From ω-regular Expressions to Büchi Automata via Partial Derivatives

Peter Thiemann[1] and Martin Sulzmann[2]

[1] Faculty of Engineering, University of Freiburg
Georges-Köhler-Allee 079, 79110 Freiburg, Germany
`thiemann@acm.org`

[2] Faculty of Computer Science and Business Information Systems
Karlsruhe University of Applied Sciences
Moltkestrasse 30, 76133 Karlsruhe, Germany
`martin.sulzmann@hs-karlsruhe.de`

Abstract. We extend Brzozowski derivatives and partial derivatives from regular expressions to ω-regular expressions and establish their basic properties. We observe that the existing derivative-based automaton constructions do not scale to ω-regular expressions. We define a new variant of the partial derivative that operates on linear factors and prove that this variant gives rise to a translation from ω-regular expressions to nondeterministic Büchi automata.

Keywords: automata and logic, omega-regular languages, derivatives

1 Introduction

Brzozowski derivatives [3] and partial derivatives [2] are well-known tools to transform regular expressions to automata and to define algorithms for equivalence and containment on them [1]. Derivatives had quite some impact on the study of algorithms for regular languages on finite words and trees [9, 4], but they received less attention in the study of ω-regular languages.

While the extension of Brzozowski derivatives to ω-regular expressions is straightforward, the corresponding automaton construction does not easily extend to ω-automata. This observation leads Park [6] to suggest resorting to a different acceptance criterion based on transitions. Redziejowski [7] remarks that "the automaton constructed from the derivative has, in general, too few transitions as well as too few states." As a remedy, Redziejowski presents a construction of a deterministic automaton where states are certain combinations of derivatives with a non-standard transition-based acceptance criterion. In subsequent work, Redziejowski [8] improves on this construction by lowering the number of states and by simplifying some technical details. To the best of our knowledge, these papers [7, 8] are the only attempts to construct ω-automata using derivatives.

In comparison, our construction and proof are much simpler, we gain new insights into the structure of linear factors as a stepping stone to partial derivatives, and we obtain a standard nondeterministic Büchi automaton. Because Brzozowski derivatives invariably lead to deterministic automata, we analyze Antimirov's partial derivatives and identify linear factors as a suitable structure on which we base the construction of a nondeterministic automaton.

Overview

Section 2 reviews the basic definitions for (ω-) regular expressions and (Büchi) automata. Section 3 reviews Brzozowski derivatives, extends them to ω-regular expressions, and demonstrates the failure of the automaton construction based on Brzozowski derivatives. Section 4 introduces Antimirov's linear factors and partial derivatives, extends them to ω-regular expressions, establishes their basic properties, and demonstrates the failure of the automaton construction based on partial derivatives. Section 5 introduces a new notion of partial derivative that operates directly on linear factors of an ω-regular expression, defines a Büchi automaton on that basis, and proves its construction correct.

2 Preliminaries

An alphabet Σ is a finite set of symbols. The set Σ^* denotes the set of finite words over Σ, $\varepsilon \in \Sigma^*$ stands for the empty word; the set Σ^ω denotes the set of infinite words over Σ. For $u \in \Sigma^*$, we write $u \cdot v$ for the concatenation of words; if $v \in \Sigma^*$, then $u \cdot v \in \Sigma^*$; if $v \in \Sigma^\omega$, then $u \cdot v \in \Sigma^\omega$. Concatenation extends to sets of words as usual: $U \cdot V = \{u \cdot v \mid u \in U, v \in V\}$ where $U \subseteq \Sigma^*$ and $V \subseteq \Sigma^*$ or $V \subseteq \Sigma^\omega$.

Given a language $U \subseteq \Sigma^*$ and $W \subseteq \Sigma^*$ or $W \subseteq \Sigma^\omega$, the *left quotient* $U^{-1}W = \{v \mid \exists u \in U : uv \in W\}$. It is a subset of Σ^* or Σ^ω depending on W. For a singleton language $U = \{u\}$, we write $u^{-1}W$ for the left quotient.

Definition 1. *The set R_Σ of regular expressions over Σ is defined inductively by $\mathbf{1} \in R_\Sigma$, $\mathbf{0} \in R_\Sigma$, $\Sigma \subseteq R_\Sigma$, and, for all $r, s \in R_\Sigma$, $(r.s)$, $(r + s)$, $r^* \in R_\Sigma$. The explicit bracketing guarantees unambiguous parsing of regular expressions.*

Definition 2. *The language denoted by a regular expression is defined inductively by $\mathcal{L} : R_\Sigma \to \wp(\Sigma^*)$ as usual. $\mathcal{L}(\mathbf{1}) = \{\varepsilon\}$. $\mathcal{L}(\mathbf{0}) = \{\}$. $\mathcal{L}(a) = \{a\}$ (singleton word) for each $a \in \Sigma$. $\mathcal{L}(r.s) = \mathcal{L}(r) \cdot \mathcal{L}(s)$. $\mathcal{L}(r + s) = \mathcal{L}(r) \cup \mathcal{L}(s)$. $\mathcal{L}(r^*) = \{u_1 \dots u_n \mid n \in \mathbb{N}, u_i \in \mathcal{L}(r)\}$.*

Definition 3. *The operations $\odot, \oplus : R_\Sigma \times R_\Sigma \to R_\Sigma$ are* smart concatenation *and* smart union *constructors for regular expressions.*

$$r \odot s = \begin{cases} \mathbf{0} & r = \mathbf{0} \lor s = \mathbf{0} \\ r & s = \mathbf{1} \\ s & r = \mathbf{1} \\ (r.s) & \textit{otherwise} \end{cases} \qquad r \oplus s = \begin{cases} r & s = \mathbf{0} \\ s & r = \mathbf{0} \\ r & r = s \\ (r + s) & \textit{otherwise} \end{cases}$$

Lemma 4. *For all r, s: $\mathcal{L}(r \odot s) = \mathcal{L}(r.s)$; $\mathcal{L}(r \oplus s) = \mathcal{L}(r + s)$.*

Definition 5. *A regular expression r is* nullable *if $\varepsilon \in \mathcal{L}(r)$. The function $N : R_\Sigma \to \{\mathbf{0}, \mathbf{1}\}$ detects nullable expressions: $N(\mathbf{1}) = \mathbf{1}$. $N(\mathbf{0}) = \mathbf{0}$. $N(a) = \mathbf{0}$. $N(r.s) = N(r) \odot N(s)$. $N(r + s) = N(r) \oplus N(s)$. $N(r^*) = \mathbf{1}$.*

Lemma 6. *For all $r \in R_\Sigma$. $N(r) = \mathbf{1}$ iff $\varepsilon \in \mathcal{L}(r)$.*

Definition 7. *The set R^ω_Σ of ω-regular expressions over Σ is defined by $\mathbf{0} \in R^\omega_\Sigma$; for all $\alpha, \beta \in R^\omega_\Sigma$, $(\alpha + \beta) \in R^\omega_\Sigma$; for all $r \in R_\Sigma$ and $\alpha \in R^\omega_\Sigma$, $(r.\alpha) \in R^\omega_\Sigma$; for all $s \in R_\Sigma$, if $\varepsilon \notin \mathcal{L}(s)$, then $s^\omega \in R^\omega_\Sigma$.*

Remark 8. Definition 7 is equivalent to an alternative definition often seen in the literature, where an ω-regular-expression has a *sum-of-product form* $\sum_{i=1}^n (r_i.s_i^\omega)$ with $\varepsilon \notin \mathcal{L}(s_i)$. An easy induction shows that every α can be rewritten in this form: cases $\mathbf{0}$, $(\alpha+\beta)$, s^ω: immediate; case $(r.\alpha)$: by induction, α can be written as $\sum_{i=1}^n (r_i.s_i^\omega)$, distributivity and associativity yield $\sum_{i=1}^n (r.r_i).s_i^\omega$ for $(r.\alpha)$. When convenient for a proof, we assume that an expression is in sum-of-product form.

Definition 9. *The language denoted by an ω-regular expression is defined inductively by $\mathcal{L}^\omega : R^\omega_\Sigma \to \wp(\Sigma^\omega)$: $\mathcal{L}^\omega(\mathbf{0}) = \emptyset$. $\mathcal{L}^\omega(\alpha + \beta) = \mathcal{L}^\omega(\alpha) \cup \mathcal{L}^\omega(\beta)$. $\mathcal{L}^\omega(r.\alpha) = \mathcal{L}(r) \cdot \mathcal{L}^\omega(\alpha)$. $\mathcal{L}^\omega(s^\omega) = \{v_1 v_2 \cdots \mid \forall i \in \mathbb{N} : v_i \in \mathcal{L}(s)\}$.*

Definition 10. *A (nondeterministic) finite automaton (NFA) is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ where Q is a finite set of states, Σ an alphabet, $\delta : Q \times \Sigma \to \wp(Q)$ the transition function, $q_0 \in Q$ the initial state, and $F \subseteq Q$ the set of final states.*

Let $w = a_0 \dots a_{n-1} \in \Sigma^*$ be a word. A run *of $\mathcal{A}$ on w is a sequence $q_0 \dots q_n$ such that, for all $0 \le i < n$, $q_{i+1} \in \delta(q_i, a_i)$. The run is* accepting *if $q_n \in F$. The language $\mathcal{L}(\mathcal{A}) = \{w \in \Sigma^* \mid \exists$ accepting run of $\mathcal{A}$ on $w\}$ is recognized by $\mathcal{A}$.*

The automaton $\mathcal{A}$ is deterministic if $|\delta(q, a)| = 1$, for all $q \in Q$, $a \in \Sigma$.

Definition 11. *A (nondeterministic) Büchi-automaton (NBA) is a tuple $\mathcal{B} = (Q, \Sigma, \delta, Q_0, F)$ where Q is a finite set of states, Σ an alphabet, $\delta : Q \times \Sigma \to \wp(Q)$ the transition function, $Q_0 \subseteq Q$ the set of initial states, and $F \subseteq Q$ the set of accepting states.*

Let $w = (a_i)_{i \in \mathbb{N}} \in \Sigma^\omega$ be an infinite word. A run *of $\mathcal{B}$ on w is an infinite sequence of states $(q_i)_{i \in \mathbb{N}}$ such that $q_0 \in Q_0$ and for all $i \in \mathbb{N}$: $q_{i+1} \in \delta(q_i, a_i)$.*

A run $(q_i)_{i \in \mathbb{N}}$ of $\mathcal{B}$ is accepting *if there exists a strictly increasing sequence $(n_j)_{j \in \mathbb{N}}$ such that $q_{n_j} \in F$, for all $j \in \mathbb{N}$. The language $\mathcal{L}^\omega(\mathcal{B}) = \{w \in \Sigma^\omega \mid \exists$ accepting run of $\mathcal{B}$ on $w\}$ is recognized by $\mathcal{B}$. The automaton $\mathcal{B}$ is deterministic if $|Q_0| = 1$ and $|\delta(q, a)| = 1$, for all $q \in Q$, $a \in \Sigma$.*

3 Regular Expressions to Finite Automata

The textbook construction to transform a regular expression into a finite automaton is taken from Kleene's work [5]. However, there is an alternative approach based on Brzozowski's idea of derivatives for regular expressions.

Given a regular expression r and a symbol $a \in \Sigma$, the derivative $r' = d_a(r)$ is a regular expression such that $\mathcal{L}(r') = \{w \mid aw \in \mathcal{L}(r)\}$, the left quotient of $\mathcal{L}(r)$ by the symbol a. The derivative can be defined symbolically by induction on regular expressions.

Definition 12 (Brzozowski derivative [3]).

$$
\begin{array}{ll}
d_a(\mathbf{0}) = \mathbf{0} & d_a(r.s) \quad = (d_a(r) \odot s) \oplus (N(r) \odot d_a(s)) \\
d_a(\mathbf{1}) = \mathbf{0} & d_a(r+s) = d_a(r) \oplus d_a(s) \\
d_a(b) = \begin{cases} \mathbf{1} & a = b \\ \mathbf{0} & a \neq b \end{cases} & d_a(r^*) \quad = d_a(r) \odot r^*
\end{array}
$$

Brzozowski proved the following representation theorem that factorizes a regular language into its ε-part and the quotient languages with respect to each symbol of the alphabet.

Theorem 13 (Representation [3]). $\mathcal{L}(r) = \mathcal{L}(N(r)) \cup \bigcup_{a\in\Sigma}\{a\} \cdot \mathcal{L}(d_a(r))$

He further proved that there are only finitely many different regular expressions derivable from a given regular expression. This finiteness result considers expressions modulo a *similarity relation* $\approx$ that contains (at least) associativity, commutativity, and idempotence of the $+$ operator as well as considering $\mathbf{0}$ as the neutral element. We further assume *associativity of concatenation.*

Definition 14 (Similarity). *Similarity* $\approx\ \subseteq R_\Sigma \times R_\Sigma$ *is the smallest compatible relation that encompasses the following elements for all* $r, s, t \in R_\Sigma$.

$$
(r+s)+t \approx r+(s+t) \qquad r+s \approx s+r \qquad r+r \approx r \qquad r+\mathbf{0} \approx r \qquad (r.s).t \approx r.(s.t)
$$

Similarity extends to $\approx^\omega\ \subseteq R^\omega_\Sigma \times R^\omega_\Sigma$ *as the smallest compatible relation that contains the following elements for all* $\alpha, \beta, \gamma \in R^\omega_\Sigma$.

$$
(\alpha+\beta)+\gamma \approx^\omega \alpha+(\beta+\gamma) \qquad \alpha+\beta \approx^\omega \beta+\alpha \qquad \alpha+\alpha \approx^\omega \alpha \qquad \alpha+\mathbf{0} \approx^\omega \alpha
$$

$$
(r.s).\alpha \approx^\omega r.(s.\alpha) \qquad r \approx s \Rightarrow (r.t^\omega) \approx^\omega (s.t^\omega) \qquad s \approx t \Rightarrow (r.s^\omega) \approx^\omega (r.t^\omega)
$$

Definition 15. *The derivative operator extends to words* $w \in \Sigma^*$ *by* $d_\varepsilon(r) = r$, $d_{aw}(r) = d_w(d_a(r))$ *and to sets of words* $W \subseteq \Sigma^*$ *by* $d_W(r) = \{d_w(r) \mid w \in W\}$.

Theorem 16 (Finiteness [3]). *For each* $r \in R_\Sigma$, *the set* $d_{\Sigma^*}(r)/_\approx$ *is finite.*

Taken together, these two theorems yield an effective transformation from a regular expression to a deterministic finite automaton.

Theorem 17 (DFA from regular expression [3]). *Define the DFA* $\mathcal{D}(r) = (Q, \Sigma, \delta, q_0, F)$ *where* $Q = d_{\Sigma^*}(r)/_\approx$, *for all* $s \in Q, a \in \Sigma$: $\delta(s, a) = \{d_a(s)\}$, $q_0 = r$, $F = \{s \in Q \mid N(s) = \mathbf{1}\}$. *Then* $\mathcal{D}(r)$ *is a deterministic finite automaton and* $\mathcal{L}(\mathcal{D}(r)) = \mathcal{L}(r)$.

Let's try to apply an analogous construction to ω-regular expressions. We first straightforwardly extend the definition of derivatives [7].

Definition 18 (Brzozowski derivative for ω-regular expressions).

$$
\begin{array}{ll}
d_a(\mathbf{0}) \qquad = \mathbf{0} & d_a(r.\alpha) = (d_a(r) \odot \alpha) \oplus (N(r) \odot d_a(\alpha)) \\
d_a(\alpha+\beta) = d_a(\alpha) \oplus d_a(\beta) & d_a(s^\omega) \ = d_a(s) \odot s^\omega
\end{array}
$$

Lemma 19. $\mathcal{L}^\omega(d_a(\alpha)) = a^{-1}\mathcal{L}^\omega(\alpha)$

Lemma 20. $\mathcal{L}^\omega(\alpha) = \bigcup_{a\in\Sigma}\{a\}\cdot\mathcal{L}^\omega(d_a(\alpha))$.

The operation $d_w(\Sigma)$ also yields finitely many derivatives modulo similarity (extended to $R^\omega_\Sigma \times R^\omega_\Sigma$ in the obvious way), but applying Brzozowski's automata construction analogously results in a *deterministic* Büchi automaton, which is known to be weaker than its nondeterministic counterpart.

Example 21. Consider the ω-regular expression $(a+b)^*.b^\omega$ that describes the language of infinite words that contain only finitely many as. It is known that this language cannot be recognized with a deterministic Büchi automaton. Applying Brzozowski's automaton construction analogously yields the following:

$$\begin{array}{ll@{\qquad}l}
Q &= \{q_0, q_1\} & \delta(q_0, a) = q_0 \\
q_0 &= (a+b)^*.b^\omega & \delta(q_0, b) = q_1 \\
q_1 &= (a+b)^*.b^\omega + b^\omega & \delta(q_1, a) = q_0 \\
Q_0 &= \{q_0\} & \delta(q_1, b) = q_1
\end{array}$$

As all states "contain" the looping expression b^ω, it is not clear which states should be accepting. Furthermore, the automaton is deterministic, so it cannot recognize $\mathcal{L}^\omega((a+b)^*.b^\omega)$, regardless.

4 Partial Derivatives

As Brzozowski's construction only results in a deterministic automaton, we next consider a construction that yields a nondeterministic automaton. It is based on Antimirov's *partial derivatives* [2]. The partial derivative $\partial_a(r)$ of a regular expression r with respect to a is a *set* of regular expressions $\{s_1,\dots,s_n\}$ such that $\bigcup_{i=1}^n \mathcal{L}(s_i) = \{w \mid aw \in \mathcal{L}(r)\}$. As a stepping stone to their definition, Antimirov introduces *linear factors* of regular expressions. A linear factor is a pair of a first symbol that can be consumed by the expression and a "remaining" regular expression. The following definition corresponds to Antimirov's definition [2, Definition 2.4], but we replace the smart constructor $\odot$ for concatenation (that elides ε) by plain concatenation to simplify the finiteness proof.

Definition 22 (Linear factors [2]).

$$\begin{array}{l@{\qquad}ll}
\textsc{lf}(\mathbf{0}) = \{\} & \textsc{lf}(r.s) & = \textsc{lf}(r).s \cup N(r) \odot \textsc{lf}(s) \\
\textsc{lf}(\mathbf{1}) = \{\} & \textsc{lf}(r+s) & = \textsc{lf}(r) \cup \textsc{lf}(s) \\
\textsc{lf}(a) = \{\langle a, \mathbf{1}\rangle\} & \textsc{lf}(r^*) & = \textsc{lf}(r).r^*
\end{array}$$

where

$$\begin{array}{l@{\qquad}l}
\mathbf{0} \odot F = \{\} & \mathbf{1} \odot F = F \\
\langle a, r\rangle.s = \langle a, r.s\rangle & F.s = \{f.s \mid f \in F\}
\end{array}$$

Defining the language of a linear factor and a set of linear factors F by

$$\mathcal{L}(\langle a, r\rangle) = a\cdot\mathcal{L}(r) \qquad \mathcal{L}(F) = \bigcup\{\mathcal{L}(f) \mid f \in F\}$$

we can prove the following results about linear factors by induction on r.

Lemma 23. *If* $\langle a, r' \rangle \in \textsc{lf}(r)$*, then* $a \cdot \mathcal{L}(r') \subseteq \mathcal{L}(r)$*.*

Lemma 24. *If* $av \in \mathcal{L}(r)$*, then there exists* $\langle a, r' \rangle \in \textsc{lf}(r)$ *such that* $v \in \mathcal{L}(r')$*.*

Lemma 25. *For all* r*,* $\mathcal{L}(\textsc{lf}(r)) = \mathcal{L}(r) \setminus \{\varepsilon\}$*.*

We label the symbol for partial derivative with ^{A} to signify Antimirov's definition. In Section 5, we define a different version of the partial derivative.

Definition 26 (Partial derivative [2]).

$$\partial^A_a(r) = \{r' \mid \langle a, r' \rangle \in \textsc{lf}(r), r' \neq \mathbf{0}\}$$

Partial derivatives extend to words and sets of words $W \subseteq \Sigma^*$ *in the usual way:*

$$\partial^A_\varepsilon(r) = \{r\} \quad \partial^A_{aw}(r) = \bigcup\{\partial^A_w(r') \mid r' \in \partial^A_a(r)\} \quad \partial^A_W(r) = \bigcup\{\partial^A_w(r) \mid w \in W\}$$

Antimirov proves [2, Theorem 3.4] that the set of all partial derivatives of a given regular expression is finite. While his definition of linear factors uses the smart concatenation $\odot$, the finiteness proof does not rely on it: it approximates smart concatenation by the standard concatenation operator.

Theorem 27. *For any* $r \in R_\Sigma$*,* $|\partial^A_{\Sigma^+}(r)| \leq ||r||$ *where* $||r||$ *is the alphabetic width of* r *(i.e., the number of occurrences of symbols from* Σ *in* r*).*

Furthermore, a language can be represented from its partial derivatives.

Lemma 28. $\mathcal{L}(r) = \mathcal{L}(N(r)) \cup \bigcup_{a \in \Sigma} a \cdot \mathcal{L}(\sum \partial^A_a(r))$*.*

Here, we write $\sum\{r_i \mid 1 \leq i \leq n\}$ for $r_1 + \cdots + r_n$, if $n > 0$, or for $\mathbf{0}$ if $n = 0$. We also have the following characterization.

Lemma 29. *If* $\partial^A_a(r) = \{s_1, \ldots, s_n\}$*, then* $\bigcup_{i=1}^n \mathcal{L}(s_i) = \{w \mid aw \in \mathcal{L}(r)\}$*.*

Antimirov defines a *nondeterministic* automaton for $\mathcal{L}(r)$ as follows.

Theorem 30 (NFA from regular expression [2]). *Define the NFA* $\mathcal{N}(r) = (Q, \Sigma, \delta, q_0, F)$ *where* $Q = \partial^A_{\Sigma^*}(r)$*, for all* $s \in Q$*,* $a \in \Sigma$*:* $\delta(s, a) = \partial^A_a(s)$*,* $q_0 = r$*,* $F = \{s \in Q \mid N(s) = \mathbf{1}\}$*. Then* $\mathcal{N}(r)$ *is an NFA and* $\mathcal{L}(r) = \mathcal{L}(\mathcal{N}(r))$*.*

Lemma 31. $w \in \mathcal{L}(r)$ *iff* $\varepsilon \in \bigcup N(\partial^A_w(r))$*.*

Proof. By induction on w.

Base case: $\varepsilon \in \mathcal{L}(r)$ iff $\varepsilon \in N(r)$ by Lemma 6. The claim follows because $N(r) = \bigcup N(\{r\}) = \bigcup N(\{\partial^A_\varepsilon(r)\})$.

Inductive case: Suppose that $aw \in \mathcal{L}(r)$ and $\partial^A_a(r) = \{r_1, \ldots, r_k\}$. By Lemma 29, $\bigcup_i \mathcal{L}(r_i) = \{v \mid av \in \mathcal{L}(r)\}$ so that $w \in \bigcup_i \mathcal{L}(r_i)$, i.e., $\exists i$: $w \in \mathcal{L}(r_i)$. By induction, $\varepsilon \in \bigcup N(\partial^A_w(r_i)) \subseteq N(\partial^A_{aw}(r))$.

For the reverse direction, suppose that $\varepsilon \in \bigcup N(\partial^A_{aw}(r)) = N(\bigcup\{\partial^A_w(r') \mid r' \in \partial^A_a(r), r' \neq \mathbf{0}\})$. Hence, there exists $r' \in \partial^A_a(r)$ such that $\varepsilon \in N(\partial^A_w(r'))$. By induction, $w \in \mathcal{L}(r')$ and thus, by Lemma 29, $aw \in \mathcal{L}(r)$. ☐

To scale the definition from Theorem 30 to ω-regular expressions we need to extend Definition 22.

Definition 32 (ω-Linear factors). *Define* $\mathrm{LF} : R^\omega_\Sigma \to \Sigma \times R^\omega_\Sigma \times \{0,1\}$ *by*

$$\begin{array}{ll}\mathrm{LF}(\mathbf{0}) \quad = \emptyset & \mathrm{LF}(r.\alpha) = \mathrm{LF}(r).\alpha \times \{0\} \cup N(r) \odot \mathrm{LF}(\alpha) \\ \mathrm{LF}(\alpha+\beta) = \mathrm{LF}(\alpha) \cup \mathrm{LF}(\beta) & \mathrm{LF}(s^\omega) \ = \mathrm{LF}(s).s^\omega \times \{1\}\end{array}$$

Compared to the linear factor of a regular expression, an ω-linear factor is a *triple* of a next symbol, an ω-regular expression, and a bit that indicates whether the factor resulted from unrolling an ω-iteration.

For an ω-linear factor define $\mathcal{L}^\omega(\langle a, \beta, g\rangle) = a \cdot \mathcal{L}^\omega(\beta)$ and for a set F of ω-linear factors accordingly $\mathcal{L}^\omega(F) = \bigcup\{\mathcal{L}^\omega(f) \mid f \in F\}$.

Each ω-regular language can be represented by its set of ω-linear factors. Compared to the finite case (Lemma 25), the empty string need not be considered because it is not an element of Σ^ω.

Lemma 33. *For all* α, $\mathcal{L}^\omega(\alpha) = \mathcal{L}^\omega(\mathrm{LF}(\alpha))$.

Proof. By induction on α. We only show one illustrative case.

Case s^ω: let $w \in \mathcal{L}^\omega(s^\omega)$. By definition, $w = v_0 v_1 \ldots$ with $\varepsilon \neq v_i \in \mathcal{L}(s)$, for all $i \in \mathbb{N}$. Suppose that $w = aw'$. Then $v_0 = av'_0$. Show that there exists $f = \langle a, s', 1\rangle \in \mathrm{LF}(s^\omega)$ such that $w' \in \mathcal{L}^\omega(s')$.

If $\mathrm{LF}(s^\omega) = \emptyset$, then $\mathcal{L}^\omega(s^\omega) = \emptyset$, which contradicts the existence of w.

Suppose next that all ω-linear factors have the form $\langle b, s', 1\rangle$ for some $b \neq a$. But then we obtain a contradiction to $av'_0 \in \mathcal{L}(s)$.

Thus, we need to examine the ω-linear factors of the form $\langle a, s'.s^\omega, 1\rangle \in \mathrm{LF}(s).s^\omega \times \{1\} = \mathrm{LF}(s^\omega)$. By Lemma 24, there must be a linear factor $\langle a, s'\rangle \in \mathrm{LF}(s)$ such that $v'_0 \in \mathcal{L}(s')$. Hence, $w' = v'_0 v_1 \cdots \in \mathcal{L}^\omega(s'.s^\omega)$ and thus $w = aw' \in \mathcal{L}^\omega(\langle a, s'.s^\omega, 1\rangle) \subseteq \mathcal{L}^\omega(\mathrm{LF}(s^\omega))$.

For the reverse direction, suppose that $w \in \mathcal{L}^\omega(\mathrm{LF}(s^\omega))$. Then there exists $\langle a, s'\rangle \in \mathrm{LF}(s)$ and hence $\langle a, s'.s^\omega, 1\rangle \in \mathrm{LF}(s).s^\omega \times \{1\} = \mathrm{LF}(s^\omega)$ such that $w \in a \cdot \mathcal{L}^\omega(s'.s^\omega) = a \cdot \mathcal{L}(s') \cdot \mathcal{L}^\omega(s^\omega)$. By Lemma 23, $a \cdot \mathcal{L}(s') \subseteq L(s)$ so that $w \in a \cdot \mathcal{L}(s') \cdot \mathcal{L}^\omega(s^\omega) \subseteq \mathcal{L}(s) \cdot \mathcal{L}^\omega(s^\omega) = \mathcal{L}^\omega(s^\omega)$. ☐

Using the obvious extension of the partial derivative operator, Lemma 29 extends to the ω-regular case.

Lemma 34. *If* $\partial^A_a(\alpha) = \{\beta_1, \ldots, \beta_n\}$, *then* $\bigcup_{i=1}^n \mathcal{L}^\omega(\beta_i) = \{w \mid aw \in \mathcal{L}^\omega(\alpha)\}$.

However, again it is not clear how to extend Antimirov's automaton construction to Büchi automata. The critical part is to come up with a characterization of the accepting states.

Example 35. Let $\alpha = (a+b)^*.b^\omega$ as in the previous example. Constructing an automaton analogously to Theorem 30 yields

$$\begin{array}{ll} q_0 \ = (a+b)^*.b^\omega & \delta(q_0, a) = \{q_0\} \\ q_1 \ = b^\omega & \delta(q_0, b) = \{q_0, q_1\} \\ Q \ = \{q_0, q_1\} & \delta(q_1, a) = \{\} \\ Q_0 = \{q_0\} & \delta(q_1, b) = \{q_1\} \end{array}$$

Thus, adopting the set of accepting states $F = \{q_1\}$ yields a nondeterministic Büchi automaton that accepts exactly $\mathcal{L}(\alpha)$. Apparently, we may categorize states of the form s^ω as accepting.

While the previous example is encouraging in that the construction leads to a correct automaton, a simple transformation of the ω-regular expression shows that the criterion for accepting states is not sufficient in the general case.

Example 36. Let $\beta = (a + b)^*.(b.b^*)^\omega$. This expression recognizes the same language as the expression of the previous example.

$$\begin{aligned}
\partial_a(\beta) &= \partial_a((a+b)^*.(b.b^*)^\omega) \\
&= \partial_a((a+b)^*).(b.b^*)^\omega \cup \partial_a(b.b^*) \odot (b.b^*)^\omega \\
&= \{(a+b)^*.(b.b^*)^\omega\} \\
\partial_b(\beta) &= \partial_b((a+b)^*.(b.b^*)^\omega) \\
&= \partial_b((a+b)^*).(b.b^*)^\omega \cup \partial_b(b.b^*) \odot (b.b^*)^\omega \\
&= \{(a+b)^*.(b.b^*)^\omega\} \cup \{b^*.(b.b^*)^\omega\} \\
\partial_b(b^*.(b.b^*)^\omega) &= \partial_b(b^*).(b.b^*)^\omega \cup \partial_b(b.b^*) \odot (b.b^*)^\omega \\
&= \{b^*.(b.b^*)^\omega\} \cup \{b^*.(b.b^*)^\omega\} \\
\partial_a(b^*.(b.b^*)^\omega) &= \{\}
\end{aligned}$$

Thus, we cannot construct a Büchi automaton for $\mathcal{L}^\omega(\beta)$ by simply classifying the states of the form s^ω as accepting because there are no such states in this automaton: thus, the automaton would accept the empty language.

Alternatively, we might be tempted to consider all expressions of the form $r.s^\omega$ where r is nullable as accepting states. This choice would classify *all states* in the example as accepting, which would cause the automaton to wrongly accept the infinite word a^ω.

5 NBA from ω-Linear Factors

The difficulties with the previous examples demonstrate that Antimirov's partial derivatives cannot be used directly as the states of a Büchi automaton. To fix these problems, we base our construction directly on the ω-linear factors that arise as an intermediate step in Antimirov's work.

Definition 37. *For an ω-linear factor (and a set F of ω-linear factors) define the partial derivative* as a set of ω-linear factors*:*

$$\partial_b(\langle a, \beta, g\rangle) = \begin{cases} \{\} & a \neq b \\ \mathrm{LF}(\beta) & a = b \end{cases} \qquad \partial_b(F) = \bigcup_{f \in F} \partial_b(f)$$

Define further the extension to words $\partial_\varepsilon(F) = F$ and $\partial_{aw}(F) = \partial_w(\partial_a(F))$ and the extension to sets of finite words $W \subseteq \Sigma^*$: $\partial_W(F) = \bigcup\{\partial_w(F) \mid w \in W\}$.

This definition of the derivative serves as the basis for defining the set of states $\mathcal{Q}(\alpha)$ for the NBA, which we are aiming to construct.

Definition 38. *Define $\mathcal{Q}(\alpha)$ inductively as the smallest set such that* $\textsc{lf}(\alpha) \subseteq \mathcal{Q}(\alpha)$ *and, for each* $a \in \Sigma$, $\partial_a(\mathcal{Q}(\alpha)) \subseteq \mathcal{Q}(\alpha)$.

Lemma 39. *If* $\langle a, \beta, g\rangle \in \mathcal{Q}(\alpha)$, *then* $\exists w \in \Sigma^*$ *such that* $\langle a, \beta, g\rangle \in \partial_w(\textsc{lf}(\alpha))$.

Proof. By induction on the construction of $\mathcal{Q}(\alpha)$.

Base case: $\langle a, \beta, g\rangle \in \textsc{lf}(\alpha) = \partial_\varepsilon(\textsc{lf}(\alpha))$.

Inductive case: $\langle a, \beta, g\rangle \in \partial_a(f)$, for some $f \in \mathcal{Q}(\alpha)$ and $a \in \Sigma$. By induction, $f \in \partial_w(\textsc{lf}(\alpha))$, for some w, and thus $\langle a, \beta, g\rangle \in \partial_a(\partial_w(\textsc{lf}(\alpha))) = \partial_{aw}(\textsc{lf}(\alpha))$. □

Proposition 40. *For each ω-regular expression α, $\mathcal{Q}(\alpha)$ is finite.*

Proof. We prove that $\mathcal{Q}(\alpha) \subseteq \Sigma \times \partial^A_{\Sigma^+}(\alpha) \times \{0, 1\}$.

Suppose that $\langle a, \alpha', g\rangle \in \mathcal{Q}(\alpha)$. There are two cases. If $\langle a, \alpha', g'\rangle \in \textsc{lf}(\alpha)$, then $a \in \Sigma$ and $\alpha' \in \partial^A_a(\alpha) \subseteq \partial^A_{\Sigma^+}(\alpha)$.

If $\langle a, \alpha', g'\rangle \in \partial_b(\langle b, \beta, g\rangle)$ for some $\langle b, \beta, g\rangle \in \mathcal{Q}(\alpha)$, then there exists some $w \in \Sigma^*$ such that $\beta \in \partial^A_{wb}(\alpha)$ and $\langle a, \alpha', g\rangle \in \textsc{lf}(\beta)$. By definition, $\alpha' \in \partial^A_{wba}(\alpha) \subseteq \partial^A_{\Sigma^+}(\alpha)$.

By Theorem 27, $|\partial^A_{\Sigma^+}(\alpha)|$ is finite and so is $|\mathcal{Q}(\alpha)| \leq |\Sigma| \cdot |\partial^A_{\Sigma^+}(\alpha)| \cdot 2$. □

Given this finiteness, we construct a non-deterministic Büchi automaton from an ω-regular expression as follows.

Definition 41 (NBA from ω-regular expression). *Define the NBA* $\mathcal{B}(\alpha) = (Q, \Sigma, \delta, Q_0, F)$ *by* $Q = \mathcal{Q}(\alpha)$; $Q_0 = \textsc{lf}(\alpha)$; $F = \{\langle a, \beta, g\rangle \in Q \mid g = 1\}$; *and* $\delta(f, a) = \partial_a(f)$.

Example 42. Consider (again) $\alpha = (a + b)^*.b^\omega$.

$$\begin{aligned}
\textsc{lf}(\alpha) &= \textsc{lf}((a+b)^*).b^\omega \cup \textsc{lf}(b^\omega)\\
&= \{\langle a, (a+b)^*.b^\omega, 0\rangle, \langle b, (a+b)^*.b^\omega, 0\rangle, \langle b, b^\omega, 1\rangle\}\\
&= Q = Q_0
\end{aligned}$$

$$\begin{aligned}
&\delta(\langle b, b^\omega, 1\rangle, a) &&= \{\}\\
&\delta(\langle b, b^\omega, 1\rangle, b) &&= \{\langle b, b^\omega, 1\rangle\}\\
&\delta(\langle a, (a+b)^*.b^\omega, 0\rangle, a) &&= \textsc{lf}((a+b)^*.b^\omega) = Q\\
&\delta(\langle a, (a+b)^*.b^\omega, 0\rangle, b) &&= \{\}\\
&\delta(\langle b, (a+b)^*.b^\omega, 0\rangle, a) &&= \{\}\\
&\delta(\langle b, (a+b)^*.b^\omega, 0\rangle, b) &&= \textsc{lf}((a+b)^*.b^\omega) = Q
\end{aligned}$$

Accepting states: $F = \{\langle b, b^\omega, 0\rangle\} = \textsc{lf}(b^\omega)$.

The resulting automaton properly accepts $\mathcal{L}^\omega(\alpha)$.

Example 43. Next consider $\beta = (a + b)^*.(b.b^*)^\omega$.

$$\begin{aligned}
\textsc{lf}(\beta) &= \textsc{lf}((a+b)^*).(b.b^\omega) \times \{0\} \cup \textsc{lf}((b.b^*)^\omega)\\
&= \textsc{lf}(a+b).(a+b)^*.(b.b^\omega) \times \{0\} \cup \textsc{lf}(b.b^*).(b.b^*)^\omega \times \{1\}\\
&= \{\langle a, (a+b)^*.(b.b^\omega), 0\rangle, \langle b, (a+b)^*.(b.b^\omega), 0\rangle\}\\
&\quad \cup \textsc{lf}(b).b^*.(b.b^*)^\omega \times \{1\}\\
&= \{\langle a, (a+b)^*.(b.b^\omega), 0\rangle, \langle b, (a+b)^*.(b.b^\omega), 0\rangle\\
&\quad , \langle b, b^*.(b.b^*)^\omega, 1\rangle\}\\
&= \{\langle a, \beta, 0\rangle, \langle b, \beta, 0\rangle, \langle b, b^*.(b.b^*)^\omega, 1\rangle\}
\end{aligned}$$

$$\begin{aligned}
\delta(\langle a, \beta\rangle, a) &= \textsc{lf}(\beta) \\
\delta(\langle b, \beta\rangle, b) &= \textsc{lf}(\beta) \\
\delta(\langle b, b^*.(b.b^*)^\omega\rangle, b) &= \textsc{lf}(b^*.(b.b^*)^\omega) \\
&= \textsc{lf}(b^*).(b.b^*)^\omega \times \{1\} \cup \textsc{lf}((b.b^*)^\omega) \\
&= \textsc{lf}(b).b^*.(b.b^*)^\omega \times \{1\} \cup \textsc{lf}(b.b^*).(b.b^*)^\omega \times \{1\} \\
&= \textsc{lf}(b).b^*.(b.b^*)^\omega \times \{1\} \cup \textsc{lf}(b).b^*.(b.b^*)^\omega \times \{1\} \\
&= \{\langle b, b^*.(b.b^*)^\omega, 1\rangle\} \\
&= \textsc{lf}((b.b^*)^\omega)
\end{aligned}$$

Accepting states:

$$F = \{\langle b, b^*.(b.b^*)^\omega, 1\rangle\} = \textsc{lf}((b.b^*)^\omega)$$

The resulting automaton properly accepts $\mathcal{L}^\omega(\beta)$ with the same number of states as in the previous example.

It remains to prove the correctness of the construction in Definition 41.

Theorem 44. *For all $\alpha \in R^\omega_\Sigma$: $\mathcal{L}^\omega(\alpha) = \mathcal{L}^\omega(\mathcal{B}(\alpha))$.*

We start with some technical lemmas.

Lemma 45. *For all $v \neq \varepsilon$, $\partial_v(\textsc{lf}(s^\omega)) = \partial_v(\textsc{lf}(s.s^\omega))$.*

Proof. By definition of ω-regular expressions, $\varepsilon \notin \mathcal{L}(s)$ that is $N(s) = \mathbf{0}$.
Observe that $\textsc{lf}(s^\omega) = \textsc{lf}(s).s^\omega \times \{1\}$,
whereas $\textsc{lf}(s.s^\omega) = \textsc{lf}(s).s^\omega \times \{0\} \cup N(s) \odot \textsc{lf}(s^\omega) = \textsc{lf}(s).s^\omega \times \{0\}$.
Because $v \neq \varepsilon$, it must be that $v = av'$, for some a.
Hence, $\partial_a(\textsc{lf}(s^\omega)) = \bigcup\{\textsc{lf}(s'.s^\omega) \mid \langle a, s'\rangle \in \textsc{lf}(s)\} = \partial_a(\textsc{lf}(s.s^\omega))$.
Hence, $\partial_{av'}(\textsc{lf}(s^\omega)) = \partial_{av'}(\textsc{lf}(s.s^\omega))$ □

The next lemma is our workhorse in proving that $\mathcal{L}^\omega(\alpha)$ is contained in the language of $\mathcal{B}(\alpha)$.

Lemma 46. *If $u \in \mathcal{L}(r)$, then $\textsc{lf}(\alpha) \subseteq \partial_u(\textsc{lf}(r.\alpha))$.*

Proof. Induction on r.
Case $r = \mathbf{0}$: contradiction because $\mathcal{L}(\mathbf{0}.\alpha) = \{\}$.
Case $r = \mathbf{1}$: Then $u = \varepsilon$ and $\partial_\varepsilon(\textsc{lf}(\mathbf{1}.\alpha)) = \textsc{lf}(\mathbf{1}.\alpha) = \textsc{lf}(\alpha)$.
Case $r = a$: Then $u = a$ and $\partial_a(\textsc{lf}(a.\alpha)) = \partial_a(\langle a, \alpha, 0\rangle) = \textsc{lf}(\alpha)$.
Case $r = r_1.r_2$: Then $u = u_1u_2$ with $u_1 \in \mathcal{L}(r_1)$ and $u_2 \in \mathcal{L}(r_2)$.
By similarity (cf. Definition 14), $\textsc{lf}((r_1.r_2).\alpha) = \textsc{lf}(r_1.(r_2.\alpha))$.
By induction on r_1, $\textsc{lf}(r_2.\alpha) \subseteq \partial_{u_1}(\textsc{lf}(r_1.(r_2.\alpha)))$.
By induction on r_2,

$$\textsc{lf}(\alpha) \subseteq \partial_{u_2}(\textsc{lf}(r_2.\alpha)) \subseteq \partial_{u_2}(\partial_{u_1}(\textsc{lf}(r_1.(r_2.\alpha)))) = \partial_u(\textsc{lf}(r.\alpha))$$

Case $r = r_1 + r_2$: Assume that $u \in \mathcal{L}(r_1) \subseteq \mathcal{L}(r)$. By induction, $\textsc{lf}(\alpha) \subseteq \partial_u(\textsc{lf}(r_1.\alpha)) \subseteq \partial_u(\textsc{lf}(r.\alpha))$. The case for r_2 is analogous.

Case $r = r_1^*$: Consider

$$\text{LF}(r_1^*.\alpha) = \text{LF}(r_1^*).\alpha \cup N(r_1^*) \odot \text{LF}(\alpha) = \text{LF}(r_1).r_1^*.\alpha \cup \text{LF}(\alpha)$$

For $u \in \Sigma^*$, $\partial_u(\text{LF}(r_1^*.\alpha)) = \partial_u(\text{LF}(r_1).r_1^*.\alpha) \cup \partial_u(\text{LF}(\alpha))$.
If $u \in \mathcal{L}(r)$, then $u = u_1 \dots u_n$, for some $n \in \mathbb{N}$, where all $u_i \neq \varepsilon$. Continue by induction on n.
If $n = 0$, $u = \varepsilon$, then clearly $\text{LF}(\alpha) \subseteq \partial_\varepsilon(\text{LF}(r_1^*.\alpha))$.
Otherwise,

$$\begin{aligned}
&\partial_u(\text{LF}(r_1^*.\alpha)) \\
&= \partial_{u_1 \dots u_n}(\text{LF}(r_1^*.\alpha)) \\
&= \partial_{u_2 \dots u_n}(\partial_{u_1}(\text{LF}(r_1^*.\alpha))) \\
&= \partial_{u_2 \dots u_n}(\partial_{u_1}(\text{LF}(r_1).r_1^*.\alpha) \cup \partial_{u_1}(\text{LF}(\alpha))) \\
&\supseteq \partial_{u_2 \dots u_n}(\partial_{u_1}(\text{LF}(r_1).r_1^*.\alpha)) \\
&\supseteq \partial_{u_2 \dots u_n}(\text{LF}(r_1^*.\alpha)) \\
&\quad \text{by induction} \\
&\supseteq \text{LF}(\alpha)
\end{aligned}$$

□

The next, final lemma is our workhorse in proving that the language of $\mathcal{B}(\alpha)$ is contained in $\mathcal{L}^\omega(\alpha)$. The proof requires the extra bit in the ω-linear factors.

Lemma 47. *Let $q_0 q_1 \dots q_n$ be a prefix of an accepting run of $\mathcal{B}(r.s^\omega)$ on $uw = a_1 \dots a_n w$ where $q_n \in \text{LF}(s^\omega)$, but $q_i \notin \text{LF}(s^\omega)$, for $0 \leq i < n$. Then $u \in \mathcal{L}(r)$.*

Proof. Induction on n.

Case 0; $u = \varepsilon$: $q_0 \in \text{LF}(s^\omega) \cap \text{LF}(r.s^\omega)$ because $q_0 \in Q_0$. Now $\text{LF}(s^\omega) = \text{LF}(s).s^\omega \times \{1\}$ and $\text{LF}(r.s^\omega) = \text{LF}(r).s^\omega \times \{0\} \cup N(r) \odot \text{LF}(s).s^\omega \times \{1\}$.

If $N(r) = \mathbf{1}$, then $q_0 \in \text{LF}(s^\omega) \subseteq \text{LF}(r.s^\omega)$ and $u = \varepsilon \in \mathcal{L}(r)$.

If $N(r) = \mathbf{0}$, then $q_0 \in \text{LF}(s).s^\omega \times \{1\} \cap \text{LF}(r).s^\omega \times \{0\} = \emptyset$ so that this case is not possible. (Without the extra bit in LF, there may be common linear factors if $\mathcal{L}(r) \cap \mathcal{L}(s^*) \neq \emptyset$.)

Case $n > 0$: $u = au'$ and $q_1 \in \partial_a(q_0)$. As $q_0 \in Q_0 = \text{LF}(r.s^\omega) = \text{LF}(r).s^\omega \times \{0\} \cup N(r) \odot \text{LF}(s^\omega)$ but $q_0 \notin \text{LF}(s^\omega)$, it must be that $q_0 \in \text{LF}(r).s^\omega \times \{0\}$.

Thus, $q_1 \in \partial_a(\text{LF}(r).s^\omega \times \{0\})$, so that there is a linear factor $\langle a, r' \rangle \in \text{LF}(r)$ such that $q_1 \in \text{LF}(r'.s^\omega)$.

Thus, $q_1 \dots q_n$ is a prefix of an accepting run of $\mathcal{B}(r'.s^\omega)$[1] on $u'w = a_2 \dots a_n w$ where $q_n \in \text{LF}(s^\omega)$, but $q_i \notin \text{LF}(s^\omega)$, for $1 \leq i < n$. By induction, $u' \in \mathcal{L}(r')$ so that $u = au' \in \mathcal{L}(r)$ by Lemma 23. □

Proof (of Theorem 44). It is sufficient to consider $\alpha = r.s^\omega$.

Case "$\subseteq$": Let $w \in \mathcal{L}^\omega(r.s^\omega)$. Then $w = uv_0v_1 \dots$ where $u \in \mathcal{L}(r)$ and $\varepsilon \neq v_i \in \mathcal{L}(s)$, for $i \in \mathbb{N}$.

Let $Q_0 = \text{LF}(r.s^\omega)$. By Lemma 46, $\text{LF}(s^\omega) \subseteq \partial_u(\text{LF}(r.s^\omega)) = \delta(Q_0, u)$.

[1] While the set Q' of states of $\mathcal{B}(r'.s^\omega)$ is a subset of the states Q of $\mathcal{B}(r.s^\omega)$, it is easy to see that the states $q_1 \dots q_n$ as well as the remaining states $q_{n+1}q_{n+2}\dots$ of the accepting run are all elements of Q'.

Furthermore, for each $i \in \mathbb{N}$, by Lemmas 45 and 46,

$$\partial_{v_i}(\textsc{lf}(s^\omega)) = \partial_{v_i}(\textsc{lf}(s.s^\omega)) \supseteq \textsc{lf}(s^\omega)$$

Hence, there exists a run of $\mathcal{B}(\alpha)$ which visits states from $F = \textsc{lf}(s^\omega)$ infinitely often.

Case "$\supseteq$": Suppose that $a_0 a_1 \cdots \in \mathcal{L}^\omega(\mathcal{B}(\alpha))$. Hence, there is a run $q_0 q_1 \cdots \in Q^\omega$ and a strictly increasing sequence $(n_i)_{i\in\mathbb{N}} \in \mathbb{N}^\omega$ such that, for all $j \in \mathbb{N}$, $q_j \in F$ iff $\exists i : j = n_i$.

Let $q = q_{n_0}$ be the first accepting state in the run and let $u = a_0 \dots a_{n_0-1}$. By construction of $\mathcal{B}(\alpha)$, $q \in \delta(Q_0, u)$ and $q \in \textsc{lf}(s^\omega) = F$. By Lemma 47, $u \in \mathcal{L}(r)$.

Next, for each $i \in \mathbb{N}$, define $v_i = a_{n_i} \dots a_{n_{i+1}}$ so that $w = uv_0v_1 \dots$.

For each i, $q_{n_i} \in F$ and $\varepsilon \neq v_i = b_i v'_i$. By construction $q_{n_i+1} \in \delta(q_{n_i}, b_i)$ so that $q_{n_i+1} \dots q_{n_{i+1}} \dots$ is a prefix of an accepting run of $\mathcal{B}(q_{n_i+1})$ where $q_{n_i+1} = \langle b_i, s'.s^\omega, 1\rangle$, for some $\langle b_i, s'\rangle \in \textsc{lf}(s)$. By Lemma 47, $v'_i \in \mathcal{L}(s')$ so that $v_i = b_i v'_i \in \mathcal{L}(s)$ by Lemma 23.

Taken together, we have shown that $w \in \mathcal{L}(r) \cdot \{v_0 v_1 \cdots \mid v_i \in \mathcal{L}(s)\} = \mathcal{L}^\omega(r.s^\omega)$. □

We believe that it is possible to reduce the number of states of $\mathcal{B}(\alpha)$ by a factor of $|\Sigma|$ by merging suitable linear factors, but we leave this for future work.

References

1. Antimirov, V.M.: Rewriting regular inequalities. In: Proc. of FCT'95. LNCS, vol. 965, pp. 116–125. Springer-Verlag (1995)
2. Antimirov, V.M.: Partial derivatives of regular expressions and finite automaton constructions. Theoretical Computer Science 155(2), 291–319 (1996)
3. Brzozowski, J.A.: Derivatives of regular expressions. J. ACM 11(4), 481–494 (1964)
4. Caron, P., Champarnaud, J.M., Mignot, L.: Partial derivatives of an extended regular expression. In: LATA. LNCS, vol. 6638, pp. 179–191. Springer (2011)
5. Kleene, S.C.: Representation of events in nerve nets and finite automata. Automata Studies (1956)
6. Park, D.: Concurrency and automata on infinite sequences. In: Theoretical Computer Science, 5th GI-Conference, Karlsruhe, Germany, March 23-25, 1981, Proceedings. LNCS, vol. 104, pp. 167–183. Springer (1981)
7. Redziejowski, R.R.: Construction of a deterministic ω-automaton using derivatives. Informatique Théorique et Applications 33(2), 133–158 (1999)
8. Redziejowski, R.R.: An improved construction of deterministic omega-automaton using derivatives. Fundam. Inform. 119(3-4), 393–406 (2012)
9. Rosu, G., Viswanathan, M.: Testing extended regular language membership incrementally by rewriting. In: Proc. of RTA'03. LNCS, vol. 2706, pp. 499–514. Springer (2003)