Bounds in ω-regularity

Mikołaj Bojańczyk
Warsaw University and LIAFA, Paris
bojan@mimuw.edu.pl

Thomas Colcombet
CNRS/IRISA, Rennes
colcombe@irisa.fr

Abstract

We consider an extension of ω-regular expressions where two new variants of the Kleene star L^* are added: L^B and L^S. These exponents act as the standard star, but restrict the number of iterations to be bounded (for L^B) or to tend toward infinity (for L^S). These expressions can define languages that are not ω-regular.

We develop a theory for these languages. We study the decidability and closure questions. We also define an equivalent automaton model, extending Büchi automata. This culminates with a — partial — complementation result.

1 Introduction

In this paper we introduce a new kind of language of infinite words. The new languages – called ωBS-regular languages – are defined using an extended form of ω-regular expressions. The extended expressions can define properties such as "words of the form $(a^*b)^\omega$ where the size of a^*b blocks is bounded". As witnessed by this example, ωBS-regular languages are a proper extension of ω-regular languages.

The expressions for ωBS-regular languages are obtained from the usual ω-regular expressions by adding two new variants of the Kleene star L^*. These are called the bounded exponent L^B and the strongly unbounded exponent L^S. The idea behind B is that the language L in the expression L^B must be iterated a bounded number of times. For instance, the language from the first paragraph is described by the expression $(a^B b)^\omega$. The idea behind S is that the iterations of the language L must tend toward infinity (i.e. have no bounded subsequence). This is not the same as being unbounded, which is more easily satisfied. In particular, the complement of the language $(a^B b)^\omega$ is the language

$$(a+b)^* a^\omega + ((a+b)^* a^S b)^\omega$$

and not the (smaller) language

$$(a+b)^* a^\omega + (a^S b)^\omega \ .$$

For instance, the word $aba^1baba^2baba^3baba^4b\cdots$ belongs to the first but not the second.

This paper is devoted to developing a theory for those new languages.

The most important concept is a new type of automaton over infinite words, called a bounding automaton. Bounding automata can be used as an alternative definition of the new languages. However, the translations between bounding automata and ωBS regular expressions are more involved than in the case of regular languages.

A bounding automaton is a finite automaton equipped with a finite number of counters. These couters can be incremented and reset, but not read. The counter values are used in the acceptance condition, which depends on their asymptotic values (whether counter values are bounded or tend toward infinity). We show that bounding automata recognize exactly the languages that can be defined using ωBS-regular expressions. Thanks to simple automata constructions, we obtain closure of ωBS-regular languages under union, intersection and projection.

Unfortunately, ωBS-regular languages are not closed under complementation, nor can the bounding automata be determinized. The following language witnesses the first statement as it is ωBS-regular, but its complement is not:

> Words of the form $a^{n_1}ba^{n_2}b\cdots$ where $n_1, n_2, \ldots$ can be split in two subsequences: one bounded, the other tending toward infinity.

Failure of complementation is bad news, especially from a logical point of view. However, we are able to identify two fragments of ωBS-regular languages that complement each other. We show that the complement of a language that only talks about bounded sequences is a language that only talks about sequences tending toward infinity; and vice versa. The difficult proof of this complementation result is the technical core of the paper.

Finally, we present a logic that expresses ωBS-regular languages. As is well known, languages defined by

Work supported by the EU-TNR network GAMES. The first author also supported by MNII grant 4 T11C 042 25 and a scholarship of the Foundation for Polish Science

ω-regular expressions are exactly the ones definable in monadic second-order logic. What extension logic corresponds to ωBS-regular expressions? One avenue is to add a new quantifier, called the bounding quantifier $\mathbb{B}$. A formula $\mathbb{B}X.\ \phi(X)$ is true if the size of sets satisfying $\phi(X)$ is bounded. Every ωBS-regular language can be defined in monadic second-order logic extended with $\mathbb{B}$. Due to failure of complementation, the converse does not hold.

Related work This work tries to continue the long lasting tradition of logic/automata correspondences [10] initiated by Büchi [4, 5] and continued by Rabin [7]. We believe that bounding properties extend the received notion of regularity and that languages defined by our extended expressions have every right to be called regular, even though they are not captured by Büchi automata. For instance, every ωBS-regular language L has a finite number of quotients $w^{-1}L, Lw^{-1}$. (Moreover, the right quotients Lw^{-1} are regular languages of finite words.) Unfortunately, our results fall short of these grand expectations, since we do not have a full complementation result.

The quantifier $\mathbb{B}$ in the logic that describes ωBS-regular languages was already introduced in [2]. Although [2] went beyond words and considered infinite trees, the satisfiability algorithm worked for a more restricted fragment of the logic with no (not even partial) complementation result. In particular, no appropriate notion of automata or regular expression was proposed.

Boundedness properties have been considered in model-checking. For instance, [3] considered systems described by push-down automata whose stack size is unbounded.

Our work on bounds can also be related to cardinality restrictions. In[6], Klaedtke and Ruess considered an extension of monadic second-order logic with cardinality extensions of the form

$$|X_1| + \cdots + |X_n| \le |Y_1| + \cdots + |Y_m| \ .$$

In general, such cardinality constraints (even $|X| \le |Y|$) lead to undecidability. Even though cardinality constraints can express all ωBS-regular languages, the decidable fragments considered in [6] are insufficient for our purposes.

Structure of the paper. In Section 2, we formally define the ωBS-regular expressions that are the subject of this paper. We introduce two restricted types of expressions (where the B and S exponents are prohibited, respectively) and overview the closure properties of the respective expressions. In Section 3, we introduce our automata models and show that they are equivalent to the regular expressions. In Section 4, we state the main technical result, which concerns closure under complementation. In Section 5, we show how our results can be applied to obtain a decision procedure for satisfiability in an extension of monadic second-order logic.

2 Regular expressions with bounds

In this section we define the different variants of ωBS-regular expressions, overview the results concerning them and show the strictness of their inclusions.

2.1 Definition

To the standard operations used in ω-regular expressions, we add two variants of the Kleene star $*$: the B and S exponents. These are used to constrain the number of iterations. When the B exponent is used, the number of iteration has to be bounded. When the S exponent is used, it has to tend toward infinity. For instance, the expression $(a^B b)^\omega$ represents the words in $(a^* b)^\omega$ where the size of sequences of consecutive a's is bounded. Similarly, the expression $(a^S b)^\omega$ requires the size of maximal sequences of consecutive a's to tend toward infinity. These new expressions are called ωBS-regular expression.

In the following we will say that a sequence of naturals is *strictly unbounded* if it tends toward the infinite, i.e. has no bounded subsequence. This behavior is denoted by the letter S, while the bounded behavior is denoted by B.

In order to formally define ωBS-regular expressions, we first use BS-regular expressions, which describe infinite sequences of finite words. Our ωBS-regular expressions are built on top of BS-regular expression just as ω-regular expressions are built on top of regular expressions. A BS-*regular expression* has the following syntax (a being some letter of the given finite alphabet Σ, and M ranging over the regular languages of finite words over Σ):

$$e = \ \emptyset \ | \ a \ | \ e.e \ | \ e+e \ | \ M \rhd e \ | \ e^* \ | \ e^B \ | \ e^S \ .$$

Except for the two extra exponents B and S and the $\rhd$ operator, these expressions coincide syntactically with the standard regular expressions. However, the semantics cannot be given in terms of languages of finite words. Instead, a BS-regular expression is evaluated to a language of sequences; by *sequence* we mean an element of $(\Sigma^*)^\omega$. We will denote by $\vec{u}$ the sequence $(u_1, u_2, \dots)$.

The semantic of BS-regular expressions is defined as follows.

- $\emptyset$ is the empty language of sequences.
- a for $a \in \Sigma$ is the language containing the single sequence $(a, a, \dots)$.
- The *concatenation* of sequence languages is defined by
$$K.L = \{(u_1v_1, u_2v_2 \dots) \ : \ \vec{u} \in K, \vec{v} \in L\} \ .$$
- The *mix* of sequence languages (which is *not* the union) is
$$K + L = \{\vec{w} \ : \ \vec{u}, \vec{v} \in K \cup L, \ \forall i. w_i \in \{u_i, v_i\}\} \ .$$

- The *finite mix* $M \triangleright L$ for M a regular language of finite words, is the set of sequences obtained by taking a sequence from L and replacing a finite number of coordinates with a word from M. This operator is redundant as established in Proposition 2.4.

- The $*$-*exponent* of a language of sequences is defined by grouping words into blocks:

$$L^* = \{(u_1 \dots u_{f(1)-1}, u_{f(1)} \dots u_{f(2)-1}, \dots) : \vec{u} \in L,\ f \text{ nondecreasing}\} .$$

- The *bounded exponent* L^B of a language of sequences is defined like L^* but we additionally require the values $f(i+1) - f(i)$ to be bounded, i.e. only factorizations of bounded size are allowed.

- The *strictly unbounded exponent* L^S of a language of sequences is defined like L^* but we additionally require the values $f(i+1) - f(i)$ to be strictly unbounded, i.e. the size of the concatenations used in the factorization must tend toward the infinite.

Languages of sequences obtained by nesting these operations — i.e. the language of sequences obtained by evaluating BS-regular expressions — are called *BS-regular languages*. The *B-regular* (resp. *S-regular*) languages correspond to the particular case where the exponent S (resp. B) is not used. When the context is clear, we do not distinguish between an expression and the corresponding language.

For instance the BS-regular (also B-regular) expression a^B represents the sequences of words from a^* where the number of a's is bounded:

$$a^B = \{(a^{f(1)}, a^{f(2)}, \dots) \ : \ f \text{ is bounded}\}$$

The sequence language $a^B.(b.a^B)^S$ consists of sequences where the number of consecutive a's is bounded, while the number of b's in each word of the sequence is strictly unbounded.

In this definition, we override the symbols $+, ., *$ from regular expressions. However, there is a strong link between the two semantics. If one takes a standard regular expression defining a language of finite words L and evaluates it as a BS-regular expression, the resulting language of sequences is simply $\{\vec{u} \ : \ \forall i. u_i \in L\}$.

Before proceeding to the definition of ωBS-regular languages, we first emphasize some closure properties of BS-regular languages.

Fact 2.1 Every BS-regular language satisfies $L = L + L$. Furthermore, for $\vec{u}$ in L and f a strictly unbounded sequence of naturals, the sequence $(u_{f(1)}, u_{f(2)}, \dots)$ also belongs to L. In particular, L is closed under taking subsequences.

Proof
Structural induction. □

We are now ready to introduce the ωBS-regular expressions. These describe languages of ω-words. From a sequence with nonempty words on infinitely many coordinates, we can construct an ω-word by concatenation of all the words:

$$(u_1, u_2, \dots)^\omega = u_1 u_2 \dots$$

This operation is naturally extended to languages of sequences by taking the ω power of every sequence in the language.

Definition 2.2 An *ωBS-regular language* is a finite union (denoted $+$) of languages of the form $M.L^\omega$, where M is a regular language of finite words and L is BS-regular. When only B-regular (resp. S-regular) languages are used for L then the resulting language is called *ωB-regular* (resp. *ωS-regular*).

This definition differs from the definition of ω-regular expressions only in that the ω is applied to BS-regular languages of sequences instead of regular word languages. As one may expect, the standard class of ω-regular languages corresponds to the case of ωBS-regular languages where neither B nor S is used.

For instance, the expression $(a^B.b)^\omega$ defines the language of ω-words containing an infinite number of b's where the possible number of consecutive a's is bounded. The language $(a^S.b)^\omega$ corresponds to the case where the length of maximal consecutive sequences of a's tends toward infinity. The language $(a+b)^* a^\omega + ((a^*.b)^*.a^S.b)^\omega$ is a bit more involved. It corresponds to the language of words where either there are finitely many b's, or the number of consecutive a's is unbounded but not necessarily strictly unbounded. This is the complement of the language $(a^B.b)^\omega$.

Fact 2.3 Emptiness is decidable for ωBS-regular languages.

Proof
An ωBS-regular language is nonempty if and only if one of the languages $M.L^\omega$ is such that the regular language M is nonempty and the BS-regular language L admits infinitely many nonempty words in a sequence. The latter can be shown decidable by structural induction. Essentially it amounts to finding a letter in the BS-expression, and verifying that none of the subexpressions containing this letter is concatenated with an empty language. □

The finite mix operator $\triangleright$ will turn out to be a convenient technical device. However, it is not necessary for describing ωBS-regular languages, as stated by:

Proposition 2.4 Every ωBS-regular expression (resp. ωB-regular and ωS-regular ones) is equivalent to one without $\rhd$ operator.

Note that this proposition does not mean that finite mix can be eliminated from BS-regular languages of sequences. For instance, the finite mix operator is necessary to define the set of sequences $a \rhd b$ where a finite number of a's is used. However, after the ω power is applied, the expression $(a \rhd b)^\omega$ can be rewritten into – and this is the subject of Proposition 2.4 – the expression $(a + b)^* b^\omega$.

2.2 Summary: The diamond

In this section we present Figure 1, which summarizes the technical contributions of this paper. We call this figure *the diamond*. Though not all the material necessary to understand this figure has been yet provided, we give it here as a reference and guide to what follows.

The diamond illustrates the four variants of languages of ω-words we consider: ω-regular, ωB-regular, ωS-regular and ωBS-regular languages. The inclusions between those four classes give a diamond shape. We show in Section 2.3 that the inclusions in the diamond are indeed strict.

To each class of languages corresponds a family of automata. The automata come in two variants: "normal automata", and the equivalent "hierarchical automata". The exact definition of these automata as well as the corresponding equivalences are the subject of Section 3 and Theorem 3.1.

All the classes are closed under union by definition. It is also easy to show that the classes are closed under projection, i.e. images under a letter to letter morphism (operation denoted by π in the figure). From the equivalence of the different families of languages with families automata we obtain closure by intersection for the four classes; see Corollary 3.2. For the closure under complement, things are not so nice. Indeed in Section 2.3 we show that ωBS-regular language are not closed under complement. However, some complementation results are still possible. Namely Theorem 4.1 establishes that complementing an ωB-regular language gives an ωS-language, and vice-versa. This is by far the most involved result of this work and we only sketch some ideas about its proof.

In Section 5 we will show how the closure results can be used to partially answer the satisfiability problem for an extension of monadic second-order logic. For this purpose, we establish Proposition 5.3 stating the closure of ωS-regular languages under a less standard operation called $\mathbb{U}$.

2.3 Limits of the diamond

In this section we show that all the inclusions depicted in the diamond are strict. Moreover, we show that there exists an ωBS-regular language whose complement is not ωBS-regular.

We start by a simple lemma.

Lemma 2.5 Every ωB-regular language over the alphabet $\{a, b\}$ which contains a word with an infinite number of b's contains a word in $(a^B b)^\omega$.

Proof
We show by a simple structural induction that a B-regular language of sequences L satisfies:

- if L contains a sequence in a^*, it contains a sequence in a^B, and,
- if L contains a sequence in $(a^* b)^+ a^*$, it contains a sequence in $(a^B b)^+ a^B$.

The statement of the lemma follows. □

Corollary 2.6 The language $(a^S b)^\omega$ is not ωB-regular. The language $(a^B b)^\omega$ is not ωS-regular.

Proof
The language $(a^S b)^\omega$ contains a word with an infinite number of b's, but its intersection with $(a^B b)^\omega$ is empty. Being ωB-regular for this language would contradict Lemma 2.5.

For the second part, assume that the language $(a^B b)^\omega$ is ωS-regular, then so is the language $(a^B b)^\omega + (a+b)^* a^\omega$. Using Theorem 4.1, its complement $((a^* b)^* a^S b)^\omega$ would be ωB-regular. But this is not possible, by the same argument as above. A proof that does not use complementation – along the same lines as in the first part – can also be given. □

We now proceed to show that ωBS-regular languages are not closed under complement. We start with a similar lemma.

Lemma 2.7 Every ωBS-regular language over the alphabet $\{a, b\}$ that contains a word with an infinite number of b's also contains a word in $(a^B b + a^S b)^\omega$.

Proof
As for Lemma 2.5, we show the following properties of a BS-regular language of sequences L by a simple structural induction:

- if L contains a sequence in a^*, it contains a sequence in $a^B + a^S$, and,
- if L contains a sequence in $(a^* b)^+ a^*$, it contains a sequence in $(a^B b + a^S b)^+ (a^B + a^S)$.

The result directly follows. □

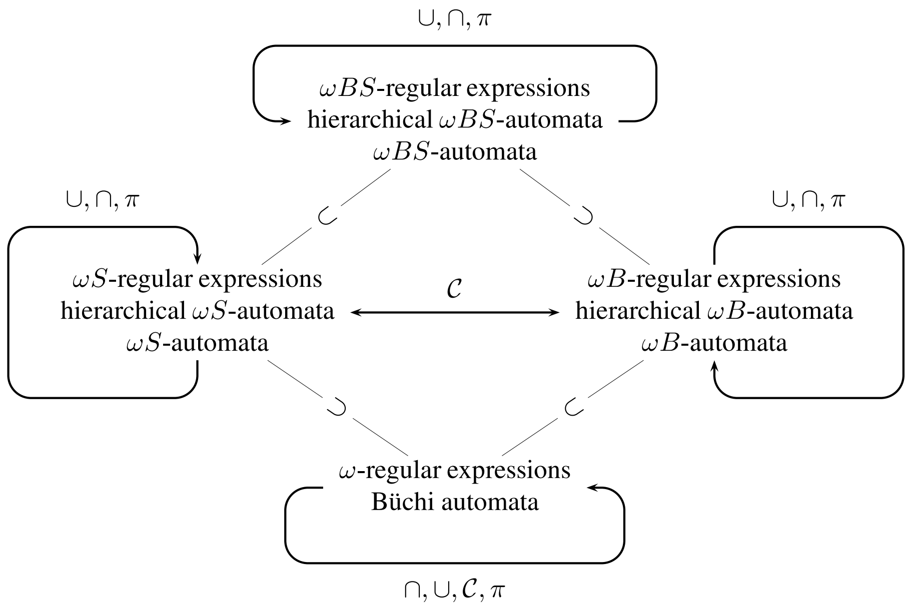


Figure 1. The diamond

Corollary 2.8 The complement of $L = (a^B b + a^S b)^\omega$ is not ωBS-regular.

Proof
The complement of L contains the word

$$a^1\ ba^1ba^2\ ba^1ba^2ba^3\ ba^1ba^2ba^3ba^4b\ \ldots\ ,$$

and consequently, assuming it is ωBS-regular, one can apply Lemma 2.7 on it. It follows that the complement of L should intersect L. Obviously a contradiction. $\square$

3 Automata

In this section we introduce new types of automata over infinite words, called ωBS-automata, and show their equivalence with ωBS-regular expressions.

3.1 Statement of the equivalences

The key equivalence result of this section is the following one.

Theorem 3.1
The following properties of a language of ω-words L are equivalent:

1. *L is ωBS-regular (resp. ωB-regular, resp. ωS-regular),*
2. *L is accepted by a hierarchical ωBS-automaton (resp. a hierarchical ωB-automaton, resp. a hierarchical ωS-automaton),*
3. *L is accepted by an ωBS-automaton (resp. an ωB-automaton, resp. an ωS-automaton).*

The necessary definitions are in the two subsequent sections, the first one defining the most general form of ωBS-automata, the second introducing their hierarchical form.

We mention here, somewhat ahead of time, an important application of this theorem: the closure under intersection of all the classes of languages.

Corollary 3.2 The classes of ωBS-regular, ωB-regular and ωS-regular languages are closed under intersection.

Proof
The corresponding automata are closed under intersection.
$\square$

3.2 General form of ωBS-automata

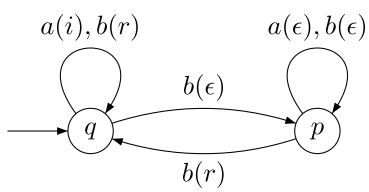
An *ωBS-automaton*, like any finite automaton, has an *input alphabet* Σ, a finite set of *states* Q and an *initial state* $q_I \in Q$. The automaton also has a set of *counters* Γ, which is partitioned into a set Γ_B of *bounding counters* (we also say *B-counters* or *counters of type B*) and a set Γ_S of *unbounding counters* (we also say *S-counters* or *counters of* *type* S). With every letter $a \in \Sigma$ the automaton associates its *transition relation*:

$$\delta_a \subseteq Q \times \{\mathrm{i}, \mathrm{r}, \epsilon\}^\Gamma \times Q .$$

The intuition is that in a transition the automaton decides what do with each counter: whether to increment it (i), reset it (r), or leave it unchanged (ϵ). When the automaton only has counters of type B (resp. of type S), then the automaton is called an *ωB-automaton* (resp. an *ωS-automaton*). The counter values are never read by the automaton; they are only used for the acceptance condition.

A *run* ρ of an ωBS-automaton over some ω-word $a_1 a_2 \dots$ is a sequence of transitions $\rho = t_1 t_2 \cdots$ such that for every i, t_i belongs to δ_{a_i}, the source state of t_1 is q_I and for each i, the target state of the transition t_i is the same as the source state of the transition t_{i+1}. During such a run, the automaton updates the values of the counters. Initially, all counters have the value 0. A counter $c \in \Gamma$ is incremented when the transition assigns i to it, it is reset to 0 when the transition assigns r to it and it is left unchanged otherwise. For a run ρ, we denote by $c(\rho)$ the sequence of values that the counter c assumes *just before being reset*. This sequence can be finite if the counter is reset only a finite number of times, or it can be infinite. A *run* ρ is *accepting* if for every counter c, the sequence $c(\rho)$ is infinite and furthermore, if c is of type S then $c(\rho)$ is strongly unbounded and if c is of type B then $c(\rho)$ is bounded.

As an example, consider the following automaton with a single counter of type B (the counter action is in the parenthesis):



This automaton accepts the language $(a^B b(a^* b)^*)^\omega$. If the counter is of type S, then the same automaton accepts the language $(a^S b(a^* b)^*)^\omega$.

Though we do not prove it here, it should be fairly clear that no deterministic ωBS-automaton can accept these languages. For this reason, we are doomed to working with non-deterministic automata. This is one of the reasons why the complementation result is difficult.

3.3 Hierarchical automata

Hierarchical ωBS-automata are a more structured version of ωBS-automata where the counters are required to be nested. They are more closely related to ωBS-regular expressions than the general form of ωBS-automata.

An ωBS-automaton is called *hierarchical* if its set of counters is $\Gamma = \{1, \dots, n\}$ and whenever a counter $i > 1$ is incremented or reset, the counters $1 \dots i-1$ are reset. It is convenient to define for a hierarchical automaton its *counter type*, defined as a word in $\{B+S\}^*$. The length of this word is the number of counters; its i-th letter is the type of counter i.

According to this definition, a transition (q, v, r) in a hierarchical automaton can be of three forms:

- either $v(l) = \epsilon$ for every $l = 1, \dots, n$, or;
- there is some k such that $v(l) = r$ for $l = 1, \dots, k$, and $v(l) = \epsilon$ for $l = k+1, \dots, n$, or;
- there is some k such that $v(l) = r$ for $l = 1, \dots, k-1$, $v(k) = i$ and $v(l) = \epsilon$ for $l = k+1, \dots, n$.

4 Complementation

The main technical result of this paper is the following complementation theorem:

Theorem 4.1
The complement of an ωS-regular language is ωB-regular. The complement of an ωB-regular language is ωS-regular.

The proof of this result is long. Here, we just try to give some ideas underlying the proof. For the sake of the explanation, we only consider the case of complementing an ωS-regular language.

First consider the simple case of a language described by an ωS-automaton $\mathcal{A}$ which has a single counter, and such that in every run, between two resets, the increments of the counter are consecutive. This means that in every run, between two resets of the counter, the counter is first left unchanged during a while, then during the n-next following steps the counter is always incremented, then it is not incremented anymore before reaching the second reset. We call *increment interval* an interval of positions in the word corresponding to a maximal sequence of increments.

We now describe an ωB-automaton $\mathcal{B}$ accepting the complement of the language recognised by $\mathcal{A}$. It uses a single B-counter which beats as a clock dividing the input ω-word into pieces of bounded size (independantly from any run of $\mathcal{A}$). We say that an interval of positions in the word is *short* (with respect to this clock) if there is at most one beat of the clock in it. If the clock beats every n steps, then short intervals have length at most $n-1$. Reciprocally, if an interval has length at most n, then it is short with respect to every clock beating with a tempo greater than n. Using those remarks, we can see the notion of being short as a fair approximation of the length of an interval.

The complement automaton $\mathcal{B}$ works by guessing the beats of a clock using non-determinism together with a B-counter, and then checks the following fact: every run of $\mathcal{A}$ which contains an infinite number of resets, contains an infinite number of short increment intervals. Once the clock is fixed, checking this is definable in monadic second-order logic. Using this remark it is simple to construct $\mathcal{B}$.

It is easy to see that if $\mathcal{B}$ accepts an ω-word, then it is not accepted by $\mathcal{A}$. The converse implication requires to remark the following: if no run of $\mathcal{A}$ is accepting, then there exists a natural N such that in every run of $\mathcal{A}$ doing an infinite number of resets, there is less than N increments between two resets infinitely often. Such a property can be established using Ramsey-like arguments.

Let us turn now to the more general case of complementing a single counter ωS-automaton (we do not constrain anymore the increments to be contiguous). Our technique uses Simon's factorisation theorem for finite semigroups [9], and reduces the problem to a bounded number of instances of the above construction. In this case, the complement ωB-automaton uses one counter for each level of the factorisation, the result being a structure of nested clocks beating with different 'granularity'. As above, once the beats of the clocks are fixed, checking if a run makes few increments is approximable in monadic second-order logic. This makes it implementable by an ωB-automaton.

Finally, for treating the general case of ωS-automata, we use automata in their hierarchical form and perform an induction on the number of counters.

5 Monadic second-order logic with bounds

In this section, we introduce the logic MSOLB. This is a strict extension of monadic second-order logic (MSOL), where the new quantifier $\mathbb{U}$ is added (the original definition in [2] uses the quantifier $\mathbb{B}$ which is the negation of $\mathbb{U}$). This quantifier expresses the fact that a property is satisfied by arbitrarily large sets. We are interested in satisfiability: the decision problem whether there exists an ω-word modeling a given formula of MSOLB. We are not able to solve this problem in its full generality. However, the diamond properties allow us to provide an interesting partial solution.

In Section 5.1 we introduce formally the logic MSOLB. In Section 5.2 we explain how ωBS-regular languages can be used to deal with intersection, complementation and existential quantification in a decision procedure for satisfiability. In Section 5.3 we deal with the quantifier $\mathbb{U}$. Finally, in Section 5.4 we present an application of this logic to ω-automatic structures.

5.1 The logic

Recall that monadic second-order logic is an extension of first-order logic by set quantification. Hence a formula of this logic is made of atomic predicates, boolean connectives ($\wedge, \vee, \neg$), first-order quantification ($\exists x.\varphi$ and $\forall x.\varphi$) and monadic second-order quantification ($\exists X.\varphi$ and $\forall X.\varphi$) together with the membership predicate $x \in X$. Over ω-words the universe is the set $\mathbb{N}$ of positions, while the atomic predicates used are: a binary predicate $x \leq y$ for order on positions, and for each letter a of the alphabet, a unary predicate $a(x)$ that tests if a position x has the label a.

In the logic MSOLB we add a new quantifier: the *existential unbounding quantifier* $\mathbb{U}$ which has the following semantics:

$$\mathbb{U}X.\varphi := \forall N \in \mathbb{N}.\ \exists X.\ (\varphi \wedge |X| \geq N)\ .$$

The quantified variable X is a set variable and $|X|$ denotes its cardinality. Informally speaking, $\mathbb{U}X.\varphi(X)$ says that the formula $\varphi(X)$ is true for sets X of arbitrarily large cardinality. If $\varphi(X)$ is true for some infinite set X, then $\mathbb{U}X.\varphi(X)$ is immediately true.

From this quantifier, we can construct other meaningful quantifiers:

- The quantifier $\mathbb{A}$ — *the universal above quantifier* — is the dual of $\mathbb{U}$, i.e. $\mathbb{A}X.\varphi$ is a shortcut for $\neg\mathbb{U}X.\neg\varphi$. It is satisfied if all the sets X above a given threshold of cardinality satisfy property φ.
- Finally, the *bounding quantifier* $\mathbb{B}$ is syntactically equivalent to the negation of the $\mathbb{U}$ quantifier. This quantifier was the first chronologically studied[2]. It says that a formula $\mathbb{B}X.\varphi$ holds if there is a bound on the cardinality of sets satisfying property φ.

Over finite structures, MSOLB and MSOL are equivalent: the quantifiers $\mathbb{U}$ is always false over finite structures and consequently can be removed. Over infinite words, MSOLB defines strictly more languages than MSOL. For instance the formula

$$\mathbb{B}X.\ [\forall x \in X.a(x)] \wedge [\forall x \leq y \leq z.x, z \in X \rightarrow y \in X]$$

corresponds over $\{a,b\}^\omega$ to the language $(a^B.b)^\omega$. Indeed, the formula says there is a bound on the size of contiguous segments made of a's. As we have seen, this language is not regular. Hence, this formula is not equivalent to any MSOL formula. This motivates the following decision problem:

> Is a given formula of MSOLB satisfied over some infinite word?

We do not know the answer to this question in its full generality. However, using the diamond (Figure 1), we can solve this question for a certain class of formulas. This is the subject of the next section.

5.2 A decidable fragment of MSOLB

The classical approach for solving satisfiability of monadic second-order logic is to translate formulas into automata. To every operation in the logic corresponds a language operation. As automata happen to be closed under those operations, and emptiness is decidable for automata, the satisfaction problem is decidable for MSOL. We use the same approach for MSOLB. Unfortunately, our automata are not closed under complement, hence we do not solve the whole logic.

Those operations are summarized in the logical view of the diamond, i.e. Figure 2. Closures under $\vee$ and $\wedge$ are a direct consequence of closure under $\cup$ and $\cap$. Closure under $\exists$ corresponds to closure under projection, which is straightforward for non-deterministic automata. Negation $\neg$ is obtained by the closure under complementation. Closures under universal quantification follow as duals of the existential quantifications. Closure under $\mathbb{U}$ of ωS-regular langages is the subject of Section 5.3, while closure under $\mathbb{A}$ is obtained by duality. We did not represent the closure under the $\mathbb{B}$ quantifier on this picture. It would go from ωS-regular languages to ωB-regular languages.

Since emptiness for BS-regular languages is decidable by Fact 2.3, we obtain:

Theorem 5.1
The satisfiability problem is decidable for the following formulas:

- B-formulas. *These include all of MSOL, are closed under* $\vee, \wedge, \forall, \exists$ *and* $\mathbb{A}$*. Moreover, the negation of an S-formula is a B-formula.*
- S-formulas. *These include all of MSOL, are closed under* $\vee, \wedge, \forall, \exists$ *and* $\mathbb{U}$*. Moreover, the negation of a B-formula is an S-formula.*
- BS-formulas. *These include all B-formulas and S-formulas, and are closed under* $\vee, \wedge, \exists$ *and* $\mathbb{U}$*.*

All ωBS-regular languages can be described by an MSOLB formulas:

Fact 5.2 Every ωBS-regular language (resp. ωB-regular, resp. ωS-regular) is definable by a BS-formula (resp. a B-formula, resp. an S-formula).

Proof
Guess a run of the automaton, and check that this run is accepting using the new quantifiers. □

The converse fails for ωBS-regular language since these languages are not closed under complementation. But it holds for ωB-regular and ωS-regular languages.

5.3 Closure under existential unbounding quantification

Here we show that the classes of ωS- and ωBS-regular languages are closed under application of the quantifier $\mathbb{U}$. This closure is settled by Proposition 5.3.

Before we proceed, we describe the quantifier $\mathbb{U}$ as a language operation, in the same way as existential quantification corresponds to projection. Let Σ be an alphabet, and consider a language $L \subseteq (\Sigma \times \{0,1\})^\omega$. Given a word $w \in \Sigma^\omega$ and a set $X \subseteq \mathbb{N}$, let $w[X] \in (\Sigma \times \{0,1\})^\omega$ be the word obtained from w by setting the second coordinate to 1 on the positions from X and to 0 on the other positions. We then define $\mathbb{U}(L)$ to be the set of those words $w \in \Sigma^\omega$ such that for every $N \in \mathbb{N}$ there is a set $X \subseteq \mathbb{N}$ of at least N elements such that $w[X]$ belongs to L.

Proposition 5.3 Both ωS and ωBS-regular languages are closed under the operation $\mathbb{U}(L)$.

We begin with a simple auxiliary result. A *partial sequence* over an alphabet Σ is a word in $\perp^*\Sigma^\omega$. A partial sequence is *defined* on the positions where it does not have value $\perp$. We say two partial sequences *meet* if there is some position where they are both defined and have the same letter.

Lemma 5.4 Let I be an infinite set of partial sequences over a finite alphabet. There is a partial sequence in I that meets infinitely many partial sequences from I.

Proof
Let Σ be the finite alphabet. A *constrainer* for I is an infinite word c over $P(\Sigma)$ such that the i-th position of every sequence in I is either undefined or belongs to c_i. The size of a constrainer is the maximal size of a set it uses infinitely.

The proof is by induction over the size of a constrainer for I. This is sufficient since every set I admits the constrainer that has Σ on every coordinate. If I admits a constrainer of size 1 then we are done. Take a set I with a constrainer c of size n. Take some sequence s in I. If s meets infinitely many sequences from I, then we are done. Otherwise let $J \subseteq I$ be the (infinite) set of sequences that do not meet s. Then one can verify that d is a constrainer for J, where d is defined by $d_i = c_i \setminus \{s_i\}$. Moreover, d is of size $n-1$. □

Let L be a language of infinite words over $\Sigma \times \{0,1\}$ recognized by an ωBS-automaton. We want to show that the language $\mathbb{U}(L)$ is also recognized by a bounding automaton. Consider the following language:

$$K = \{w[X] : w[Y] \in L, \text{ for some } X \subseteq Y\} .$$

This language is downward closed in the sense that if $w[X]$ belongs to K, then $w[Y]$ belongs to K for every $Y \subseteq X$.

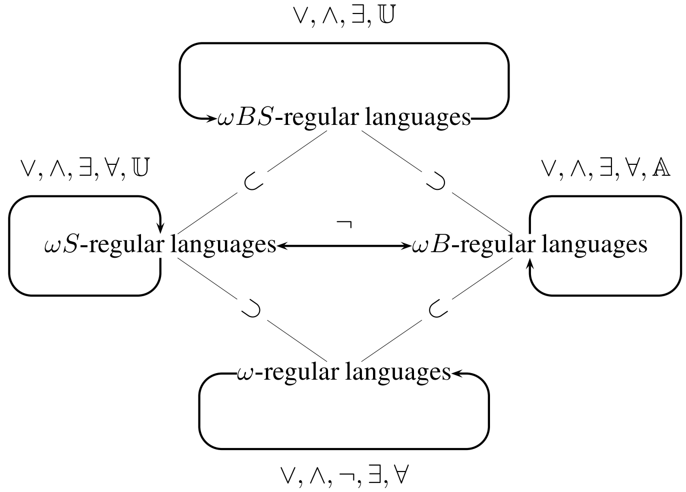


Figure 2. Logical view of the diamond

Furthermore, clearly $\mathbb{U}(L) = \mathbb{U}(K)$. Moreover, if L is recognized by a ωBS-automaton (resp. ωS-automaton), then so is K. Let $\mathcal{A}$ be an ωBS-automaton recognizing K. We will construct a ωBS-automaton recognizing $\mathbb{U}(K)$.

Given a word $w \in \Sigma^\omega$, a sequence of sets $X_1, X_2, \ldots \subseteq \mathbb{N}$ is *an unbounding witness* for K if for every i, the word $w[X_i]$ belongs to K and the sizes of the sets are unbounded. An unbounding witness is *sequential* if there is a sequence of numbers $a_1 < a_2 < \cdots$ such that all members of X_i are between a_i and a_{i+1}.

The following lemma is a simple application of the property of K being downward closed.

Lemma 5.5 A word admitting an unbounding witness for K admits a sequential one.

Let $X_1, X_2, \ldots$ be a sequential unbounding witness and let $a_1 < a_2 \cdots$ be the appropriate sequence of numbers. Let $\rho_1, \rho_2, \ldots$ be accepting runs of the automaton $\mathcal{A}$ over the words $w[X_1], w[X_2], \ldots$ Such runs exist by definition of unbounding witness. The sequence $X_1, X_2, \ldots$ is a *good witness* if every two runs ρ_i and ρ_j agree on almost all positions.

Lemma 5.6 A word belongs to $\mathbb{U}(K)$ if and only if it admits a good witness.

Proof
By Lemma 5.5, a word belongs to $\mathbb{U}(K)$, if and only if it admits a sequential witness. For every i, let s_i be the partial sequence that has $\perp$ on positions before a_{i+1} and agrees with ρ_i after a_{i+1}. By applying Lemma 5.4 to the set $\{s_1, s_2 \ldots\}$, we can find a run ρ_i and a set $J \subseteq \mathbb{N}$ such that for every $j \in J$, the runs ρ_i and ρ_j agree on some position x_j after a_{j+1}. For $j \in J$, let ρ'_j be a run that is defined as ρ_j on positions before x_j and is defined as ρ_i on positions after x_j. Since modifying the counter values over a finite set of positions does not violate the acceptance condition, the run ρ'_j is also an accepting run over the word $w[X_j]$. For every $j, k \in J$, the runs ρ'_j and ρ'_k agree on almost all positions (i.e. positions after both x_j and x_k). Therefore the witness obtained by using only the sets X_j with $j \in J$ is a good witness. □

Lemma 5.7 Words admitting a good witness can be recognized by a bounding automaton.

Proof
Given a word w, the automaton is going to guess a sequential witness

$$a_1 < a_2 < \cdots \qquad X_1, X_2, \ldots \subseteq \mathbb{N}$$

and a run ρ of $\mathcal{A}$ over w and verify the following properties:

- The run ρ is accepting;
- There is no bound on the size of the X_i's;
- For every i, some run over $w[X_i]$ agrees with ρ on almost all positions.

The first property can be obviously verified by a ωBS-automaton. For the second property, the automaton nondeterministically chooses a subsequence of $X_1, X_2, \ldots$ where the sizes are strongly unbounded. The third property is a regular property. The statement of the lemma then follows by closure of bounding automata under projection and intersection. □

5.4 An example: unbounded out-degree

Let $\varphi(X, Y)$ be a formula of MSOLB with two free set variables. This formula can be seen as an edge relation on sets. We show here that MSOLB can be used to say that this edge relation has unbounded out-degree.

We begin by defining the notion of an X-witness. This is a set witnessing that there are many successors of the set X under φ. (The actual successors of X form a set of sets, something MSOLB cannot talk about directly.) An X-*witness* is a set Y such that every two elements $x, y \in Y$ can be separated by a successor of X, that is:

$$\forall x, y \in Y \; \exists Z. \varphi(X, Z) \wedge (x \in Z \Leftrightarrow y \notin Z) \; .$$

(Therefore being an X-witness can be defined by an MSOLB formula.) We claim that the graph of φ has unbounded out-degree if and only if there are X-witnesses of arbitrarily large cardinality (for different sets X). This claim follows from the following fact:

Fact 5.8 If X has more than 2^n successors, then it has an X-witness of size at least n. If X has n successors, then all X-witnesses have size at most 2^n.

Proof
For the first statement, we first show that X has at least n successors that are boolean independent (none is a boolean combination of the others). From n boolean independent successors one can then construct by induction an X-witness of size n.

For the second statement, consider X with n successors as well as an X-witness. To each element w of the X-witness, associate the characteristic function of '$w \in Y$' for Y ranging over the successors of X. If the X-witness had more than 2^n elements, then at least two would give the same characteristic function, contradicting the definition of an X-witness. $\square$

An ω-automatic graph is one where each vertex is a set of naturals, and the edge relation is defined by a formula $\varphi(X, Y)$ of MSOL over the naturals with successor (see [1]). In this particular case, the existence of arbitrarily large X-witnesses is expressed by a formula that belongs to one of the classes with decidable satisfiability from Theorem 5.1. This shows:

Proposition 5.9 It is decidable if an ω-automatic graph has unbounded out-degree.

6 Future work

We conclude the paper with some open questions.

As we have defined them, ωBS-regular languages are not closed under complementation. Can we find a larger class that is? What are the appropriate automata?

Are there natural deterministic automata? The automata in this paper seem to be inherently nondeterministic.

Our complementation proof is very complicated. It would be worthwhile to find a simpler version. In particular, the computational complexity of the construction could be reduced. In the present version, a single complementation step gives a non-elementary blowup of the automaton's state space.

Are there other meaningful and decidable extensions of monadic second-order logic? For instance does adding the predicate "the set of positions X is ultimately periodic" lead to an undecidable logic? (This predicate can be used to define the language $(ab^B)^\omega$.)

Is there an algebraic model for ωBS-regular languages? Can ω-semigroups be appropriately extended? Is there a link with tropical semirings [8]?

Is there a corresponding (decidable) temporal logic?

References

[1] A. Blumensath and E. Grädel. Finite presentations of infinite structures: Automata and interpretations. *Theory of Computing Systems*, 37:641 – 674, 2004.

[2] M. Bojańczyk. A bounding quantifier. In *Computer Science Logic*, volume 3210 of *Lecture Notes in Computer Science*, pages 41–55, 2004.

[3] A. Bouquet, O. Serre, and I. Walukiewicz. Pushdown games with unboundedness and regular conditions. In *Foundations of Software Technology and Theoretical Computer Science*, volume 2914 of *Lecture Notes in Computer Science*, pages 88–99, 2003.

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

[5] J. R. Büchi. On a decision method in restricted second-order arithmetic. In *Proc. 1960 Int. Congr. for Logic, Methodology and Philosophy of Science*, pages 1–11, 1962.

[6] F. Klaedtke and H. Ruess. Parikh automata and monadic second–order logics with linear cardinality constraints. Technical Report 177, Institute of Computer Science at Freiburg University, 2002.

[7] M. O. Rabin. Decidability of second-order theories and automata on infinite trees. *Transactions of the AMS*, 141:1–23, 1969.

[8] I. Simon. Recognizable sets with multiplicites in the tropical semiring. In *Mathematical Foundations of Computer Science*, volume 324 of *Lecture Notes in Computer Science*, pages 107 – 120, 1988.

[9] I. Simon. Factorization forests of finite height. *Theoretical Computer Science*, 72:65 – 94, 1990.

[10] W. Thomas. Languages, automata, and logic. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Language Theory*, volume III, pages 389–455. Springer, 1997.