Decidability of Difference Logics with Unary Predicates

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

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

Abstract

We investigate the decidability of a family of logics mixing difference-logic constraints and unary uninterpreted predicates. The focus is set on logics whose domain of interpretation is $\mathbb{R}$, but the language has a recognizer for integer values. We first establish the decidability of the logic allowing unary uninterpreted predicates, order constraints between real and integer variables, and difference-logic constraints between integer variables. Afterwards, we prove the undecidability of the logic where unary uninterpreted predicates and difference-logic constraints between real variables are allowed.

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 were later introduced in SMT solvers (e.g., [2, 3, 4]) 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.

In a previous paper [5] we discussed how to mix arithmetic and predicate symbols while avoiding undecidability. We focused on logics allowing uninterpreted unary predicates and difference-logic constraints, and investigated the expressive power of such logics over the domains $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$ and $\mathbb{R}$. We highlighted that when the domain is $\mathbb{N}$ or $\mathbb{Z}$, the logic is equivalent to the monadic second-order theory of $\mathbb{N}$ with one successor relation (usually referred to as S1S). It is thus decidable and some effective decision procedures already exist. We also provided some directions to solve the case where the domain of interpretation is $\mathbb{R}$, namely developing an encoding for the models of the predicates variables, and advocating for the use of the automata defined in [6].

SC^2 2022: The 7th International Workshop on Satisfiability Checking and Symbolic Computation, August 12, 2022, Haifa, Israel

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

© 2022 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings (CEUR-WS.org)

CEUR-WS.org/Vol-3458/paper3.pdf

In this paper we show that difference logic on the reals together with unary uninterpreted predicates is already undecidable. However, the mixed integer-real theory of order with unary uninterpreted predicates is decidable, even when allowing difference-logic constraints on integer variables. Basically, adding the “+1” function on the reals makes the logic undecidable.

After some prerequisites and the introduction of the studied fragments in Section 2, we first prove in Section 3 the decidability of the fragment with uninterpreted unary predicates, order constraints between real and integer variables and difference-logic constraints involving only integer variables. Section 4 is dedicated to a proof of undecidability of the fragment interpreted over $\mathbb{R}$, where uninterpreted unary predicates, order between real values, and difference-logic constraints on real variables are allowed.

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 ($\vee$, $\wedge$, $\neg$, $\Rightarrow$, $\Leftrightarrow$) and first-order quantification $\exists x.\, \varphi$ and $\forall x.\, \varphi$, respectively equivalent to writing $\exists x\, (\varphi)$ and $\forall x\, (\varphi)$, i.e., the dot stands for an opening parenthesis that is closed at the end of the formula. Variable symbols are denoted by x, y, z, $\ldots$

Our signature contains interpreted arithmetic symbols 0, 1, $+$, $-$, $<$, $\leq$, $\geq$, $>$, and constants in $\mathbb{N}$ that stand for terms $1 + 1 + \cdots + 1$. We furthermore use a unary interpreted predicate $x \in \mathbb{Z}$ to denote that x has an integer value. The signature also contains uninterpreted predicate symbols P, Q, $\ldots$ In the whole article, we only consider unary predicate symbols. Our language is the set of all well-formed formulas (in the usual sense) built using symbols from the signature. Further specific restrictions will be introduced later.

An interpretation specifies a domain (i.e., a set of elements), assigns a value in the domain to each free variable, and assigns relations of appropriate arity on the domain to predicate symbols in the signature. Throughout the article, the interpretation domain is always $\mathbb{R}$. The arithmetic symbols 0, 1, $+$, $-$, $<$, $\leq$, $\geq$, $>$ are interpreted as expected on $\mathbb{R}$, and $x \in \mathbb{Z}$ is true if and only if x has an integer value[2]. An interpretation assigns an arbitrary subset of the domain $\mathbb{R}$ to each unary predicate. By extension, an interpretation assigns a value in $\mathbb{R}$ to every term, and a truth value to every formula. We denote the interpretation I of a variable x by $I[x]$, and the interpretation of a predicate P by $I[P]$. A model of a formula is an interpretation that assigns true to this formula. A formula is satisfiable on a domain (here $\mathbb{R}$) if it has a model on that domain.

2.1. Difference arithmetic with unary predicates

We consider several fragments where the language is restricted, in particular in the way that the arithmetic relations can be used. A fragment is decidable if there exists an effective procedure to check whether a given formula in this fragment is satisfiable.

In the various fragments introduced below, all arithmetic atoms are of the following form:

[2] In the current context, this choice of notation for mixed integer-real arithmetic is simpler than using a multi-sorted logic.

- *order constraints* of the form $x \bowtie y$, where x and y are variables and $\bowtie \in \{<, \leq, =, \geq, >\}$;
- *difference-logic constraints* of the form $x - y \bowtie c$, where x and y are variables, c is a constant in $\mathbb{Z}$, and $\bowtie \in \{<, \leq, =, \geq, >\}$.

As a reminder, the language of our formulas only contain *unary predicates.* The only atoms besides the arithmetic ones are of the form $P(x)$ where P is an uninterpreted predicate symbol and x is a variable, and $x \in \mathbb{Z}$ where x is a variable. For convenience, the set of comparison operators is not minimal (we allow negations in formulas). Also, we sometimes write difference-logic constraints in their equivalent form $x \bowtie y + c$.

We now introduce our fragments of interest. Their names are derived from the SMT-LIB nomenclature, where acronyms stand for the theories that appear in the combinations:

- *UF1*: the theory of uninterpreted functions, with the restriction that only monadic (i.e., unary) predicates are allowed;
- *RO*: the theory of order on the reals only;
- *IRO*: the theory of order on the reals and integers;
- *IDL*: difference logic on the integers;
- *RDL*: difference logic on the reals.

UF1·RO. The fragment *UF1·RO* is the fragment with unary uninterpreted predicates and order constraints between variables interpreted over $\mathbb{R}$. Difference-logic constraints and atoms of the form $x \in \mathbb{Z}$ are not allowed.

Example: The formula

$$\forall x \, \exists y, z \, . \, y < x < z \wedge \forall t \, . \, (y < t < z \wedge P(t)) \Rightarrow t = x$$

describes a predicate P that is true only on isolated real numbers.

UF1·IRO. The fragment *UF1·IRO* is the extension of *UF1·RO* where atoms of the form $x \in \mathbb{Z}$ are allowed. This fragment can express order relations between real and integer variables.

Example: The formula

$$\forall x, y. \left[x < y \wedge x \in \mathbb{Z} \wedge y \in \mathbb{Z}\right] \Rightarrow \exists v. \, x < v < y \wedge P(v)$$

describes a predicate P that is true for at least one value located between any two integers.

UF1·IDL·IRO. The fragment *UF1·IDL·IRO* is an extension of the fragment *UF1·IRO* (and therefore of *UF1·RO*). It is also interpreted over $\mathbb{R}$. Order constraints between variables and atoms of the form $x \in \mathbb{Z}$ are allowed. Additionally, difference-logic constraints are allowed, but they can only involve *integer-guarded* variables.

In order to enforce this integer-guard restriction on difference-logic constraints, *UF1·IDL·IRO* formulas must be *well-guarded*, i.e., difference-logic constraints can only appear in the two following contexts:

- $x \in \mathbb{Z} \wedge y \in \mathbb{Z} \wedge x - y \bowtie c$,

- $(x \in \mathbb{Z} \land y \in \mathbb{Z}) \Rightarrow x - y \bowtie c$,

where x and y are variables, $c \in \mathbb{N}$ is a constant, and $\bowtie \in \{<, \leq, =, \geq, >\}$.

Example: The formula

$$\begin{array}{ll} & \big[\forall x, y.\, \big(x \in \mathbb{Z} \land y \in \mathbb{Z} \land y - x = 2\big) \Rightarrow \big(P(x) \Leftrightarrow P(y)\big)\big] \\ \land & \big[\exists x, y.\, x \in \mathbb{Z} \land y \in \mathbb{Z} \land P(x) \land \neg P(y)\big] \\ \land & \big[\forall z.\, \neg(z \in \mathbb{Z}) \Rightarrow P(z)\big] \end{array}$$

describes a predicate that is either true on all odd numbers and false on all even numbers, or the opposite, as well as true on all non-integer numbers.

UF1·RDL. The fragment *UF1·RDL* is the fragment interpreted over $\mathbb{R}$, where order constraints, difference-logic constraints and unary predicate atoms are allowed without any restriction. The use of atoms of the form $x \in \mathbb{Z}$ is forbidden. Since order constraints are a special case of difference-logic constraints, the name of the fragment only refers to *RDL* and not *RO*.

Example: The formula

$$\forall x\, \exists y.\, y - x > 0 \land y - x < 3 \land P(y)$$

describes a predicate P such that any sub-interval of $\mathbb{R}$ of length greater or equal to 3 contains a value for which P is true.

Note: It might appear to the reader that a missing logic in this nomenclature is *UF1·IRDL*, with difference-logic constraints on both real and integer variables. We will later show that *UF1·RDL* is already undecidable, so it makes little sense to introduce any extension of it.

3. Decidability of *UF1·IDL·IRO*

It has been established in [8, 9, 10, 11, 12] that the monadic first-order logic of order over $\mathbb{R}$ (*UF1·RO*) is decidable. We show here that its extension *UF1·IDL·IRO* (and therefore *UF1·IRO*) is also decidable, by reduction to *UF1·RO*.

Theorem 1. *UF1·IDL·IRO and UF1·IRO are decidable.*

Note that the decidability of *UF1·IRO* is a direct consequence of the decidability of *UF1·IDL·IRO*, since *UF1·IDL·IRO* is an extension of *UF1·IRO*. The remaining of this section is thus dedicated to the proof that *UF1·IDL·IRO* is decidable.

3.1. Recognizing integer values

We first show how to define in *UF1·RO* a predicate P_{int} over $\mathbb{R}$ that is $<$-isomorphic to $\mathbb{Z}$, i.e., there exists a bijection between the sets described by P_{int} and $\mathbb{Z}$ that preserves the order relation over their elements. Integer guards in *UF1·IDL·IRO* will later be translated using P_{int}. Intuitively, an integer-guarded variable in a *UF1·IDL·IRO* formula will correspond to a variable taking its value in the set described by P_{int} in the translated *UF1·RO* formula.

We axiomatize P_{int} in *UF1·RO* as follows:

- Every element of P_{int} is isolated:
  $\forall x \, \exists y, z.\, y < x < z \wedge \forall t.\, [y < t < z \wedge P_{int}(t)] \Rightarrow t = x$.
- Every point in $\mathbb{R}$ has a successor in P_{int}:
  $\forall x \, \exists y.\, x < y \wedge P_{int}(y) \wedge \forall t.\, x < t < y \Rightarrow \neg P_{int}(t)$.
- Similarly, every point in $\mathbb{R}$ has a predecessor in P_{int}:
  $\forall x \, \exists y.\, y < x \wedge P_{int}(y) \wedge \forall t.\, y < t < x \Rightarrow \neg P_{int}(t)$.

Notice that the successor and predecessor axioms ensure that P_{int} is not the empty predicate. Furthermore, it is worth mentioning that defining the successor (resp. predecessor) axiom by stating that every point in P_{int} has a successor (resp. predecessor) in P_{int} would not be correct, as P_{int} would also describe sets that are not $<$-isomorphic to $\mathbb{Z}$, e.g., the set $\{(k,i) \mid k \in \mathbb{Z},\, i \in \{1,2\}\}$ with the order relation $(k,i) < (\ell, j)$ if and only if either $i < j$, or $i = j$ and $k < \ell$.

The set of all integers is a model for P_{int}, therefore the above axiomatization is consistent. The set of elements satisfying P_{int} is necessarily infinite and does not admit a maximal or a minimal element. This is a direct consequence of the successor and predecessor axioms. More interestingly, this set is also necessarily countable. Indeed, since each point is isolated, there exists an application that maps the elements satisfying P_{int} to disjoint open intervals. Any set of disjoint intervals in $\mathbb{R}$ with non-zero length is necessarily countable [13], since each of them contains a rational value that does not belong to the others.

It is now possible to define a predecessor and a successor relation on the real numbers satisfying P_{int} with the following formulas:

- $\mathit{pred}(x,y) = \; P_{int}(x) \wedge P_{int}(y) \wedge x < y \wedge \forall z.\, x < z < y \Rightarrow \neg P_{int}(z)$,
  i.e., x is the predecessor of y;
- $\mathit{succ}(x,y) = \; P_{int}(x) \wedge P_{int}(y) \wedge y < x \wedge \forall z.\, y < z < x \Rightarrow \neg P_{int}(z)$,
  i.e., x is the successor of y.

Note that these two definitions are redundant, i.e., the formula $\mathit{pred}(x,y)$ holds if and only if $\mathit{succ}(y,x)$ holds.

We next prove that the set of elements satisfying P_{int} is $<$-isomorphic to the integers. For convenience in the proof, we define 0_{int} as an arbitrary existentially quantified value that belongs to the set described by P_{int}.

Lemma 1. *For any model M of P_{int}, the set $\{x \mid x \in M[P_{int}]\}$ is $<$-isomorphic to $\mathbb{Z}$.*

Proof. Given a model M of the axiomatization of P_{int}, we need to define a bijection between the set $\mathcal{P} = \{x \mid x \in M[P_{int}]\}$ and $\mathbb{Z}$ that preserves order.

Let us define an application f from $\mathcal{P}$ to $\mathbb{Z}$. We set $f(0_{int}) = 0$, and then define recursively:

- $f(y) = f(x) + 1$ for each $x, y \in \mathcal{P}$ such that $y > 0_{int}$ and $\mathit{pred}(x,y)$,
- $f(y) = f(x) - 1$ for each $x, y \in \mathcal{P}$ such that $y < 0_{int}$ and $\mathit{succ}(x,y)$.

Thanks to the fact that every point has a unique predecessor and successor, it follows that f ranges over the whole set $\mathbb{Z}$. It is clear that f preserves order. It remains to show that f is well defined for every element in $\mathcal{P}$.

If there exists some element $y \in \mathcal{P}$ for which f is not defined, it means that f is not well-founded, in the sense that there exists either an element $y > 0_{int}$ such that the interval $[0_{int}, y]$ contains an infinite number of elements satisfying P_{int}, or there exists an element $y < 0_{int}$ such that the interval $[y, 0_{int}]$ contains an infinite number of elements satisfying P_{int}. Since both cases are symmetric, we only address the former. There must exist a strictly increasing infinite series of elements in $\mathcal{P}$ bounded by y. Let us consider its limit $z \in \mathbb{R}$. Because there must exist an element of $\mathcal{P}$ smaller than z and arbitrarily close to z, it follows that z cannot have a predecessor, which contradicts an axiom. Therefore f is well-defined, and every element of $\mathcal{P}$ is associated to an integer number. The application f is therefore a bijection. $\square$

3.2. Translating formulas

We are now able to describe the satisfiability-preserving translation of formulas from *UF1·IDL·IRO* to *UF1·RO*. Consider a *UF1·IDL·IRO* formula φ. Without loss of generality, we assume that P_{int} does not appear in φ. The translation of φ is defined as

$$\mathit{AXIOMS}(P_{int}) \wedge [\![\varphi]\!]$$

where $\mathit{AXIOMS}(P_{int})$ is the conjunction of the axioms of P_{int}, and $[\![\cdot]\!]$ is a translation operator. This translation operator $[\![\cdot]\!]$ distributes over all Boolean operators and quantifiers, and corresponds to the identity for most considered atoms, except for:

- $[\![x \in \mathbb{Z}]\!] = P_{int}(x)$;
- $[\![x - y \bowtie c]\!] = \exists z_0, \dots z_c \,.\, (y = z_0) \wedge (x \bowtie z_c) \wedge \bigwedge_{0 \leq i < c} \mathit{succ}(z_{i+1}, z_i)$,
  for $c \in \mathbb{N}$ and $\bowtie \in \{<, \leq, =, \geq, >\}$. We assume that $z_0, \dots z_c$ are fresh variables w.r.t. x and y.

Example:

$$[\![x - y \leq 2]\!] = \exists z_0, z_1, z_2.\; y = z_0 \wedge \mathit{succ}(z_1, z_0) \wedge \mathit{succ}(z_2, z_1) \wedge x \leq z_2.$$

Notice that we only deal with the case $c \in \mathbb{N}$ since every atom of the form $x - y \bowtie c$ with $c \in \mathbb{Z} \setminus \mathbb{N}$ and $\bowtie \in \{<, \leq, =, \geq, >\}$ can be rewritten as $y - x \bowtie' -c$ with the following correspondences: $(\bowtie, \bowtie') \in \{(=,=), (<,>), (>,<), (\geq,\leq), (\leq,\geq)\}$.

3.3. Establishing equisatisfiability

Given a *UF1·IDL·IRO* formula φ, the translation that we have introduced generates a corresponding *UF1·RO* formula ψ. To establish that they are equisatisfiable, we need to prove that if φ admits a model, then ψ also admits one, and reciprocally. This is quite straightforward:

- If φ is satisfiable, let M be one of its models. Then, since ψ shares the same free variables and predicates than φ with the only addition of P_{int}, we can directly construct a model M' of ψ that is similar to M for the shared variables and predicates, and P_{int} is interpreted so that $P_{int}(x)$ holds whenever $x \in \mathbb{Z}$. This is always possible since the only constraints on P_{int} generated by the construction of ψ are the axioms stated above.

- If ψ is satisfiable, then there exists a model M of ψ. Let us construct a model M' of φ. Let $0_{int} \in \mathbb{R}$ be an arbitrary element of $M[P_{int}]$ (similarly to before). We define an automorphism g of $\mathbb{R}$, such that $g(0_{int}) = 0$, and recursively $g(y) = g(x) + 1$ for $x, y \in M[P_{int}]$, $y > 0_{int}$, and *pred*(x, y); and $g(y) = g(x) - 1$ for $x, y \in M[P_{int}]$, $y < 0_{int}$, and *succ*(x, y). The automorphism g maps each open interval between the k-th and $(k+1)$-th successors (resp. predecessors) of 0_{int} in $M[P_{int}]$, onto the open interval $(k, k+1)$ (resp. $(-(k+1), k)$) while preserving order.
  M' is defined by $M'[x] = g(M[x])$ for each free variable x of the formula φ, and $M'[P] = \{g(x) \mid x \in M[P]\}$ for each uninterpreted predicate P of φ. No unary predicate atom can be violated by M' by definition. Furthermore, no order constraint can be violated by M' either since g preserves order. Regarding the difference-logic constraints, the intermediate variables z_i introduced in the translation are necessarily mapped to values in $M[P_{int}]$ since the *succ* relation enforces this property. Hence for each such variable, we have $g(M[z_i]) \in \mathbb{Z}$. Intuitively, this ensures that in M' the difference between the values taken by the integer variables is consistent with the difference-logic constraints. Hence M' is a model of φ.

4. Undecidability of *UF1·RDL*

The result presented in the previous section establishes a lower bound of some sort for the decidability of our family of fragments. A natural follow up problem is to establish a corresponding upper bound, i.e., to define a slight extension of this logic that yields non-trivial undecidability. We show here that as soon as difference-logic constraints on reals are allowed, the logic becomes undecidable. More precisely, we establish the undecidability of *UF1·RDL* by reducing the halting problem of a Turing machine to the satisfiability problem over *UF1·RDL*.

Theorem 2. *Satisfiability is undecidable for UF1·RDL.*

The remaining of this section is dedicated to the proof of the undecidability of *UF1·RDL*. We consider w.l.o.g. Turing machines defined over an alphabet with only two symbols and no explicit blank symbol [14]. This choice leads to a simpler proof, but the same approach can easily be extended to Turing machines with larger alphabets.

4.1. Turing machine

The proof is by reduction from the halting problem for a Turing machine starting from a blank tape (i.e., a tape filled with the symbol 0). Consider a Turing machine $\mathcal{M} = (Q, \Sigma, q_I, q_F, \delta)$, where

- Q is a finite set of states,
- the alphabet Σ is w.l.o.g. assumed to be $\{0, 1\}$,
- $q_I \in Q$ is the initial state,
- $q_F \in Q$ is the halting state,
- $\delta : (Q \setminus \{q_F\}) \times \Sigma \to Q \times \Sigma \times \{L, R\}$ is the transition relation, assumed to be total.

A *configuration* C of such a Turing machine is a triplet containing the current state q, the content of the tape $t = \{0,1\}^{\mathbb{Z}}$ and finally the position of the head $h \in \mathbb{Z}$. Since the machine starts from a blank tape, the initial configuration is $C_0 = (q_I, \{0\}^{\mathbb{Z}}, 0)$.

A *run* ρ of length $n \in \mathbb{N} \cup \{+\infty\}$ of such a Turing machine is a (possibly infinite) sequence of configurations $(C_i)_{i\in[0,n]}$, such that for any two consecutive configurations $C_i = (q_i, t_i, h_i)$ and $C_{i+1} = (q_{i+1}, t_{i+1}, h_{i+1})$ there exists a transition $(q, \alpha, q', \alpha', \lambda) \in \delta$ such that:

- $q = q_i$ and $q' = q_{i+1}$,
- $t_i[h_i] = \alpha$, i.e., the tape cell at position h_i contains the symbol α,
- $t_{i+1}[h_i] = \alpha'$,
- $t_{i+1}[k] = t_i[k]$, for every $k \in \mathbb{Z}$, $k \neq h_i$,
- $h_{i+1} = h_i + 1$ if $\lambda = R$, and $h_{i+1} = h_i - 1$ if $\lambda = L$.

A *halting run* is a finite run such that the state of its last configuration is the halting state q_F, and such that no previous configuration has q_F as its state.

4.2. Encoding a run with predicates

The reduction consists in encoding a run ρ of $\mathcal{M}$ of length $n \in \mathbb{N} \cup \{+\infty\}$ with predicates interpreted over real values. For this, we need to be able to represent configurations, and verify whether there exists a transition of $\mathcal{M}$ that allows to go from one to another.

This requires to be able to represent the state, the head position and the entire tape content for each consecutive configuration of the run ρ. Furthermore, we also need to compare two consecutive contents of the tape and ensure that the only change occurs at the position of the head, as well as to handle the motion of the head.

This can be done as follows:

- Let $N = \lceil \log_2(|Q|) \rceil$. Every state $q \in Q$ can be encoded without ambiguity by a tuple $(b_{q,1}, b_{q,2}, \dots, b_{q,N})$ of Boolean values. The states visited by ρ can thus be described by N predicates Q_1, Q_2, …, Q_N such that for each $i \in [0,n]$, the tuple $(Q_1(i), Q_2(i), \dots, Q_N(i))$ encodes the state reached by ρ after i steps. For convenience we introduce the formula $\mathit{State}_q(i) = \bigwedge_{1\le j\le N} Q_j(i) = b_{q,j}$ that is true if and only if the machine is in state q at step i, where the shorthand $Q_j(i) = b_{q,j}$ means $\neg Q_j(i)$ if $b_{q,j} = \bot$ and $Q_j(i)$ otherwise.
- For each $i \in [0,n]$, the content of the tape after i steps is described by the value of a predicate T interpreted over the open interval $(i, i+1)$. This is achieved by defining an order-preserving mapping ζ between $\mathbb{Z}$ and the open interval $(0,1)$, and replicating this mapping in every interval $(i, i+1)$. In other words, for every step $i \in [0,n]$ and tape position $j \in \mathbb{Z}$, the value of $T(i + \zeta(j))$ provides the content of the cell at position j of the tape after i steps.
- The position of the head is described by the value of a predicate H interpreted in the same way as T: For every step $i \in [0,n]$ and tape position $j \in \mathbb{Z}$, we have $H(i + \zeta(j)) = \top$ if and only if the head is at position j after i steps.

Recall that in this fragment the use of integer guards is forbidden. In order to have similar landmarks as natural and integer numbers, we will have to construct our own sets of elements that are $<$-isomorphic to $\mathbb{N}$ and $\mathbb{Z}$.

4.3. Enforcing a valid run

We now show how to translate $\mathcal{M}$ into a formula of the logic, that is satisfiable if and only if $\mathcal{M}$ admits a halting run.

First, we define 0 as an arbitrary existentially quantified value in $\mathbb{R}$, and 1 as the real such that $0 + 1 = 1$.

Then we define a predicate P_{nat} that is $<$-isomorphic to $\mathbb{N}$. We construct P_{nat} as follows:

$$\forall x \,.\, P_{nat}(x) \Leftrightarrow [x = 0 \vee (x \geq 1 \wedge \exists y \,.\, P_{nat}(y) \wedge x = y + 1)]$$

The state of the initial configuration is encoded on 0, and the states of the successive configurations are encoded on the successive elements belonging to the set described by P_{nat}.

We also define a predicate Z with a characteristic set that maps $\mathbb{Z}$ into $(0, 1)$: each x such that $Z(x)$ holds must satisfy $0 < x < 1$, be isolated, and every point in the open interval $(0, 1)$ must admit a successor and a predecessor within $(0, 1)$ on which Z is true. All these constraints are clearly expressible in *UF1·RDL*. This is done in a similar way as the definition of P_{int} in the previous section. The only differences are that we forbid Z to be true for any value outside of $(0, 1)$, and that the successor and predecessor axioms are adapted in the following way:

- Successor:
  $\forall x \,.\, (0 < x < 1) \Rightarrow (\exists y \,.\, x < y < 1 \wedge Z(y) \wedge \forall t \,.\, x < t < y \Rightarrow \neg Z(t))$.
- Predecessor:
  $\forall x \,.\, (0 < x < 1) \Rightarrow (\exists y \,.\, 0 < y < x \wedge Z(y) \wedge \forall t \,.\, y < t < x \Rightarrow \neg Z(t))$.

Notice that the successor and predecessor axioms ensure that Z is not the empty predicate.

We can now extend Z into $\overline{Z}$ over the union of intervals $(i, i+1)$ for i such that $P_{nat}(i)$ holds:

$$\forall x \,.\, \overline{Z}(x) \;\Leftrightarrow\; \big[(0 < x < 1 \wedge Z(x)) \;\vee (x > 1 \wedge \exists y \,.\, x = y + 1 \wedge \overline{Z}(y))\big]$$

The conditions on the initial configuration of $\mathcal{M}$ are encoded by the following formula:

$$\begin{aligned} \mathit{START}_{\mathcal{M}} = \mathit{State}_{q_I}(0) \wedge (\forall x \,.\, Z(x) \;\Rightarrow\; \neg T(x)) \\ \wedge\; \exists x \,.\, H(x) \wedge Z(x) \wedge \forall y \,.\, (Z(y) \wedge H(y)) \;\Rightarrow\; y = x \end{aligned}$$

Notice that we do not specify the value of the head and tape predicates outside of the set described by $\overline{Z}$. In other words, we ignore how these predicates behave outside of $\overline{Z}$. The same goes for the value of the state predicates $(Q_j)_{1 \leq j \leq N}$ outside of P_{nat}.

The conditions on the transition relation of $\mathcal{M}$ are more complex. Intuitively, if at a given step $i \in [1, n]$ we have not yet encountered the halting state q_F, then we must ensure that the configuration at step i can be obtained from the configuration at the previous step $i - 1$, by following a transition $(q, \alpha, q', \alpha', \lambda) \in \delta$. The overall formula for these conditions is the following:

$$\mathit{STEP}_{\mathcal{M}} = \forall y \,.\, \big(y > 0 \wedge P_{nat}(y) \wedge \mathit{NotEnded}_{\mathcal{M}}(y)\big) \Rightarrow \exists x \,.\, y = x{+}1 \wedge \mathit{Transition}_{\mathcal{M}}(x, y)$$

The subformula $\mathit{NotEnded}_{\mathcal{M}}(y)$ expresses that no previous landmark to y (i.e., a value x such that $x < y$ and $P_{\mathit{nat}}(x)$ holds) encodes the halting state. The formula is defined by:

$$\mathit{NotEnded}_{\mathcal{M}}(y) = \forall x \,.\, (x < y \wedge P_{\mathit{nat}}(x)) \Rightarrow \neg \mathit{State}_{q_F}(x)$$

The subformula $\mathit{Transition}_{\mathcal{M}}(x, y)$ expresses that there exists a transition $(q, \alpha, q', \alpha', \lambda) \in \delta$ that allows to move from the configuration corresponding to the landmark x to the configuration corresponding to y in one step. For convenience, we decompose the conditions on the transition relation as follows:

$$\begin{aligned} \mathit{Transition}_{\mathcal{M}}(x, y) = {} & \mathit{UniqueHead}_{\mathcal{M}}(x, y) \\ \wedge & \bigvee_{(q,\alpha,q',\alpha',\lambda)\in\delta} \Big[\mathit{States}_{q,q'}(x, y) \wedge \mathit{Tape}_{\alpha,\alpha'}(x, y) \wedge \mathit{Head}_{\lambda}(x, y) \Big] \end{aligned}$$

For a given transition $(q, \alpha, q', \alpha', \lambda) \in \delta$, the conditions on the states, tape and head are expressed as follows:

- The landmarks x and y should be such that $Q_1(x), Q_2(x), \ldots, Q_N(x)$ encode q and $Q_1(y), Q_2(y), \ldots, Q_N(y)$ encode q':

$$\mathit{States}_{q,q'}(x, y) = \mathit{State}_{q}(x) \wedge \mathit{State}_{q'}(y)$$

- The tape must contain α at the current position of the head for the step corresponding to x. Additionally, for the step corresponding to y, the tape contains α' at the previous position of the head, and is unchanged at all other positions.

$$\begin{aligned} & \mathit{Tape}_{\alpha,\alpha'}(x, y) = \\ & \qquad \big[\forall z. \, (x < z < y \wedge H(z) \wedge \overline{Z}(z)) \Rightarrow \, (T(z) = \alpha \wedge \exists z'. \, z' = z{+}1 \wedge T(z') = \alpha')\big] \\ & \quad \wedge \quad \big[\forall z. \, (x < z < y \wedge \neg H(z) \wedge \overline{Z}(z)) \Rightarrow \exists z'. \, z' = z{+}1 \wedge T(z') = T(z)\big] \end{aligned}$$

- The head is moved in the direction specified by λ. This can be expressed by exploiting the predecessor and successor relations on $\overline{Z}$ (defined in the exact same manner than $\mathit{pred}(x, y)$ and $\mathit{succ}(x, y)$ for P_{int} in the previous section).

$$\begin{aligned} \mathit{Head}_{\lambda}(x, y) = {} & \forall z. \, \big[(x < z < y \wedge H(z) \wedge \overline{Z}(z)) \\ & \qquad \Rightarrow \exists z', z''. \, (z' = z + 1) \wedge f_{\lambda}(z'', z') \wedge H(z'')\big] \end{aligned}$$

  where

$$f_{\lambda} = \begin{cases} \mathit{succ} & \text{if } \lambda = R \\ \mathit{pred} & \text{if } \lambda = L \end{cases}$$

- We must also ensure that the position of the head is unique for the configuration corresponding to x:

$$\begin{aligned} \mathit{UniqueHead}_{\mathcal{M}}(x, y) = {} & \exists z. \big[x < z < y \wedge \overline{Z}(z) \wedge H(z) \\ & \qquad \wedge \forall z'. \, \big(x < z' < y \wedge \overline{Z}(z') \wedge H(z')\big) \Rightarrow z' = z\big] \end{aligned}$$

Finally, the existence of a halting run is expressed by the formula:

$$\mathit{END}_{\mathcal{M}} = \exists x.\, P_{\mathit{nat}}(x) \wedge \mathit{State}_{q_F}(x)$$

The global formula that expresses that the Turing machine $\mathcal{M}$ halts on some run encoded by the predicates Q_j ($j = 1, \dots N$), T and H is the following:

$$\mathit{HALT}_{\mathcal{M}}(Q_1, \dots Q_N, T, H) = \mathit{START}_{\mathcal{M}} \wedge \mathit{STEP}_{\mathcal{M}} \wedge \mathit{END}_{\mathcal{M}}$$

This concludes the proof of Theorem 2.

5. Conclusion and future work

In this work, we have established a lower and an upper bound for decidability in a family of logics mixing weak forms of arithmetic and uninterpreted unary predicates.

We proved the decidability of the fragment *UF1·IDL·IRO*, where uninterpreted unary predicates, order constraints between real and integer variables and difference-logic constraints between integer variables are allowed. This result is a consequence of the already established decidability of its restriction *UF1·RO* (where only uninterpreted unary predicates and order constraints between real values are allowed).

The other result establishes the undecidability of the fragment *UF1·RDL*, where uninterpreted unary predicates and difference-logic constraints between real variables are allowed. Notice that this result can be adapted straightforwardly to the logic interpreted over the domain $\mathbb{Q}$.

Since the submission of this article, we additionally proved that the restriction of *UF1·RDL* where only a single unary uninterpreted predicate is allowed is also undecidable. This further refines the decidability frontier.

A complexity upper bound for a decision procedure of *UF1·RO* is also known [11, 12]. Our long term goal is to design a practical decision procedure for this decidable logic, and to adapt it for *UF1·IDL·IRO*.

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).

[5] B. Boigelot, P. Fontaine, B. Vergain, Deciding satisfiability for fragments with unary predicates and difference arithmetic, in: Proceedings, 6th International Workshop on Satisfiability Checking and Symbolic Computation, 2021.

[6] V. Bruyère, O. Carton, Automata on linear orderings, Journal of Computer and System Sciences 73 (2007) 1–24.

[7] H. B. Enderton, A mathematical introduction to logic (2nd edition), Academic Press, Boston, 2001.

[8] J. P. Burgess, Y. Gurevich, The decision problem for linear temporal logic, Notre Dame Journal of Formal Logic 26 (1985) 115–128.

[9] H. Läuchli, J. Leonard, On the elementary theory of linear order, Fundamenta Mathematicae 59 (1966) 109–116.

[10] W. Thomas, Ehrenfeucht games, the composition method, and the monadic theory of ordinal words, in: Structures in Logic and Computer Science, Springer, 1997, pp. 118–143.

[11] M. Reynolds, The complexity of temporal logic over the reals, Annals of Pure and Applied Logic 161 (2010) 1063–1096.

[12] A. Rabinovich, Temporal logics over linear time domains are in PSPACE, Information and Computation 210 (2012) 40–67.

[13] W. Sierpiński, Cardinal and ordinal numbers (2nd edition), PWN, Warszawa, 1965.

[14] C. E. Shannon, A universal Turing machine with two internal states, Automata studies 34 (1956) 157–165.