Deciding Satisfiability for Fragments with Unary Predicates and Difference Arithmetic

Bernard Boigelot[1], Pascal Fontaine[1] and Baptiste Vergain[1]

[1]*Montefiore Institute, B28, Université de Liège, Belgium.*

Abstract

We report our work in progress[2] to build decision procedures for highly expressive languages mixing arithmetic and uninterpreted predicates. More precisely, we study a language combining difference-logic constraints and unary predicates. The decision problem is impacted by the domain of interpretation $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$ or $\mathbb{R}$. For the integer domains $\mathbb{N}$ and $\mathbb{Z}$, the problem reduces to deciding the monadic second-order theory of $\mathbb{N}$ with the successor relation (S1S), which is known to be feasible by translating formulas into automata, and then checking the emptiness of their accepted language. We also advocate the use of automata as an appropriate tool to represent the set of models for the real domain $\mathbb{R}$.

Keywords

Satisfiability, First-order logic, Unary uninterpreted predicates, Difference logic

1. Introduction

SMT (satisfiability modulo theories) solving has been very successfully used in various applications, most notably in verification (see e.g., [1]). Most SMT solvers were conceived as decision procedures for quantifier-free fragments including interpreted symbols and arithmetic operators. Support for quantifiers was mainly based on heuristics. Although some techniques in SMT solvers (e.g., [2, 3, 4]) were later introduced to reach decidability for quantified but purely arithmetic fragments, that is, without uninterpreted predicates, there has been little attention to the problem of decidability of quantified fragments mixing uninterpreted symbols and arithmetic.

There is a simple explanation to this: mixing arithmetic and predicate symbols quickly leads to undecidability. For instance, Presburger arithmetic [5] is decidable, but adding a single ternary uninterpreted predicate symbol (or a binary function) to the language augments the expressive power to Peano arithmetic, since this uninterpreted symbol can be used to model multiplication. In fact, even adding a single unary uninterpreted predicate to Presburger arithmetic makes it undecidable [6]. It thus seems hopeless to find interesting decidable fragments with uninterpreted predicates for arbitrarily quantified formulas, on the basis of full linear arithmetic on integers.

[2]Our knowledge on this matter has significantly improved since this paper was written. An update on this work will be provided in a following article.

SC^2 *2021: 6th International Workshop on Satisfiability Checking and Symbolic Computation, August 19–20, 2021*

Bernard.Boigelot@uliege.be (B. Boigelot); Pascal.Fontaine@uliege.be (P. Fontaine); bvergain@uliege.be (B. Vergain)

However, if we weaken the expressiveness of arithmetic, there exist decidable quantified fragments with uninterpreted symbols, without restrictions on quantifiers. In Section 4 we recall that Büchi automata (a type of finite automata working on infinite words) provide a decision procedure for the monadic second-order theory of $\mathbb{N}$ with the successor relation (usually referred to as S1S), a language which is essentially equivalent to difference logic with uninterpreted predicates on the domain of the natural numbers $\mathbb{N}$. In this extended abstract, we report on our work in progress to study the decidability of difference logic with uninterpreted predicates on various domains. For the domains of natural numbers $\mathbb{N}$ and integer numbers $\mathbb{Z}$, this easily reduces to checking the emptiness of the language described by Büchi automata. For the real numbers however, the problem is significantly more complicated. We provide here some insights and directions to solve it. Our motivation is to precisely draw the frontier of decidability for satisfiability checking in presence of uninterpreted symbols and arithmetic.

After some prerequisites and notations in Section 2, we introduce in Section 3 the considered language and informally discuss its expressive power. Section 4 provides a decision procedure for this language when the domain is interpreted as the set of natural numbers $\mathbb{N}$, restating a well established equivalence with checking the emptiness of the language accepted by Büchi automata. The following section adapts this result to the case of integers $\mathbb{Z}$. We finally discuss in Section 6 our work in progress on this language interpreted over the real numbers $\mathbb{R}$, before concluding.

2. Preliminaries

We refer to e.g. [7] for a general introduction to first-order logic with equality, and assume that the reader is familiar with the notions of signature, term, variable, and formula. We use the usual logical connectives ($\vee$, $\wedge$, $\neg$, $\Rightarrow$, $\equiv$) and first-order quantification ($\exists x.\, \varphi$ and $\forall x.\, \varphi$). Our signature essentially contains arithmetic symbols 0, 1, $+$, $-$, $<$, $\leq$, $\geq$, $>$, and uninterpreted symbols P, Q,.... A constant in $\mathbb{N}$ stands for a term $1 + 1 + \ldots 1$. In the following, we only consider unary predicate symbols. The set of all formulas built using variables and symbols from a signature is the language of formulas. An interpretation specifies a domain (i.e., a set of elements), assigns a value in the domain to each free variable, and assigns function and predicate relations of appropriate arity to the function and predicate symbols in the signature. We only consider interpretations whose domain is $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$ or $\mathbb{R}$. The arithmetic symbols are interpreted on the chosen domain, that is, an interpretation always assigns its usual meaning on the domain to each arithmetic symbol. The uninterpreted symbols are left free, that is, an interpretation can assign to a unary predicate an arbitrary subset of the domain on which the predicate is true, the predicate being false on the complement of this set in the domain.

By extension, an interpretation assigns a value in the domain to every term, and a truth value to every formula. A model is an interpretation that assigns true to the formula. A formula is satisfiable on a domain ($\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$ or $\mathbb{R}$) if it has a model on that domain.

A fragment is a first-order language of formulas, possibly with further syntactical restrictions, and a domain of interpretation. We say that a fragment is decidable if there is an effective procedure to check whether a given formula in the language is satisfiable on this domain.

3. Difference arithmetic with unary predicates

We consider a first-order language of formulas, built using usual logical connectives ($\vee$, $\wedge$, $\neg$, $\Rightarrow$, $\equiv$) and first-order quantification ($\exists x.\, \varphi$ and $\forall x.\, \varphi$). Atoms are either difference-logic constraints of the form $x - y \bowtie c$, where x and y are variables, c is a constant in $\mathbb{N}$, and $\bowtie \in \{<, \leq, =, \geq, >\}$, or unary predicate formulas $P(x)$, where P is an uninterpreted predicate symbol and x is a variable. The formulas are interpreted on a given domain, equal to either $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$ or $\mathbb{R}$. For the domains $\mathbb{Z}$ and $\mathbb{R}$, our fragments are actually subsets of the SMT-LIB logics *UFIDL* and *UFRDL* [8], i.e., the logics that mix *Uninterpreted Functions* and *Difference Logic* over the *Integers* or the *Reals*. Since they only allow unary predicates, we call our fragments *UF1NDL*, *UF1IDL*, *UF1QDL* and *UF1RDL* when the domain of interpretation is $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$ and $\mathbb{R}$ respectively.

These fragments are actually quite expressive, as we illustrate now:

- $P(0) \wedge \neg P(1) \wedge \forall x \forall y\,.\, (x - y = 2) \Rightarrow (P(x) \equiv P(y))$
  Over either $\mathbb{N}$ or $\mathbb{Z}$, this formula states that the predicate P is true for all even numbers.
- $\forall x \forall y \exists z\,.\, x < y \Rightarrow (x < z \ \wedge\ z < y \ \wedge P(z))$
  This formula enforces the set of values that satisfy P to be dense within the domain of interpretation. It is obviously unsatisfiable if the domain itself is not dense, e.g., for the domains $\mathbb{N}$ and $\mathbb{Z}$.
- $\forall x \forall y \exists z\,.\, x < y \Rightarrow (x < z \ \wedge\ z < y)$
  Since no predicate variable is used here, this sentence is valid if and only if the domain is dense, which holds for $\mathbb{Q}$ and $\mathbb{R}$.
- $\forall x \forall y \exists u \exists v\,.\, x < y \Rightarrow (x < u < y \ \wedge\ x < v < y \ \wedge P(u) \wedge \neg P(v))$
  This property (that we call *chaoticity*) states that both the predicate P and its complement $\neg P$ are dense within the domain. In other words, there does not exist a non-empty open interval of the domain in which P is either uniformly true or uniformly false.
- $(\exists i \forall x.\, P(x) \Rightarrow i < x)$
  $\wedge\ (\forall b.\, (\forall x.\, P(x) \Rightarrow b \leq x) \Rightarrow (\exists y.\, (\forall x.\, P(x) \Rightarrow y \leq x) \wedge (y > b)))$
  This states that the subset described by P admits a lower bound, but no maximal lower bound; this partly expresses the negation of the *Dedekind completeness* property [9].

Notice that this logical formalism is sufficiently expressive to discriminate between the various domains of interpretation. Namely, the sentence describing the density of the domain is false on $\mathbb{N}$ and $\mathbb{Z}$, and true on $\mathbb{Q}$ and $\mathbb{R}$. It is easy to imagine a formula discriminating $\mathbb{N}$ from $\mathbb{Z}$ by stating the existence of a smaller element. We are able to discriminate between $\mathbb{Q}$ and $\mathbb{R}$ thanks to the last property.

4. The fragment *UF1NDL*

4.1. Satisfiability

The satisfiability problem for *UF1NDL*, i.e., our fragment interpreted over the natural numbers, reduces to the satisfiability of S1S, that is, the monadic second-order theory of $\mathbb{N}$ with the successor relation [10]. Indeed, the order relation $x \leq y$ can be reconstructed as follows in S1S:

$$\forall X\,.\, (X(x) \wedge \forall z\,.\, X(z) \Rightarrow X(z+1)) \Rightarrow X(y)$$

It is also known that the language of models of a given S1S formula is recognizable by an infinite-word automaton. Every formula of S1S can indeed be translated by structural induction into a Büchi automaton that represents the set of its models [11]. Checking the satisfiability of a given S1S formula therefore reduces to checking the emptiness of the language accepted by the corresponding Büchi automaton, which is decidable [11, 10]. This shows that satisfiability is decidable for *UF1NDL*.

4.2. From *UF1NDL* formulas to infinite-word automata

A related but slightly different way of deciding the satisfiability of the fragment *UF1NDL* consists in translating directly *UF1NDL* formulas into infinite-word automata representing their models. We now provide some details on this translation procedure.

A predicate $P : \mathbb{N} \to \{\top, \bot\}$ can be encoded by an infinite word $a_0 a_1 a_2 \ldots \in \{0,1\}^\omega$, where for all $i \in \mathbb{N}$, $a_i = 1$ iff $P(i) = \top$. Similarly, the value of a variable $x \in \mathbb{N}$ can be encoded by the word $0^x 10^\omega$. For a formula $\varphi(P_1, \ldots, P_n, x_1, \ldots, x_m)$ with predicates $P_1, \ldots, P_n$ and free variables $x_1, \ldots, x_m$, an interpretation of these variables and predicates over $\mathbb{N}$ is encoded by an infinite word over the alphabet $\{0,1\}^{n+m}$, every symbol of which provides one digit of the encoding of each predicate and each variable value.

For a given *UF1NDL* formula φ, the set of encodings of the models of φ forms a language $L(\varphi)$. It is easily shown that this language is ω-regular, and that a Büchi automaton that accepts this language can effectively be built, by proceeding by induction on the structure of φ. The atoms $x_i - x_j \bowtie c$ and $P_i(x_j)$ directly translate into elementary automata. Then, similarly to the translation of S1S formulas into automata, one applies Boolean connectives and quantifiers by means of automata manipulations that essentially amount to performing the same operation on their accepted languages [11]. The problem of deciding whether φ is satisfiable then becomes equivalent to checking whether the resulting automaton accepts a non-empty language.

This decision procedure has the advantage of simplicity, but requires to be able to complement Büchi automata, which is a costly operation in practice [12, 13].

4.3. From infinite-word automata to *UF1NDL* formulas

The translation procedure from *UF1NDL* to Büchi automata sketched in Section 4.2 suffices to establish that satisfiability is decidable for *UF1NDL*. It is however worth mentioning that the reverse translation is also possible, which shows that *UF1NDL* and Büchi automata share the same expressive power.

More precisely, given a Büchi automaton accepting a language L, one can build a formula of *UF1NDL* that is satisfiable iff L is not empty. The main principle of this translation consists in employing the appropriate number of predicates for encoding the sequence of states visited by the automaton during a run. One then constructs a formula that checks that every step in this sequence satisfies the transition relation of the automaton, that its first element corresponds to the initial state of the automaton, and that it describes an accepting run [10].

5. The fragment *UF1IDL*

The decision procedure for *UF1NDL* outlined in Section 4.2 straightforwardly generalizes to the integer domain. The only difficulty is to design a scheme for encoding values in $\mathbb{Z}$ and predicates over $\mathbb{Z}$ as infinite words.

A solution consists in encoding $k \in \mathbb{Z}$ such that $k \geq 0$ as $0^{2k}10^{\omega}$, and $k < 0$ as $0^{-2k-1}10^{\omega}$. The same idea can be adapted to represent predicates. With this encoding, the atoms $x_i - x_j \bowtie c$ and $P_i(x_j)$ translate into elementary automata that are identical or only slightly more complex than the ones obtained for *UF1NDL*. The same approach then reduces the decidability of *UF1IDL* to the problem of checking whether the automaton constructed from a given formula accepts a non-empty language.

6. The fragment *UF1RDL*

We now address the problem of deciding satisfiability for *UF1RDL* formulas, i.e., our fragment interpreted over the domain $\mathbb{R}$. As a first step, we only consider a restricted fragment that we call *Weak UF1RDL* (*WUF1RDL*), in which

- each formula involves at most one unary uninterpreted predicate P,
- difference-logic atoms are of the form $x - y \bowtie 0$, i.e., the only constant allowed is 0.

Notice that even with such restrictions, all the properties illustrated in Section 3 but the first one remain expressible.

Our goal is to follow the same approach as in Sections 4 and 5 for the domains $\mathbb{N}$ and $\mathbb{Z}$, which consists in designing an encoding scheme for the predicate and variable values, building a symbolic representation of the set of models of a given formula, and checking whether this set is empty or not.

6.1. Encoding predicates

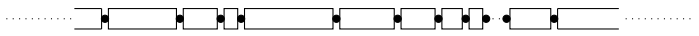
Unlike the case of the integer domains $\mathbb{N}$ and $\mathbb{Z}$, a predicate $P : \mathbb{R} \to \{\top, \bot\}$ cannot be unambiguously encoded as an infinite word $a_0a_1a_2\ldots$ over some finite alphabet. Such a precise representation of P is however not needed. Indeed, the fragment *WUF1RDL* does not make it possible to express constraints about the absolute difference between real values, but only about their relative order. This means that if two predicates P and P' admit characteristic sets $\{x \in \mathbb{R} \mid P(x)\}$ and $\{x \in \mathbb{R} \mid P'(x)\}$ whose elements are ordered in the same way over $\mathbb{R}$, then for every formula φ of *WUF1RDL* involving one predicate and m free variables, $\varphi(P, x_1, \ldots, x_m)$ is satisfiable iff $\varphi(P', x_1, \ldots, x_m)$ is satisfiable as well.

Instead of designing an encoding scheme that precisely describes a given predicate, we thus opt for an abstract encoding relation that maps multiple predicate interpretations onto the same representation. Our goal is to design this relation in such a way that if two predicates P and P' share the same representation, then for every formula φ of *WUF1RDL* involving one predicate and m free variables, $\varphi(P, x_1, \ldots, x_m)$ will be satisfiable iff $\varphi(P', x_1, \ldots, x_m)$ is satisfiable as well.

Consider an arbitrary predicate $P : \mathbb{R} \to \{\top, \bot\}$. We define the set $T_P \subseteq \mathbb{R}$ as the set of all real values that belong to an open interval in which P is uniformly true. Similarly, we define $F_P \subseteq \mathbb{R}$ as the set of all real values that belong to an open interval in which P is uniformly false. Finally, we define $C_P \subseteq \mathbb{R}$ as the set of all real values that belong to an open interval that has an empty intersection with $T_P \cup F_P$. In such an interval, the behavior of P is necessarily chaotic (cf. Section 3), since this interval cannot contain a sub-interval of non-zero length in which P has a constant value.

Each of the sets T_P, F_P and C_P is thus equal to a union of non-empty open intervals in $\mathbb{R}$, with the additional property that those intervals do not overlap. This union is therefore necessarily countable [14], since each of them contains a rational value that does not belong to the others. Let I_P denote the set of those intervals. The set $T_P \cup F_P \cup C_P$ entirely covers $\mathbb{R}$ except for a (possibly uncountable) number of discrete points, that we gather in a set E_P. The elements of E_P correspond to the endpoints of the intervals in I_P, together with the limits of all convergent sequences of such endpoints.

The set $I_P \cup E_P$ thus forms a totally ordered partition of $\mathbb{R}$. The following picture depicts such a partition, in which the elements of I_P are shown as rectangles and those of E_P as black dots. It is possible that an infinite number of intervals of I_P appear within a bounded interval of the real line; the smaller dots appearing between two black dots in the right part of the picture illustrate such a behavior.



In order to obtain an abstract representation for the predicate P, we use the notion of word over linear orderings introduced in [15], that generalizes the concept of infinite word. Given a totally ordered set X and a finite alphabet Σ, a word over X is a mapping from the ordering of X to Σ, i.e., a function that associates each element of the ordering with an element of the alphabet.

For a predicate $P : \mathbb{R} \to \{\top, \bot\}$, we define the abstract encoding of P as the word w over the totally ordered set $I_P \cup E_P$ that maps every element of I_P onto the alphabet $\{T, F, C\}$, and every element of E_P onto $\{\top, \bot\}$. For every $I \in I_P$, one has $w(I) = T$ if $I \subseteq T_P$, $w(I) = F$ if $I \subseteq F_P$, and $w(I) = C$ if $I \subseteq C_P$. For every $x \in E_P$, one has $w(x) = \top$ if $P(x) = \top$ and $w(x) = \bot$ if $P(x) = \bot$.

6.2. Encoding variables

The abstract encoding of variable values must contain enough information to represent their relative order, as well as their relation with the predicate P. We encode each variable value as a word over the same totally ordered set $I_P \cup E_P$ as for the predicate P, or over an arbitrarily chosen ordering in the case of a formula with no predicate. For a variable, every element of I_P is mapped onto a single symbol F, and every element x of E_P onto the alphabet $\{\top, \bot\}$, such that $w(x) = \top$ iff the value of the variable is equal to x.

With this encoding scheme, one cannot encode a variable value that does not belong to E_P. This is not problematic, since intervals can easily be split. If one needs to express that a variable has a value that belongs to an interval $(a, b) \in I_P$, this interval can be replaced in I_P by both

(a, c) and (c, b), where c is any value such that $a < c < b$, and c is then added to E_P. Since the number of variables is finite, this operation has only to be carried out finitely many times.

Finally, just like in the case of the integer domains $\mathbb{N}$ and $\mathbb{Z}$, the abstract encodings of the predicates and all variables can be combined into a single word over the totally ordered set $I_P \cup E_P$. Each element of I_P is mapped onto the alphabet $\{T, F, C\} \times \{F\}^m$, where m is the number of variables, and each element of E_P is mapped onto $\{\top, \bot\}^{1+m}$.

6.3. Representing the set of models of a formula

A notion of automaton operating on words over any total ordering (hence possibly uncountable ones) has been introduced in [15]. We conjecture that this formalism makes it possible to represent symbolically the set of models of any *WUF1RDL* formula. More precisely, we aim at establishing that for every formula φ with one predicate and m free variables:

- for every predicates P and P' that share the same abstract encoding (cf. Section 6.1), $\varphi(P, x_1, \ldots, x_m)$ is satisfiable iff $\varphi(P', x_1, \ldots, x_m)$ is satisfiable as well,
- the set of encodings of the models of φ can be recognized by an automaton over total orderings, and such an automaton can effectively be built by structural induction on φ,
- the formula φ is satisfiable iff its corresponding automaton accepts a non-empty language.

Those properties hold for the atomic formulas of *WUF1RDL*. In order to be able to apply Boolean connectives and to perform quantification, one needs to provide corresponding operations on the automata representing the set of models of the formula being handled, and show that these operations preserve the properties of interest. The main difficulty revolves around the complementation operation, which is needed in particular for handling universal quantification.

The problem of complementing automata over total orderings has been investigated in [16] and [17]. A solution has been obtained for automata operating over a restricted class of total orderings called *finite-rank* orderings. Whether this class is suited for representing models of *WUF1RDL* formulas remains an open problem, the question being to establish that for every formula φ, the set of finite-rank models of φ is non-empty iff φ is satisfiable. Our preliminary results about this problem are promising. Finally, the problem of checking the emptiness of the language accepted by an automaton over a total ordering is solved in [18].

7. Conclusions and future work

In this work, we have investigated highly expressive first-order languages that stand close to the frontier of decidability. These fragments mix difference-logic constraints and uninterpreted unary predicates, and their expressive power varies with their domain of interpretation.

For the domains of natural and integer numbers, there exists a natural translation of formulas expressed in the corresponding fragments into finite automata. Such automata can effectively be built by structural induction over the formulas, and provide a symbolic representation of their set of models. Checking whether a given formula is satisfiable amounts to checking whether the language accepted by the corresponding automaton is empty or not.

The case of the rational and real domains is more involved, since the expressive power of even a single unary predicate over a dense domain makes it possible to describe structures that are more complex than for the natural and integer domains. One could be startled by this, since arithmetic theories over the reals are very often simpler to handle than those over the integers, and usually admit decision procedures of lower complexity [19]. This is not the case for the fragments studied here, for which it actually turns out that the problem of deciding their satisfiability over $\mathbb{N}$ or $\mathbb{Z}$ reduces to the same problem over $\mathbb{R}$, and not the other way around. Indeed, one can easily define a predicate over $\mathbb{R}$ that is only true for isolated values, and express the property that each of those values admit exactly one successor, which essentially allows to simulate natural or integer numbers with real variables.

We have presented our work in progress aimed at developing a decision procedure for the logic *UF1RDL*. Our approach relies on the automata over total orderings introduced in [15]. Our first-priority goal is to establish that these automata provide suitable symbolic representations for the sets of models of formulas that involve a single unary predicate and difference-logic constraints of the form $x - y \bowtie 0$.

The generalization of this result to formulas with multiple predicates seems to be straightforward; it essentially amounts to synchronizing the abstract encodings of all predicates with respect to the same total order, and adding equality and disequality constraints over them in each elementary interval of the representation. Allowing constraints of the form $x - y \bowtie c$ for arbitrary values of c seems to be more difficult, and would require to adapt the encoding of predicate and variable values so as to be able to reason about the absolute difference between values.

Our motivation is here to precisely draw the frontier of decidability for satisfiability checking of formulas containing uninterpreted symbols and arithmetic. This could also give us insights for quantifier elimination, or rather more generally for handling quantifiers for fragments mixing uninterpreted symbols and arithmetic.

Finally, another perspective would be to investigate the possibility of lifting our results to the domain of rational numbers.

References

[1] C. Barrett, D. Kroening, T. Melham, Problem Solving for the 21st Century: Efficient Solvers for Satisfiability Modulo Theories, Technical Report 3, London Mathematical Society and Smith Institute for Industrial Mathematics and System Engineering, 2014. URL: http://theory.stanford.edu/~barrett/pubs/BKM14.pdf, Knowledge Transfer Report.

[2] P. Rümmer, A constraint sequent calculus for first-order logic with linear integer arithmetic, in: I. Cervesato, H. Veith, A. Voronkov (Eds.), Proceedings, 15th International Conference on Logic for Programming, Artificial Intelligence and Reasoning, volume 5330 of *Lecture Notes in Computer Science*, Springer, 2008, pp. 274–289.

[3] N. Bjørner, Linear quantifier elimination as an abstract decision procedure, in: J. Giesl, R. Hähnle (Eds.), Automated Reasoning, 5th International Joint Conference, IJCAR 2010, Edinburgh, UK, July 16-19, 2010. Proceedings, volume 6173 of *Lecture Notes in Computer Science*, Springer, 2010, pp. 316–330.

[4] E. Ábrahám, J. H. Davenport, M. England, G. Kremer, Deciding the consistency of non-linear real arithmetic constraints with a conflict driven search using cylindrical algebraic coverings, Journal of Logical and Algebraic Methods in Programming 119 (2021) 100633.
[5] M. Presburger, On the completeness of a certain system of arithmetic of whole numbers in which addition occurs as the only operation, History and Philosophy of Logic 12 (1991) 225–233. English translation of the original paper from 1929.
[6] J. Y. Halpern, Presburger arithmetic with unary predicates is Π_1^1 complete, The Journal of Symbolic Logic 56 (1991) 637–642.
[7] H. B. Enderton, A mathematical introduction to logic, Elsevier, 2001.
[8] C. Barrett, P. Fontaine, C. Tinelli, The Satisfiability Modulo Theories Library (SMT-LIB), www.SMT-LIB.org, 2016.
[9] E. J. McShane, Partial orderings and Moore-Smith limits, The American Mathematical Monthly 59 (1952) 1–11.
[10] W. Thomas, Languages, automata, and logic, in: Handbook of formal languages, Springer, 1997, pp. 389–455.
[11] J. R. Büchi, On a decision method in restricted second order arithmetic, Logic, Methodology and Philosophy of Science, 1962.
[12] S. Safra, Complexity of automata on infinite objects, Ph.D. thesis, Weizmann Institute of Science, 1989.
[13] A. P. Sistla, M. Y. Vardi, P. Wolper, The complementation problem for Büchi automata with applications to temporal logic, Theoretical Computer Science 49 (1987) 217–237.
[14] W. Sierpiński, Cardinal and ordinal numbers (2nd edition), PWN, Warszawa, 1965.
[15] V. Bruyère, O. Carton, Automata on linear orderings, Journal of Computer and System Sciences 73 (2007) 1–24.
[16] C. Rispal, O. Carton, Complementation of rational sets on countable scattered linear orderings, International Journal of Foundations of Computer Science 16 (2005) 767–786.
[17] C. Rispal, Automates sur les ordres linéaires: Complémentation, Ph.D. thesis, Université de Marne la Vallée, 2004.
[18] O. Carton, Accessibility in automata on scattered linear orderings, in: International Symposium on Mathematical Foundations of Computer Science, Springer, 2002, pp. 155–164.
[19] J. Ferrante, C. W. Rackoff, The computational complexity of logical theories, volume 718, Springer, 2006.