# An interval temporal logic characterization of extended $\omega$-regular languages[☆]

Dario Della Monica[a,*], Angelo Montanari[a], Pietro Sala[b]

[a] *Università di Udine, Udine (Italy)*
[b] *Università di Verona, Verona (Italy)*

---

**Abstract**

Some extensions of $\omega$-regular languages have been proposed in the literature to express asymptotic properties of $\omega$-words which are not captured by $\omega$-regular languages. They include $\omega B$-regular languages, that extend $\omega$-regular languages with boundedness, $\omega S$-regular languages, that enrich $\omega$-regular ones with strong unboundedness, $\omega BS$-regular languages, that combine $\omega B$- and $\omega S$-regular ones, and $\omega T$-regular languages, that include meaningful languages which are not $\omega BS$-regular. Formal definitions of extended $\omega$-regular languages have been given in terms of both suitable classes of automata and extended $\omega$-regular expressions, while satisfactory temporal logic counterparts are still missing. In this paper, we give a characterization of them in terms of interval temporal logics by providing an explicit encoding of expressions into formulas.

*Keywords:* Temporal logic, $\omega$-regular expressions, expressiveness
*2010 MSC:* 00-01, 99-00

---

## 1. Introduction

In this paper, we explore the relationships between extended $\omega$-regular languages and temporal logic by providing an encoding of language expressions into formulas of suitable interval temporal logics.

[☆]This work was supported by the Department Strategic Plan (PSD) of the University of Udine – Interdepartmental Project on Artificial Intelligence (2021-25) and by the GNCS 2022 project "Elaborazione del Linguaggio Naturale e Logica Temporale per la Formalizzazione di Testi". The paper is an extended and revised version of [1] and [2].

[*]`dario.dellamonica@uniud.it` (corresponding author)

$\omega$-regular languages are a natural setting for the specification and verification of nonterminating finite-state systems. Since the seminal work by Büchi, McNaughton, and Rabin in the sixties [3, 4, 5], much has been done on the theory and the application of $\omega$-regular languages. Equivalent characterizations of $\omega$-regular languages have been given in terms of formal languages, automata, and classical and temporal logic. However, while the consensus on what features regular languages of finite words must exhibit is unanimous (it largely relies on Myhill-Nerode theorem [6]), the notion of $\omega$-regular languages is more controversial. In the last years, it has been shown that $\omega$-regular languages can be extended in meaningful ways, preserving their decidability and (some of their) closure properties [7, 8, 9, 10].

The proposed extensions pair the Kleene star $(.)^*$ with some variants of it. The bounding exponent $B$ of $\omega B$-regular languages, denoted by $(.)^B$, constrains the language $L$ in the expression $L^B$ to be iterated only a bounded number of times, the bound being fixed for the whole $\omega$-word [10]. The unbounding exponent $S$ of $\omega S$-regular languages, denoted by $(.)^S$, when applied to a language $L$, forces the number of iterations of $L$ to tend to infinity, that is, for every $k > 0$, it constrains the number of times the argument $L$ is repeated at most $k$ times to be finite [10]. The exponents $(.)^B$ and $(.)^S$ can be freely mixed in $\omega BS$-regular languages (the combination of $\omega B$- and $\omega S$-regular ones) [10]. The union of the classes of $\omega B$- and $\omega S$-regular languages is properly included in the class of $\omega BS$-regular ones [9], as witnessed by the $\omega BS$-regular language $\mathcal{L}$ over the alphabet $\{a, b\}$ consisting of those $\omega$-words featuring infinitely many occurrences of $b$'s and such that the sequence of the distances between consecutive $b$'s contains only finitely many values occurring infinitely often. As it will become clear when formal definitions will be given, such a language is captured by the $\omega BS$-regular expression $(a^B b + a^S b)^\omega$, but it cannot be encoded by means of the union of languages generated by $\omega B$- and $\omega S$-regular expressions. The existence of non-$\omega BS$-regular languages (like, e.g., the complement $\bar{\mathcal{L}}$ of $\mathcal{L}$ above) that are the complements of some $\omega BS$-regular ones and express natural asymptotic behaviours motivated the search for other classes of extended $\omega$-regular languages.

In [11], $\omega T$-regular languages, which are based on a different extension of $(.)^*$, denoted by $(.)^T$, and include meaningful non-$\omega BS$-regular languages such as $\bar{\mathcal{L}}$, have been studied.

Besides those in terms of $\omega B$-, $\omega S$-, $\omega BS$-, and $\omega T$-regular expressions, equivalent characterizations of the above languages have been given in terms of automata and classical logic (extensions of the monadic second-order theory of one successor S1S). Temporal logic counterparts are still missing. As a matter of fact, encodings of $\omega B$- and $\omega S$-regular languages in interval temporal logics were proposed in [1] and [12], respectively. Unfortunately, as we will show later, both of them are flawed. Here, we provide a fix, and, in addition, give an interval temporal logic characterization of $\omega T$-regular languages.

Interval temporal logic (ITL) is a general framework for representing and reasoning about time. ITLs are characterized by high expressiveness (they overcome various limitations of point-based temporal logics) and high computational complexity (formulas translate into binary relations over the underlying linear order). One of the first ITLs proposed in the literature is Moszkowski's Propositional ITL (PITL), which was successfully applied to hardware specification and verification [13]. The application of interval-based formalisms to temporal reasoning in AI was first investigated by Allen [14]. A systematic logical study of interval representation and reasoning started with Halpern and Shoham's work on the logic HS featuring one modality for each Allen relation [15]. While decidability is a common feature of point-based temporal logics, undecidability rules over ITLs. The first such undecidability results were obtained for PITL by Moszkowski [16]. General undecidability results for HS are given in [15] and further sharpened in [17]. For a long time, these results have discouraged the search for practical applications and further theoretical investigation on ITLs. This bleak picture started lightening up in the last years when various non-trivial decidable fragments of HS have been identified (see, e.g., [18, 19, 20, 21]).

In this paper, we focus on the HS fragment $AB$, whose modalities correspond to Allen's relations *meets* (modality $\langle A\rangle$) and *begun by* (modality $\langle B\rangle$), and some extensions of it with modalities for the inverse relations *met by* (modality $\langle\bar{A}\rangle$)

and *begins* (modality $\langle \bar{B} \rangle$). In [1], Montanari and Sala have proved that regular (resp., $\omega$-regular) languages can be defined in $AB$, interpreted over finite linear orders (resp., $\mathbb{N}$).[1] Here, we show that extended $\omega$-regular languages can be captured by suitable extensions of $AB$, by means of formulas that pair atomic propositions corresponding to the elements of the alphabet of the extended $\omega$-regular language and auxiliary atomic propositions. More precisely, we show that (i) $\omega B$-regular languages can be encoded in $AB\bar{A}$, that extends $AB$ with the past modality $\langle \bar{A} \rangle$, (ii) $\omega S$-regular languages can be encoded in $AB$ enriched with an equivalence relation $\sim$, namely $AB{\sim}$, and (iii) $\omega T$-regular languages are captured by $AB\bar{A}{\sim}$, the extension of $AB$ with both modality $\langle \bar{A} \rangle$ and equivalence relation $\sim$. A distinctive feature of the encodings is that they do not resort to any counter, that is, checking the satisfaction of boundedness/unboundedness conditions in ITL does not require the precision in length measurements given by counters (in fact, some abstraction over counters, that allows one to consider orders of magnitude rather than exact values, is exploited also in the automaton-based characterizations of extended $\omega$-regular languages).

The paper is organized as follows. In Section 2, we provide some background knowledge on extended $\omega$-regular languages and ITLs. Then, in Section 3, we prove some useful properties of extended ($\omega$-)regular languages. Next, in Section 4, we describe in detail the encodings of regular and $\omega$-regular languages in $AB$. In Section 5, we point out the main issues that must be addressed to lift the encoding of Section 4 to extended $\omega$-regular languages. Finally, in Sections 6, 7, and 8, we show how to enrich the encoding of $\omega$-regular languages into $AB$ in order to capture the increased expressive power of extended $\omega$-regular languages. Conclusions provide an assessment of the work done and outline directions of future work.

[1] In fact, they make use of the logic $AB\bar{B}$, but it can be shown that modality $\langle \bar{B} \rangle$ simplifies the encoding, but it is inessential.

## 2. Preliminaries

In this section, we provide some background knowledge about extended $\omega$-regular languages and interval temporal logics. Let $\mathbb{N}$ be the set of natural numbers and $\mathbb{N}_{>0} = \mathbb{N} \setminus \{0\}$. Further, for an infinite sequence $\vec{u}$ and $i \in \mathbb{N}_{>0}$, we denote by $u_i$ its $i$-th element.

### 2.1. Extended $\omega$-regular languages

In the following, we give a short account of extended $\omega$-regular languages in terms of the extended $\omega$-regular expressions that define them. For a detailed one, we refer the reader to [11]. Extended $\omega$-regular expressions are built on top of the corresponding extended regular ones, just as $\omega$-regular expressions are built on top of regular ones. Intuitively, extended regular expressions differ from regular ones as they allow constructors from the set $\{(.)^B, (.)^S, (.)^T\}$. Formally, let $\Sigma$ be a finite, nonempty alphabet. Then, *$BST$-regular expressions* over $\Sigma$ are captured by the grammar:

$$e \ ::= \ \emptyset \ \mid \ a \ \mid \ e \cdot e \ \mid \ e + e \ \mid \ e^* \ \mid \ e^B \ \mid \ e^S \ \mid \ e^T, \text{ where } a \in \Sigma.$$

Sometimes, we will omit the operator $\cdot$, thus writing, e.g., $ee$ for $e \cdot e$.

In the following, we provide the semantics of $BST$-regular expressions. Unlike standard regular expressions, the semantics of extended regular ones is given in terms of languages of infinite sequences of finite words, that make it possible to force suitable constraints that capture the intended meaning of $(.)^B$, $(.)^S$, and $(.)^T$. Intuitively, according to its standard semantics, a regular expression $e$ corresponds to a regular language of finite words, say it $\mathcal{L}_e^{RE}$. According to the semantics given in this paper, instead, the regular expression $e$ identifies the set $\mathcal{L}(e)$ of infinite sequences whose elements are finite words from $\mathcal{L}_e^{RE}$, i.e., $\mathcal{L}(e) = \{\vec{w} \mid w_i \in \mathcal{L}_e^{RE}\}$. As an example, we have that $\mathcal{L}(a) = \{(a, a, a, \ldots)\}$ and $\mathcal{L}(a^*) = \{\vec{w} \mid w_i \text{ is a sequence of } a\text{'s of any length}\}$. Roughly speaking, the constructor $(.)^*$ produces sequences of words by grouping together arbitrarily many consecutive elements of a sequence generated by the argument language. The constructors $(.)^B$, $(.)^S$, and $(.)^T$ behave similarly, the difference being that the number of consecutive elements that are grouped together is not arbitrary, but suitably constrained in the limit (see the formal definition below for more details). As an example, we have that $\mathcal{L}(a^B) = \{\vec{w} \mid w_i$ is a sequence of $a$'s and there is an upper bound to the length of $w_i$, for all $i\}$.

In order to ease the definition of the semantics of extended regular expressions, we introduce the notions of *concatenation* and *shuffle* of two word sequences, as well as the one of *$f$-aggregation* of a word sequence, for a given nondecreasing function $f : \mathbb{N} \to \mathbb{N}_{>0}$, with $f(0) = 1$. The first two notions are used in the semantic clauses for the operators $\cdot$ and $+$, respectively, while the third one comes handy in the definition of the semantics for the constructors $(.)^*$, $(.)^B$, $(.)^S$, and $(.)^T$.

The *concatenation* of two word sequences $\vec{u}$ and $\vec{v}$, denoted by $\vec{u} \odot \vec{v}$, is the word sequence $\vec{w} = (u_1 \cdot v_1, u_2 \cdot v_2, \ldots)$, that is, $w_i = u_i \cdot v_i$ for all $i \in \mathbb{N}_{>0}$, where $\cdot$ is the classic word concatenation operator from regular expressions. Roughly speaking, $\vec{w}$ is obtained from the component-wise application of the word concatenation operator $\cdot$ to $\vec{u}$ and $\vec{v}$.

The notion of shuffle [11] of two word sequences is based on the notion of *selection function*, namely a function $g : \mathbb{N}_{>0} \to \{1, 2\}$. Intuitively, given a selection function $g$, the $g$-shuffle of word sequences $\vec{v}^1$ and $\vec{v}^2$, denoted by $\vec{v}^1 +_g \vec{v}^2$, is the word sequence whose $i$-th element is taken from $\vec{v}^1$ if $g(i) = 1$ and from $\vec{v}^2$ otherwise. The order in which elements of $\vec{v}^1$ (resp., $\vec{v}^2$) appear in $\vec{v}^1 +_g \vec{v}^2$ is the same as they appear in $\vec{v}^1$ (resp., $\vec{v}^2$), but possibly at different positions. As an example, if $\vec{v}^1 = (a, aa, aaa, \ldots)$, $\vec{v}^2 = (b, bb, bbb, \ldots)$, and $g$ is a selection function such that $g(1) = g(3) = 1$ and $g(2) = 2$, then we have that $\vec{v}^1 +_g \vec{v}^2$ is a sequence of the form $(a, b, aa, \ldots)$, where the 1st element of $\vec{v}^2$ (resp., 2nd element of $\vec{v}^1$) is the 2nd (resp., 3rd) element of the $g$-shuffle of $\vec{v}^1$ and $\vec{v}^2$. This is formalized as follows. First, we denote by $\mathtt{1's\text{-}upto}(g, i)$ the number of positions, up to $i$, where the value of function $g$ is 1, i.e., $\mathtt{1's\text{-}upto}(g, i) = |\{j \mid g(j) = 1 \text{ and } 1 \leq j \leq i\}|$; analogously, we denote by $\mathtt{2's\text{-}upto}(g, i)$ the number of positions, up to $i$, where the value of function $g$ is 2. Intuitively, $\mathtt{1's\text{-}upto}(g, i)$ (resp., $\mathtt{2's\text{-}upto}(g, i)$) denotes the number of element of $\vec{u}$ (resp.,

$\vec{v}$) that have been selected by $g$ to appear in the prefix of $\vec{u} +_g \vec{v}$ of length $i$. Therefore, they can be used to determine (the position of) the word in sequence $\vec{u}$ (resp., $\vec{v}$) that appears in position $i$ of sequence $\vec{w}$.

The *$g$-shuffle* $\vec{u} +_g \vec{v}$ is the word sequence $\vec{w}$, where, for all $i \in \mathbb{N}_{>0}$,

$$w_i = \begin{cases} u_{1\text{'s-upto}(g,i)} & \text{if } g(i) = 1 \\ v_{2\text{'s-upto}(g,i)} & \text{if } g(i) = 2 \end{cases}$$

We say that an infinite word sequence $\vec{w}$ is a *shuffle* of $\vec{u}$ and $\vec{v}$ if there is a selection function $g$ such that $\vec{w}$ is the $g$-shuffle of $\vec{u}$ and $\vec{v}$. Notice that the set of selection functions includes those $g$ that eventually converge to either 1 or 2, i.e., there exists $k \in \mathbb{N}_{>0}$ such that $g(x) = 1$ (resp., $g(x) = 2$) for all $x > k$.

Finally, given a nondecreasing function $f : \mathbb{N} \to \mathbb{N}_{>0}$, with $f(0) = 1$, the *$f$-aggregation* of a word sequence $\vec{u}$ is the sequence $(u_{f(0)}u_{f(0)+1} \ldots u_{f(1)-1}, u_{f(1)} \ldots u_{f(2)-1}, \ldots)$. For the sake of readability, we denote by $\mathcal{F}$ the set of nondecreasing functions $f : \mathbb{N} \to \mathbb{N}_{>0}$, with $f(0) = 1$. Given a function $f \in \mathcal{F}$, it is convenient to denote by $\delta_f = \langle \delta_f(i) \rangle_{i \in \mathbb{N}_{>0}}$ the sequence of the *deltas* of $f$, that is, the difference between consecutive values returned by $f$. Formally, $\delta_f(i) = f(i) - f(i-1)$.

In order to provide the semantics of $BST$-regular expressions, we need to precisely state the notions of $B$-, $S$-, and $T$-sequences.

An infinite sequence $\langle n_i \rangle_{i \in \mathbb{N}_{>0}}$ of natural numbers is said to be

- a *$B$-sequence* if it is *bounded*, i.e., there exists $b \in \mathbb{N}$ such that $n_i < b$ for all $i \in \mathbb{N}_{>0}$;
- an *$S$-sequence* if it is *strongly unbounded*, i.e., its limit inferior is infinite (equivalently, no value occurs infinitely often in the sequence), or, more formally, for every $n \in \mathbb{N}$ there is $k \in \mathbb{N}$ such that $n_i > n$ for all $i > k$;
- a *$T$-sequence* if it features infinitely many values occurring infinitely often, i.e., there exist infinitely many $n \in \mathbb{N}$ and infinitely many $i \in \mathbb{N}_{>0}$ such that $n_i = n$.

We are now ready to define the formal semantics of $BST$-regular expressions:

- $\mathcal{L}(\emptyset) = \emptyset$;
- for $a \in \Sigma$, $\mathcal{L}(a)$ only contains the infinite sequence of the one-letter word $a$, that is, $\mathcal{L}(a) = \{(a, a, a, \ldots)\}$;
- $\mathcal{L}(e_1 \cdot e_2) = \{\vec{w} \mid \vec{w}$ is the concatenation of $\vec{u}$ and $\vec{v}$, with $\vec{u} \in \mathcal{L}(e_1)$ and $\vec{v} \in \mathcal{L}(e_2)\}$;
- $\mathcal{L}(e_1 + e_2) = \{\vec{w} \mid \vec{w}$ is a shuffle of $\vec{u}$ and $\vec{v}$, with $\vec{u}, \vec{v} \in \mathcal{L}(e_1) \cup \mathcal{L}(e_2)\}$;[2]
- $\mathcal{L}(e^*) = \{\vec{w} \mid \vec{w}$ is the $f$-aggregation of $\vec{u}$, with $\vec{u} \in \mathcal{L}(e)$ and $f \in \mathcal{F}\}$;
- $\mathcal{L}(e^B) = \{\vec{w} \mid \vec{w}$ is the $f$-aggregation of $\vec{u}$, with $\vec{u} \in \mathcal{L}(e)$ and $f \in \mathcal{F}$ such that $\delta_f$ is a $B$-sequence$\}$;
- $\mathcal{L}(e^S) = \{\vec{w} \mid \vec{w}$ is the $f$-aggregation of $\vec{u}$, with $\vec{u} \in \mathcal{L}(e)$ and $f \in \mathcal{F}$ such that $\delta_f$ is an $S$-sequence$\}$;
- $\mathcal{L}(e^T) = \{\vec{w} \mid \vec{w}$ is the $f$-aggregation of $\vec{u}$, with $\vec{u} \in \mathcal{L}(e)$ and $f \in \mathcal{F}$ such that $\delta_f$ is a $T$-sequence$\}$.

The $\omega$-*constructor* $(.)^\omega$ turns languages of infinite word sequences into languages of $\omega$-words by simply concatenating the words in the sequence into a single (infinite) word. Formally:

- $\mathcal{L}(e^\omega) = \{w \mid |w| = \infty,\ w = u_1 u_2 u_3 \ldots$, and $\vec{u} \in \mathcal{L}(e)\}$.

[2] It is worth pointing out that previous work, e.g., [9], uses a different semantics for the operator $+$, based on the *mixing*, rather than the shuffling, of two sequences. Given a selection function $g$, the $g$-*mix* of two sequences $\vec{v}^1$ and $\vec{v}^2$ features elements taken from either $\vec{v}^1$ or $\vec{v}^2$, according to $g$, analogously to the $g$-shuffle; however, unlike the $g$-shuffle, the position of elements of $\vec{v}^1$ and $\vec{v}^2$ included in the $g$-mix is preserved, but some elements of one or both sequences can be discarded. For instance, consider again the example where $\vec{v}^1 = (a, aa, aaa, \ldots)$, $\vec{v}^2 = (b, bb, bbb, \ldots)$, and $g$ is a selection function such that $g(1) = g(3) = 1$ and $g(2) = 2$. The $g$-*mix* of $\vec{v}^1$ and $\vec{v}^2$ is a sequence of the form $(a, bb, aaa, \ldots)$, where the 1st (resp., 2nd, 3rd) element of the $g$-mix is the 1st element of $\vec{v}^1$ (resp., 2nd element of $\vec{v}^2$, 3rd element of $\vec{v}^1$), and the 1st and 3rd element of $\vec{v}^2$ and the 2nd element of $\vec{v}^1$ are discarded, that is, they do not appear in the $g$-mix. Finally, we observe that the two semantics for the $+$ operator are equivalent (in the sense that the same languages are generated) in the context of $\omega BS$-expressions [11]. The $g$-shuffle operator was introduced in [11] to avoid some anomalous behaviors caused by the constructor $(.)^T$, that was first proposed in that paper. As an example, the equivalence $e^T = e^T + e^T$ only holds with the semantics of $+$ based on the shuffle operation (see [11] for an in-depth discussion).

It is worth noticing that it is possible for a language to contain word sequences featuring an infinite suffix of the empty words, e.g., the word sequence $\vec{v} = (\varepsilon, \varepsilon, \varepsilon, \ldots)$ belongs to the language generated by the expression $a^*$. By blindly concatenating words in $\vec{v}$, we obtain the empty word, which should not be included in the language of $(e^*)^\omega$. This explains the presence of the condition $|w| = \infty$ in the definition above.

$\omega BST$*-expressions* are defined by the following grammar:

$$E ::= E + E \mid R \cdot E \mid e^\omega$$

where $R$ is a regular expression, $e$ is a $BST$-regular expression, and $+$ and $\cdot$ denote, respectively, union and concatenation of word languages (formally, $\mathcal{L}(E_1 + E_2) = \mathcal{L}(E_1) \cup \mathcal{L}(E_2)$ and $\mathcal{L}(E_1 \cdot E_2) = \{u \cdot v \mid u \in \mathcal{L}(E_1), v \in \mathcal{L}(E_2)\}$).[3] From now on, to keep the notation light and with a little abuse of notation, we sometimes identify a language $\mathcal{L}(e)$ with the expression $e$ defining it, and, as we did for languages of word sequences, we sometimes omit the operator $\cdot$ between word languages. Moreover, in order to distinguish, through the notation, languages of words and languages of word sequence, we often use lowercase (resp., uppercase) letters $e$, $e_1, \ldots,$ (resp., $E$, $E_1, \ldots,$ $R$, $R_1, \ldots$) for languages of word sequences (resp., words). Finally, when referring to an $\omega BST$-regular expression, without loss of generality, we assume that it has the form $R_1 e_1^\omega + \ldots + R_k e_k^\omega$, where $R_i$ is a regular expression and $e_i$ is a $BST$-regular expression, for all $i$.

A $B$-regular expression (resp., $\omega B$-regular expression) is a $BST$-regular expression (resp., $\omega BST$-regular expression) with no occurrences of constructors $(.)^S$ and $(.)^T$. Other classes of extended regular and extended $\omega$-regular expressions, namely $S$-, $T$-, $BS$-, $BT$-, $ST$-, $\omega S$-, $\omega T$-, $\omega BS$-, $\omega BT$-, and $\omega ST$-regular expressions, are defined analogously.

[3] Notice the abuse of notation with respect to the previous definition of the operators $+$ and $\cdot$ over languages of word sequences.

### 2.1.1. Parse trees

In order to prove the correctness of the proposed encodings, we introduce and formally define the notion of $E$ *parse tree for* $w$, with $w$ being a ($\omega$-)word belonging to the language defined by the ($\omega$-)regular expression $E$.

Hereafter, for a $(\omega)BST$-regular expression $E$, we fix a sequence, denoted by $sub(E)$, of its sub-expressions partially ordered according to their complexity (sub-expression relation), i.e., $sub(E)$ is a sequence $\langle e_1, e_2, \ldots, e_n \rangle$, where $e_n = E$ and if $e_i$ is a sub-expression of $e_j$, then $i < j$. Notice that, in general, there are more than one such sequences; any of them can be used. Moreover, each occurrence of the same sub-expression in $E$ has a distinct corresponding element in $sub(E)$, that is, if the same sub-expression occurs more than once in $E$, then it occurs more than once in $sub(E)$ as well. Formally, $sub(E)$ is a topological sort of the directed acyclic graph representing the sub-expression relation (with repetitions) of the expression $E$. If, for instance (see also Figure 1), $E = (a^*ba^*c)^\omega$, then we can fix $sub(E) = \langle a, b, a, a^*, a^*, a^*b, a^*ba^*, c, a^*ba^*c, (a^*ba^*c)^\omega \rangle$, where the two sub-expressions $a$ and $a^*$ occur twice; more precisely, $e_1 = a$ refers, say, to the first occurrence of $a$ in $E$, $e_2 = b$ to the only occurrence of $b$, and $e_3 = a$ to the second occurrence of $a$; similarly, $e_4$ and $e_5$ refer to the first and second occurrence of $a^*$ in $E$, respectively, while $e_{10}$ refers to the whole expression.

Given a (possibly infinite) word $w = w_1 w_2 \ldots$ and two indexes $i, j \in \mathbb{N}_{>0}$, with $i, j \leq |w| + 1$ ($|w| = \infty$ if $w$ is an $\omega$-word), we define the finite sub-word $w[i, j) = w_i \ldots w_{j-1}$ ($w[i, j) = \varepsilon$ if $j \leq i$). Moreover, we denote by $w[i, \omega)$ the (possibly infinite) suffix of $w$ starting at $w_i$.

For a ($\omega$-)word $w$ and a ($\omega$-)regular expression $E$, with $w \in \mathcal{L}(E)$, we say that a tuple $\tau^E_w = (Nodes, Edges, e\text{-}idx, s, f)$ is an $E$ *parse tree for* $w$ if the following conditions hold:

- the pair $(Nodes, Edges)$ is a tree;
- $e\text{-}idx : Nodes \to \{1, \ldots, |sub(E)|\}$;

- $s, f : Nodes \to \{1, \ldots, |w|+1\}$ such that $s(n) \leq f(n)$ for all $n \in Nodes$;[4]

- if $r$ is the root of the tree $(Nodes, Edges)$, then $e_{e\text{-}idx(r)} = E$ and $w[s(r), f(r)) = w$ (note that $f(r) - s(r) = |w|$);

- for each $n \in Nodes$, it holds that $s(n) < \omega$, and, additionally,

  $(i)$ if $e_{e\text{-}idx(n)} = a$, for some $a \in \Sigma$, then $n$ is a leaf, $w_{s(n)} = a$, and $f(n) = s(n) + 1$;

  $(ii)$ if $e_{e\text{-}idx(n)} = \varepsilon$, then $n$ is a leaf and $f(n) = s(n)$;

  $(iii)$ if $e_{e\text{-}idx(n)} = e_j + e_k$, then $n$ has exactly one child $n'$ in the tree $(Nodes, Edges)$ such that $e\text{-}idx(n') \in \{j, k\}$, and $(s(n), f(n)) = (s(n'), f(n'))$;

  $(iv)$ if $e_{e\text{-}idx(n)} = e_j e_k$, then $n$ has exactly two children $n', n''$ in the tree $(Nodes, Edges)$ such that $e\text{-}idx(n') = j$, $e\text{-}idx(n'') = k$, $f(n') = s(n'')$, and $(s(n), f(n)) = (s(n'), f(n''))$;

  $(v)$ if $e_{e\text{-}idx(n)} = e_j^*$, then either $n$ is a leaf and $s(n) = f(n)$ or $f(n) < \omega$ and $n$ has exactly $h$ children $n^1, \ldots, n^h$, with $h \in \mathbb{N}_{>0}$, in the tree $(Nodes, Edges)$, such that $e\text{-}idx(n^1) = \ldots = e\text{-}idx(n^h) = j$, $f(n^k) = s(n^{k+1})$, for all $k \in \{1, \ldots, h-1\}$, and $(s(n), f(n)) = (s(n^1), f(n^h))$;

  $(vi)$ if $e_{e\text{-}idx(n)} = e_j^\omega$, then $n$ has infinitely many children $\langle n^h \rangle_{h \in \mathbb{N}_{>0}}$ such that $e\text{-}idx(n^h) = j$, $f(n^h) = s(n^{h+1})$, for every $h \in \mathbb{N}_{>0}$, and $(s(n), f(n)) = (s(n^1), \omega)$.

An example of the proposed notation is shown in Figure 1, which depicts the $E$ parse tree for $w = aabaaacbac\ldots$ and $E = (a^*ba^*c)^\omega$. Intuitively, an $E$ parse trees for $w$ *witnesses* the membership of the ($\omega$-)word $w$ in the ($\omega$-)regular language $\mathcal{L}(E)$.

In order to formally state the relationship between a $(\omega)BST$-regular expression $E$ and an $E$ parse tree, we need to identify (in the parse tree) the

[4]Note that if $w$ is an infinite word, then the co-domain of both $s$ and $f$ is $\mathbb{N}_{>0} \cup \{\omega\}$.

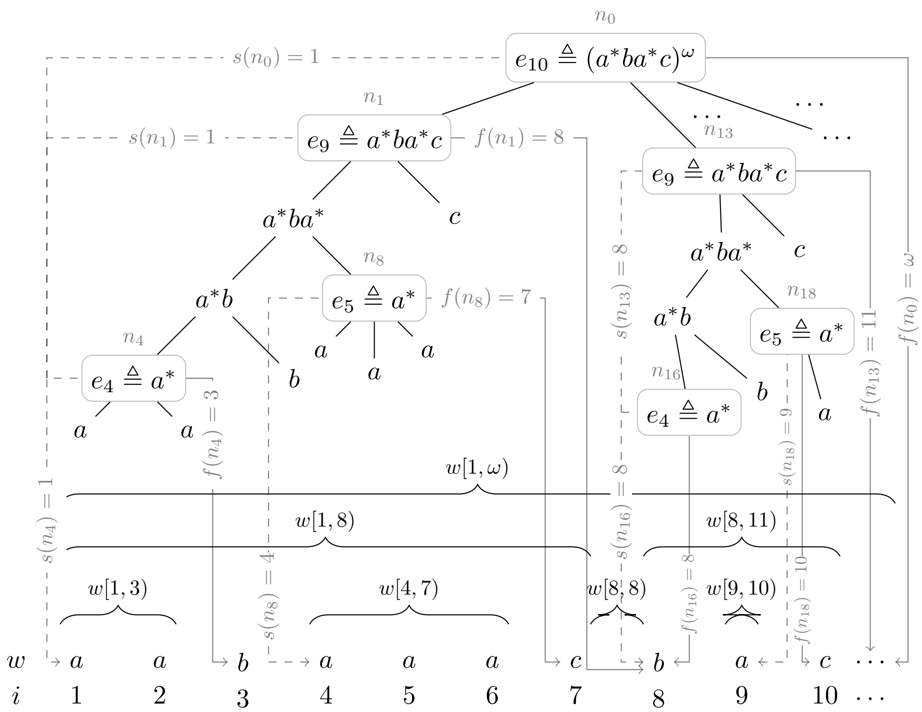


Figure 1: Parse tree witnessing the membership of $w = aabaaacbac\ldots$ in $\mathcal{L}(E)$, with $E = (a^*ba^*c)^\omega$.

sequences of the number of iterations generated by the iteration constructors $(.)^*$, $(.)^B$, $(.)^S$, and $(.)^T$. To this end, for every $e_i, e_j \in sub(E)$, with $e_i = (e_j)^{op}$ and $op \in \{*, B, S, T\}$, and every word $w \in \mathcal{L}(E)$, with $\tau$ being an $E$ parse tree for $w$, we denote by $\tau\text{-}count(i)$ the sequence of the numbers of children of nodes corresponding to $e_i$, that is, nodes $n$ with $e\text{-}idx(n) = i$, ordered according to a DFS visit of $\tau$. We will often omit the prefix specifying the parse tree, and simply write, e.g., $count(i)$ for $\tau\text{-}count(i)$.

As an example, consider once more the word $w = aabaaacbac\ldots$ belonging to the language of $E = (a^*ba^*c)^\omega$ (see Figure 1), and the above-given sequence $sub(E) = \langle a, b, a, a^*, a^*, a^*b, a^*ba^*, c, a^*ba^*c, (a^*ba^*c)^\omega \rangle$. The root node of the (unique) parse tree witnessing the membership of $w$ in $\mathcal{L}(E)$ corresponds to $e_{10} = E$; the root features an infinite number of children, each of them corresponding to $e_9 = (a^*ba^*c)$. The complete structure of the first two children of the root is depicted in Figure 1. The sequences $count(4)$ and $count(5)$ are, respectively, $\langle 2, 0, \ldots \rangle$ and $\langle 3, 1, \ldots \rangle$.

We are now ready to formalize the relationship between words in $\mathcal{L}(E)$ and $E$ parse trees, for any given $(\omega)BST$-regular expression $E$, through the following lemma, whose simple proof is omitted, where $E_*$ denotes the expression obtained from $E$ by replacing $B$-, $S$-, and $T$-constructors by $*$-constructors.

**Lemma 1.** *Let $w$ be a ($\omega$-)word. Then,*

(a) *if $E$ is a ($\omega$-)regular expression, then $w \in \mathcal{L}(E)$ if and only if there exists an $E$ parse tree for $w$;*

(b) *if $E$ is an $\omega B$-regular expression, then $w \in \mathcal{L}(E)$ if and only if there exists an $E_*$ parse tree for $w$ such that $count(i)$ is a $B$-sequence for every $e_i \in sub(E)$ with $e_i = e_j^B$;*

(c) *if $E$ is an $\omega S$-regular expression, then $w \in \mathcal{L}(E)$ if and only if there exists an $E_*$ parse tree for $w$ such that $count(i)$ is either a finite sequence or an $S$-sequence, for every $e_i \in sub(E)$ with $e_i = e_j^S$;*

(d) *if $E$ is an $\omega T$-regular expression, then $w \in \mathcal{L}(E)$ if and only if there exists an $E_*$ parse tree for $w$ such that $count(i)$ is either a finite sequence or a $T$-sequence, for every $e_i \in sub(E)$ with $e_i = e_j^T$.*

When dealing with the $B$-constructor, we can ignore empty strings generated by the argument expression, as formalized by the following lemma, whose simple proof is omitted. Let $E$ be a ($\omega$-)regular expression, $w$ be a ($\omega$-)word, and $\tau = (Nodes, Edges, e\text{-}idx, s, f)$ be an $E$ parse tree for $w$. We denote by $*\text{-}\varepsilon\text{-}children(E,\tau)$ the set of nodes $n$ corresponding to an expression that is the argument of a $*$-constructor in $E$ and such that $s(n) = f(n)$. Formally, we have:

$$*\text{-}\varepsilon\text{-}children(E,\tau) = \{n \in Nodes \mid s(n) = f(n) \text{ and } e\text{-}idx(n) = j \\ \text{for some } e_i \in sub(E), \text{ with } e_i = e_j^*\}$$

Moreover, we denote by $\tau_{\varepsilon\text{-}free}$ the tree structure obtained from $\tau$ by removing nodes in $*$-$\varepsilon$-*children*$(E, \tau)$.

**Lemma 2.** *Let $E$ be an ($\omega$-)regular expression and $w$ be a ($\omega$-)word. If $\tau$ is an $E$ parse tree for $w$, then so is $\tau_{\varepsilon\text{-}free}$. Moreover, if $\tau$-count$(i)$ is a $B$-sequence, then so is $\tau_{\varepsilon\text{-}free}$-count$(i)$, for all $i$.*

### 2.2. Interval temporal logics $AB$, $AB\bar{A}$, $AB{\sim}$, and $AB\bar{A}{\sim}$

In what follows, we define syntax and semantics of the interval temporal logics $AB$, $AB\bar{A}$, $AB{\sim}$, and $AB\bar{A}{\sim}$. As a preliminary step, we define the notion of (labeled) interval structure, which is common to all the logics we consider. We identify any given ordinal $N \leq \omega$ with the prefix of $\mathbb{N}$ of length $N$, that is, $N = \{0, 1, \ldots, N-1\}$ if $N < \omega$, and $N = \mathbb{N}$ if $N = \omega$, and we accordingly define the associated *interval structure* (or, simply, *structure*) $\mathbb{I}(N)$ as the set of all closed intervals $[i, j]$, with $i, j \in N$ and $i \leq j$. A special role will be played by *point intervals* (or, simply, *points*) and *unit intervals*, i.e., intervals of the forms $[i, i]$ and $[i, i+1]$, for some $i \in N$, respectively. Given a nonempty set $\mathcal{P}rop$ of proposition letters, a *labeled interval structure* over $\mathcal{P}rop$ is a pair $(\mathbb{I}(N), V)$, where $\mathbb{I}(N)$ is a (possibly infinite) interval structure and $V : \mathbb{I}(N) \to \mathcal{P}(\mathcal{P}rop)$ is a *valuation function* providing an interpretation of proposition letters, i.e., a function that assigns to every interval the set of proposition letters that are true on it.

#### 2.2.1. The logic $AB$

$AB$ features modalities $\langle A \rangle$ and $\langle B \rangle$, that correspond to Allen's relations *meets* (denoted by $A$) and *begun by* (denoted by $B$), respectively. It is a relatively small, but quite expressive, fragment of the Halpern and Shoham's interval temporal logic HS [15], and its satisfiability problem is EXPSPACE-complete over both finite linear orders and $\mathbb{N}$ [20]. Formally, given a nonempty set $\mathcal{P}rop$ of proposition letters, formulas of $AB$ are defined as follows:

$$\varphi := p \mid \varphi \vee \varphi \mid \neg\varphi \mid \langle A \rangle\varphi \mid \langle B \rangle\varphi,$$

where $p \in \mathcal{P}rop$. We use the shorthands $\varphi \wedge \psi$ for $\neg(\neg\varphi \vee \neg\psi)$, $[X]\varphi$ for $\neg\langle X\rangle\neg\varphi$, with $X \in \{A, B\}$, $\perp$ for $p \wedge \neg p$, and $\top$ for $p \vee \neg p$. Formulas of $AB$ are interpreted over labeled interval structures endowed with Allen's relations $A$ and $B$. Allen's relations $A$ and $B$ are defined as follows. Given two intervals $[i, j], [i', j'] \in \mathbb{I}(N)$, we say that: *(a)* $[i, j]A[i', j']$ if and only if $j = i'$; *(b)* $[i, j]B[i', j']$ if and only if $i = i'$ and $j' < j$. $AB$ semantics is given in terms of *interval models* (or simply *models*) $M = \langle \mathbb{I}(N), A, B, V\rangle$, where $(\mathbb{I}(N), V)$ is a (possibly infinite) labeled interval structure. Truth of $AB$ formulas over an interval $[i, j]$ belonging to a model $M$ is inductively defined as follows:

- $M, [i, j] \models p$ if and only if $p \in V([i, j])$, for $p \in \mathcal{P}rop$;
- $M, [i, j] \models \neg\varphi$ if and only if it is not the case that $M, [i, j] \models \varphi$;
- $M, [i, j] \models \varphi \vee \psi$ if and only if $M, [i, j] \models \varphi$ or $M, [i, j] \models \psi$;
- $M, [i, j] \models \langle X\rangle\varphi$ if and only if there exists an interval $[i', j']$ such that $[i, j]X[i', j']$ and $M, [i', j'] \models \varphi$, for $X \in \{A, B\}$.

Given $M = \langle \mathbb{I}(N), A, B, V\rangle$ and $\varphi$, $M$ satisfies $\varphi$ if there is $[i, j] \in \mathbb{I}(N)$ such that $M, [i, j] \models \varphi$, and $\varphi$ is satisfiable if there is an interval model $M$ that satisfies it.

It is immediate to see that point and unit intervals are captured by $AB$ (and thus by all the logics we deal with) by means of formulas $\pi \triangleq [B]\perp$ and $unit \triangleq \langle B\rangle\top \wedge [B][B]\perp$, respectively.[5]

Hereafter, we use modalities $[G]$ (*globally*) and $[init]$ (*every initial interval*), which are definable in $AB$ as follows: *(i)* $[G]\varphi \triangleq [B][A]\varphi \wedge [A][A]\varphi$, and *(ii)* $[init]\varphi \triangleq [B](\pi \to [A]\varphi) \wedge (\pi \to [A]\varphi)$. When evaluated on $[x, y]$, $[G]\varphi$ forces $\varphi$ to be true over all intervals $[w, z]$, for some $w, z$, with $w \geq x$; in particular, when evaluated on $[0, y]$, it forces $\varphi$ to be true on *all* intervals. When evaluated on $[x, y]$, $[init]\varphi$ forces $\varphi$ to be true on all intervals $[x, z]$, for some

[5] Even though the symbol $\pi$ for point intervals is not particularly evocative, it is a long-established notation in the context of the interval temporal logic HS.

$z$; in particular, when evaluated on $[0, y]$, it forces $\varphi$ to be true on *all initial* intervals, that is, all prefixes of the linear order.

### 2.2.2. The logic $AB\bar{A}$

$AB\bar{A}$ is obtained from $AB$ by adding the (past) modality $\langle \bar{A} \rangle$ for the Allen relation *met by* (denoted by $\bar{A}$). Unlike what happens with point-based temporal logics, the addition of past operators to interval ones usually increases both their expressiveness and their computational complexity (see, for instance, [22]). This is the case with $AB\bar{A}$: its satisfiability problem is still decidable, but non-primitive recursive, over finite linear orders, and undecidable over $\mathbb{N}$ [21]. $AB\bar{A}$ syntax extends that of $AB$ in the obvious way. $AB\bar{A}$ formulas are interpreted on models $M = \langle \mathbb{I}(N), A, B, \bar{A}, V \rangle$, and the truth of a formula $\varphi$ over an interval $[i, j]$ of $M$ is defined by means of the semantic clauses for $AB$ defined above, together with the following one:

- $M, [i, j] \models \langle \bar{A} \rangle \varphi$ if and only if there exists an interval $[i', j']$ such that $[i, j] \bar{A} [i', j']$ and $M, [i', j'] \models \varphi$.

where, for any pair of intervals $[i, j], [i', j'] \in \mathbb{I}(N)$, $[i, j] \bar{A} [i', j']$ if and only if $i = j'$.

### 2.2.3. The logic $AB\sim$

$AB\sim$ is obtained from $AB$ by adding an equivalence relation $\sim$ over the points of the model. From the computational point of view, $AB\sim$ behaves similarly to $AB\bar{A}$: the satisfiability problem for $AB\sim$ is non-primitive recursive over finite linear orders, while decidability is lost over $\mathbb{N}$ [12]. Formally, the language of $AB$ is extended with a new symbol $\sim$, and formulas are built according to the syntax:

$$\varphi := p \mid \sim \mid \varphi \vee \varphi \mid \neg \varphi \mid \langle A \rangle \varphi \mid \langle B \rangle \varphi,$$

where $p \in \mathcal{P}rop$. The semantics of $AB\sim$ formulas is given in terms of models $M = \langle \mathbb{I}(N), A, B, \sim, V \rangle$, where $\sim$ is an equivalence relation on $N$. Truth is defined as for $AB$ formulas, with an additional semantic clause for $\sim$:

- $M, [i, j] \models \sim$ if and only if $i \sim j$.

Notice that, since $\sim$ is an equivalence relation, for every model $M$ and points $i, j, k$ in $M$, the following properties hold:

1. $M, [i, i] \models \sim$ (by reflexivity of $\sim$), and

2. if $M, [i, j] \models \sim$ and $M, [j, k] \models \sim$, then $M, [i, k] \models \sim$ (by transitivity of $\sim$).

*2.2.4. The logic $AB\bar{A}\sim$*

Syntax and semantics of $AB\bar{A}\sim$ are obtained from those of $AB\bar{A}$ and $AB\sim$ by merging them in the obvious way.

*2.3. Linking $\omega$-words and interval structures*

In order to encode word languages into logical formulas, we need to establish a correspondence between words and models of the considered logic. In the following, we show how to interpret ($\omega$-)words as labeled interval structures, and vice versa.

In order to represent ($\omega$-)words by means of labeled interval structures, we introduce a proposition letter for every symbol of the alphabet (thus, $\Sigma \subseteq \mathcal{P}rop$), and then we define a suitable formula (see formula $\varphi_\Sigma$, defined in Section 4) to restrict to interval models built over labeled interval structures where exactly one symbol of the alphabet $\Sigma$ holds true in each unit interval, so to have a natural mapping from models to words over $\Sigma$.

For a (possibly infinite) word $w = w_1 w_2 \dots$ over a finite alphabet $\Sigma$ and a labeled interval structure $S = \langle \mathbb{I}(N), V \rangle$ over $\mathcal{P}rop$, we say that $w$ and $S$ are *compatible*, denoted by $w \approx S$ (or, equivalently, $S \approx w$), if $N = |w| + 1$, $\Sigma \subseteq \mathcal{P}rop$, and $V : \mathbb{I}(N) \to \mathcal{P}(\mathcal{P}rop)$ is such that on each unit interval only the proper letter (among those in $\Sigma$) holds, that is, $V([i-1, i]) \cap \Sigma = \{w_i\}$ for every $i \in \{1, \dots, |w|\}$, and no letter from $\Sigma$ holds over any non-unit interval, that is, $V([i, j]) \cap \Sigma = \emptyset$, for every $i, j$ with $j - i \neq 1$.

This notion can be lifted to cope with models in the natural way. We say that a word $w$ and an interval model $M$ are *compatible*, denoted by $w \approx M$ (or, equivalently, $M \approx w$) if $w$ and the labeled interval structure over which $M$ is built are compatible.

## 3. Some useful properties of $BST$-regular languages

In this section, we prove some properties of $BST$-regular languages that will be later exploited to analyze the proposed encodings. Proofs can be found in Appendix A.

To begin with, we note that operations $+_g$, for any selection function $g$, are not commutative, i.e., $\vec{u} +_g \vec{v}$ is in general not the same as $\vec{v} +_g \vec{u}$. However, for every pair of word sequences $\vec{u}, \vec{v}$ and every selection function $g$ there is a selection function $g'$, defined as $g'(i) = 3 - g(i)$ for all $i \in \mathbb{N}_{>0}$, such that $\vec{u} +_g \vec{v} = \vec{v} +_{g'} \vec{u}$. Therefore, the shuffle operation is indeed *commutative*, that is, if $\vec{w}$ is a shuffle of $\vec{u}$ and $\vec{v}$, then it is also a shuffle of $\vec{v}$ and $\vec{u}$, which amounts to say $e_1 + e_2 = e_2 + e_1$ for every pair of $BST$-regular expressions $e_1$ and $e_2$. Similarly, it can be easily shown that the shuffle operation is also *associative*: given three word sequences $\vec{u}$, $\vec{v}$, and $\vec{w}$, and two selection functions $f$ and $f'$, it holds that $(\vec{u} +_f \vec{v}) +_{f'} \vec{w} = \vec{u} +_g (\vec{v} +_{g'} \vec{w})$ for suitably defined selection functions $g$ and $g'$. Thus, we have that $(e_1 + e_2) + e_3 = e_1 + (e_2 + e_3)$ for every triple of $BST$-regular expressions $e_1$, $e_2$, and $e_3$.

Next, we first demonstrate the idempotence of the shuffle operator, that is, $\mathcal{L}(e) = \mathcal{L}(e + e)$ holds for every BST-regular expression $e$ (Corollary 1), which immediately follows from the next proposition. Then, we present an additional result (Corollary 2), that follows from Proposition 2 and shows that the constraints imposed by $S$- and $T$-constructors can be ignored, to a certain extent, when applied to word sequences featuring infinitely many empty strings. This will be made clearer in Sections 7 and 8. We conclude the section by remarking that $BST$-regular expressions enjoy *prefix independence*, which makes it possible to ignore the constraints imposed by $B$-, $S$-, and $T$-constructors in specific situations that will be clarified later.

**Proposition 1.** *Let $e$ be a BST-regular expression. If $\vec{u}, \vec{v} \in \mathcal{L}(e)$ and $\vec{w}$ is a shuffle of $\vec{u}$ and $\vec{v}$, then $\vec{w} \in \mathcal{L}(e)$ as well.*

**Corollary 1** (shuffle idempotence)**.** *$\mathcal{L}(e) = \mathcal{L}(e + e)$, for every BST-regular expression $e$.*

We now establish a technical result that will be useful in the following. Let $\vec{\varepsilon} = (\varepsilon, \varepsilon, \varepsilon, \varepsilon, \ldots)$ be the infinite sequence of empty strings. Moreover, let us say that a selection function $g : \mathbb{N}_{>0} \to \{1, 2\}$ is *non-$i$-convergent*, with $i \in \{1, 2\}$, if for every $j \in \mathbb{N}_{>0}$, there is $k > j$ such that $g(k) \neq i$.

We define the *$\varepsilon$-pumpings* of a word sequence $\vec{u}$ as the word sequences $\vec{u} +_g \vec{\varepsilon}$, for all non-2-convergent selection functions $g$, if $\vec{u}$ features infinitely many empty strings; otherwise, the only *$\varepsilon$-pumping* of $\vec{u}$ is $\vec{u}$ itself. Intuitively, an $\varepsilon$-pumping of a word sequence featuring infinitely many empty strings is obtained by injecting (possibly infinitely many) finite sequences of $\varepsilon$'s at arbitrary positions of the original sequence.

The following result states that $BST$-regular languages are closed under the operation of "pumping" (possibly infinitely many) empty strings at arbitrary positions of word sequences featuring infinitely many empty strings. Let $\mathcal{L}_\varepsilon(e) = \{\vec{v} \mid \vec{v}$ is an $\varepsilon$-pumping of $\vec{u}$ and $\vec{u} \in \mathcal{L}(e)\}$ be the language that extends $\mathcal{L}(e)$ with the $\varepsilon$-pumpings of all of its sequences.

**Proposition 2.** *It hold that $\mathcal{L}(e) = \mathcal{L}_\varepsilon(e)$, for every $BST$-regular expression $e$.*

Thanks to the above proposition, we can now state the following property.

**Corollary 2.** *Let $e$ be a $BST$-regular expression. If $\vec{u}$ is the $f$-aggregation of $\vec{v}$, for a function $f \in \mathcal{F}$ and a word sequence $\vec{v} \in \mathcal{L}(e)$ featuring infinitely many empty strings, then $\vec{u} \in \mathcal{L}(e^S)$. If, in addition, there is at least one value occurring infinitely often in $\delta_f$, then $\vec{u} \in \mathcal{L}(e^T)$ as well.*

Before concluding the section, we believe it is useful to point out that $BST$-regular expressions enjoy the property of *prefix independence*, which, intuitively, states that, in order to verify that a word sequence behaves according to the

$B$-, $S$-, or $T$-constructor, one can basically focus on any of its suffixes (see Proposition 6 in [11]), or, equivalently, it is not possible to refute a word sequence by just looking at any finite prefix. As a consequence, when encoding a sub-expression $e_i = e_j^{op}$, with $op \in \{*, B, S, T\}$, of a $BST$-regular expression $E$, it is not necessary to guarantee the satisfaction of the constraints imposed by $B$-, $S$-, and $T$-constructors (they can be treated as the standard $*$-constructor) over models featuring only finitely many occurrences of $expr_i$ intervals. Intuitively, models of this kind arise from expressions where $e_i$ occurs under the scope of the shuffle operator. As an example, consider the expression $E = (a^{op}b + c^*d)^\omega$, with $op \in \{*, B, S, T\}$. There are words in $\mathcal{L}(E)$ featuring only finitely many $b$'s, as the shuffle operator can postpone forever the selection of sub-words in the language of one of its operands, specifically $a^{op}b$, thus possibly ignoring an entire (infinite) suffix of a word sequence belonging to the language of $a^{op}b$. Since, by prefix independence, a word sequence cannot be refuted due to a finite prefix, when encoding such an expression in a logical formula, we can treat $a^{op}b$ as it were $a^*b$ for those models with only finitely many $b$'s. This property will be exploited in Sections 7 and 8, by imposing suitable guards on the formulas that encode the $S$- and $T$-constructors, respectively, so that they only affect models featuring infinitely many $b$'s.

## 4. Encoding regular and $\omega$-regular languages in $AB$

In this section, we provide a detailed account of the encodings of regular and $\omega$-regular languages in $AB$ (Theorems 2 and 3 – a short account can be found in [1]). These encodings produce $AB$ formulas of the form $\varphi_\Sigma \wedge \varphi_E$, where $\varphi_\Sigma$ is a very simple formula ensuring that each unit interval $[i, i+1]$ has a unique atomic proposition (from $\Sigma$) true on it, and $\varphi_E$ is obtained, in a bottom-up fashion, from the given ($\omega$-)regular expression $E$. The intuitive idea here is to use, for each occurrence $i$ of a sub-expression $e_i$ of $E$, two auxiliary atomic propositions, $expr_i$ and $expr_i^{end}$, to identify the interval whose associated finite word should be recognized by the sub-expression $e_i$. A suitable combination of the $AB$ modalities then ensures that the word is indeed correctly recognized. The translation of the atomic constructs and the alternation is immediate; that of concatenation and Kleene star is quite more elaborated.

In order to map interval temporal logic formulas into ($\omega$-)languages, in Section 2.3 we have established a correspondence between words and interval models that is based on the assumption that interval models are built over a set $\mathcal{P}rop$ of proposition letters which includes a proposition letter for every symbol of the alphabet $\Sigma$ (in symbols, $\Sigma \subseteq \mathcal{P}rop$) and that exactly one such letter holds true in each unit interval (and no interval other than unit intervals satisfies any such proposition letter). Thus, the first step of our encoding consists in defining formula $\varphi_\Sigma$, which forces models to satisfy these assumptions:

$$\varphi_\Sigma = [G]\left(\left(unit \ \leftrightarrow \ \bigvee_{a\in\Sigma} a\right) \ \wedge \ \bigwedge_{a\in\Sigma}\left(a \ \rightarrow \ \bigwedge_{b\in\Sigma\setminus\{a\}} \neg b\right)\right).$$

Next, to give a logical characterization of an expression $E$, we make use of two proposition letters $expr_i$ and $expr_i^{end}$ for each sub-expression $e_i$ in $sub(E)$, including $E$ itself. Let us stress that two occurrences $e_i$ and $e_j$ of the same sub-expression are associated with two different pairs of proposition letters ($expr_i/expr_i^{end}$ and $expr_j/expr_j^{end}$). Suitable formulas are then exploited to force the propagation of such proposition letters in a top-down fashion following the semantics of ($\omega$-)regular expressions.

As an example, if an interval $[a,b]$ is labeled with $expr_i$ (meaning that $expr_i$ is true on it) and $e_i$ is the expression $e_j + e_k$, then, through a suitable formula, we constrain $[a,b]$ to be labeled with $expr_j$ or $expr_k$ as well; if, instead, $e_i$ is the expression $(e_j)^*$, then, by means of a different formula, we force $[a,b]$ to be partitioned into sub-intervals labeled with $expr_j$, that is, we force the existence of finitely many points $c_0, c_1, \ldots, c_m$, with $a = c_0 < c_1 < \ldots < c_m = b$, such that $[c_h, c_{h+1}]$ is labeled with $expr_j$ for each $h \in \{0,\ldots,m-1\}$, unless $a = b$.

By suitably combining all such formulas, we encode an ($\omega$-)regular expression $E$ into a formula $\varphi$ that is satisfied exactly by those interval models that are compatible (according to the definition given in Section 2.3) with words belonging to the language of $E$.

### *4.1. Encoding regular languages in AB*

Let $R$ be a regular expression on $\Sigma$. We show how to encode $R$ into an $AB$ formula over the finite set of proposition letters $\mathcal{P}rop$, which includes $\Sigma$.

As anticipated, for each $e_i \in sub(E)$, we introduce two proposition letters $expr_i$ and $expr_i^{end}$. For each $i$, we force $expr_i^{end}$ to be true exactly at the right endpoint of $expr_i$ intervals and we prevent points that are strictly contained in an $expr_i$ interval to satisfy $expr_i^{end}$ (this also implies that an $expr_i$ interval cannot end inside another one). This condition is expressed by the formula:

$$\begin{aligned}\varphi^{end}_{expr_i} = \quad & [G]((expr_i^{end} \ \rightarrow \ \pi) \wedge \\ & \qquad (expr_i \ \rightarrow \ \langle A\rangle expr_i^{end} \ \wedge \ [B](\neg\pi \rightarrow [A]\neg expr_i^{end}))) \wedge \\ & [init](\langle A\rangle expr_i^{end} \ \rightarrow \ \langle A\rangle(\pi \ \wedge \ expr_i) \vee \langle B\rangle\langle A\rangle(\neg\pi \ \wedge \ expr_i)) \wedge \\ & [G](\langle A\rangle expr_i^{end} \ \wedge \ \langle B\rangle(\neg\pi \ \wedge \ expr_i) \ \rightarrow \\ & \qquad \langle B\rangle(\neg\pi \ \wedge \ \langle A\rangle(\neg\pi \ \wedge \ expr_i))).\end{aligned}$$

The first conjunct (lines 1 and 2) forces *(i)* $expr_i^{end}$ to hold at point intervals only, *(ii)* the right endpoint of any $expr_i$ interval to be labeled with $expr_i^{end}$, and *(iii)* no point strictly contained in an $expr_i$ interval to be labeled with $expr_i^{end}$. The rest of the formula forces every $expr_i^{end}$ point interval to be the right endpoint of an $expr_i$ interval. More precisely, the second conjunct (line 3) constrains every $expr_i^{end}$ point interval $x$ to be an $expr_i$ point interval as well or to have an $expr_i$ non-point interval $[y, y']$, with $y < y'$, that starts before it, that is, $y < x$. Notice that $y' \leq x$; otherwise, the $expr_i^{end}$ point $x$ would fall strictly inside the $expr_i$ interval $[y, y']$, which is not possible. Finally, the third conjunct (lines 4 and 5) forces $x$ to be the right endpoint of an $expr_i$ interval. To prove it, let $[y, y']$ be the unique $expr_i$ non-point interval such that there is no $expr_i$ non-point interval $[w, z]$, with $y < w < x$. Towards a contradiction, assume that $[y, y']$ does not end in $x$, that is, $y' < x$, and consider the interval $[y, x]$. It satisfies the antecedent of the implication $(\langle A\rangle expr_i^{end} \ \wedge \ \langle B\rangle(\neg\pi \ \wedge \ expr_i))$ and thus it must satisfy the consequent as well $(\langle B\rangle(\neg\pi \ \wedge \ \langle A\rangle(\neg\pi \ \wedge \ expr_i)))$, which imposes the existence of an $expr_i$ non-point interval $[w, z]$, with $y < w < x$, thus leading to a contradiction.

The next formula $\varphi^{\not\sqcap}_{expr_i}$ prevents an $expr_i$ interval from starting within another one (thus, two $expr_i$ intervals cannot intersect each other):

$$\varphi^{\not\sqcap}_{expr_i} = [G](expr_i \ \rightarrow \ [B](\neg\pi \ \rightarrow \ [A]\neg expr_i)).$$

Finally, formulas $\varphi_{expr_i}$ are defined by induction on the complexity of the corresponding expressions $e_i$.

- If $e_i = \emptyset$, we put $\varphi_{expr_i} = [G](expr_i \ \rightarrow \ \bot)$.
- If $e_i = a$, for some $a \in \Sigma$, we put $\varphi_{expr_i} = [G](expr_i \ \rightarrow \ a)$.
- If $e_i = \varepsilon$, we put $\varphi_{expr_i} = [G](expr_i \ \rightarrow \ \pi)$.
- If $e_i = e_j + e_k$, we put $\varphi_{expr_i} = [G](expr_i \ \leftrightarrow \ (expr_j \ \vee \ expr_k))$.
- If $e_i = e_j e_k$, then we constrain every $expr_i$ interval to be partitioned in two adjacent sub-intervals satisfying $expr_j$ and $expr_k$, respectively. This is done by means of the formula:

  **Theorem 1.** *d*

  **Proposition 1.** *Let $e$ be a BST-regular expression. If $\vec{u}, \vec{v} \in \mathcal{L}(e)$ and $\vec{w}$ is a shuffle of $\vec{u}$ and $\vec{v}$, then $\vec{w} \in \mathcal{L}(e)$ as well.*

  **Theorem 1.** *d*

$$\begin{aligned}
\varphi_{expr_i} = \ & [G](expr_j \ \rightarrow \ \langle A\rangle expr_k) \\
\wedge \ & [G](expr_k \ \rightarrow \ (expr_j^{end} \vee \langle B\rangle expr_j^{end}) \ \wedge \ \langle A\rangle expr_i^{end}) \\
\wedge \ & [G]((expr_j \vee expr_k) \ \rightarrow \ [B](\neg\pi \ \rightarrow \ [A]\neg expr_i^{end})) \\
\wedge \ & [G]((expr_j \ \wedge \ expr_k) \ \rightarrow \ \pi \wedge expr_i) \\
\wedge \ & [G](expr_i \ \rightarrow \ ((\langle B\rangle(\pi \ \wedge \ expr_j) \wedge expr_k)) \\
& \qquad\qquad \vee \langle B\rangle(\neg\pi \wedge expr_j) \\
& \qquad\qquad \vee (expr_j \ \wedge \ \langle A\rangle(\pi \ \wedge \ expr_k)) \\
\wedge \ & [G]( \quad (\langle A\rangle expr_j \ \rightarrow \ \langle A\rangle expr_i) \\
& \qquad \wedge (\langle A\rangle(\neg\pi \ \wedge \ expr_j) \ \rightarrow \ \langle A\rangle(\neg\pi \ \wedge \ expr_i)) \ \ ) \\
\wedge \ & [G](expr_k \ \wedge \ \langle B\rangle expr_i^{end} \ \rightarrow \ expr_i).
\end{aligned}$$

  The first four conjuncts (lines 1–4) state properties of the two sub-intervals: every $expr_j$ interval is followed by an $expr_k$ interval (line 1), every $expr_k$ interval is preceded by an $expr_j$ interval (i.e., there are $expr_j$ intervals ending at all starting points of $expr_k$ ones) and it ends where an $expr_i$ interval ends (line 2), no $expr_i$ interval ends (strictly) inside an $expr_j$ or

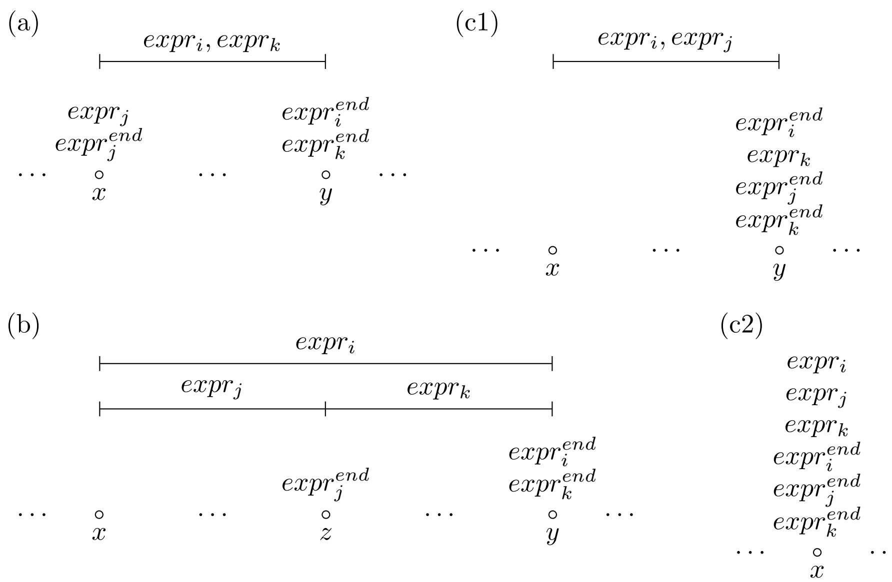


Figure 2: Interval configurations for the encoding of the concatenation operator.

an $expr_k$ interval (line 3), and an $expr_j$ interval is not an $expr_k$ interval (and vice versa) unless it is a point interval and satisfies $expr_i$ as well (line 4). By making use of the above properties, the fifth conjunct (lines 5–7) distinguishes three possible ways of partitioning an $expr_i$ interval: (i) it is started by an $expr_j$ point and coincides with an $expr_k$ interval (line 5 – Figure 2(a)), (ii) it is started by a non-point $expr_j$ interval (line 6 – Figure 2(b)), or (iii) it coincides with an $expr_j$ interval and it is ended by an $expr_k$ point, including the case in which there is a point interval satisfying $expr_i$, $expr_j$, and $expr_k$ (line 7 – Figures 2(c1) and 2(c2)). In case (ii), the existence of an $expr_k$ non-point interval, adjacent to the $expr_j$ interval and ending exactly where the $expr_i$ interval ends, is a consequence of the first four conjuncts. The next to last conjunct (lines 8 and 9) ensures that every $expr_j$ interval occurs as a (not necessarily strict) prefix of an $expr_i$ interval (recall that no $expr_i$ interval ends inside an $expr_j$ one), while

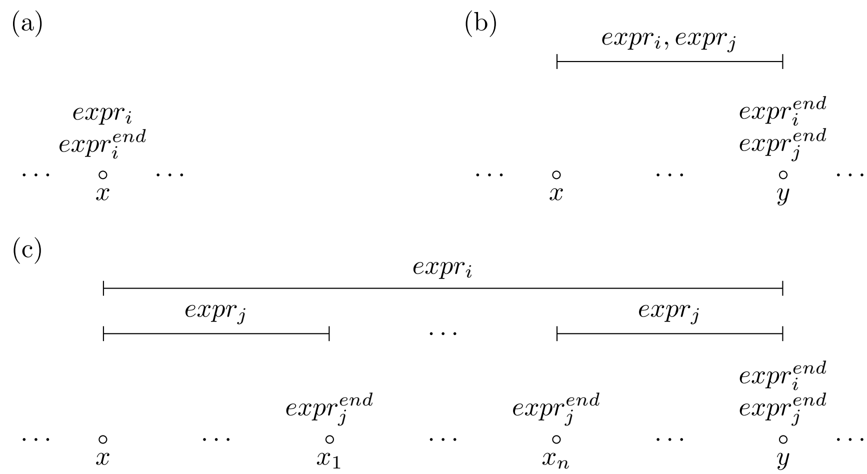


Figure 3: Interval configurations for the encoding of the Kleene star contructor.

the last conjunct (line 10) states that if an $expr_k$ interval is immediately preceded by an $expr_i$ one, then it is itself an $expr_i$ interval. This last property, together with the previous ones, guarantees that every $expr_k$ interval occurs as a (not necessarily strict) suffix of an $expr_i$ interval. This is due to the facts that *(i)* every $expr_k$ interval is preceded by an $expr_j$ one, *(ii)* every $expr_j$ interval is a prefix of an $expr_i$ one, and *(iii)* no $expr_i$ interval ends inside an $expr_k$ one. Observe that, as a consequence, we have that an $expr_j$ interval and an $expr_k$ one do not intersect (except for the intersections consisting of a single point that is not inside any of the two intervals), and thus the partition into $expr_j$ and $expr_k$ intervals is unique for every $expr_i$ interval.

- If $e_i = e_j^*$, then we constrain every $expr_i$ interval to be partitioned into a finite number of adjacent $expr_j$ sub-intervals. We distinguish three cases: (i) zero $expr_j$ intervals, that is, the $expr_i$ interval is a point interval, corresponding to the empty string, that does not contain any $expr_j$ interval (Figure 3(a)), (ii) one $expr_j$ intervals, that is, the $expr_i$ interval is also an

$expr_j$ interval (Figure 3(b)), and (iii) an arbitrary, but finite, number of $expr_j$ intervals (Figure 3(c)). Formally, we state the requested conditions by means of the formula:

$$
\begin{aligned}
\varphi_{expr_i} = {} & [G](expr_i \to \pi \vee expr_j \vee (\langle B\rangle expr_j \wedge \\
& \qquad [B](\langle A\rangle expr_j^{end} \to \langle A\rangle(\neg\pi \wedge expr_j)))) \\
& \wedge [G](expr_j \to [B](\neg\pi \to [A]\neg expr_i^{end})) \\
& \wedge [init](\langle A\rangle expr_j \wedge \neg\langle A\rangle expr_i \to \langle B\rangle\langle A\rangle(\neg\pi \wedge expr_i)) \\
& \wedge [G](\langle A\rangle expr_j \wedge \langle B\rangle(\neg\pi \wedge expr_i) \to \\
& \qquad \langle A\rangle expr_i \vee \langle B\rangle(\neg\pi \wedge \langle A\rangle(\neg\pi \wedge expr_i))) \\
& \wedge [G](expr_i \wedge \langle A\rangle(\neg\pi \wedge expr_j) \to \langle A\rangle expr_i) \\
& \wedge [G](\langle A\rangle(\neg\pi \wedge expr_j) \wedge \langle A\rangle expr_i \to \langle A\rangle(\neg\pi \wedge expr_i)).
\end{aligned}
$$

The first conjunct (lines 1 and 2) encodes the three above cases via three disjuncts (one for each possible scenario). The rest of the formula guarantees that every interval on which $expr_j$ holds occurs inside an interval on which $expr_i$ holds. More precisely, the second conjunct (line 3) states that no $expr_i$ interval ends (strictly) inside an $expr_j$ interval. The third, fourth, and fifth conjuncts (lines 4–7) guarantee that for every $expr_j$ interval $[x, y]$ there is an $expr_i$ interval $[w, z]$ for which at least one of the following properties holds: $(i)$ $[w, z]$ starts at $x$ (i.e., $w = x$) $(ii)$ $[w, z]$ ends at $x$ and $[x, y]$ is a point interval (i.e., $z = x = y$) $(iii)$ $[w, z]$ contains (strictly) $x$ (i.e., $w < x < z$). Assume, towards a contradiction, that for some $expr_j$ interval $[x, y]$ there is no such an $expr_i$ interval $[w, z]$. Then, the third conjunct (line 4) imposes the existence of an $expr_i$ non-point interval $[w, z]$, with $w < x$. Without loss of generality, let $[w, z]$ be the unique $expr_i$ non-point interval such that there is no other $expr_i$ non-point interval $[w', z']$, with $w < w' < x$. Since, by assumption, $[w, z]$ does not contain $x$, it holds that $z \leq x$. If $z < x$, then interval $[w, x]$ satisfies the antecedent of the implication in the fourth conjunct (lines 5 and 6), namely $\langle A\rangle expr_j \wedge \langle B\rangle(\neg\pi \wedge expr_i)$ (line 5), and thus it must also satisfy its consequent, namely $\langle A\rangle expr_i \vee \langle B\rangle(\neg\pi \wedge \langle A\rangle(\neg\pi \wedge expr_i))$ (line 6). The latter imposes the existence of an $expr_i$ interval starting at $x$ or an

$\mathit{expr}_i$ non-point interval $[w', z']$, with $w < w' < x$, thus leading to a contradiction. If, instead, $z = x$, then, by assumption, $[x, y]$ cannot be a point interval. Thus, interval $[w, x]$ satisfies the antecedent of the implication in the fifth conjunct (line 7), namely $\mathit{expr}_i \wedge \langle A\rangle(\neg\pi \wedge \mathit{expr}_j)$, and thus it must also satisfy its consequent, namely $\langle A\rangle \mathit{expr}_i$. The latter imposes the existence of an $\mathit{expr}_i$ interval starting at $x$, thus leading to a contradiction. This allows us to conclude that, for every $\mathit{expr}_j$ interval $[x, y]$ there is an $\mathit{expr}_i$ interval $[w, z]$ that satisfies at least one among properties $(i)$, $(ii)$, and $(iii)$ above. Clearly, this suffices to guarantee that every $\mathit{expr}_j$ point interval occurs within an $\mathit{expr}_i$ interval. To guarantee that the property also holds for $\mathit{expr}_j$ non-point interval, let us assume $x < y$. In such a case, either property $(i)$ $(w = x)$ or property $(iii)$ $(w < x < z)$ above holds. Thanks to the last conjunct (line 8), if $x = w$, then there is an $\mathit{expr}_i$ interval $[x, z']$, with $z' > x$. Hence, we can conclude that there exists an $\mathit{expr}_i$ interval $[w, z]$, with $w \leq x < z$. The desired property follows from the fact that no $\mathit{expr}_i$ interval ends inside an $\mathit{expr}_j$ one.

Now, let $\varphi_R$ be the following formula:

$$\varphi_R = \ \mathit{expr}_n \ \wedge \ [A]\pi \ \wedge \bigwedge_{e_i \in sub(R)} \varphi_{\mathit{expr}_i} \ \wedge \bigwedge_{e_i \in sub(R)} \varphi^{end}_{\mathit{expr}_i} \ \wedge \bigwedge_{e_i \in sub(R)} \varphi^{\not\cap}_{\mathit{expr}_i}.$$

The following theorem holds (the proof is given in Appendix B).

**Theorem 2.** *Let $R$ be a regular expression over $\Sigma$. Then, $\mathcal{L}(R) = \{w \in \Sigma^* \mid w \approx M$ and $M = \langle \mathbb{I}(N), A, B, V\rangle$ is a model such that $M, [0, N-1] \models \varphi_R \wedge \varphi_\Sigma\}$.*

### *4.2. Encoding $\omega$-regular languages in AB*

The encoding of regular expressions can be lifted to $\omega$-regular ones. Since we are forced to work with finite intervals, the formula encoding an $\omega$-regular expression intuitively behaves as follows. An $\omega$-regular expression $E$ can be seen as the alternation $(+)$ of a finite number of expressions of the form $Re^\omega$, i.e., $E = R_1 e_1^\omega + \ldots + R_k e_k^\omega$, where, for all $i$, $R_i$ is regular. Formulas encoding expressions $E_i = R_i e_i^\omega$, with $i \in \{1, \ldots, k\}$, are meant to hold true on a certain finite prefix of $\mathbb{N}$, that represents the finite word captured by $R_i$, and use modality $\langle A\rangle$ to describe properties of the infinite suffix. Then, the encoding of $E$ consists of the disjunction of the formulas encoding the sub-expressions $E_i$.

Formally, the encoding of an $\omega$-regular expression $E$ into an $AB$ formula is defined inductively. The base case is given by regular sub-expressions, whose encoding has been illustrated in the previous section. Thus, we only need to specify how to handle the $\omega$-constructor, as well as alternation and concatenation (when the second operand is an $\omega$-regular expression).

- If $e_i = e_j + e_k$, where $e_j$ and $e_k$ are $\omega$-regular expressions, then $\varphi_{expr_i} = \varphi_{expr_j} \vee \varphi_{expr_k}$.
- If $e_i = e_j e_k$, where $e_j$ is a regular expression and $e_k$ is an $\omega$-regular one, then $\varphi_{expr_i} = expr_j \wedge \langle A\rangle\varphi_{expr_k}$.
- If $e_i = e_j^{\omega}$, where $e_j$ is a regular expression, then $\varphi_{expr_i} = expr_j \wedge \langle A\rangle(\neg\pi \wedge expr_j) \wedge [A][A](expr_j \rightarrow \langle A\rangle(\neg\pi \wedge expr_j))$.

Now, let $\varphi_E$ be the formula:

$$\varphi_E = \bigwedge_{e_i \in sub(E)} \varphi_{expr_i} \wedge \bigwedge_{e_i \in sub(E)} \varphi^{end}_{expr_i} \wedge \bigwedge_{e_i \in sub(E)} \varphi^{\not\emptyset}_{expr_i}.$$

The following theorem holds [1].

**Theorem 3.** *Let $E$ be an $\omega$-regular expression over $\Sigma$. Then, $\mathcal{L}(E) = \{w \in \Sigma^\omega \mid w \approx M$ and $M$ is a model such that $M, [0, n] \models \varphi_E \wedge \varphi_\Sigma$ for some $n \in \mathbb{N}\}$.*

## 5. Beyond $\omega$-regular languages

In the next sections, we provide the encodings of $\omega B$-, $\omega S$-, and $\omega T$-regular expressions into suitable extensions of $AB$ by building on the encodings of regular and $\omega$-regular expressions in $AB$ given in Section 4. The only new ingredients when stepping from $\omega$-regular languages to $\omega B$-, $\omega S$-, and $\omega T$-regular ones are the $B$-, $S$-, and $T$-constructor, respectively. Thus, the main problem is to define formulas $\varphi_{expr_i}$ encoding expressions $e_i$ of the form $e_j^B$, $e_j^S$, and $e_j^T$. Any such formula is a conjunction of two sub-formulas, a local one, which is the same used for expressions of the form $e_j^*$, and a global one, which guarantees the fulfillment of the constraints imposed by the $B$-, $S$-, and $T$-constructor, respectively.

Analogously to what we have done in Section 4.1 for regular languages (Theorem 2) and in Section 4.2 for $\omega$-regular ones (Theorem 3), in the following sections we provide the three main results of the paper, that is, we show how to encode $\omega B$-regular, $\omega S$-regular, and $\omega T$-regular expressions by means of, respectively, $AB\bar{A}$, $AB\sim$, and $AB\bar{A}\sim$ formulas. In Section 6, we show that for every $\omega B$-regular expression $E$, there is an $AB\bar{A}$ formula $\varphi$ such that $L(E) = L(\varphi)$ (Theorem 4), where $L(\varphi)$ is the $\omega$-language of words that are compatible with interval models satisfying $\varphi$, according to the relation $\approx$ of compatibility formalized in Section 2.3. Analogous results are presented for $\omega S$-regular expressions in Section 7 (Theorem 5) and $\omega T$-regular expressions in Section 8 (Theorem 6). Detailed proofs are given in Appendix C, Appendix D, and Appendix E.

Formulas resulting from the encodings given in Sections 6, 7, and 8 have the form $\varphi = \varphi_\Sigma \wedge \varphi_{E_*} \wedge \bigwedge_{(i,j)\in Z(E)} \Phi_Z^{(i,j)}$, with $Z \in \{B, S, T\}$, where:

- $\varphi_\Sigma \wedge \varphi_{E_*}$ is the $AB$ formula encoding, as shown in Section 4, the $\omega$-regular expression $E_*$, obtained from the $\omega Z$-regular expression $E$ by replacing each application of a $Z$-constructor $(.)^Z$ by the Kleene star $(.)^*$;
- $Z(E)$ is the set of indexes $(i, j)$ for which there are sub-expressions $e_i$ and $e_j$ in $E$ such that $e_i = (e_j)^Z$;
- $\Phi_Z^{(i,j)}$ is an $AB\bar{A}$, $AB\sim$, or $AB\bar{A}\sim$ formula (depending on whether $Z$ is equal to $B$, $S$, or $T$, respectively), forcing the constraints on the number of occurrences of words recognized by the sub-expression $e_j$, as prescribed by the semantics of the $Z$-constructor.

It is worth noticing that the correctness of the encodings of $\omega S$- and $\omega T$-regular expressions hinge upon the property of $S$- and $T$-regular languages stated in Corollary 2, as well as the property of prefix independence (see Section 3).

*Remark.* As a matter of fact, an encoding of $\omega B$-regular (resp., $\omega S$-regular) expressions in $AB\bar{A}$ (resp., $AB{\sim}$) formulas was proposed in [1] (resp., [12]).[6] Unfortunately, both encodings were flawed. In [2], we provide a counter-example showing that the encoding of $\omega B$-regular expressions in $AB\bar{A}$ formulas proposed in [1] is incorrect (a similar counterexample can be given for the encoding of $\omega S$-regular expressions in $AB{\sim}$ formulas proposed in [12]).

## 6. $\omega B$-regular languages in $AB\bar{A}$

In this section, we build, for every $\omega B$-expression $E$ and every $e_i, e_j \in sub(E)$, with $e_i = e_j^B$, a formula $\Phi_B^{(i,j)}$ that forces models to satisfy the boundedness constraint that the $B$-constructor imposes on $\omega$-words.

Let $B(E) = \{(i,j) \mid e_i, e_j \in sub(E), \text{with } e_i = (e_j)^B\}$. To force the proper behaviour of the $B$-constructor, for every $(i,j) \in B(E)$ we partition the interval model into intervals so that, eventually, the number of $expr_j$ intervals starting in the elements (i.e., intervals) of the partition is non-increasing. We also impose that no $expr_i$ interval contains an entire element of such partition. The boundedness constraint imposed by the $B$-constructor is then verified, since every $expr_i$ interval spans at most two elements of the partition and thus there is a bound to the number of $expr_j$ intervals in it. Intuitively, to force such a configuration we use the additional proposition letters $ph_j$, $bl_j$, and $p_j$. Proposition letter $bl_j$ defines the partition of the interval model, with each pair of consecutive $bl_j$ points identifying an element of the partition, while $ph_j$ is used to label points where $expr_j$ intervals start. Then, proposition letter $p_j$ defines a sequence of surjective functions (one for each element of the partition) from $ph_j$ points of an element of the partition to $ph_j$ points of the next element of the partition, thus ensuring that the number of $expr_j$ intervals starting in the elements of the partition is non-increasing. Technically, this is done by forcing

[6] The encodings given in [1] and [12] actually use languages $AB\bar{B}\bar{A}$ and $AB\bar{B}{\sim}$, that extend, respectively, $AB\bar{A}$ and $AB{\sim}$ with modality $\langle \bar{B} \rangle$. Such a modality simplifies the encodings, but it is not necessary.

the interval model to satisfy the following properties, expressed by means of suitable $AB\bar{A}$ formulas:

1. $ph_j$ and $bl_j$ may only label left endpoints of $expr_j$ intervals which are not left endpoints of $expr_i$ ones, but they cannot label the same points:
$$[G]((ph_j \vee bl_j \rightarrow \pi \wedge \langle A\rangle expr_j \wedge \neg\langle A\rangle expr_i) \wedge (ph_j \rightarrow \neg bl_j));$$
2. there exists $n \in \mathbb{N}$ such that every $n' > n$ which is the left endpoint of an $expr_j$ interval, but not the left endpoint of an $expr_i$ one, is labeled with either $ph_j$ or $bl_j$:
$$\langle A\rangle[A](\langle A\rangle expr_j \wedge [A]\neg expr_i \rightarrow \langle A\rangle(ph_j \vee bl_j));$$
3. in between two consecutive $bl_j$ points $x$ and $y$, with $x < y$, there exists at least one point $z$, with $x < z < y$, such that $z$ is the left endpoint of an $expr_i$ interval:
$$[G](\langle B\rangle bl_j \wedge \langle A\rangle bl_j \rightarrow \langle B\rangle(\neg\pi \wedge \langle A\rangle expr_i));$$
4. every $ph_j$ point is the left endpoint of exactly one $p_j$ interval:
$$[G](ph_j \rightarrow \langle A\rangle p_j) \wedge [G](p_j \rightarrow \neg\langle B\rangle p_j);$$
5. every $p_j$ interval is begun by a $ph_j$ point and strictly contains exactly one $bl_j$ point:
$$[G](p_j \rightarrow \langle B\rangle ph_j \wedge \langle B\rangle(\neg\pi \wedge \langle A\rangle bl_j) \wedge [B](\langle A\rangle bl_j \rightarrow \neg\langle B\rangle\langle A\rangle bl_j));$$
6. every $ph_j$ point $x$ such that there exists a $bl_j$ point $y$, with $y < x$, is the right endpoint of at least one $p_j$ interval:
$$[G](\langle A\rangle ph_j \wedge \langle B\rangle bl_j \rightarrow \langle A\rangle\langle\bar{A}\rangle p_j).$$

Figure 4 gives a graphical account of the above properties. Properties 1–2 guarantee that, from a point on, say it $n$, the points that are the left endpoint of an $expr_j$ interval, but not of an $expr_i$ one, are exactly those labeled with either $ph_j$ or $bl_j$. The suffix starting at $n$ can be seen as a (possibly finite or even empty) sequence of slices $[n_0, n_1], [n_1, n_2] \ldots$, where $\{n_0 < n_1 < \ldots\}$ is the

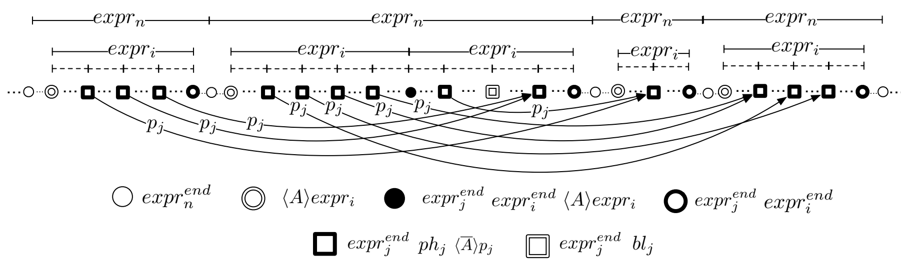


Figure 4: Example of the structure we are enforcing by means of formula $\Phi_B^{(i,j)}$ for an expression $E = (e_n)^\omega$, where $e_n$ contains the sub-expression $e_i = e_j^B$ (dashed intervals represent $expr_j$ intervals).

ordered set of $bl_j$ points greater than $n$. Now, let $[n_k, n_{k+1}]_{ph_j}$ be the set of $ph_j$ points $x$ laying strictly in between $n_k$ and $n_{k+1}$; recall that, by properties 1–2, these are exactly the left endpoints of $expr_j$ intervals that are not left endpoints of $expr_i$ ones. By properties 4–6, $p_j$ encodes a series of surjective functions $f_k : [n_k, n_{k+1}]_{ph_j} \to [n_{k+1}, n_{k+2}]_{ph_j}$, with $k \geq 0$, linking the $ph_j$ points of pairs of consecutive slices.[7] It follows that $|[n_0, n_1]_{ph_j}| \geq |[n_1, n_2]_{ph_j}| \geq \ldots$, that is, the sequence is not increasing. Finally, property 3 imposes that, for every $k$, there is at least one point $x$, with $n_k < x < n_{k+1}$, which is the left endpoint of an $expr_i$ interval. Then, every $expr_i$ interval starting after $n$ spans at most two adjacent slices, and thus it contains at most $|[n_0, n_1]_{ph_j}| * 2$ many $expr_j$ intervals, thus providing a bound, as required by the $B$-constructor.

Now, for every $(i,j) \in B(E)$, let $\Phi_B^{(i,j)}$ be the conjunction of the above formulas. The following theorem holds, where, in conformity with the notation introduced in Section 2.1.1, immediately before Lemma 1, we denote by $E_*$ the expression obtained from $E$ by replacing $B$-constructors with $*$-constructors.

**Theorem 4.** *Let $E$ be an $\omega B$-regular expression over $\Sigma$. Then, $\mathcal{L}(E) = \{w \in \Sigma^\omega \mid w \approx M$ and $M$ is a model such that $M, [0,n] \models \varphi_{E_*} \wedge \varphi_\Sigma \wedge \bigwedge_{(i,j)\in B(E)} \Phi_B^{(i,j)}$ for some $n \in \mathbb{N}\}$.*

[7] As a matter of fact, the image of one such function $f_k$ might also include elements not belonging to $[n_{k+1}, n_{k+2}]_{ph_j}$; however, properties 4–6 guarantee that $[n_{k+1}, n_{k+2}]_{ph_j}$ is included in the image, which is enough for our purposes.

## 7. $\omega S$-regular languages in $AB\sim$

In analogy to the previous section, for every $\omega S$-expression $E$ and every $e_i, e_j \in sub(E)$, with $e_i = (e_j)^S$, we build a formula $\Phi_S^{(i,j)}$ that forces models to satisfy the strongly unboundedness constraint the $S$-constructor imposes on $\omega$-words. As already pointed out at the end of Section 3, it makes sense to force the behavior of the $S$-constructor (the same applies with the $T$-constructor in the next section) only in those models featuring infinitely many $expr_i$ intervals. Scenarios where there are only finitely many $expr_i$ intervals are easier to deal with, as one can simply ignore the constraints imposed by the $S$-constructor, by suitably guarding the formula that encodes them. Roughly speaking, one can treat the $S$-constructor as if it were the $*$-constructor in all those models featuring only finitely many $expr_i$ intervals. Thus, the formulas we present in the following assume that there are infinitely many $expr_i$ intervals, and scenarios with only finitely many $expr_i$ intervals are dealt with by suitably guarding them.

An additional, analogous simplification applies when considering models featuring infinitely many $expr_j$ points, which correspond to empty strings belonging to the language of $e_j$. Thanks to Corollary 2, if a model features infinitely many $expr_j$ points, then the $S$-constructor behaves as the $*$-constructor, and, as a consequence, we can ignore the constraint that it imposes on $\omega$-words. Thus, formulas $\Phi_S^{(i,j)}$, that we are going to define, also assume that there are only finitely many $expr_j$ points, and are suitably guarded in order to have no effect over models featuring infinitely many $expr_j$ points.

Let $S(E) = \{(i,j) \mid e_i, e_j \in sub(E), \text{with } e_i = (e_j)^S\}$. To force the proper behaviour of the $S$-constructor, we make use of the atomic symbol $\sim$, which is a special proposition letter encoding some equivalence relation between points of the interval structure. Moreover, for every $(i,j) \in S(E)$, we introduce two proposition letters, namely $ph_j$ and $new_j$. The idea of the encoding is to label with $ph_j$ some of the point intervals inside $expr_i$ intervals and use such $ph_j$ intervals to establish a lower bound to the number of $expr_j$ intervals occurring inside $expr_i$ intervals. More precisely, if an $expr_i$ interval contains $n$ points labeled with $ph_j$, then it must contain at least $n$ intervals labeled with $expr_j$. Then, by forcing the sequence of the numbers of $ph_j$ points contained in each $expr_i$ interval to be nondecreasing and (by means of the proposition letter $new_j$) unbounded, we guarantee the strong unboundedness constraint imposed by the $S$-constructor. More technically, by means of suitable $AB\sim$ formulas, we force the following properties:

1. $ph_j$ may only label left endpoints of $expr_j$ intervals which are neither left nor right endpoints of $expr_i$ ones (together with the fact that an $expr_j$ interval can only occur inside an $expr_i$ one, this implies that a $ph_j$ point can only occur strictly inside an $expr_i$ interval):
$$[G](ph_j \rightarrow \pi \wedge \langle A\rangle expr_j \wedge \neg\langle A\rangle expr_i \wedge \neg expr_i^{end});$$
2. $ph_j$ intervals are $\sim$-equivalent to other $ph_j$ intervals only, and if two distinct $ph_j$ points $x$ and $y$ belong to the same $expr_i$ interval, then $x \not\sim y$ (equivalently, if $x \sim y$, then $x$ and $y$ belong to two distinct $expr_i$ intervals):
$$[G](\sim\rightarrow ((\langle B\rangle ph_j \leftrightarrow \langle A\rangle ph_j) \wedge (\langle B\rangle ph_j \rightarrow \langle B\rangle(\neg\pi \wedge \langle A\rangle expr_i^{end})));$$
3. for every $expr_i$ interval $[n, n']$ that strictly contains at least one $ph_j$ point, there is another $expr_i$ interval that starts not earlier than $n'$ and contains a $ph_j$ point; moreover, for each $ph_j$ point in $[n, n']$, there is a $ph_j$ point $y$, with $x \sim y$, belonging to the next $expr_i$ interval $[n'', n''']$, that is, there is not another $expr_i$ interval starting in between $n$ and $n'''$:
$$[G](ph_j \rightarrow \langle A\rangle(\neg\pi \wedge \sim \wedge [B](\langle A\rangle expr_i^{end} \rightarrow [B][A]\neg expr_i^{end})));$$
4. $new_j$ points are $ph_j$ points; moreover, if $x$ is a $new_j$ point, then there is no point $y$ such that $y < x$ and $y \sim x$; finally, every $ph_j$ point is eventually followed by a distinct $new_j$ point:
$$[G]\,((new_j \rightarrow ph_j) \wedge (\neg\pi \wedge \ \sim\rightarrow [A]\neg new_j) \wedge (ph_j \rightarrow \langle A\rangle(\neg\pi \ \wedge \ \langle A\rangle new_j)))\,.$$

A graphical account of the properties imposed by the above formulas is given in Figure 5. Thanks to properties 1 and 2, every $ph_j$ point must fall strictly inside an $expr_i$ interval and it is only $\sim$-equivalent to other $ph_j$ points.

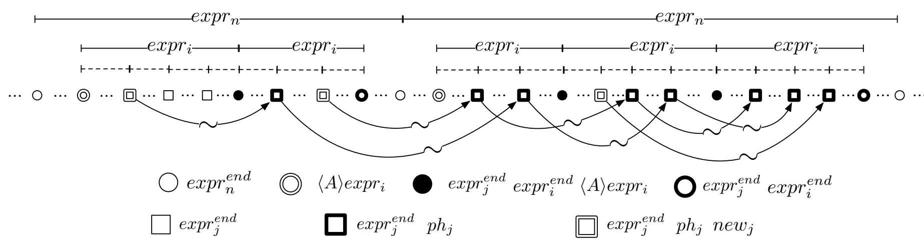


Figure 5: Example of the structure we are enforcing by means of formula $\Phi_S^{(i,j)}$ for an expression $E = (e_n)^\omega$, where $e_n$ contains the sub-expression $e_i = e_j^S$ (dashed intervals represent $expr_j$ intervals).

Therefore, thanks to property 3, we are able to guarantee that, if there is a $ph_j$ point (which must fall strictly inside an $expr_i$ interval), then there is an infinite sequence of non-overlapping $expr_i$ intervals, each of them containing at least one $ph_j$ point (and thus being a non-point interval). For each such $expr_i$ intervals and each $ph_j$ point inside it, there is a $ph_j$ point $y$ in the next $expr_i$ interval in the sequence such that $x \sim y$. Additionally, thanks again to property 2, such a $ph_j$ point $y$ is unique, that is, there cannot be another $ph_j$ point $y'$ in the same $expr_i$ interval as $y$ such that $x \sim y'$; analogously, there cannot be two distinct $ph_j$ points $x, x'$ belonging to the same $expr_i$ interval such that both $x \sim y$ and $x' \sim y$ hold. Roughly speaking, $\sim$ establishes infinitely many injective functions $f_k$ ($k \in \mathbb{N}$) from the set of $ph_j$ points in the $k$th $expr_i$ interval to the set of $ph_j$ points in the $(k+1)$th $expr_i$ interval.

As a consequence, properties 1–3 ensure that, as long as there is a $ph_j$ point in the model, the sequence of the numbers of $ph_j$ points contained in each $expr_i$ interval is nondecreasing. Notice that properties 1–3 also force the existence of a point in the model starting from which no more $expr_i$ points occur. Property 4 forces the sequence of numbers of $ph_j$ points in each $expr_i$ to be unbounded as well, by forcing the existence of infinitely many special $ph_j$ points, labeled with $new_j$, that are $\sim$-equivalent only to points that follow them in the model (formally, if $x$ is a $new_j$ point, then there is no $y < x$ with $y \sim x$). The existence of infinitely many $new_j$ points produces the effect of having infinitely many non-surjective functions in the aforementioned set $\{f_h\}_{h \in \mathbb{N}}$ of injective functions, thus implying that the numbers of $ph_j$ points contained in every

$expr_i$ is unbounded.

Finally, thanks to property 1, we have that the number of $expr_j$ interval contained in an $expr_i$ is actually greater than the number of $ph_j$ points contained in that $expr_i$ interval. Therefore, as long as there is a $ph_j$ point in the model, the behaviour of the $S$-constructor is correctly captured.

As a last observation, we emphasize that establishing a nondecreasing unbounded sequence that represents a lower bound to the number of $expr_j$ intervals occurring in an $expr_i$ one is enough to satisfy the constraints imposed by the $S$-constructor. Indeed, as shown in Figure 5, there can be $expr_j$ intervals whose left endpoint is not labeled with $ph_j$, meaning that the number of $expr_j$ intervals can be greater than the number of $ph_j$ points; more precisely, it can fluctuate but it cannot go below a certain threshold which eventually grows indefinitely.

As pointed out above, the presence of a $ph_j$ point causes the existence of infinitely many $expr_i$ intervals and finitely many $expr_i$ points. Thus, to complete the encoding, the formula we are building must also admit models featuring finitely many $expr_i$ intervals or infinitely many $expr_i$ points. As for the former class of models, featuring only finitely many $expr_i$ intervals, we made already clear that we can simply treat the $S$-constructor as the $*$-constructor. As for the second class of models, instead, observe that the presence of infinitely many $expr_i$ points that are not $expr_j$ points would break the strongly unboundedness constraint imposed by the $S$-constructor. Thus, models featuring infinitely many $expr_i$ points must feature infinitely many $expr_j$ points as well. As already noted, also in this case it is possible to ignore the constraints imposed by the $S$-constructor, thanks to Corollary 2.

Therefore, to complete the encoding, it suffices to add a formula (to be put in conjunction with the above ones) that constrains the model to feature at least one $ph_j$ point if it features infinitely many $expr_i$ intervals, but only finitely many $expr_j$ points:

$$[G]\langle A\rangle\langle A\rangle expr_i \ \wedge\ \langle A\rangle[A][A](expr_j \ \to\ \neg\pi) \to \langle B\rangle\langle A\rangle\langle A\rangle ph_j.$$

For all $(i,j) \in S(E)$, let $\Phi_S^{(i,j)}$ be the conjunction of the above formulas. The following theorem holds (in analogy to the previous sections, we denote by $E_*$ the expression obtained from $E$ by replacing $S$-constructors with $*$-constructors).

**Theorem 5.** *Let $E$ be an $\omega S$-regular expression over $\Sigma$. Then, $\mathcal{L}(E) = \{w \in \Sigma^\omega \mid w \approx M$ and $M$ is a model such that $M, [0, n] \models \varphi_{E_*} \wedge \varphi_\Sigma \wedge \bigwedge_{(i,j) \in S(E)} \Phi_S^{(i,j)}$ for some $n \in \mathbb{N}\}$.*

## 8. $\omega T$-regular languages in $AB\bar{A}{\sim}$

As in the previous sections, we build here, for every $\omega T$-expression $E$ and every $e_i, e_j \in \mathit{sub}(E)$, with $e_i = (e_j)^T$, a formula $\Phi_T^{(i,j)}$ that forces models to satisfy the constraint the $T$-constructor imposes on $\omega$-words.

To start with, we observe that, as in the case of the $S$-constructor, it is not necessary to impose any constraint over models featuring only finitely many $\mathit{expr}_i$ intervals. Additionally, by Corollary 2, if a model features infinitely many $\mathit{expr}_j$ points, then we are allowed to verify a simpler constraint, that is, there are infinitely many $\mathit{expr}_i$ intervals containing exactly $k$ many $\mathit{expr}_j$ intervals, for some $k > 0$. This will be explained in more detail later in the section.

Let $T(E) = \{(i, j) \mid e_i, e_j \in \mathit{sub}(E), \text{with } e_i = (e_j)^T\}$. To encode $\omega T$-regular languages in $AB\bar{A}{\sim}$, we first show that a particular class of models over $\mathbb{N}$ can be captured by a conjunction of $AB\bar{A}{\sim}$ formulas $\Phi_\infty^{(i,j)}$, for $(i, j) \in T(E)$, which make use of proposition letters $ph_j, bl_j, p_j, q_j$, and $\mathit{conf}_j$, as well as the proposition letter $\sim$, representing an equivalence relation over $\mathbb{N}$. Models of such a class (see Figure 6) are partitioned, for every $(i, j) \in T(E)$, into *configurations* (intervals whose endpoints are consecutive $\mathit{conf}_j$ points). Every configuration is partitioned in *blocks* (intervals whose endpoints are consecutive $bl_j$ points), which, in turn, contain $ph_j$ points. Proposition letter $\sim$ is used to force $ph_j$ points belonging to the same block to be equivalent, and $ph_j$ points belonging to different blocks of the same configuration not to be equivalent. Propositional letter $p_j$ is then used to encode partial surjective functions from $ph_j$ points of a block to $ph_j$ points of the next block in the same configuration, if any; this ensures that, within each configuration, blocks contain a decreasing number of $ph_j$ points. Every block (belonging to a configuration) is associated to a block in

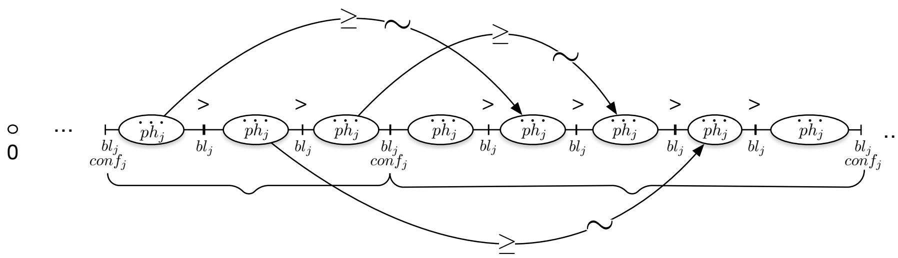


Figure 6: Example of the type of structure we enforce by means of formula $\Phi_{\infty}^{(i,j)}$.

the next configuration, and $ph_j$ points belonging to associated blocks are forced to be equivalent (using proposition letter $\sim$). This ensures that two blocks cannot be associated to the same block in the next configuration. Moreover, formula $\Phi_{\infty}^{(i,j)}$ constrains every configuration to contain at least one block not associated to any block in previous configurations. Thus, configurations feature an increasing number of blocks, and there are infinitely many infinite chains of blocks belonging to consecutive configurations. Similarly to $p_j$, propositional letter $q_j$ encodes (possibly partial) surjective functions from $ph_j$ points of a block to $ph_j$ points of the block associated with it in the next configuration; this ensures that the number of $ph_j$ points in a block is not smaller than the number of $ph_j$ points in its associated block (in the next configuration). Therefore, given an infinite chain of associated blocks, the number of $ph_j$ points contained in its blocks eventually converges to a constant value. We have, then, infinitely many chains, each of which converges to a number of $ph_j$ points.

At this point, for every $(i,j) \in T(E)$, we can force, by means of formula $\Phi^{in_j}$, every chain to have infinitely many bijective correspondences between the $ph_j$ points of one of its blocks and the $expr_j^{end}$ intervals contained in an $expr_i$ interval. This amounts to force the behaviour of the $T$-constructor. Formally, we want to characterize, through $AB\bar{A}\sim$ formula $\Phi_{\infty}^{(i,j)}$, the models that satisfy the following properties:

1. $ph_j, bl_j$, and $conf_j$ only appear as labels of points, $ph_j$ and $bl_j$ never occur together in the same labeling, and $conf_j$ only appears in a labeling containing also $bl_j$, that is, a *configuration* (i.e., an interval whose endpoints

are consecutive $\mathit{conf}_j$ points) features one or more *blocks* (i.e., intervals whose endpoints are consecutive $bl_j$ points):

$$[G]((bl_j \vee ph_j \to \pi) \wedge (\mathit{conf}_j \to bl_j) \wedge (ph_j \to \neg bl_j));$$

2. there are infinitely many $\mathit{conf}_j$ points, that is, there are infinitely many configurations:

$$[G]\langle A\rangle\langle A\rangle \mathit{conf}_j;$$

3. between two consecutive $bl_j$ points there is at least one $ph_j$ point and all $ph_j$ points falling inside the same block also belong to the same equivalence class, that is, each block is associated with exactly one equivalence class of $ph_j$ points:

$$[G](\langle B\rangle bl_j \ \wedge \ \langle A\rangle bl_j \ \to \ \langle B\rangle(\neg\pi \wedge \langle A\rangle ph_j)) \wedge$$
$$[G](\langle B\rangle ph_j \wedge \langle A\rangle ph_j \wedge [B][A]\neg bl_j \to\sim);$$

4. $ph_j$ points are only $\sim$-equivalent to other $ph_j$ points and, for every pair of distinct $ph_j$ points $x$ and $y$, if there is a $bl_j$ point but no $\mathit{conf}_j$ point between them, then $x \not\sim y$, that is, pairs of distinct blocks in the same configuration represent distinct equivalence classes of $ph_j$ points:

$$[G](\sim \to \ (\langle B\rangle ph_j \ \leftrightarrow \ \langle A\rangle ph_j)) \wedge$$
$$[G](\sim \wedge\langle B\rangle ph_j \ \wedge \ \langle B\rangle(\neg\pi \wedge \langle A\rangle bl_j) \to \langle B\rangle\langle A\rangle \mathit{conf}_j);$$

5. $p_j$ intervals connect $ph_j$ points belonging to consecutive blocks inside the same configuration; more precisely, for every $ph_j$ point $x$ of a block that is not the first block of a configuration, there is a distinguished $ph_j$ point $y$ in the previous block (belonging to the same configuration) such that $[y, x]$ is a $p_j$ interval, where by *distinguished* we mean that there cannot be two distinct $ph_j$ points $x, x'$ and a point $y$ such that $[y, x]$ and $[y, x']$ are $p_j$ intervals; every block contains at least one $ph_j$ point (the last one) that is not connected to any $ph_j$ point in the future (notice that, as a consequence, the number of $ph_j$ points in a block is greater than the number of $ph_j$ points in the next block of the same configuration, if any,

i.e., for every configuration, the finite sequence given by the numbers of $ph_j$ points featured in each block of that configuration is strictly decreasing):

$$[G](p_j \rightarrow \langle B\rangle ph_j \wedge \langle A\rangle ph_j \wedge [B]\neg p_j \wedge [B][A]\neg \mathit{conf}_j \wedge$$

$$\langle B\rangle\langle A\rangle bl_j \wedge [B](\langle A\rangle bl_j \rightarrow [B][A]\neg bl_j)) \wedge$$

$$[G](ph_j \wedge \langle \bar{A}\rangle(\langle B\rangle bl_j \wedge [B][A]\neg \mathit{conf}_j) \rightarrow \langle \bar{A}\rangle p_j);$$

$$[G](ph_j \wedge [A](\neg\pi \wedge \sim\rightarrow \langle B\rangle\langle A\rangle bl_j) \rightarrow [A]\neg p_j) \wedge$$

6. for every $ph_j$ point $x$ there is a $ph_j$ point $y > x$ such that $x \sim y$ and there is exactly one $\mathit{conf}_j$ point between $x$ and $y$, that is, an equivalence class (corresponding to a block) in a configuration is witnessed in all the following configurations:

$$[G](ph_j \rightarrow \langle A\rangle(\sim \wedge\langle B\rangle\langle A\rangle \mathit{conf}_j \wedge [B](\langle A\rangle \mathit{conf}_j \rightarrow [B][A]\neg \mathit{conf}_j)));$$

7. every configuration contains at least one $ph_j$ point $x$ such that there is no point $y$ with $y < x$ and $y \sim x$ (observe that this implies that every configuration features a block that *starts* a new equivalence class, and thus the infinite sequence of numbers of blocks in configurations is strictly increasing):

$$[G](\mathit{conf}_j \rightarrow \langle A\rangle(\langle A\rangle ph_j \wedge [B](\neg\pi \rightarrow [A]\neg \mathit{conf}_j) \wedge \langle A\rangle[\bar{A}]\neg \sim));$$

8. for every $ph_j$ point $x$ belonging to a block $b$ that does not start a new equivalence class, that is, such that there is a unique block $b'$ associated with the same equivalence class as $b$ in the previous configuration, there is a distinguished $ph_j$ point $y$ belonging to block $b'$ such that $[y, x]$ is a $q_j$ interval (once again, here *distinguished* means that there cannot be two points $x, x'$ such that $[y, x]$ and $[y, x']$ are $q_j$ intervals for some $y$ – this means that, for every equivalence class, the sequence given by the numbers of $ph_j$ points contained in each block of that equivalence class is non-increasing):

$$[G](q_j \rightarrow \sim \wedge[B]\neg q_j \wedge \langle B\rangle\langle A\rangle \mathit{conf}_j \wedge [B](\langle A\rangle \mathit{conf}_j \rightarrow [B][A]\neg \mathit{conf}_j)) \wedge$$

$$[G](ph_j \wedge \langle \bar{A}\rangle(\sim \wedge\langle B\rangle\langle A\rangle bl_j) \rightarrow \langle \bar{A}\rangle q_j).$$

Let $\Phi_\infty^{(i,j)}$ be the conjunction of the above formulas. A graphical account of the structure enforced by $\Phi_\infty^{(i,j)}$ is given in Figure 6. Notice that there may be points not labeled with any of $ph_j, bl_j$, and $conf_j$.

Thanks to $\Phi_\infty^{(i,j)}$, a model can be seen as an infinite sequence of configurations $[conf_j^0, conf_j^1], [conf_j^1, conf_j^2], \ldots$. For every $x \in \mathbb{N}$, $[conf_j^x, conf_j^{x+1}]$ contains a finite sequence of $n_j(x)+1$, with $n_j : \mathbb{N} \to \mathbb{N}$, sets $blk_j^{x,0}, \ldots, blk_j^{x,n_j(x)}$ of $ph_i$ points each one associated with exactly one equivalence class, i.e., points in $blk_j^{x,y}$ belong to the same equivalence class, for every $y \in \{0, \ldots, n_j(x)\}$. More precisely, $n_j(x)+1$ is the number of blocks in the $x$-th configuration $[conf_j^x, conf_j^{x+1}]$ and $blk_j^{x,y}$ is the set of $ph_j$ points in the $y$-th block of the $x$-th configuration. For every $(i,j) \in T(E)$, the following properties hold:

(P1) function $n_j(x)$ is strictly increasing (property 7);

(P2) for every $x \in \mathbb{N}$, sequence $\langle |blk_j^{x,y}| \rangle_{0 \le y \le n(x)}$ is strictly decreasing (property 5);

(P3) for every $ph_j$ point $w$, it is possible to identify a configuration index $x$ (referring to the $x$-th configuration in the model) and an infinite sequence of indexes $\langle y_k \rangle_{k \in \mathbb{N}}$ (referring to positions of blocks in consecutive configurations starting from the $x$-th configuration, i.e., $y_k$ refers to the $y_k$-th block in the $(x+k)$-th configuration), such that $[w]_\sim = \bigcup_{k \in \mathbb{N}} blk_j^{x+k,y_k}$ (property 6) and sequence $\langle |blk_j^{x+k,y_k}| \rangle_{k \in \mathbb{N}}$ is non-increasing (property 8);

Property $(P3)$ states that, for every equivalence class $[w]_\sim$ of $ph_j$ points, there is a configuration such that $[w]_\sim$ is witnessed by exactly one block in each of the successive configurations. Moreover, it states that the blocks that witness $[w]_\sim$ feature a non-increasing number of points. Let $x$ and $\langle y_0, y_1, y_2, \ldots \rangle$ be, respectively, the index and the infinite sequence of indexes such that $[w]_\sim = \bigcup_{k \in \mathbb{N}} blk_j^{x+k,y_k}$, whose existence is guaranteed by property $(P3)$. Since the number of points in each block (in particular, in $blk_j^{x,y_0}$) is finite, there is $k' \in \mathbb{N}$ for which $|blk_j^{x+k',y_{k'}}| = |blk_j^{x+k'+1,y_{k'+1}}| = \ldots$, i.e., sequence $\langle |blk_j^{x+k,y_k}| \rangle_{k \in \mathbb{N}}$ converges to a single value, called the *value of the equivalence class* $[w]_\sim$ and denoted by $val(w)$. By $(P2)$, it holds that for any two $ph_j$ points $w$ and $w'$, with $w \not\sim w'$, i.e., $w$ and $w'$ belong to distinct equivalence classes, it holds that $val(w) \neq val(w')$; otherwise, there would eventually be a configuration featuring two distinct blocks with the same number of $ph_j$ points, which contradicts $(P2)$. Finally, $(P1)$ guarantees that the number of distinct equivalence classes is infinite. Therefore, the image of $val$ is infinite, i.e., there are infinitely many natural numbers $n$ with $val(w) = n$ for some $w$.

We say that a block is *instantiated* with an $expr_i$ interval when the block contains an $expr_i$ interval that, in turn, embeds all the $ph_j$ points falling in that block and, in addition, the set of $ph_j$ points in the block and the set of points starting an $expr_j$ interval within the $expr_i$ interval coincide. An *instantiation of an equivalence class* with an $expr_i$ interval is an instantiation of a block witnessing that equivalence class with the $expr_i$ interval. Since an instantiation of a block with an $expr_i$ interval establishes a bijective correspondence between the $ph_j$ points in the block and the $expr_j$ intervals in the $expr_i$ interval, it is clear that enforcing the behaviour imposed by the $T$-constructor amounts to force all the equivalence classes to have infinitely many instantiations with $expr_i$ intervals. Indeed, if an equivalence class $[w]_\sim$ is instantiated infinitely often, there are infinitely many $expr_i$ intervals containing exactly $val(w)$ many $expr_j$ intervals. Since the number of equivalence classes is infinite and they have all distinct values $val(\cdot)$, the behaviour of the $T$-constructor is correctly encoded.

In what follows we show how to force all the equivalence classes to be instantiated infinitely many times with $expr_i$ intervals by means of $AB\bar{A}{\sim}$ formula $\Phi^{in_j}$. For an $expr_i$ interval $[x, y]$, let $points_j([x, y]) = \{z \mid x \leq z \leq y$ and $M, [z, z'] \models expr_j$ for some $z'\}$, and, for a $ph_j$ point $w$, let us denote by $x_w$ and $\sigma_w = \langle y_0, y_1, y_2, \ldots\rangle$, respectively, the index and the infinite sequence of indexes such that $[w]_\sim = \bigcup_{k\in\mathbb{N}} blk_j^{x_w+k, y_k}$ (existence of $x_w$ and $\sigma_w$ is guaranteed by property $(P3)$). For every equivalence class $[w]_\sim$ of $ph_j$ points, formula $\Phi^{in_j}$ forces the existence of an infinite sub-sequence $\langle y_{k_1}, y_{k_2}, y_{k_3}, \ldots\rangle$ of $\sigma_w$ such that the $y_{k_h}$-th block of the $(x_w + k_h)$-th configuration is instantiated with an $expr_i$

interval, i.e., each equivalence class is instantiated infinitely many times. To this end, we use proposition letter $in_j$ to mark the infinite sequence of blocks to be instantiated with an $expr_i$ interval; more precisely, we force $in_j$ to hold true exactly on points starting $expr_j$ intervals contained in blocks of the relevant sequence. For $(i,j) \in T(E)$, let $\Phi^{in_j}$ be the conjunction of the following formulas:

- $in_j$ appears only as the label of $ph_j$ points that begin $expr_j$ intervals:

$$[G](in_j \rightarrow ph_j \wedge \langle A \rangle expr_j);$$

- either none or all of the $ph_j$ points in a block are $in_j$ points as well:

$$[G](((\langle B \rangle ph_j \wedge \langle A \rangle ph_j \wedge [B][A]\neg bl_j) \rightarrow (\langle A \rangle in_j \leftrightarrow \langle B \rangle in_j));$$

- if an $expr_i$ interval contains an $in_j$ point, then the $expr_j$ intervals within it begin with an $in_j$ point:

$$[G](expr_i \wedge \langle B \rangle \langle A \rangle in_j \rightarrow [B]((\langle A \rangle expr_j \rightarrow \langle A \rangle in_j)).$$

- every block containing $in_j$ points encloses an $expr_i$ interval that, in turn, contains all the $in_j$ points belonging to that block:

$$\begin{aligned}[G](expr_i \wedge \langle B \rangle \langle A \rangle in_j \rightarrow {} & [B][A]\neg bl_j \wedge \\ & [\bar{A}]([B][A]\neg bl_j \rightarrow [B][A]\neg in_j) \wedge \\ & [A]([B][A]\neg bl_j \rightarrow [B][A]\neg in_j));\end{aligned}$$

At this point, formula $[G](ph_j \rightarrow \langle A \rangle(\neg\pi \wedge \sim \wedge \langle A \rangle in_j))$ forces every equivalence class, that is, every $ph_j$ point, to be instantiated infinitely many times with $expr_i$ intervals. Thus, the conjunction of this last formula with formulas $\Phi_\infty^{(i,j)}$ and $\Phi^{in_j}$ above forces models to behave accordingly to the $T$-constructor. However, as it is the case with the $S$-constructor, there are models that do not satisfy such a conjunction but, still, may encode words belonging to the language of the $\omega T$-regular expression we are trying to encode. This is the case with models featuring only finitely many $expr_i$ intervals and models featuring infinitely many $expr_j$ points. Obviously, models in the former class do not satisfy the above conjunction (that forces the existence of infinitely many $expr_i$

intervals), but they can anyway correspond to words belonging to the language because, as already pointed out, in these cases the $T$-constructor behaves as the $*$-constructor, due to prefix independence property. Consider, instead, models that feature infinitely many $expr_j$ points and do not satisfy the above conjunction, i.e., do not instantiate *all* equivalence classes infinitely many times with $expr_i$ intervals. Thanks to Corollary 2, in this scenario the behaviour enforced by the $T$-constructor is preserved as long as *at least one* equivalence class is instantiated infinitely many times with $expr_i$ intervals.

Thus, we can now define formulas $\Phi_T^{(i,j)}$, for $(i,j) \in T(E)$, as follows, where the first disjunct captures models featuring only finitely many $expr_i$ intervals, the second one models featuring infinitely many $expr_j$ points, and the third one deals with all other scenarios.

$$\begin{aligned}
&\langle A\rangle[A][A]\neg expr_i \vee \\
&\big(\Phi_\infty^{(i,j)} \wedge \Phi^{in_j} \wedge [G]\langle A\rangle\langle A\rangle(\pi \ \wedge \ expr_j) \wedge \\
&\qquad\qquad \langle B\rangle\langle A\rangle\langle A\rangle in_j \ \wedge \ [G](in_j \rightarrow \langle A\rangle(\neg\pi \wedge \ \sim \ \wedge \langle A\rangle in_j))\big) \vee \\
&\big(\Phi_\infty^{(i,j)} \wedge \Phi^{in_j} \ \wedge \ \langle A\rangle[A][A](expr_j \rightarrow \neg\pi) \ \wedge \ [G](ph_j \rightarrow \langle A\rangle(\neg\pi \wedge \ \sim \wedge \langle A\rangle in_j))\big).
\end{aligned}$$

The following theorem holds (once more, $E_*$ is obtained from $E$ by replacing $T$-constructors by $*$-constructors).

**Theorem 6.** *Let $E$ be an $\omega T$-regular expression over $\Sigma$. Then, $\mathcal{L}(E) = \{w \in \Sigma^\omega \mid w \approx M$ and $M$ is a model such that $M, [0,n] \models \varphi_{E_*} \wedge \varphi_\Sigma \wedge \bigwedge_{(i,j)\in T(E)} \Phi_T^{(i,j)}$ for some $n \in \mathbb{N}\}$.*

## 9. Conclusions

In this paper, we filled a gap in the study of extended $\omega$-regular languages by providing a temporal logic characterization of $\omega B$-, $\omega S$-, and $\omega T$-regular languages. We identified interval temporal logic as a suitable candidate for such a role. We first provided an encoding of regular and $\omega$-regular languages into the interval temporal logic $AB$ of Allen's relations *meets* and *begun by*. Then, we showed how to enrich $AB$ in order to turn $\omega B$-, $\omega S$-, and $\omega T$-regular expressions into formulas of suitable interval temporal logics. We focused on $B$-, $S$-, and

$T$-constructors in isolation, but the proposed encodings can be easily merged to deal with their combinations ($\omega BS$-, $\omega BT$-, $\omega ST$-, and $\omega BST$-regular expressions). As for future work, we are looking for syntactic and/or semantic fragments of the considered interval temporal logics that preserve (un)satisfiability of the resulting formulas and behave better from a computational point of view.

## Appendix A. Proofs of Section 3

**Proposition 1.** *Let $e$ be a BST-regular expression. If $\vec{u}, \vec{v} \in \mathcal{L}(e)$ and $\vec{w}$ is a shuffle of $\vec{u}$ and $\vec{v}$, then $\vec{w} \in \mathcal{L}(e)$ as well.*

*Proof.* The proof is by induction on the size of $BST$-regular expressions. If $e = \emptyset$, then the claim follows straightforwardly, since the antecedent of the implication is false. If $e = a$ for some $a \in \Sigma$, then we have that $\vec{u} = \vec{v} = \vec{w} = (a, a, a, \ldots) \in \mathcal{L}(a)$.

Now, let $\vec{u}, \vec{v} \in \mathcal{L}(e)$ and $\vec{w} = \vec{u} +_g \vec{v}$, for a selection function $g$. This means that $\vec{w} \in \mathcal{L}(e + e)$. We show that $\vec{w} \in \mathcal{L}(e)$.

If $e = e_1 \cdot e_2$, then there are word sequences $\vec{u'}, \vec{v'} \in \mathcal{L}(e_1)$ and $\vec{u''}, \vec{v''} \in \mathcal{L}(e_2)$ such that $\vec{u} = \vec{u'} \odot \vec{u''}$ and $\vec{v} = \vec{v'} \odot \vec{v''}$, i.e., $\vec{u}$ (resp., $\vec{v}$) corresponds to the application component-wise of the word concatenation operator $\cdot$ to $\vec{u'}$ and $\vec{u''}$ (resp., $\vec{v'}$ and $\vec{v''}$). It is easy to see that $\vec{w} = (\vec{u'} +_g \vec{v'}) \odot (\vec{u''} +_g \vec{v''})$. By inductive hypothesis, $\vec{u'} +_g \vec{v'} \in \mathcal{L}(e_1)$ and $\vec{u''} +_g \vec{v''} \in \mathcal{L}(e_2)$, thus $\vec{w} \in \mathcal{L}(e_1 \cdot e_2) = \mathcal{L}(e)$.

If $e = e_1 + e_2$, then we have that $\vec{w} \in \mathcal{L}((e_1 + e_2) + (e_1 + e_2))$. By commutativity and associativity of the shuffle operation, it holds that $\mathcal{L}((e_1 + e_2) + (e_1 + e_2)) = \mathcal{L}((e_1 + e_1) + (e_2 + e_2))$, which means that there are word sequences $\vec{w'}, \vec{w''} \in \mathcal{L}(e_1 + e_1) \cup \mathcal{L}(e_2 + e_2)$ such that $\vec{w}$ is a shuffle of $\vec{w'}$ and $\vec{w''}$. In turn, $\vec{w'}$ is a shuffle of two word sequences $\vec{u'}$ and $\vec{v'}$, both belonging to $\mathcal{L}(e_1)$ or both belonging to $\mathcal{L}(e_2)$; similarly, $\vec{w''}$ is a shuffle of two word sequences $\vec{u''}$ and $\vec{v''}$, both belonging to $\mathcal{L}(e_1)$ or both belonging to $\mathcal{L}(e_2)$. By inductive hypothesis, $\vec{w'}, \vec{w''} \in \mathcal{L}(e_1) \cup \mathcal{L}(e_2)$, and, since $\vec{w}$ is a shuffle of $\vec{w'}$ and $\vec{w''}$, we can conclude that $\vec{w} \in \mathcal{L}(e_1 + e_2) = \mathcal{L}(e)$.

If $e = (e_1)^{op}$, with $op \in \{*, B, S, T\}$, then there are word sequences $\vec{t}, \vec{z} \in \mathcal{L}(e_1)$ and functions $f, f' \in \mathcal{F}$, such that $\vec{u}$ is the $f$-aggregation of $\vec{t}$ and $\vec{v}$ is the $f'$-aggregation of $\vec{z}$. Moreover, if $op = B$ (resp., $S, T$), then $\delta_f$ and $\delta_{f'}$ are $B$-sequences (resp., $S$-, $T$-sequences).

It is possible to define a selection function $g'$ and a function $f'' \in \mathcal{F}$ such that $\vec{w}$ is the $f''$-aggregation of $\vec{t} +_{g'} \vec{z}$. Intuitively, $g'$ chooses elements from sequences $\vec{t}$ and $\vec{z}$ so to reflect the order, established by the selection function $g$, in which such elements appear in $\vec{w}$ (even though in $\vec{w}$ they appear *aggregated* according to $f$, for elements of $\vec{t}$, and $f'$, for elements of $\vec{z}$). In other words, $g'$ chooses elements from $\vec{t}$ and $\vec{z}$ so that the infinitary concatenation, into an infinite word, of all finite words of the resulting word sequence $\vec{t} +_{g'} \vec{z}$ is equal to the infinite word resulting from the infinitary concatenation of all finite words of the word sequence $\vec{w}$. Analogously, $f''$ aggregates elements of $\vec{t} +_{g'} \vec{z}$ so to reflect the concatenation produced by $f$ and $f'$, thus obtaining exactly $\vec{w}$, that is, $f''$ emulates $f$ (resp., $f'$) when aggregating consecutive elements of $\vec{t} +_{g'} \vec{z}$ belonging to $\vec{t}$ (resp., $\vec{z}$).

Towards a formal definition of $g'$ and $f''$, recall that $\texttt{1's-upto}(g, i)$ (resp., $\texttt{2's-upto}(g, i)$) determines the position of the word in $\vec{u}$ (resp., $\vec{v}$) that appears in position $i$ of sequence $\vec{w}$. First, $g' : \mathbb{N}_{>0} \to 1, 2$ is the function corresponding to the sequence $s$ over $\{1, 2\}$ built as follows. Start with the empty sequence $s_0$, and, for every $i \in \mathbb{N}_{>0}$, sequence $s_i$ is obtained from $s_{i-1}$ by appending

- $\delta_f(\texttt{1's-upto}(g, i))$ many 1's, if $g(i) = 1$,
- $\delta_{f'}(\texttt{2's-upto}(g, i))$ many 2's, if $g(i) = 2$.

Next, $f''$ is defined as follows:

- $f''(0) = 1$,
- $f''(i) = \begin{cases} f''(i-1) + \delta_f(\texttt{1's-upto}(g, i)) & \text{if } g(i) = 1 \\ f''(i-1) + \delta_{f'}(\texttt{2's-upto}(g, i)) & \text{if } g(i) = 2 \end{cases}$, for all $i \in \mathbb{N}_{>0}$.

We show that if $\delta_{f'}$ are $B$-sequences (resp., $S$-, $T$-sequences), so is $\delta_{f''}$. Clearly, every value in $\delta_{f''}$ also appears in $\delta_f$ or $\delta_{f'}$. Therefore, if $\delta_f$ and $\delta_{f'}$ are $B$-sequences, so is $\delta_{f''}$. As a matter of fact, the above property can be generalized to suffixes of $\delta_f$, $\delta_{f'}$, and $\delta_{f''}$, as follows: for every $i \in \mathbb{N}_{>0}$ there is $j \in \mathbb{N}_{>0}$ such that every value in the suffix of $\delta_{f''}$ starting at position $j$ also appears in the suffix of $\delta_f$ starting at position $i$ or in the one of $\delta_{f'}$ starting at the same position. Therefore, if $\delta_f$ and $\delta_{f'}$ are $B$-sequences, so is $\delta_{f''}$. Moreover, observe that for at least one among $\delta_f$ and $\delta_{f'}$, let us call it $\hat{\delta}$, it holds that every value in $\hat{\delta}$ also appears in $\delta_{f''}$. Therefore, if $\hat{\delta}$ is a $T$-sequence so is $\delta_{f''}$.

Finally, since $\vec{w}$ is the $f''$-aggregation of the word sequence $\vec{t} +_{g'} \vec{z}$, which, by inductive hypothesis, belongs to $\mathcal{L}(e_1)$, we conclude that $\vec{w} \in \mathcal{L}(e)$. $\square$

**Corollary 1** (shuffle idempotence)**.** *$\mathcal{L}(e) = \mathcal{L}(e + e)$, for every BST-regular expression $e$.*

*Proof.* $\mathcal{L}(e) \subseteq \mathcal{L}(e+e)$ holds trivially, while $\mathcal{L}(e+e) \subseteq \mathcal{L}(e)$ follows immediately from Proposition 1. $\square$

**Proposition 2.** *It hold that $\mathcal{L}(e) = \mathcal{L}_\varepsilon(e)$, for every $BST$-regular expression $e$.*

*Proof.* Clearly, it holds that $\mathcal{L}(e) \subseteq \mathcal{L}_\varepsilon(e)$. To prove the converse inclusion, that is, $\mathcal{L}_\varepsilon(e) \subseteq \mathcal{L}(e)$, we proceed by induction on the size of $BST$-regular expressions.

If $e = \emptyset$ (resp., $e = a$), then $\mathcal{L}(e) = \emptyset = \mathcal{L}_\varepsilon(e)$ (resp., $\mathcal{L}(e) = \{(a, a, a, \ldots)\} = \mathcal{L}_\varepsilon(e)$), and the thesis follows trivially.

Let $\vec{u} \in \mathcal{L}(e)$ and $\vec{v}$ be an $\varepsilon$-pumping of $\vec{u}$. We show that $\vec{v} \in \mathcal{L}(e)$. If $\vec{u}$ does not feature infinitely many empty strings, then the unique $\varepsilon$-pumping of $\vec{u}$ is $\vec{u}$ itself, and the thesis trivially follows. Thus, let us assume that $\vec{u}$ features infinitely many empty strings, and let $g$ be the non-2-convergent selection function such that $\vec{v} = \vec{u} +_g \vec{\varepsilon}$.

If $e = e_1 \cdot e_2$, then there are word sequences $\vec{u'} \in \mathcal{L}(e_1)$ and $\vec{u''} \in \mathcal{L}(e_2)$ such that $\vec{u}$ is the concatenation of $\vec{u'}$ and $\vec{u''}$, i.e., $\vec{u} = \vec{u'} \odot \vec{u''}$. Since $\vec{u}$ features infinitely many empty strings, so do both $\vec{u'}$ and $\vec{u''}$. Therefore, $(\vec{u'} +_g \vec{\varepsilon})$ (resp., $(\vec{u''} +_g \vec{\varepsilon})$) is an $\varepsilon$-pumping of $\vec{u'}$ (resp., $\vec{u''}$), which means that $(\vec{u'} +_g \vec{\varepsilon}) \in \mathcal{L}_\varepsilon(e_1)$ and $(\vec{u''} +_g \vec{\varepsilon}) \in \mathcal{L}_\varepsilon(e_2)$. By inductive hypothesis, we have that $(\vec{u'} +_g \vec{\varepsilon}) \in \mathcal{L}_\varepsilon(e_1) = \mathcal{L}(e_1)$ and $(\vec{u''} +_g \vec{\varepsilon}) \in \mathcal{L}_\varepsilon(e_2) = \mathcal{L}(e_2)$. It is not difficult to see that $\vec{u} +_g \vec{\varepsilon} = (\vec{u'} +_g \vec{\varepsilon}) \odot (\vec{u''} +_g \vec{\varepsilon})$, hence $\vec{v} = \vec{u} +_g \vec{\varepsilon} \in \mathcal{L}(e)$.

If $e = e_1 + e_2$, then there are word sequences $\vec{u'}, \vec{u''} \in \mathcal{L}(e_1) \cup \mathcal{L}(e_2)$ such that $\vec{u}$ is a shuffle of $\vec{u'}$ and $\vec{u''}$, i.e., $\vec{u} = \vec{u'} +_{g'} \vec{u''}$ for a selection function $g'$. Since $\vec{u}$ features infinitely many empty strings, so does at least one among $\vec{u'}$ and $\vec{u''}$. Assume, without loss of generality, that $\varepsilon$ occurs infinitely often in $\vec{u'}$ and that $\vec{u'} \in \mathcal{L}(e_1)$. By commutativity and associativity of the shuffle operation (see Section 3), there are two selection functions $g''$ and $g'''$ such that $(\vec{u'} +_{g'} \vec{u''}) +_g \vec{\varepsilon} = (\vec{u'} +_{g''} \vec{\varepsilon}) +_{g'''} \vec{u''}$. It is also not difficult to convince oneself that function $g''$ can be defined so to be non-2-convergent. Therefore, since $\vec{u'}$ features infinitely many empty strings, $\vec{u'} +_{g''} \vec{\varepsilon} \in \mathcal{L}_\varepsilon(e_1)$. By inductive hypothesis, we have that $\vec{u'} +_{g''} \vec{\varepsilon} \in \mathcal{L}_\varepsilon(e_1) = \mathcal{L}(e_1) \subseteq \mathcal{L}(e_1) \cup \mathcal{L}(e_2)$, which implies $(\vec{u'} +_{g''} \vec{\varepsilon}) +_{g'''} \vec{u''} \in \mathcal{L}(e)$. The thesis follows from the observation that $\vec{v} = \vec{u} +_g \vec{\varepsilon} = (\vec{u'} +_{g'} \vec{u''}) +_g \vec{\varepsilon}$.

If $e = (e_1)^{op}$, with $op \in \{*, B, T\}$, then $\vec{u}$ is the $f$-aggregation of $\vec{u'}$, for a sequence $\vec{u'} \in \mathcal{L}(e_1)$ and a function $f \in \mathcal{F}$, with $\delta_f$ being a $B$-sequence (resp., $T$-sequence) if $op = B$ (resp., $op = T$). It is not difficult to devise a function $f' \in \mathcal{F}$ such that $\vec{u} +_g \vec{\varepsilon}$ is the $f'$-aggregation of $\vec{u'}$. Intuitively, $f'$ creates new empty strings via *vacuous aggregations*, that is, aggregating together 0 words from sequence $\vec{u'}$ into empty strings. This results in a sequence $\delta_{f'}$ that can be obtained from $\delta_f$ by inserting 0's in correspondence of the empty strings added to $\vec{u}$ by $g$ to obtain $\vec{v}$ (via the operation $\vec{u} +_g \vec{\varepsilon}$), which means that, if $\delta_f$ is a $B$-sequence (resp., $T$-sequence), so is $\delta_{f'}$. Therefore, we have that $\vec{v} = \vec{u} +_g \vec{\varepsilon}$ is the $f'$-aggregation of $\vec{u'}$, hence $\vec{v} \in \mathcal{L}(e)$.

If $e = (e_1)^S$, then $\vec{u}$ is the $f$-aggregation of $\vec{u'}$, for a sequence $\vec{u'} \in \mathcal{L}(e_1)$ and a function $f \in \mathcal{F}$, with $\delta_f$ being an $S$-sequence. Since $\vec{u}$ features infinitely many empty strings, so does $\vec{u'}$, or $\delta_f$ would contain infinitely many 0's, which is in contradiction with it being an $S$-sequence. It is not difficult to devise a non-2-convergent selection function $g'$ such that $\vec{u'} +_{g'} \vec{\varepsilon}$ contains finite sub-sequences of empty strings of increasing lengths at positions corresponding to empty strings added to $\vec{u}$ by $g$ to obtain $\vec{v}$ (via the operation $\vec{u} +_g \vec{\varepsilon}$). In other words, $g'$ creates in $\vec{u'} +_{g'} \vec{\varepsilon}$ a finite sub-sequence of consecutive $\varepsilon$'s in correspondence of each of the empty string added in $\vec{v}$ by $g$, and such sub-sequences have increasing lengths. Then, there is a function $f'$ such that $\vec{u} +_g \vec{\varepsilon}$ is the $f'$-aggregation of $\vec{u'} +_{g'} \vec{\varepsilon}$. Intuitively, $f'$ mimics $f$ when aggregating words of $\vec{u'}$ to form words occurring in $\vec{u}$, while it aggregates into empty strings the sub-sequences of consecutive $\varepsilon$'s created by $g'$ via the operation $\vec{u'} +_{g'} \vec{\varepsilon}$. Since such sequences have increasing lengths, $\delta_{f'}$ preserves the property of being an $S$-sequence. Moreover, $\vec{u'} +_{g'} \vec{\varepsilon} \in \mathcal{L}_\varepsilon(e_1)$, because $\vec{u'}$ features infinitely many empty strings. By inductive hypothesis, $\vec{u'} +_{g'} \vec{\varepsilon} \in \mathcal{L}_\varepsilon(e_1) = \mathcal{L}(e_1)$, and thus $\vec{v} = \vec{u} +_g \vec{\varepsilon} \in \mathcal{L}(e)$. $\square$

**Corollary 2.** *Let $e$ be a BST-regular expression. If $\vec{u}$ is the $f$-aggregation of $\vec{v}$, for a function $f \in \mathcal{F}$ and a word sequence $\vec{v} \in \mathcal{L}(e)$ featuring infinitely many empty strings, then $\vec{u} \in \mathcal{L}(e^S)$. If, in addition, there is at least one value occurring infinitely often in $\delta_f$, then $\vec{u} \in \mathcal{L}(e^T)$ as well.*

*Proof.* To begin with, observe that, since $\vec{v}$ features infinitely many empty strings, the sequence obtained injecting sequences of empty strings of increasing lengths after every word in $\vec{v}$ is an $\varepsilon$-pumping of $\vec{v}$. More formally, the sequence $\vec{v'} = (v_1, \varepsilon, v_2, \varepsilon, \varepsilon, v_3, \varepsilon, \varepsilon, \varepsilon, \ldots)$ is an $\varepsilon$-pumping of $\vec{v}$. It is not difficult to see that there is a function $f'$, with $\delta_{f'}$ being an $S$-sequence, such that the $f'$-aggregation of $\vec{v'}$ coincides with the $f$-aggregation of $\vec{v}$; therefore, $\vec{u}$ is the $f'$-aggregation of $\vec{v'}$. Clearly, $\vec{v'} \in \mathcal{L}_\varepsilon(e)$, and, by Proposition 2, $\vec{v'} \in \mathcal{L}(e)$, hence $\vec{u} \in \mathcal{L}(e^S)$.

In order to conclude the proof, note that the existence of a value $k$ occurring infinitely often in $\delta_f$ means that $\vec{v}$ contains infinitely many sub-sequences of $k$ many consecutive words that $f$ aggregates together into a word in $\vec{u}$. Let $\vec{v^i} = (v^1_1, v^1_2, \ldots, v^1_k)$, for $i \in \mathbb{N}_{>0}$, be all such sub-sequences of $\vec{v}$. Further, let $\vec{\varepsilon^i}$ be the finite sequence featuring $i$ many empty strings, for all $i \in \mathbb{N}_{>0}$. Finally, consider the sequence $\vec{v'}$ obtained by injecting sequences $\vec{\varepsilon^1}$, $\vec{\varepsilon^1}$, $\vec{\varepsilon^2}$, $\vec{\varepsilon^1}$, $\vec{\varepsilon^2}$, $\vec{\varepsilon^3}$, $\vec{\varepsilon^1}$, $\vec{\varepsilon^2}$, $\vec{\varepsilon^3}$, , $\vec{\varepsilon^4}$, $\ldots$, immediately after, respectively, $\vec{v^1}$, $\vec{v^2}$, $\vec{v^3}$, $\vec{v^4}$, $\vec{v^5}$, $\vec{v^6}$, $\vec{v^7}$, $\vec{v^8}$, $\vec{v^9}$, $\vec{v^{10}}$, $\ldots$. Since $\vec{v}$ features infinitely many empty strings, we have that $\vec{v'}$ is an $\varepsilon$-pumping of $\vec{v}$, meaning that $\vec{v'} \in \mathcal{L}_\varepsilon(e)$. It is not difficult, now, to see that there is a function $f'$, with $\delta_{f'}$ being a $T$-sequence, such that the $f'$-aggregation of $\vec{v'}$ coincides with the $f$-aggregation of $\vec{v}$; therefore, $\vec{u}$ is the $f'$-aggregation of $\vec{v'}$. Intuitively, $f'$ aggregates the newly added sequences of empty strings together with the corresponding sub-sequences $\vec{v^i}$, and thus $\delta_{f'}$ features infinitely many occurrences of $k+i$, for every $i \in \mathbb{N}_{>0}$. By Proposition 2, $\vec{v'} \in \mathcal{L}_\varepsilon(e)$ implies $\vec{v'} \in \mathcal{L}(e)$, hence $\vec{u} \in \mathcal{L}(e^T)$. $\square$

## Appendix B. Soundness of the encoding of regular expressions

Thanks to Lemma 1(a), proving Theorem 2 amounts to establishing the following correspondence between interval models and $R$ parse trees for finite words, with $R$ being a regular expression (Lemmas 3 and 4 below).

**Lemma 3.** *Let $R$ be a regular expression over $\Sigma$ and $w \in \Sigma^*$ be a finite word. If there exists an $R$ parse tree for $w$, then there is an interval model $M = \langle \mathbb{I}(N), A, B, V \rangle$ such that $w \approx M$ and $M, [0, N-1] \models \varphi_R \wedge \varphi_\Sigma$.*

*Proof.* First of all, observe that if $R = \emptyset$, then no $R$ parse tree for $w$ exists, and the claim is vacuously true.

Then, assume $R \neq \emptyset$, and let $w = w_1 w_2 \ldots w_{|w|} \in \Sigma^*$ and $\tau_w^R = (Nodes, Edges, e\text{-}idx, s, f)$ be an $R$ parse tree for $w$. We define model $M = \langle \mathbb{I}(N), A, B, V \rangle$ and we show that $w \approx M$ and $M, [0, N-1] \models \varphi_R \wedge \varphi_\Sigma$. First, we set $N = |w| + 1 = f(r)$, where $r$ is the root of $\tau_w^R$. Recall that, if $N < \omega$, then $\mathbb{I}(N) = \{[x, y] \mid x, y \in \mathbb{N} \text{ and } x \leq y < N\}$.

For every $[x, y] \in \mathbb{I}(N)$, let $\mathit{expr\text{-}propositions}_{[x,y]} = \{\mathit{expr}_{e\text{-}idx(n)} \mid n \in Nodes \text{ and } [x, y] = [s(n) - 1, f(n) - 1]\}$; intuitively, it is meant to collect all propositions $\mathit{expr}_i$, for $e_i \in sub(R)$, that hold true in $[x, y]$. The valuation function $V$ of the model $M = \langle \mathbb{I}(N), A, B, V \rangle$ can be defined as follows. For every $[x, y] \in \mathbb{I}(N)$,

$$V([x,y]) = \begin{cases} \mathit{expr\text{-}propositions}_{[x,y]} & \text{if } y - x > 1 \\ \mathit{expr\text{-}propositions}_{[x,y]} \cup \{w_y\} & \text{if } y - x = 1 \\ \mathit{expr\text{-}propositions}_{[x,y]} \cup \left\{\mathit{expr}^{end}_{e\text{-}idx(n)} \mid n \in Nodes, y = f(n) - 1\right\} & \\ & \text{if } x = y \end{cases}$$

As a general observation, notice that, since the labeling (of intervals with proposition letters) imposed by $V$ preserves the tree structure of $\tau_w^R$ and since it is never the case that two nodes $n, n'$ of the same type (i.e., $e\text{-}idx(n) = e\text{-}idx(n')$) are one the ancestor of the other, we have that $\mathit{expr}_i$ intervals are pairwise disjoint, for every $e_i \in sub(R)$.

By definition of $V$, it immediately follows that $M \approx w$, which, in turn, implies $M, [0, |w|] \models \varphi_\Sigma$ (see definition of $\varphi_\Sigma$ at page 21). To conclude the proof, we still need to show that $M, [0, |w|] \models \varphi_R$. To this end, we show that $M, [0, |w|]$ makes true each conjunct of formula

$$\varphi^R = \mathit{expr}_n \ \wedge \ [A]\pi \ \wedge \bigwedge_{e_i \in \mathit{sub}(R)} \varphi_{\mathit{expr}_i} \ \wedge \bigwedge_{e_i \in \mathit{sub}(R)} \varphi^{\mathit{end}}_{\mathit{expr}_i} \ \wedge \bigwedge_{e_i \in \mathit{sub}(R)} \varphi^{\not\cap}_{\mathit{expr}_i}.$$

We begin with the simplest cases. It clearly holds that $M, [0, |w|] \models [A]\pi$ since $[0, |w|]$ is the maximal interval and then its only adjacent-to-the-right interval in $\mathbb{I}(N)$ is the point $[|w|, |w|]$. Let us prove now that $M, [0, |w|] \models \mathit{expr}_n$, that is, $\mathit{expr}_n \in V([0, |w|])$. Since $\tau^R_w$ is an $R$ parse tree for $w$, then for the root $r$ of $\tau^R_w$ it holds that $e\text{-}\mathit{idx}(r) = n$, $s(r) = 1$, and $f(r) = |w| + 1$. Thus, by definition of $\mathit{expr}\text{-}\mathit{propositions}_{[x,y]}$, it follows that $\mathit{expr}_n \in V([0, |w|])$. Instead, the fact that $M, [0, |w|] \models \bigwedge_{e_i \in \mathit{sub}(R)} \varphi^{\not\cap}_{\mathit{expr}_i}$ immediately follows from the fact that $\mathit{expr}_i$ intervals are pairwise disjoint, for every $e_i \in \mathit{sub}(R)$.

Let us now prove that $M, [0, |w|] \models \varphi^{\mathit{end}}_{\mathit{expr}_i}$ for every $e_i \in \mathit{sub}(R)$ (see definition of $\varphi^{\mathit{end}}_{\mathit{expr}_i}$ at page 22). To this end, let us consider a generic index $i$ associated with a sub-expression $e_i \in \mathit{sub}(R)$. From the definition of $V$ (case $x = y$), it follows that $\mathit{expr}^{\mathit{end}}_i$ holds exactly on points where an $\mathit{expr}_i$ interval ends. Moreover, since $\mathit{expr}_i$ intervals do not intersect each other, it is easy to see that $M, [0, |w|] \models \varphi^{\mathit{end}}_{\mathit{expr}_i}$.

Finally, let us prove that $M, [0, |w|] \models \varphi_{\mathit{expr}_i}$ for every $e_i \in \mathit{sub}(R)$. Let $i$ be a generic index associated with a sub-expression $e_i \in \mathit{sub}(R)$. Recall that an interval $[x, y]$ satisfies a proposition letter $\mathit{expr}_i$ if and only if there is a node $n$ with $e\text{-}\mathit{idx}(n) = i$, $x = s(n) - 1$ and $y = f(n) - 1$ (by definition of $\mathit{expr}\text{-}\mathit{propositions}_{[x,y]}$). We proceed case by case.

- If $e_i = a$, for some $a \in \Sigma$, then we have $\varphi_{\mathit{expr}_i} = [G](\mathit{expr}_i \ \rightarrow \ a)$. Let $[x, y]$ be an $\mathit{expr}_i$ interval and $n$ be such that $e\text{-}\mathit{idx}(n) = i$, $x = s(n) - 1$ and $y = f(n) - 1$. We show that $a$ holds true in $[x, y]$ as well. By definition of parse tree, it holds that $s(n) + 1 = f(n)$ and that $w_{s(n)} = a$. Therefore, we have $x = y - 1$ and $y = s(n)$. By definition of $V$ (case $y - x = 1$), we have that $a = w_{s(n)} = w_y \in V([x, y])$.

- If $e_i = \varepsilon$, then we have $\varphi_{expr_i} = [G](expr_i \rightarrow \pi)$. Let $[x, y]$ be an $expr_i$ interval and $n$ be such that $e\text{-}idx(n) = i$, $x = s(n) - 1$ and $y = f(n) - 1$. By definition of parse tree, it holds $s(n) = f(n)$, which implies $x = y$. Therefore, $\pi$ holds true in $[x, y]$ as well.

- If $e_i = e_j + e_k$, then we have $\varphi_{expr_i} = [G](expr_i \leftrightarrow (expr_j \vee expr_k))$. Since nodes $n$ with $e\text{-}idx(n) \in \{j, k\}$ only appear in $\tau_w^R$ as children of nodes $n'$ with $e\text{-}idx(n') = i$, we have that every $expr_j$ (resp., $expr_k$) interval is an $expr_i$ interval as well, thus proving the right-to-left direction of the equivalence. Now, let $[x, y]$ be an $expr_i$ interval and let $n$ be such that $e\text{-}idx(n) = i$, $x = s(n) - 1$ and $y = f(n) - 1$. By definition of parse tree, $n$ has exactly one child $n'$ such that $e\text{-}idx(n') \in \{j, k\}$, and $(s(n), f(n)) = (s(n'), f(n'))$. It immediately follows from the definition of $V$ that either $expr_j$ or $expr_k$ holds true in $[x, y]$.

- If $e_i = e_j e_k$, then we have

$$
\begin{aligned}
\varphi_{expr_i} = \ & [G](expr_j \rightarrow \langle A\rangle expr_k) \\
\wedge \ & [G](expr_k \rightarrow (expr_j^{end} \vee \langle B\rangle expr_j^{end}) \wedge \langle A\rangle expr_i^{end}) \\
\wedge \ & [G]((expr_j \vee expr_k) \rightarrow [B](\neg\pi \rightarrow [A]\neg expr_i^{end})) \\
\wedge \ & [G]((expr_j \wedge expr_k) \rightarrow \pi \wedge expr_i) \\
\wedge \ & [G](expr_i \rightarrow (\langle B\rangle(\pi \wedge expr_j) \wedge expr_k)) \\
& \qquad\qquad \vee \langle B\rangle(\neg\pi \wedge expr_j) \\
& \qquad\qquad \vee (expr_j \wedge \langle A\rangle(\pi \wedge expr_k)) \\
\wedge \ & [G]( \quad (\langle A\rangle expr_j \rightarrow \langle A\rangle expr_i) \\
& \qquad \wedge (\langle A\rangle(\neg\pi \wedge expr_j) \rightarrow \langle A\rangle(\neg\pi \wedge expr_i)) \quad ) \\
\wedge \ & [G](expr_k \wedge \langle B\rangle expr_i^{end} \rightarrow expr_i).
\end{aligned}
$$

  By definition of parse tree, we have that every node $n$ with $e\text{-}idx(n) = i$ has two children $n', n''$ with $e\text{-}idx(n') = j$ and $e\text{-}idx(n'') = k$, and such that $s(n) = s(n')$, $f(n') = s(n'')$, and $f(n'') = f(n)$; moreover, no other node $n'''$ exists with $e\text{-}idx(n''') \in \{j, k\}$. To satisfy the first conjunct of $\varphi_{expr_i}$ it is enough to observe that every $expr_j$ interval is immediately followed by an $expr_k$ interval (by the definition of $V$ and the one of parse tree). The satisfaction of the second conjunct follows from the facts that every $expr_k$ interval is immediately preceded by an $expr_j$ interval (by the definition of $V$ and the one of parse tree) and that every $expr_k$ interval ends where an $expr_i$ interval ends (due to $f(n'') = f(n)$). The third conjunct holds because no $expr_i$ interval ends inside an $expr_j$ or an $expr_k$ interval (as it is never the case that a node $n'$, with $e\text{-}idx(n') \in \{j, k\}$, is an ancestor of a node $n$, with $e\text{-}idx(n) = i$). As for the fourth conjunct, if an interval $[x, y]$ satisfies both $expr_j$ and $expr_k$ then there must be two children $n', n''$ of a node $n$, with $e\text{-}idx(n) = i$, $e\text{-}idx(n') = j$, and $e\text{-}idx(n'') = k$, such that $s(n') = s(n'')$ and $f(n') = f(n'')$. Therefore, it holds that $s(n) = s(n') = s(n'') = f(n') = f(n'') = f(n)$, which means that $expr_i$ is true on $[x, y]$ and that $x = y$, and thus $\pi$ holds true over $[x, y]$ as well. To see that the fifth conjunct is satisfied, let $[x, y]$ be an $expr_i$ interval and $n, n', n''$ be three nodes such that $e\text{-}idx(n) = i$, $e\text{-}idx(n') = j$, $e\text{-}idx(n'') = k$, $s(n) = s(n')$, $f(n') = s(n'')$, and $f(n'') = f(n)$. By the definition of $V$ and the one of parse tree, there is a point $z$, with $x \leq z \leq y$, such that $[x, z]$ is an $expr_j$ interval and $[z, y]$ is an $expr_k$ interval. It is easy to see that if $x = z$ (resp., $x < z < y$, $z = y$), then $\langle B \rangle(\pi \wedge expr_j) \wedge expr_k$ (resp., $\langle B \rangle(\neg\pi \wedge expr_j)$, $expr_j \wedge \langle A \rangle(\pi \wedge expr_k)$) is true on $[x, y]$, and thus the implication holds true in $[x, y]$ as well. To verify that the sixth conjunct is true, recall that nodes $n'$ with $e\text{-}idx(n') = j$ can only occur in $\tau_w^R$ as left children of nodes $n$ with $e\text{-}idx(n) = i$; then, it immediately follows that $expr_j$ intervals can only occur in $M$ as (not necessarily strict) prefixes of $expr_i$ intervals. Finally, to check that the seventh conjunct is satisfied, let $[x, y]$ be an $expr_k$ interval, with $x < y$ and $x$ being an $expr_i^{end}$ point. Recall that nodes $n''$ with $e\text{-}idx(n'') = k$ can only occur in $\tau_w^R$ as right children of nodes $n$ with $e\text{-}idx(n) = i$; then, it immediately follows that $expr_k$ intervals can only occur in $M$ as (not necessarily strict) suffixes of $expr_i$ intervals. Thus, $[z, y]$ is an $expr_i$ interval, for some $z \leq x$; however, since $x$ is an $expr_i^{end}$ point, it cannot be $z < x$ (or, there would be an $expr_i^{end}$ point inside an $expr_i$ interval, and thus two $expr_i$ intervals that intersect). Thus, it must be $z = x$, meaning that $[x, y]$ is an $expr_i$ interval.

- if $e_i = e_j^*$ we have:

$$\begin{aligned}
\varphi_{expr_i} = {} & [G](expr_i \rightarrow \pi \ \vee \ expr_j \ \vee \ (\langle B\rangle expr_j \ \wedge \\
& \qquad [B](\langle A\rangle expr_j^{end} \ \rightarrow \ \langle A\rangle(\neg\pi \ \wedge \ expr_j)))) \\
& \wedge [G](expr_j \ \rightarrow \ [B](\neg\pi \ \rightarrow \ [A]\neg expr_i^{end})) \\
& \wedge [init](\langle A\rangle expr_j \wedge \neg\langle A\rangle expr_i \ \rightarrow \ \langle B\rangle\langle A\rangle(\neg\pi \ \wedge \ expr_i)) \\
& \wedge [G](\langle A\rangle expr_j \ \wedge \ \langle B\rangle(\neg\pi \ \wedge \ expr_i) \ \rightarrow \\
& \qquad \langle A\rangle expr_i \vee \langle B\rangle(\neg\pi \ \wedge \ \langle A\rangle(\neg\pi \ \wedge \ expr_i))) \\
& \wedge [G](expr_i \ \wedge \ \langle A\rangle(\neg\pi \ \wedge \ expr_j) \ \rightarrow \ \langle A\rangle expr_i) \\
& \wedge [G](\langle A\rangle(\neg\pi \ \wedge \ expr_j) \ \wedge \ \langle A\rangle expr_i \ \rightarrow \ \langle A\rangle(\neg\pi \ \wedge \ expr_i)).
\end{aligned}$$

By definition of parse tree, for every node $n$ with $e\text{-}idx(n) = i$, it holds that either $n$ is a leaf and $s(n) = f(n)$ or $f(n) < \omega$ and $n$ has $h$ children $n^1, \ldots, n^h$, with $h \in \mathbb{N}_{>0}$, such that $e\text{-}idx(n^1) = \ldots = e\text{-}idx(n^h) = j$, $f(n^k) = s(n^{k+1})$, for all $k \in \{1, \ldots, h-1\}$, and $(s(n), f(n)) = (s(n^1), f(n^h))$. Clearly, the first conjunct holds true, with the three disjuncts of the right-hand side of the implication corresponding to a node $n$, with $e\text{-}idx(n) = i$, having zero, one, or more than one children. The second conjunct is satisfied because no $expr_i$ interval ends inside an $expr_j$ interval. As for the third conjunct, every node $n$, with $e\text{-}idx(n) = j$, is a child of a node $n'$ with $e\text{-}idx(n') = i$, that means that for every $expr_j$ interval $[x, y]$ there is an $expr_i$ interval $[w, z]$, with $w \leq x \leq y \leq z$, and thus the third conjunct holds true. In order to verify that the fourth conjunct is true, consider an interval $[x, y]$ on which it holds $\langle A\rangle expr_j \ \wedge \ \langle B\rangle(\neg\pi \ \wedge \ expr_i)$. Thus, there are two nodes $n, n'$, with $e\text{-}idx(n) = i$, $e\text{-}idx(n') = j$, $s(n) - 1 = x < f(n) - 1 < s(n') - 1 = y$. By definition of parse tree, node $n'$ is a child of a node $n''$, with $e\text{-}idx(n'') = i$. Thus, we have $s(n'') \leq s(n') \leq f(n') \leq f(n'')$. Since $expr_i$ intervals do not intersect, it must be $f(n) \leq s(n'')$. If $s(n'') = s(n')$, then $\langle A \rangle expr_i$ holds; if $s(n'') < s(n')$, then we have that $x = s(n) - 1 < f(n) - 1 \leq s(n'') - 1 < s(n') - 1 = y \leq f(n'') - 1$, and $\langle B \rangle(\neg\pi \ \wedge \ \langle A \rangle(\neg\pi \ \wedge \ expr_i))$ holds over $[x, y]$. To check that the fifth conjunct is satisfied, it is enough to observe that, if $[w, x]$ is an $expr_i$ interval and an $expr_j$ interval $[x, y]$, with $x < y$, starts at $x$, then there are nodes $n, n'$, with $e\text{-}idx(n) = i$, $e\text{-}idx(n') = j$, and $f(n) - 1 = s(n') - 1 = x < y = f(n') - 1$. If $[w, x]$ is a point interval, then $\langle A \rangle expr_i$ holds on $[w, z]$, and the fifth conjunct is verified. Thus, we assume $s(n) < f(n)$. By definition of parse tree, node $n'$ is a child of a node $n''$, with $e\text{-}idx(n'') = i$ and $s(n'') \leq s(n') < f(n') \leq f(n'')$. Since $f(n) < f(n'')$ holds and $expr_i$ intervals do not end one inside the other, it cannot be $s(n'') < f(n)$, which implies $s(n') = f(n) \leq s(n'') \leq s(n')$. Thus, it holds that $s(n') = s(n'')$; consequently, $\langle A \rangle expr_i$ holds true in every interval ending in $x = s(n') - 1$. Finally, the sixth conjunct follows from the fact that a node $n'$, with $e\text{-}idx(n') = j$, is a child of a node $n$, with $e\text{-}idx(n) = i$, and the fact that $expr_i$ intervals do not overlap. $\square$

**Lemma 4.** *Let $R$ be a regular expression over $\Sigma$ and $w \in \Sigma^*$ be a finite word. If there is an interval model $M = \langle \mathbb{I}(N), A, B, V \rangle$ such that $w \approx M$ and $M, [0, N-1] \models \varphi_R \ \wedge \ \varphi_\Sigma$, then there exists an $R$ parse tree for $w$.*

*Proof.* Recall that $M, [0, N-1] \models \varphi_R$ implies $M, [0, N-1] \models expr_n \wedge \varphi_{expr_i}$, for every $e_i \in sub(R)$, and that $R = e_n \in sub(R)$. To begin with, observe that if $R = \emptyset$, then $\varphi_{expr_n} = [G](expr_n \ \rightarrow \ \bot)$, and thus no interval model $M$ exists such that $M, [0, N-1] \models \varphi_R \ \wedge \ \varphi_\Sigma$, and the claim is vacuously true.

Therefore, assume $R \neq \emptyset$. We build, from $M$, an $R$ parse tree for $w$. To this end, we first show a more general property: for every $[x, y] \in \mathbb{I}(N)$ and every $e_i \in sub(R)$, with $expr_i \in V([x, y])$, there exists an $e_i$ parse tree for $w[x+1, y+1)$. Then, given that $R = e_n$, the claim follows from the facts that $N = |w| + 1$, by definition of $w \approx M$, and $expr_n \in V([0, N-1])$, since $expr_n$

appears as a conjunct in $\varphi_R$. The proof is by structural induction on $e_i$.

- If $e_i = a$, for some $a \in \Sigma$, then $\varphi_{expr_i} = [G](expr_i \rightarrow a)$. By $expr_i \in V([x,y])$ and $M,[0,N-1] \models \varphi_{expr_i}$, we have that $a \in V([x,y])$. By $M,[0,N-1] \models \varphi_\Sigma$, it holds that $y = x+1$ and $a' \notin V([x,y])$, for any $a' \in \Sigma \setminus \{a\}$. Moreover, by definition of $w \approx M$, we have that $w_{x+1} = a$. Let $\tau = (\{r\}, \emptyset, e\text{-}idx, s, f)$, where $e\text{-}idx(r) = i$, $s(r) = x+1$, and $f(r) = s(r)+1 = y+1$. Clearly, $\tau$ is an $e_i$ parse tree for $w[s(r), f(r)) = w[x+1, y+1) = a$.

- If $e_i = \varepsilon$, then $\varphi_{expr_i} = [G](expr_i \rightarrow \pi)$. By $expr_i \in V([x,y])$ and $M,[0,N-1] \models \varphi_{expr_i}$, we have that $x = y$. Let $\tau = (\{r\}, \emptyset, e\text{-}idx, s, f)$, where $e\text{-}idx(r) = i$, and $s(r) = f(r) = x+1$. Clearly, $\tau$ is an $e_i$ parse tree for $w[s(r), f(r)) = w[x+1, y+1) = \varepsilon$.

- If $e_i = e_j + e_k$, then $\varphi_{expr_i} = [G](expr_i \leftrightarrow (expr_j \vee expr_k))$. By $expr_i \in V([x,y])$ and $M,[0,N-1] \models \varphi_{expr_i}$, we have that $\{expr_j, expr_k\} \cap V([x,y]) \neq \emptyset$. Let us assume, without loss of generality, that $expr_j \in V([x,y])$. By inductive hypothesis, there exists an $e_j$ parse tree $\tau' = (N', E', e\text{-}idx', s', f')$ for $w[x+1, y+1)$. Let $r'$ be the root of $\tau'$ and $r$ be a fresh node, i.e., $r \notin N'$. We define $\tau = (N' \cup \{r\}, E' \cup \{(r, r')\}, e\text{-}idx, s, f)$, where $e\text{-}idx$, $s$, and $f$ extend $e\text{-}idx'$, $s'$, and $f'$, respectively, to the new set of nodes $N$ as follows: $e\text{-}idx(r) = i$, $s(r) = s(r')$, and $f(r) = f(r')$. Clearly, $\tau$ is an $e_i$ parse tree for $w[s(r), f(r)) = w[s(r'), f(r')) = w[x+1, y+1)$.

- If $e_i = e_j e_k$, then

$$
\begin{aligned}
\varphi_{expr_i} = \; & [G](expr_j \rightarrow \langle A\rangle expr_k) \\
\wedge \; & [G](expr_k \rightarrow (expr_j^{end} \vee \langle B\rangle expr_j^{end}) \wedge \langle A\rangle expr_i^{end}) \\
\wedge \; & [G]((expr_j \vee expr_k) \rightarrow [B](\neg\pi \rightarrow [A]\neg expr_i^{end})) \\
\wedge \; & [G]((expr_j \wedge expr_k) \rightarrow \pi \wedge expr_i) \\
\wedge \; & [G](expr_i \rightarrow (\langle B\rangle(\pi \wedge expr_j) \wedge expr_k)) \\
& \qquad\qquad \vee \langle B\rangle(\neg\pi \wedge expr_j) \\
& \qquad\qquad \vee (expr_j \wedge \langle A\rangle(\pi \wedge expr_k)) \\
\wedge \; & [G]( \quad (\langle A\rangle expr_j \rightarrow \langle A\rangle expr_i) \\
& \qquad \wedge (\langle A\rangle(\neg\pi \wedge expr_j) \rightarrow \langle A\rangle(\neg\pi \wedge expr_i)) \quad ) \\
\wedge \; & [G](expr_k \wedge \langle B\rangle expr_i^{end} \rightarrow expr_i).
\end{aligned}
$$

By $expr_i \in V([x,y])$ and $M,[0,N-1] \models \varphi_{expr_i}$, the fifth conjunct of $\varphi_{expr_i}$ implies that there is a point $z$, with $x \leq z \leq y$, such that $expr_j \in V([x,z])$. If $x = z$ or $z = y$, the same conjunct also implies that $expr_k \in V([z,y])$. If $x < z < y$, then the first conjunct guarantees the existence of a point $z' \geq z$ such that $expr_k \in V([z,z'])$. We show that $z' = y$. On the one hand, the second conjunct forces $z'$ to coincide with a point where an $expr_i$ interval ends, and thus it cannot be $z' < y$, as $expr_i$ intervals do not end inside each other. On the other hand, it cannot be $z' > y$, or there would be an $expr_i$ interval ending inside an $expr_k$ one, which is prevented by the third conjunct. Thus, we have that $z$ partitions $[x,y]$ in two intervals $[x,z]$ and $[z,y]$, with $expr_j \in V([x,z])$ and $expr_k \in V([z,y])$. By inductive hypothesis, there are an $e_j$ parse tree $\tau' = (N', E', e\text{-}idx', s', f')$ for $w[x+1, z+1)$ and an $e_k$ parse tree $\tau'' = (N'', E'', e\text{-}idx'', s'', f'')$ for $w[z+1, y+1)$. Let $r'$ and $r''$ be the roots of $\tau'$ and $\tau''$, respectively, and let $r$ be a fresh node, i.e., $r \notin N' \cup N''$. We define $\tau = (N' \cup N'' \cup \{r\}, E' \cup E'' \cup \{(r,r'),(r,r'')\}, e\text{-}idx, s, f)$, where $e\text{-}idx$, $s$, and $f$ extend $e\text{-}idx' \cup e\text{-}idx''$, $s' \cup s''$, and $f' \cup f''$, respectively, to the new set of nodes $N$ as follows: $e\text{-}idx(r) = i$, $s(r) = s(r')$, and $f(r) = f(r'')$. Clearly, $\tau$ is an $e_i$ parse tree for $w[s(r), f(r)) = w[s(r'), f(r'')) = w[x+1, y+1)$. Note that we did not make use of some of the conjuncts of $\varphi_{expr_i}$. As a matter of fact, they are needed to guarantee that $expr_j$ and $expr_k$ intervals are, respectively, prefixes and suffixes of $expr_i$ intervals, which is useful for the encodings given in the next sections.

- If $e_i = e_j^*$, then

$$
\begin{aligned}
\varphi_{expr_i} = \ & [G](expr_i \rightarrow \pi \ \vee \ expr_j \ \vee \ (\langle B \rangle expr_j \wedge \\
& \qquad [B](\langle A \rangle expr_j^{end} \ \rightarrow \ \langle A \rangle(\neg\pi \ \wedge \ expr_j)))) \\
& \wedge [G](expr_j \ \rightarrow \ [B](\neg\pi \ \rightarrow \ [A]\neg expr_i^{end})) \\
& \wedge [init](\langle A \rangle expr_j \wedge \neg\langle A \rangle expr_i \ \rightarrow \ \langle B \rangle \langle A \rangle(\neg\pi \ \wedge \ expr_i)) \\
& \wedge [G](\langle A \rangle expr_j \ \wedge \ \langle B \rangle(\neg\pi \ \wedge \ expr_i) \ \rightarrow \\
& \qquad \langle A \rangle expr_i \vee \langle B \rangle(\neg\pi \ \wedge \ \langle A \rangle(\neg\pi \ \wedge \ expr_i))) \\
& \wedge [G](expr_i \ \wedge \ \langle A \rangle(\neg\pi \ \wedge \ expr_j) \ \rightarrow \ \langle A \rangle expr_i) \\
& \wedge [G](\langle A \rangle(\neg\pi \ \wedge \ expr_j) \ \wedge \ \langle A \rangle expr_i \ \rightarrow \ \langle A \rangle(\neg\pi \ \wedge \ expr_i)).
\end{aligned}
$$

By $expr_i \in V([x,y])$ and $M, [0, N-1] \models \varphi_{expr_i}$, the first and the second conjunct of $\varphi_{expr_i}$ imply three possibilities: $x = y$, $expr_j \in V([x,y])$, or there are finitely many points $z_0, z_1, \ldots, z_k$, for $k \geq 2$,with $x = z_0 < z_1 < \ldots < z_k = y$ and $expr_j \in V([z_{i-1}, z_i])$, for every $i \in \{1, \ldots, k\}$. If $x = y$, we define $\tau = (\{r\}, \emptyset, e\text{-}idx, s, f)$, where $e\text{-}idx(r) = i$, and $s(r) = f(r) = x+1$. Clearly, $\tau$ is an $e_i$ parse tree for $w[s(r), f(r)) = w[x+1, y+1) = \varepsilon$. If $expr_j \in V([x,y])$, then, by inductive hypothesis, there exists an $e_j$ parse tree $\tau' = (N', E', e\text{-}idx', s', f')$ for $w[x+1, y+1)$. Let $r'$ be the root of $\tau'$ and $r$ be a fresh node, i.e., $r \notin N'$. We define $\tau = (N' \cup \{r\}, E' \cup \{(r, r')\}, e\text{-}idx, s, f)$, where $e\text{-}idx$, $s$, and $f$ extend $e\text{-}idx'$, $s'$, and $f'$, respectively, to the new set of nodes $N$ as follows: $e\text{-}idx(r) = i$, $s(r) = s(r')$, and $f(r) = f(r')$. Clearly, $\tau$ is an $e_i$ parse tree for $w[s(r), f(r)) = w[s(r'), f(r')) = w[x+1, y+1)$. Finally, if there are finitely many points $z_0, z_1, \ldots, z_k$, for $k \geq 2$, with $x = z_0 < z_1 < \ldots < z_k = y$ and $expr_j \in V([z_{i-1}, z_i])$, for every $i \in \{1, \ldots, k\}$, then, by inductive hypothesis, for every $i \in \{1, \ldots, k\}$ there is an $e_j$ parse tree $\tau_i = (N_i, E_i, e\text{-}idx_i, s_i, f_i)$ for $w[z_{i-1}{+}1, z_i{+}1)$, with $r_i$ being the root of $\tau_i$. Let $r$ be a fresh node, i.e., $r \notin$

$\bigcup_{i\in\{1,\ldots,k\}} N_i$. We define $\tau = (\bigcup_{i\in\{1,\ldots,k\}} N_i \cup \{r\}, \bigcup_{i\in\{1,\ldots,k\}} E_i \cup \{(r, r_i) \mid i \in \{1,\ldots,k\}\}, e\text{-}idx, s, f)$, where $e\text{-}idx$, $s$, and $f$ extend $\bigcup_{i\in\{1,\ldots,k\}} e\text{-}idx_i$, $\bigcup_{i\in\{1,\ldots,k\}} s_i$, and $\bigcup_{i\in\{1,\ldots,k\}} f_i$, respectively, to the new set of nodes $N$ as follows: $e\text{-}idx(r) = i$, $s(r) = s(r_1)$, and $f(r) = f(r_k)$. Clearly, $\tau$ is an $e_i$ parse tree for $w[s(r), f(r)) = w[s(r_1), f(r_k)) = w[x+1, y+1)$. Once again, we did not make use of some of the conjuncts, which are needed to force $expr_j$ intervals to only occur inside $expr_i$ ones. This property will come handy for the encodings given in the next sections. $\square$

Theorem 2 immediately follows from Lemmas 1(a), 3, and 4.

**Theorem 2.** *Let $R$ be a regular expression over $\Sigma$. Then, $\mathcal{L}(R) = \{w \in \Sigma^* \mid w \approx M$ and $M = \langle \mathbb{I}(N), A, B, V\rangle$ is a model such that $M, [0, N-1] \models \varphi_R \wedge \varphi_\Sigma\}$.*

### Appendix C. Soundness of the encoding of $\omega B$-regular expressions

In order to prove the soundness of the encoding of $\omega B$-regular expressions, we establish a correspondence between interval models and $E$ parse trees for $\omega$-words, with $E$ being an $\omega B$-regular expression, similar to the one given in Appendix B.

**Lemma 5.** *Let $E$ be an $\omega B$-regular expression over $\Sigma$ and $w \in \Sigma^\omega$ be an infinite word. If there exists an $E_*$ parse tree for $w$ such that $count(i)$ is a $B$-sequence, for every $e_i \in sub(E)$, with $e_i = e_j^B$, then there is an interval model $M$ such that $w \approx M$ and $M, [0, n] \models \varphi_{E_*} \wedge \varphi_\Sigma \wedge \bigwedge_{(i,j)\in B(E)} \Phi_B^{(i,j)}$ for some $n \in \mathbb{N}$.*

*Proof.* Let $\tau$ be an $E_*$ parse tree for $w$ such that $count(i)$ is a $B$-sequence for every $e_i \in sub(E)$, with $e_i = e_j^B$. Thanks to Lemma 2, we can assume, without loss of generality, that $\tau$ does not contain nodes $n$ such that $s(n) = f(n)$ and $e\text{-}idx(n) = j$, for any $e_i \in sub(E)$, with $e_i = e_j^*$.

By Theorem 3 and Lemma 1(a), there is a model $M' = \langle \mathbb{I}(\mathbb{N}), A, B, \bar{A}, V'\rangle$ such that $w \approx M'$ and $M', [0, n] \models \varphi_{E_*} \wedge \varphi_\Sigma$, for some $n \in \mathbb{N}$. We define a new valuation function $V$ that extends $V'$ (i.e., $V'([x, y]) \subseteq V([x, y])$ for all $[x, y] \in \mathbb{I}(\mathbb{N})$) by providing an interpretation of the new proposition letters

$ph_j, bl_j$, and $p_j$, used in the encoding given in Section 6, so that the resulting model $M = \langle \mathbb{I}(\mathbb{N}), A, B, \bar{A}, V\rangle$ is such that $w \approx M$ and $M, [0, n] \models \varphi_{E_*} \wedge \varphi_\Sigma \wedge \bigwedge_{(i,j)\in B(E)} \Phi_B^{(i,j)}$. Since $V$ extends $V'$, it clearly holds that $w \approx M$ and $M, [0, n] \models \varphi_{E_*} \wedge \varphi_\Sigma$. Thus, we only have to show that $M, [0, n] \models \Phi_B^{(i,j)}$, for every $(i, j) \in B(E)$. For the sake of conciseness, we only show how to define, for a generic element $(i, j) \in B(E)$, a valuation $V$ such that the resulting model satisfies $\Phi_B^{(i,j)}$. Clearly the model resulting from the union of all valuations defined for $(i, j)$ ranging over $B(E)$ satisfies $\bigwedge_{(i,j)\in B(E)} \Phi_B^{(i,j)}$. Thus, let $(i, j) \in B(E)$ and $B$ be the largest number occurring in $count(i)$. Towards the definition of $V$, we define sets $V_{bl_j/ph_j}$, $V_{bl_j}$, $V_{ph_j}$, and $V_{p_j}$, which, intuitively, are meant to keep information about the (point) intervals where the new proposition letters hold. More precisely, we let $V_{bl_j/ph_j} = \{s(n) : n \in Nodes, e\text{-}idx(n) = j\} \setminus \{s(n) : n \in Nodes, e\text{-}idx(n) = i\}$ be the set of candidate points where $bl_j$ or $ph_j$ must hold true, that is, the points from which an $expr_j$, but no $expr_i$ interval, start. Note that, by assumption, it holds that $s(n) < f(n)$ for every node $n \in Nodes$ with $e\text{-}idx(n) = j$. Therefore, $V_{bl_j/ph_j}$ is the set of children, excluding the leftmost ones, of nodes $n$ with $e\text{-}idx(n) = i$. We define now sets $V_{bl_j}$ and $V_{ph_j}$ as two disjoint subsets of $V_{bl_j/ph_j}$. If $V_{bl_j/ph_j}$ is finite, the we let $V_{bl_j} = V_{ph_j} = V_{p_j} = \emptyset$. Otherwise, let $seq_{V_{bl_j/ph_j}} = \langle x_1, x_2, \ldots\rangle$ be the infinite increasing sequence of elements of $V_{bl_j/ph_j}$. We place in $V_{bl_j}$ one element every $B$ many elements, with the remaining ones being placed in $V_{ph_j}$; formally, we define $V_{bl_j} = \{x_h \in seq_{V_{ph_j/bl_j}} \mid h \equiv 0 \ (\texttt{mod}\ B)\}$ and $V_{ph_j} = V_{bl_j/ph_j} \setminus V_{bl_j}$. Note that there are $(B-1)$ many $ph_j$ points between consecutive $bl_j$ points. Finally, we define $V_{p_j} = \{[x_h, x_{h+B}] \in \mathbb{I}(\mathbb{N}) \mid x_h \in seq_{V_{ph_j/bl_j}} \text{ and } h \not\equiv 0 \ (\texttt{mod}\ B)\}$. Notice that, if $V_{ph_j} = \emptyset$ then $V_{p_j} = \emptyset$ as well. Intuitively, $p_j$ will hold true over intervals connecting corresponding $ph_j$ points in consecutive blocks of $ph_j$ points enclosed between consecutive $bl_j$ points, i.e., if $x, y, z$ are three consecutive $bl_j$ points, with $x < y < z$, and $x_1, \ldots, x_{B-1}$ (resp., $y_1, \ldots, y_{B-1}$) are the $ph_j$ points in between $x$ and $y$ (resp., in between $y$ and $z$) increasingly ordered, then $[x_h, y_h]$ is a $p_j$ interval, for every $h \in \{1, \ldots, B-1\}$.

We are now ready to define the new valuation function $V$. For every $[x, y] \in$

$\mathbb{I}(\mathbb{N})$, we define $V([x,y])$ as the unique set such that

- $V'([x,y]) \subseteq V([x,y])$,
- $bl_j \in V([x,y])$ if and only if $x = y$ and $x \in V_{bl_j}$,
- $ph_j \in V([x,y])$ if and only if $x = y$ and $x \in V_{ph_j}$,
- $p_j \in V([x,y])$ if and only if $[x,y] \in V_{p_j}$.

It is easy to verify that, thanks to this definition of $V$, the interval model $M = \langle \mathbb{I}(\mathbb{N}), A, B, \bar{A}, V \rangle$ is such that formulas encoding properties 1–6 in Section 6 hold true on $[0,k]$ for every $k \in \mathbb{N}$. In particular, observe that formula associated with property 3 in Section 6, i.e., an $expr_i$ interval starts in between every pair of consecutive $bl_j$ points, is satisfied. Indeed, let $x, y$ be a generic pair of consecutive $bl_j$ point, with $x < y$, and recall that every $bl_j$ (resp., $ph_j$) point interval $[x,x]$ corresponds to a node $n'$, with $e\text{-}idx(n') = j$, that is a child, but not the leftmost one, of a node $n$ with $e\text{-}idx(n) = i$; more precisely, it holds $[x,x] = [s(n'), s(n')]$. Then, since there are exactly $(B-1)$ many $ph_j$ points in between $x$ and $y$, and given that every node $n$, with $e\text{-}idx(n) = i$, has at most $(B-1)$ children, excluding the leftmost one, we have that there are, in between $x$ and $y$, at least two $ph_j$ point intervals corresponding to nodes $n'$ and $m'$ having different parent nodes, say $n$ and $m$, respectively. Thus, we have $x < s(n') < f(n') \leq f(n) \leq s(m)$ and $s(m') < y$. Moreover, it holds that $s(m) \leq s(m')$, and property 3 is fulfilled, as an $expr_i$ interval starts from point $s(m)$, with $x < s(m) < y$.

Therefore $M, [0,n] \models \Phi_B^{(i,j)}$, and the thesis follows. □

**Lemma 6.** *Let $E$ be an $\omega B$-regular expression over $\Sigma$ and $w \in \Sigma^\omega$ be an infinite word. If there is an interval model $M$ such that $w \approx M$ and $M, [0,n] \models \varphi_{E_*} \wedge \varphi_\Sigma \wedge \bigwedge_{(i,j) \in B(E)} \Phi_B^{(i,j)}$ for some $n \in \mathbb{N}$, then there exists an $E_*$ parse tree for $w$ such that $count(i)$ is a $B$-sequence, for every $e_i \in sub(E)$, with $e_i = e_j^B$.*

*Proof.* Since $w \approx M$ and $M, [0,n] \models \varphi_{E_*} \wedge \varphi_\Sigma$, by Theorem 3 and Lemma 1(a) it is possible to build from $M$ an $E_*$ parse tree $\tau$ for $w$. In particular, it is possible to build $\tau = (Nodes, Edges, e\text{-}idx, s, f)$ so that, for every $(i,j) \in B(E)$ and every node $n \in Nodes$, with $e\text{-}idx(n) = i$, the number of children of $n$ coincides with the number of $expr_j$ intervals contained in the $expr_i$ interval $[s(n)-1, f(n)-1]$. Now, let $(i,j) \in B(E)$. By $M, [0,n] \models \Phi_B^{(i,j)}$, we have that properties 1–6 from Section 6 hold with respect to $M$. Consequently, as shown in Section 6 itself, it is possible to find a bound $K' \in \mathbb{N}$ and a point $x \in \mathbb{N}$ such that every $expr_i$ interval starting after $x$ contains at most $K'$ many $expr_j$ intervals. Since there are only finitely many $expr_i$ intervals starting not later than $x$ (as $expr_i$ intervals do not intersect each other), there is a bound $K \in \mathbb{N}$ such that every $expr_i$ interval contains at most $K$ many $expr_j$ intervals. Thus, it holds that $\max(count(i)) \leq K$, which means that $count(i)$ is a $B$-sequence. $\square$

Theorem 4 immediately follows from Lemmas 1(b), 5, and 6.

**Theorem 4.** *Let $E$ be an $\omega B$-regular expression over $\Sigma$. Then, $\mathcal{L}(E) = \{w \in \Sigma^\omega \mid w \approx M$ and $M$ is a model such that $M, [0,n] \models \varphi_{E_*} \wedge \varphi_\Sigma \wedge \bigwedge_{(i,j)\in B(E)} \Phi_B^{(i,j)}$ for some $n \in \mathbb{N}\}$.*

## Appendix D. Soundness of the encoding of $\omega S$-regular expressions

In this appendix, we prove the soundness of the encoding of $\omega S$-regular expressions in $AB\!\sim$. We proceed analogously to the previous appendix.

**Lemma 7.** *Let $E$ be an $\omega S$-regular expression over $\Sigma$ and $w \in \Sigma^\omega$ be an infinite word. If there exists an $E_*$ parse tree for $w$ such that $count(i)$ is either a finite sequence or an $S$-sequence, for every $e_i \in sub(E)$, with $e_i = e_j^S$, then there is an interval model $M$ such that $w \approx M$ and $M, [0,n] \models \varphi_{E_*} \wedge \varphi_\Sigma \wedge \bigwedge_{(i,j)\in S(E)} \Phi_S^{(i,j)}$ for some $n \in \mathbb{N}$.*

*Proof.* Let $\tau = (Nodes, Edges, e\text{-}idx, s, f)$ be an $E_*$ parse tree for $w$ such that $count(i)$ is either finite or an $S$-sequence, i.e., no number occurs infinitely often in it, for every $e_i \in sub(E)$, with $e_i = e_j^S$

By Theorem 3 and Lemma 1(a), there is a model $M' = \langle \mathbb{I}(\mathbb{N}), A, B, \sim, V' \rangle$ such that $w \approx M'$ and $M', [0,n] \models \varphi_{E_*} \wedge \varphi_\Sigma$, for some $n \in \mathbb{N}$. We define a new valuation function $V$ that extends $V'$ (i.e., $V'([x,y]) \subseteq V([x,y])$ for all $[x,y] \in \mathbb{I}(\mathbb{N})$) by providing an interpretation of the new proposition letters $ph_j$ and $new_j$, as well as the one of $\sim$, used in the encoding given in Section 7, so that the resulting model $M = \langle \mathbb{I}(\mathbb{N}), A, B, \sim, V\rangle$ is such that $w \approx M$ and $M, [0,n] \models \varphi_{E_*} \wedge \varphi_\Sigma \wedge \bigwedge_{(i,j)\in S(E)} \Phi_S^{(i,j)}$. Since $V$ extends $V'$, it clearly holds that $w \approx M$ and $M, [0,n] \models \varphi_{E_*} \wedge \varphi_\Sigma$. Thus, we only have to show that $M, [0,n] \models \Phi_S^{(i,j)}$, for every $(i,j) \in S(E)$. As in the previous section, it is enough to show how to define a valuation function for a generic element of $S(E)$. Thus, let $(i,j) \in S(E)$. If $M', [0,n] \not\models [G]\langle A\rangle\langle A\rangle \mathit{expr}_i \;\wedge\; \langle A\rangle[A][A](\mathit{expr}_j \;\rightarrow\; \neg\pi)$, that is, there are only finitely many $\mathit{expr}_i$ intervals or infinitely many $\mathit{expr}_j$ points, then, for every $[x,y] \in \mathbb{I}(\mathbb{N})$, we define $V([x,y]) = V'([x,y])$ if $x \neq y$, and $V([x,y]) = V'([x,y]) \cup \{\sim\}$ otherwise , and it is immediate to see that $\Phi_S^{(i,j)}$ is satisfied. Now, assume $M', [0,n] \models [G]\langle A\rangle\langle A\rangle \mathit{expr}_i \;\wedge\; \langle A\rangle[A][A](\mathit{expr}_j \;\rightarrow\; \neg\pi)$, that is, there are infinitely many $\mathit{expr}_i$ intervals but only finitely many $\mathit{expr}_j$ points. Since $M'$ is built from $\tau$, we have that $\mathit{count}(i)$ is infinite and there are only finitely many nodes $n$ such that $e\text{-}\mathit{idx}(n) = j$ and $s(n) = f(n)$. Towards the definition of $V$, we introduce the following notation. For a node $n \in \mathit{Nodes}$, we denote by $|n|$ the number of children of $n$ in $\tau$. For $h \in \mathbb{N}_{>0}$, we denote by $n_h$ the $h$th node $n$ such that $e\text{-}\mathit{idx}(n) = i$, according to the ordering produced by a DFS visit of $\tau$. Moreover, we denote by $n_h^k$ the $k$th child of $n_h$, for every $k \in \{1, \ldots, |n_h|\}$. Notice that $e\text{-}\mathit{idx}(n_h^k) = j$, for all $h, k$, and that, since there are only finitely many $\mathit{expr}_j$ points in $M$, there exists an index $h'$ such that $s(n_h^k) < f(n_h^k)$ holds for every $h \geq h'$ and $k \in \{1, \ldots, |n_h|\}$. Observe, also, that $\mathit{count}(i) = \langle |n_h| \rangle_{h\in\mathbb{N}_{>0}}$. Since $\mathit{count}(i)$ is an $S$-sequence, for every natural number $k$ there is a suffix of $\langle |n_h| \rangle_{h\in\mathbb{N}_{>0}}$ that only features numbers greater than $k$. Thus, there exists an infinite increasing sequence of indexes $\mathcal{I} = \langle I_1, I_2, \ldots\rangle$ such that $I_1 > h'$ and, for every $I_m \in \mathcal{I}$, the suffix $\langle |n_h| \rangle_{h \geq I_m}$ of $\langle |n_h| \rangle_{h\in\mathbb{N}_{>0}}$ starting at $I_m$ only features numbers greater than $m$; formally $\min(\langle |n_h| \rangle_{h\geq I_m}) > m$. Intuitively, if $n_h$ is such that $h \geq I_m$, then $n_h$ has more than $m$ children, and each such children $n_h^k$ is such that $s(n_h^k) < f(n_h^k)$.

We are now ready to define the new valuation function $V$. For every $[x,y] \in$

$\mathbb{I}(\mathbb{N})$, we define $V([x,y])$ as the unique set such that

- $V'([x,y]) \subseteq V([x,y])$,
- $ph_j \in V([x,y])$ if and only if there is an index $I_m \in \mathcal{I}$ and a node $n_h^k$, with $h \geq I_m$ and $2 \leq k \leq m$, such that $x = y = s(n_h^k)$,
- $new_j \in V([x,y])$ if and only if there is an index $I_m \in \mathcal{I}$ such that $x = y = s(n_{I_m}^m)$,
- $\sim \in V([x,y])$ if $ph_j \in V[x,x]$, $ph_j \in V[y,y]$, $new_j \notin V[y,y]$, and there are nodes $n_h^k$ and $n_{h'}^{k'}$, with $h' = h+1$, $k' = k$, $x = s(n_h^k)$, and $y = s(n_{h'}^{k'})$ – note that we only provide sufficient condition for the definition of $\sim$; the full valuation is obtained by applying transitive closure (recall that $\sim$ encodes an equivalence relation).

It is easy to verify that, thanks to this definition of $V$, the interval model $M = \langle \mathbb{I}(\mathbb{N}), A, B, \sim, V \rangle$ is such that formulas encoding properties 1–4 in Section 7 hold true on $[0,k]$ for every $k \in \mathbb{N}$.

Therefore $M, [0,n] \models \Phi_S^{(i,j)}$, and the thesis follows. $\square$

**Lemma 8.** *Let $E$ be an $\omega S$-regular expression over $\Sigma$ and $w \in \Sigma^\omega$ be an infinite word. If there is an interval model $M$ such that $w \approx M$ and $M, [0,n] \models \varphi_{E_*} \wedge \varphi_\Sigma \wedge \bigwedge_{(i,j) \in S(E)} \Phi_S^{(i,j)}$ for some $n \in \mathbb{N}$, then there exists an $E_*$ parse tree for $w$ such that $count(i)$ is either a finite sequence or an $S$-sequence, for every $e_i \in sub(E)$, with $e_i = e_j^S$.*

*Proof.* Since $w \approx M$ and $M, [0,n] \models \varphi_{E_*} \wedge \varphi_\Sigma$, by Theorem 3 and Lemma 1(a) it is possible to build from $M$ an $E_*$ parse tree $\tau$ for $w$. In particular, it is possible to build $\tau = (Nodes, Edges, e\text{-}idx, s, f)$ so that, for every $(i,j) \in S(E)$ and every node $n \in Nodes$, with $e\text{-}idx(n) = i$, the number of children of $n$ coincides with the number of $expr_j$ intervals contained in the $expr_i$ interval $[s(n)-1, f(n)-1]$. Now, let $(i,j) \in S(E)$. By $M, [0,n] \models \Phi_S^{(i,j)}$, we have that $M, [0,n] \models [G]\langle A\rangle\langle A\rangle expr_i \ \wedge \ \langle A\rangle[A][A](expr_j \ \rightarrow \ \neg\pi) \rightarrow \langle B\rangle\langle A\rangle\langle A\rangle ph_j$. We distinguish two possibilities, depending on whether $M, [0,n] \models [G]\langle A\rangle\langle A\rangle expr_i \ \wedge$

$\langle A\rangle[A][A](expr_j \rightarrow \neg\pi)$ or not. In the former case, $M,[0,n] \models \langle B\rangle\langle A\rangle\langle A\rangle ph_j$ holds as well, meaning that $M$ features at least one $ph_j$ point. We have already shown in Section 7 that, as long as a model $M$ features at lease one $ph_j$ point, properties 1–4 from Section 7 force it to also feature an infinite sequence of $expr_i$ intervals that behave correctly according to the $S$-constructor. Therefore, such a model $M$ encodes an $E_*$ parse tree for $w$ such that $count(i)$ is an $S$-sequence. If, instead, it is the case that $M,[0,n] \not\models [G]\langle A\rangle\langle A\rangle expr_i \wedge \langle A\rangle[A][A](expr_j \rightarrow \neg\pi)$, then there are only finitely many $expr_i$ intervals or infinitely many $expr_j$ points. In the former case, $count(i)$ is clearly finite, hence the thesis holds. In the latter one, the thesis follows from Corollary 2 and Lemma 1(c). $\square$

Theorem 5 immediately follows from Lemmas 1(c), 7, and 8.

**Theorem 5.** *Let $E$ be an $\omega S$-regular expression over $\Sigma$. Then, $\mathcal{L}(E) = \{w \in \Sigma^\omega \mid w \approx M$ and $M$ is a model such that $M,[0,n] \models \varphi_{E_*} \wedge \varphi_\Sigma \wedge \bigwedge_{(i,j)\in S(E)} \Phi_S^{(i,j)}$ for some $n \in \mathbb{N}\}$.*

## Appendix E. Soundness of the encoding of $\omega T$-regular expressions

In this appendix, we prove the soundness of the encoding of $\omega T$-regular expressions in $AB\bar{A}{\sim}$. We follow the same path we followed in the previous proofs of soundness.

**Lemma 9.** *Let $E$ be an $\omega T$-regular expression over $\Sigma$ and $w \in \Sigma^\omega$ be an infinite word. If there exists an $E_*$ parse tree for $w$ such that $count(i)$ is either a finite sequence or a $T$-sequence, for every $e_i \in sub(E)$, with $e_i = e_j^S$, then there is an interval model $M$ such that $w \approx M$ and $M,[0,n] \models \varphi_{E_*} \wedge \varphi_\Sigma \wedge \bigwedge_{(i,j)\in T(E)} \Phi_T^{(i,j)}$ for some $n \in \mathbb{N}$.*

*Proof.* Let $\tau = (Nodes, Edges, e\text{-}idx, s, f)$ be an $E_*$ parse tree for $w$ such that $count(i)$ is either finite or a $T$-sequence, i.e., it features infinitely many values occurring infinitely often, for every $e_i \in sub(E)$, with $e_i = e_j^T$.

By Theorem 3 and Lemma 1(a), there is a model $M' = \langle \mathbb{I}(\mathbb{N}), A, B, \bar{A}, \sim, V'\rangle$ such that $w \approx M'$ and $M',[0,n] \models \varphi_{E_*} \wedge \varphi_\Sigma$, for some $n \in \mathbb{N}$. We define a new valuation function $V$ that extends $V'$ (i.e., $V'([x,y]) \subseteq V([x,y])$ for all $[x,y] \in \mathbb{I}(\mathbb{N})$) by providing an interpretation of the new proposition letters used in the encoding given in Section 8, so that the resulting model $M = \langle \mathbb{I}(\mathbb{N}), A, B, \bar{A}, \sim, V\rangle$ is such that $w \approx M$ and $M, [0,n] \models \varphi_{E_*} \wedge \varphi_\Sigma \wedge \bigwedge_{(i,j)\in T(E)} \Phi_T^{(i,j)}$. Since $V$ extends $V'$, it clearly holds that $w \approx M$ and $M, [0,n] \models \varphi_{E_*} \wedge \varphi_\Sigma$. Thus, we only have to show that $M, [0,n] \models \Phi_T^{(i,j)}$, for every $(i,j) \in T(E)$. As in the previous section, it is enough to show how to define a valuation function for a generic element of $T(E)$. Thus, let $(i,j) \in T(E)$. If $M', [0,n] \models \langle A\rangle[A][A]\neg \mathit{expr}_i$ (the first disjunct of $\Phi_T^{(i,j)}$ is satisfied), then $M', [0,n] \models \Phi_T^{(i,j)}$ and we are done. Otherwise, there are infinitely many $\mathit{expr}_i$ intervals, and thus $\mathit{count}(i)$ is a $T$-sequence, meaning that it features infinitely many values occurring infinitely often. It is possible to provide an evaluation of proposition letters $\sim$, $ph_j$, $bl_j$, $p_j$, $q_j$, $\mathit{conf}_j$, and $in_j$, so that both $\Phi_\infty^{(i,j)}$ and $\Phi^{in_j}$ are satisfied. A formal definition would be too much pedantic and technical, so we omit it. Intuitively, the interval model is divided into configurations (intervals whose endpoints are consecutive $\mathit{conf}_j$ points), and each configuration features one block (interval whose endpoints are consecutive $bl_j$ points) more than the previous one. Each block in each configuration is instantiated (using proposition letters $ph_j$ and $in_j$) with an $\mathit{expr}_i$ interval whose number of children is one of the values occurring infinitely often in $\mathit{count}(i)$. It is not difficult to convince oneself that one such models satisfies $\Phi_\infty^{(i,j)}$ and $\Phi^{in_j}$. Then, if model $M'$ features infinitely many $\mathit{expr}_j$ points, then $M$ satisfies the second disjunct of $\Phi_T^{(i,j)}$ ($\Phi_\infty^{(i,j)} \wedge \Phi^{in_j} \wedge [G]\langle A\rangle\langle A\rangle(\pi \wedge \mathit{expr}_j) \wedge \langle B\rangle\langle A\rangle\langle A\rangle in_j \wedge [G](in_j \rightarrow \langle A\rangle(\neg\pi \wedge \sim \wedge \langle A\rangle in_j))$), otherwise it satisfies the third one ($\Phi_\infty^{(i,j)} \wedge \Phi^{in_j} \wedge \langle A\rangle[A][A](\mathit{expr}_j \rightarrow \neg\pi) \wedge [G](ph_j \rightarrow \langle A\rangle(\neg\pi \wedge \sim \wedge \langle A\rangle in_j))$).

Therefore $M, [0,n] \models \Phi_S^{(i,j)}$, and the thesis follows. $\square$

**Lemma 10.** *Let $E$ be an $\omega T$-regular expression over $\Sigma$ and $w \in \Sigma^\omega$ be an infinite word. If there is an interval model $M$ such that $w \approx M$ and $M, [0,n] \models \varphi_{E_*} \wedge \varphi_\Sigma \wedge \bigwedge_{(i,j)\in T(E)} \Phi_T^{(i,j)}$ for some $n \in \mathbb{N}$, then there exists an $E_*$ parse tree for $w$ such that $\mathit{count}(i)$ is either a finite sequence or a $T$-sequence, for every $e_i \in \mathit{sub}(E)$, with $e_i = e_j^S$.*

*Proof.* Since $w \approx M$ and $M, [0, n] \models \varphi_{E_*} \wedge \varphi_\Sigma$, by Theorem 3 and Lemma 1(a) it is possible to build from $M$ an $E_*$ parse tree $\tau$ for $w$. In particular, it is possible to build $\tau = (Nodes, Edges, e\text{-}idx, s, f)$ so that, for every $(i, j) \in T(E)$ and every node $n \in Nodes$, with $e\text{-}idx(n) = i$, the number of children of $n$ coincides with the number of $expr_j$ intervals contained in the $expr_i$ interval $[s(n) - 1, f(n) - 1]$. Now, let $(i, j) \in T(E)$. Since $M, [0, n] \models \Phi_T^{(i,j)}$, we have three possibilities, depending on whether the first, second, or third disjunct of $\Phi_T^{(i,j)}$ holds true.

If the first disjunct ($\langle A\rangle[A][A]\neg expr_i$) is true, then $count(i)$ is finite, and we are done. If the second disjunct $(\Phi_\infty^{(i,j)} \wedge \Phi^{in_j} \wedge [G]\langle A\rangle\langle A\rangle(\pi \wedge expr_j) \wedge \langle B\rangle\langle A\rangle\langle A\rangle in_j \wedge [G](in_j \to \langle A\rangle(\neg\pi \wedge \sim \wedge \langle A\rangle in_j)))$ is satisfied, then there are infinitely many $expr_j$ points and there is at least one value occurring infinitely often in $count(i)$; the thesis follows from Corollary 2, which establishes that $\tau$ can be suitably adapted so to make $count(i)$ a $T$-sequence. Finally, if the third disjunct $(\Phi_\infty^{(i,j)} \wedge \Phi^{in_j} \wedge \langle A\rangle[A][A](expr_j \to \neg\pi) \wedge [G](ph_j \to \langle A\rangle(\neg\pi \wedge \sim \wedge\langle A\rangle in_j)))$ is fulfilled, then, as we have already shown in Section 8, there are infinitely many values occurring infinitely often in $count(i)$, meaning that $count(i)$ is a $T$-sequence. $\square$

Theorem 6 below immediately follows from Lemmas 1(d), 9, and 10.

**Theorem 6.** *Let $E$ be an $\omega T$-regular expression over $\Sigma$. Then, $\mathcal{L}(E) = \{w \in \Sigma^\omega \mid w \approx M$ and $M$ is a model such that $M, [0, n] \models \varphi_{E_*} \wedge \varphi_\Sigma \wedge \bigwedge_{(i,j)\in T(E)} \Phi_T^{(i,j)}$ for some $n \in \mathbb{N}\}$.*

[1] A. Montanari, P. Sala, Interval logics and $\omega$B-regular languages, in: Proc. of the 7th LATA, Vol. 7810 of LNCS, Springer, 2013, pp. 431–443. `doi:10.1007/978-3-642-37064-9\_38`.

[2] D. Della Monica, A. Montanari, P. Sala, Extended omega-regular languages and interval temporal logic, 22nd Italian Conference on Theoretical Computer Science (ICTCS) (2021).

[3] J. R. Büchi, On a decision method in restricted second order arithmetic, in: Logic, Methodology and Philosophy of Science (Proc. 1960 Internat. Congr.), Stanford Univ. Press, Stanford, Calif., 1962, pp. 1–11.

[4] R. McNaughton, Testing and generating infinite sequences by a finite automaton, Inf. Control. 9 (5) (1966) 521–530. `doi:10.1016/S0019-9958(66)80013-X`.
URL `https://doi.org/10.1016/S0019-9958(66)80013-X`

[5] M. O. Rabin, Decidability of second-order theories and automata on infinite trees, Transactions of the American Mathematical Society 141.
URL `http://www.jstor.org/stable/1995086`

[6] J. Hopcroft, Introduction to automata theory, languages, and computation, Addison-Wesley, Boston, 2001.

[7] M. Bojańczyk, A bounding quantifier, in: CSL, Vol. 3210 of LNCS, Springer, 2004, pp. 41–55. `doi:10.1007/978-3-540-30124-0_7`.

[8] M. Bojańczyk, Weak MSO with the unbounding quantifier, Theory of Computing Systems 48 (3) (2011) 554–576. `doi:10.1007/s00224-010-9279-2`.

[9] M. Bojańczyk, T. Colcombet, Bounds in $\omega$-regularity, in: LICS, 2006, pp. 285–296. `doi:10.1109/LICS.2006.17`.

[10] M. Bojańczyk, T. Colcombet, Boundedness in languages of infinite words, Logical Methods in Computer Science Volume 13, Issue 4.

[11] D. Barozzini, D. de Frutos-Escrig, D. Della Monica, A. Montanari, P. Sala, Beyond $\omega$-regular languages: $\omega T$-regular expressions and their automata and logic counterparts, Theor. Comput. Sci. 813 (2020) 270–304.

[12] A. Montanari, P. Sala, Adding an equivalence relation to the interval logic $\mathrm{AB\bar{B}}$: complexity and expressiveness, in: Proc. of the 28th LICS, IEEE Computer Society, 2013, pp. 193–202. doi:10.1109/LICS.2013.25.

[13] B. C. Moszkowski, Z. Manna, Reasoning in interval temporal logic, in: Proc. of Workshop on Logic of Programs, Vol. 164 of LNCS, Springer, 1983, pp. 371–382. doi:10.1007/3-540-12896-4_374.

[14] J. F. Allen, Maintaining knowledge about temporal intervals, Comm. of the ACM 26 (11) (1983) 832–843. doi:10.1145/182.358434.

[15] J. Y. Halpern, Y. Shoham, A propositional modal logic of time intervals, Journal of the ACM 38 (4) (1991) 935–962. doi:10.1145/115234.115351.

[16] B. C. Moszkowski, Reasoning about digital circuits, Ph.D. thesis, Department of Computer Science, Stanford University, Stanford, CA (1983).

[17] K. Lodaya, Sharpening the undecidability of interval temporal logic, in: Proc. of the 6th Asian Computing Science Conference – Advances in Computing Science – ASIAN, Vol. 1961 of LNCS, Springer, 2000, pp. 290–298. doi:10.1007/3-540-44464-5_21.

[18] D. Bresolin, D. Della Monica, A. Montanari, P. Sala, G. Sciavicco, Interval temporal logics over strongly discrete linear orders: Expressiveness and complexity, Theor. Comput. Sci. 560 (2014) 269–291.

[19] D. Bresolin, D. Della Monica, A. Montanari, P. Sala, G. Sciavicco, Decidability and complexity of the fragments of the modal logic of Allen's relations over the rationals, Inf. Comput. 266 (2019) 97–125. doi:10.1016/j.ic.2019.02.002.

[20] A. Montanari, G. Puppis, P. Sala, G. Sciavicco, Decidability of the interval temporal logic $\mathrm{AB\bar{B}}$ over the natural numbers, in: Proc. of the 27th STACS, Vol. 5 of LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2010, pp. 597–608. `doi:10.4230/LIPIcs.STACS.2010.2488`.

[21] A. Montanari, G. Puppis, P. Sala, Maximal decidable fragments of Halpern and Shoham's modal logic of intervals, in: Proc. of the 37th ICALP, Part II, Vol. 6199 of LNCS, Springer, 2010, pp. 345–356. `doi:10.1007/978-3-642-14162-1\_29`.

[22] D. Della Monica, A. Montanari, P. Sala, The importance of the past in interval temporal logics: The case of propositional neighborhood logic, in: Logic Programs, Norms and Action - Essays in Honor of Marek J. Sergot on the Occasion of His 60th Birthday, Vol. 7360 of LNCS, Springer, 2012, pp. 79–102. `doi:10.1007/978-3-642-29414-3\_6`.