified odd S$\{1\}$-ranking $\{f_j\}$ for $\mathcal{G}$ and let $\pi$ be an arbitrary trace of $\mathcal{G}$. Since, $\{f_j\}$ is odd, either $\pi$ visits infinitely many $G$ states, in which case $\pi$ is accepting, or $\pi$ converges to a stratum $S_j$ and $f_j(\pi)$ converges to an odd value. In this second case, there is $l$ such that for all $l' > l$, $f_j(\pi(l')) = f_j(\pi(l))$ and $f_j(\pi(l))$ is odd. By definition of stratified S$\{1\}$-ranking (point $(i)$), $\pi(l')$ cannot be a $B$ state, and consequently $\pi$ visits only finitely many $B$ states. Hence, $\pi$ is an accepting trace.

"$\Rightarrow$" Assume now that $\mathcal{G}$ is an accepting run for $\mathcal{A}$. The construction of each $f_j$ works at each stratum independently. Fix $S_j$. First, one removes all $G$ vertices and all those vertices not in $S_j$. The algorithm works exactly as with the proof of Lemma 6 by stages, at each stage first removing those states that cannot access $B$ nodes,

and then remove those states that only access finitely many nodes. Since at each stage one removes at least one element from all cuts at a sufficiently large $l$, at the width of elements from $S_j$ is at most $|S_j|$, the algorithm is guaranteed to finish in $|S_j|$ rounds, generating an odd S$\{1\}$-ranking for stratum $S_j$. It is routine to check that $\{f_j\}$ is indeed a stratified S$\{1\}$-ranking. $\square$

$\blacksquare$

**Theorem 11** *Let $\mathcal{A}$ be a stratified* ASW$\{1\}$ *and $N$ the corresponding NBW using stratified rankings. Then $w \in \mathcal{L}(\mathcal{A})$ if and only of $w \in \mathcal{L}(N)$.*

*Proof:* The proof is analogous to Theorem 11.

$\blacksquare$

Fig. 2 depicts the translation of RLTL into APW.
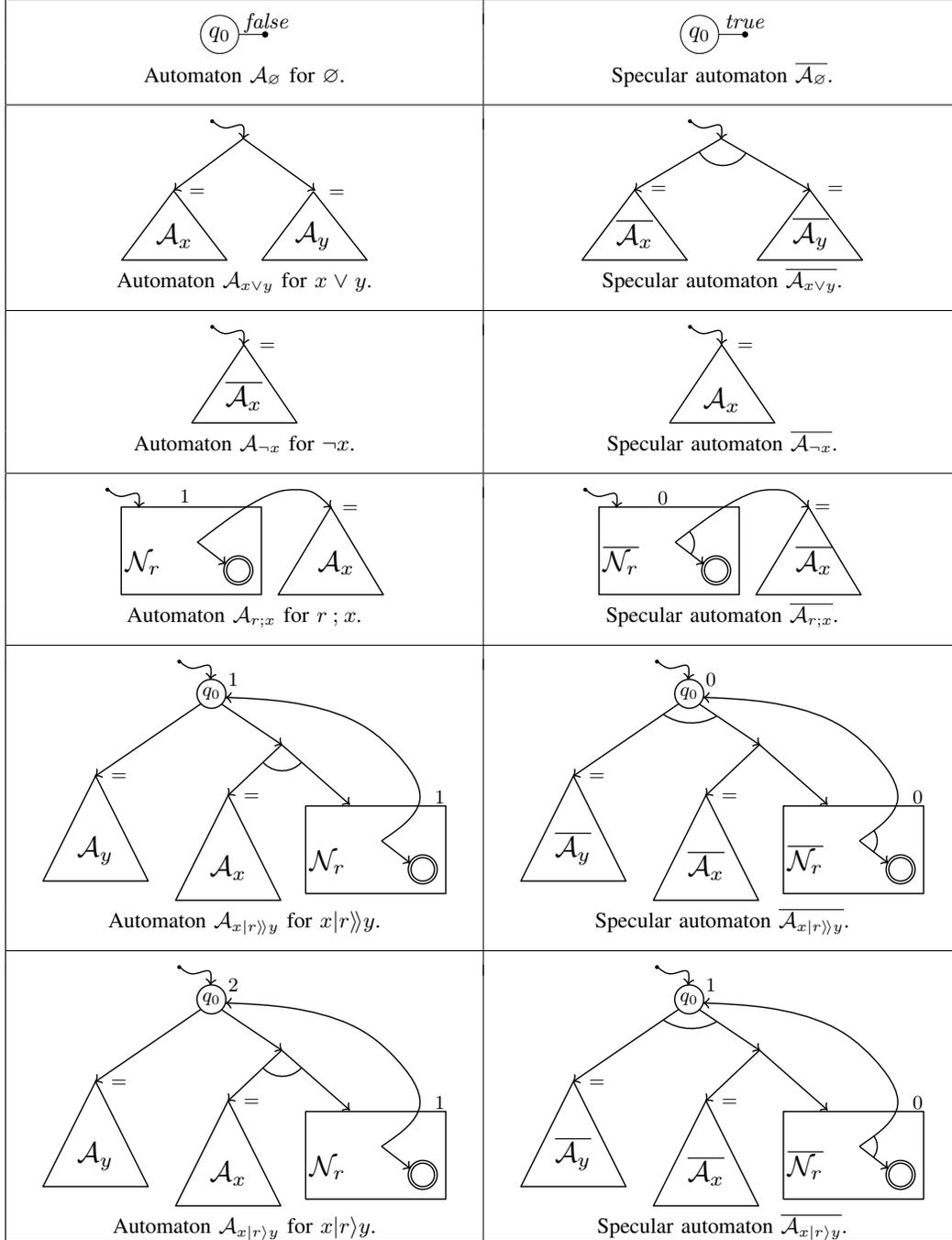
Figure 2. Specular automata pairs for $\varnothing$, $x \vee y$, $\neg x$, $x \,;\, y$, $x|r\rangle\!\rangle y$ and $x|r\rangle y$.