An optimal decision procedure for Right Propositional Neighborhood Logic

Davide Bresolin and Angelo Montanari
(bresolin,montana@dimi.uniud.it)
Dipartimento di Matematica e Informatica,
Università degli Studi di Udine,
Via Delle Scienze, 208, I-33100 - Udine, ITALY

Guido Sciavicco (guido@um.es)
Department of Information Engineering and Communications,
Faculty of Informatics, University of Murcia,
Campus de Espinardo, E-30100 - Espinardo (Murcia) SPAIN

Abstract. Propositional interval temporal logics are quite expressive temporal logics that allow one to naturally express statements that refer to time intervals. Unfortunately, most such logics turned out to be (highly) undecidable. To get decidability, severe syntactic and/or semantic restrictions have been imposed to interval-based temporal logics that make it possible to reduce them to point-based ones. The problem of identifying expressive enough, yet decidable, new interval logics or fragments of existing ones which are genuinely interval-based is still largely unexplored. In this paper, we focus our attention on interval logics of temporal neighborhood. We address the decision problem for the future fragment of Neighborhood Logic (Right Propositional Neighborhood Logic, RPNL for short) and we positively solve it by showing that the satisfiability problem for RPNL over natural numbers is NEXPTIME-complete. Then, we develop a sound and complete tableau-based decision procedure and we prove its optimality.

1. Introduction

Propositional interval temporal logics are quite expressive temporal logics that provide a natural framework for representing and reasoning about temporal properties in several areas of computer science, including artificial intelligence (reasoning about action and change, qualitative reasoning, planning, and natural language processing), theoretical computer science (specification and automatic verification of programs) and databases (temporal and spatio-temporal databases).

Various propositional and first-order interval temporal logics have been proposed in the literature (a recent comprehensive survey can be found in [10]). The most significant propositional ones are Halpern and Shoham's Modal Logic of Time Intervals (HS) [12], Venema's CDT logic, interpreted over linear and partial orders [8, 11, 21], Moszkowski's

Propositional Interval Temporal Logic (PITL) [18], and Goranko, Montanari, and Sciavicco's Propositional Neighborhood Logic (PNL) [9].

HS features four basic operators: $\langle B \rangle$ (*begins*) and $\langle E \rangle$ (*ends*), and their transposes $\langle \overline{B} \rangle$ and $\langle \overline{E} \rangle$. Given a formula φ and an interval $[d_0, d_1]$, $\langle B \rangle \varphi$ holds at $[d_0, d_1]$ if φ holds at $[d_0, d_2]$, for some $d_2 < d_1$, and $\langle E \rangle \varphi$ holds at $[d_0, d_1]$ if φ holds at $[d_2, d_1]$, for some $d_2 > d_0$. HS has been shown to be undecidable for several classes of linear and branching orders, including natural numbers [12]. The fragment of HS with the two modalities $\langle B \rangle$ and $\langle E \rangle$ only has been proved to be undecidable when interpreted over dense linear orders by sharpening the original Halpern and Shoham's result [14].

CDT has three binary operators C (*chop*), D, and T, which correspond to the ternary interval relations occurring when an extra point is added in one of the three possible distinct positions with respect to the two endpoints of the current interval (*before, between*, and *after*), plus a modal constant π which holds at a given interval if and only if it is a point-interval. CDT is powerful enough to embed HS, and thus it is undecidable (at least) over the same classes of orders.

PITL provides two modalities, namely, $\bigcirc$ (*next*) and C (the specialization of the *chop* operator for discrete structures). In PITL an interval is defined as a finite or infinite sequence of states. Given two formulas φ, ψ and an interval $s_0, \ldots, s_n$, $\bigcirc\varphi$ holds over $s_0, \ldots, s_n$ if and only if φ holds over $s_1, \ldots, s_n$, while $\varphi \, C \, \psi$ holds over $s_0, \ldots, s_n$ if and only if there exists i, with $0 \leq i \leq n$, such that φ holds over $s_0, \ldots, s_i$ and ψ holds over $s_i, \ldots, s_n$. In [18], Moszkowski proves the undecidability of PITL over discrete linear orders, while its undecidability over dense linear orders has been shown by Lodaya [14].

PNL has two modalities for right and left interval neighborhoods, namely, the *after* operator $\langle A \rangle$, such that $\langle A \rangle \varphi$ holds over $[d_0, d_1]$ if φ holds over $[d_1, d_2]$ for some $d_2 > d_1$, and its transpose $\langle \overline{A} \rangle$ [9]. While the undecidability of first-order Neighborhood Logic (NL) can be easily proved by embedding HS in it, the satisfiability problem for PNL has been recently shown to be decidable in NEXPTIME with respect to several class of linear orders [16]. The proof basically reduces the problem to the satisfiability problem for the decidable 2-variable fragment of first-order logic extended with a linear order [19]. As a matter of fact, such a reduction does not provide us with a viable decision procedure for PNL.

In summary, propositional interval temporal logics are very expressive (it can be shown that both HS and CDT are strictly more expressive than every point-based temporal logic on linear orders), but in general (highly) undecidable. They make it possible to express properties of *pairs* of time points (think of intervals as constructed out of

points), rather than *single* time points, and, in most cases, this feature prevents one from the possibility of reducing interval-based temporal logics to (decidable) point-based ones and of benefitting from the good computational properties of point-based logics.

To make such a reduction possible, severe syntactic and/or semantic restrictions must be imposed to interval temporal logics [15].

One can get decidability by making a suitable choice of the interval modalities. This is the case with the $\langle B\rangle\langle\overline{B}\rangle$ (*begins/begun by*) and and $\langle E\rangle\langle\overline{E}\rangle$ (*ends/ended by*) fragments of HS. Consider the case of $\langle B\rangle\langle\overline{B}\rangle$ (the case of $\langle E\rangle\langle\overline{E}\rangle$ is similar). As shown by Goranko et al. [10], the decidability of $\langle B\rangle\langle\overline{B}\rangle$ can be obtained by embedding it into the propositional temporal logic of linear time LTL[F,P] with temporal modalities F (sometime in the future) and P (sometime in the past). The formulae of $\langle B\rangle\langle\overline{B}\rangle$ are simply translated into formulae of LTL[F,P] by a mapping that replaces $\langle B\rangle$ by P and $\langle\overline{B}\rangle$ by F. LTL[F,P] has the finite model property and is decidable.

As an alternative, decidability can be achieved by constraining the classes of temporal structures over which the interval logic is interpreted. This is the case with the so-called Split Logics (SLs) [17]. SLs are propositional interval logics equipped with operators borrowed from HS and CDT, but interpreted over specific structures, called split structures. The distinctive feature of split structures is that every interval can be 'chopped' in at most one way. The decidability of various SLs has been proved by embedding them into first-order fragments of monadic second-order decidable theories of time granularity (which are proper extensions of the well-known monadic second-order theory of one successor S1S).

Finally, another possibility is to constrain the relation between the truth value of a formula over an interval and its truth value over subintervals of that interval. As an example, one can constrain a propositional letter to be true over an interval if and only if it is true at its starting point (*locality*) or can constrain it to be true over an interval if and only it it is true over all its subintervals (*homogeneity*). A decidable fragment of PITL extended with quantification over propositional letters (QPITL) has been obtained by imposing the *locality* constraint [18]. By exploiting such a constraint, decidability of QPITL can be proved by embedding it into quantified LTL. (In fact, as already noticed by Venema, the locality assumption yields decidability even in the case of the interval logics HS and CDT [21].)

A major challenge in the area of propositional interval temporal logics is thus to identify *genuinely interval-based* decidable logics, that is, logics which are not explicitly translated into point-based logics and

not invoking locality or other semantic restrictions, and to provide them with actual decision procedures.

In this paper, we propose an implicit and incremental tableau-based decision procedure for the future fragment of PNL, that we call Right PNL (RPNL for short), interpreted over natural numbers. While various tableau methods have been developed for linear and branching time point-based temporal logics [6, 7, 13, 20, 22], not much work has been done on tableau methods for interval-based temporal logics. One reason for this disparity is that operators of interval temporal logics are in many respects more difficult to deal with [11]. As an example, there exist straightforward inductive definitions of the basic operators of point-based temporal logics, while inductive definitions of interval modalities turn out to be much more involved (consider, for instance, the one for the *chop* operator given in [2]).

In [8, 11], Goranko et al. propose a general tableau method for CDT, interpreted over partial orders. It combines features of the classical tableau method for first-order logic with those of explicit tableau methods for modal logics with constraint label management, and it can be easily tailored to most propositional interval temporal logics proposed in the literature. However, it only provides a semi-decision procedure for unsatisfiability. In [4], Bresolin and Montanari propose an implicit and declarative tableau-based decision procedure for RPNL that combines syntactic restrictions (future temporal operators) and semantic ones (the domain of natural numbers). However, such a procedure is in EXPSPACE and, thus, since the decision problem for PNL is in NEXPTIME [16], it is far to be optimal.

In this paper, we devise an optimal tableaux-based decision procedure for RPNL. Unlike the case of the $\langle B\rangle\langle\overline{B}\rangle$ and $\langle E\rangle\langle\overline{E}\rangle$ fragments, we cannot abstract way from the left endpoint of intervals: there can be contradictory formulae that hold over intervals that have the same right endpoint, but a different left one. The proposed tableau method partly resembles the tableau-based decision procedure for LTL [22]. However, while the latter takes advantage of the so-called fix-point definition of temporal operators, which makes it possible to proceed by splitting every temporal formula into a (possibly empty) part related to the current state and a part related to the next state, and to completely forget the past, our method must also keep track of universal and (pending) existential requests coming from the past.

The paper is organized as follows. In Section 2 we introduce syntax and semantics of RPNL. We distinguish two possible semantics, namely, a *strict* one, which excludes intervals with coincident endpoints (*point-intervals*), and a *non-strict* one, which includes them. In Section

3 we give an intuitive account of the proposed decision method, in the case of strict semantics, and then, in Section 4, we formalize it. In Section 5 we prove the NEXPTIME-completeness of the satisfiability problem for RPNL, while in Section 6 we devise an optimal tableau-based decision procedure and we prove its soundness and completeness. Finally, in Section 7 we briefly show how to adapt the method to the case of non-strict semantics. Conclusions provide an assessment of the work and outline future research directions.

2. Right Propositional Neighborhood Logics

In this section, we give syntax and semantics of RPNL interpreted over the set $\mathbb{N}$ of natural numbers or over a prefix of it. To this end, we introduce some preliminary notions. Let $\mathbb{D} = \langle D, < \rangle$ be a strict linear order isomorphic to $\mathbb{N}$ or to a prefix of it. An *interval* on $\mathbb{D}$ is an ordered pair $[d_i, d_j]$ such that $d_i, d_j \in D$ and $d_i \leq d_j$. We say that $[d_i, d_j]$ is a *strict interval* if $d_i < d_j$, and that it is a *point interval* if $d_i = d_j$. The set of all strict intervals will be denoted by $\mathbb{I}(\mathbb{D})^-$, while the set of all intervals on $\mathbb{D}$ will be denoted by $\mathbb{I}(\mathbb{D})^+$. With $\mathbb{I}(\mathbb{D})$ we denote either of these. The pair $\langle \mathbb{D}, \mathbb{I}(\mathbb{D})^- \rangle$ is called a *strict interval structure*, while the pair $\langle \mathbb{D}, \mathbb{I}(\mathbb{D})^+ \rangle$ is called a *non-strict interval structure*. For every pair of intervals $[d_i, d_j], [d'_i, d'_j] \in \mathbb{I}(\mathbb{D})$, we say that $[d'_i, d'_j]$ is a *right neighbor* of $[d_i, d_j]$ if and only if $d_j = d'_i$.

The language of *Strict Right Propositional Neighborhood Logic* (RPNL$^-$ for short) consists of a set AP of propositional letters, the classical connectives $\neg$ and $\vee$, and the modal operator $\langle A \rangle$, the dual of which is denoted $[A]$. The remaining classical connectives, as well as the logical constants $\top$ (true) and $\bot$ (false), can be defined as usual. The *formulae* of RPNL$^-$, denoted by $\varphi, \psi, \ldots$, are recursively defined by the following grammar:

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

The language of *Non-strict Right Propositional Neighborhood Logic* (RPNL$^+$ for short) differs from the language of RPNL$^-$ only in the notation for the modalities: $\langle A \rangle$ and $[A]$ are replaced by $\Diamond_r$ and $\Box_r$, respectively. We use different notations for the modalities in RPNL$^-$ and RPNL$^+$ only to reflect their historical links and to make it easier to distinguish between the two semantics from the syntax. The *formulae* of RPNL$^+$, are recursively defined by the following grammar:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid \Diamond_r \varphi.$$

We denote by $|\varphi|$ the length of φ, that is, the number of symbols in φ (in the following, we shall use $|\ |$ to denote the cardinality of a set as well). Whenever there are no ambiguities, we call an RPNL formula just a formula. A formula of the forms $\langle A\rangle\psi$ or $\neg\langle A\rangle\psi$ (resp., $\Diamond_r\psi$ or $\neg\Diamond_r\psi$) is called a *temporal formula* (from now on, we identify $\neg\langle A\rangle\psi$ with $[A]\neg\psi$ and $\neg\Diamond_r\psi$ with $\Box_r\psi$).

A *model* for an RPNL$^-$ (resp., RPNL$^+$) formula is a pair $\mathbf{M} = \langle\langle\mathbb{D}, \mathbb{I}(\mathbb{D})\rangle, \mathcal{V}\rangle$, where $\langle\mathbb{D}, \mathbb{I}(\mathbb{D})\rangle$ is a strict (resp., non-strict) interval structure and $\mathcal{V} : \mathbb{I}(\mathbb{D}) \longrightarrow 2^{AP}$ is a *valuation function* assigning to every interval the set of propositional letters true on it. Given a model $\mathbf{M} = \langle\langle\mathbb{D}, \mathbb{I}(\mathbb{D})\rangle, \mathcal{V}\rangle$ and an interval $[d_i, d_j] \in \mathbb{I}(\mathbb{D})$, the semantics of RPNL$^-$ (resp., of RPNL$^+$) is defined recursively by the *satisfiability relation* $\Vdash$ as follows:

- for every propositional letter $p \in AP$, $\mathbf{M}, [d_i, d_j] \Vdash p$ iff $p \in \mathcal{V}([d_i, d_j])$;
- $\mathbf{M}, [d_i, d_j] \Vdash \neg\psi$ iff $\mathbf{M}, [d_i, d_j] \nVdash \psi$;
- $\mathbf{M}, [d_i, d_j] \Vdash \psi_1 \vee \psi_2$ iff $\mathbf{M}, [d_i, d_j] \Vdash \psi_1$, or $\mathbf{M}, [d_i, d_j] \Vdash \psi_2$;
- $\mathbf{M}, [d_i, d_j] \Vdash \langle A\rangle\psi$ (resp., $\Diamond_r\psi$) iff $\exists d_k \in D$, $d_k > d_j$ (resp., $d_k \geq d_j$), such that $\mathbf{M}, [d_j, d_k] \Vdash \psi$.

We place ourselves in the most general setting and we do not impose any constraint on the valuation function. In particular, given interval $[d_i, d_j]$, it may happen that $p \in \mathcal{V}([d_i, d_j])$ and $p \notin \mathcal{V}([d'_i, d'_j])$ for all intervals $[d'_i, d'_j]$ (strictly) contained in $[d_i, d_j]$.

Let d_0 be the initial point of D and let d_1 be its successor. Since our logic has only future time operators, we can restrict our attention to the *initial interval* $[d_0, d_1]$ of $\mathbb{I}(\mathbb{D})$. From now on, we shall say that a formula φ is *satisfiable* if and only if there exists a model $\mathbf{M} = \langle\langle\mathbb{D}, \mathbb{I}(\mathbb{D})\rangle, \mathcal{V}\rangle$ such that $\mathbf{M}, [d_0, d_1] \Vdash \varphi$, where $[d_0, d_1]$ is the initial interval of $\mathbb{I}(\mathbb{D})$.

3. An intuitive account of the proposed solution

In this section we give an intuitive account of the proposed solution to the satisfiability problem for RPNL$^-$. More precisely, we introduce the main features of a model building process that, given a formula φ to be checked for satisfiability, generates a model for it (if any) step by step. Such a process takes into consideration one element of the temporal domain at a time and, at each step, it progresses from one time point to the next one. For the moment, we completely ignore the problem of termination. In the following, we shall show how to turn this process into an effective procedure.

Let $D = \{d_0, d_1, d_2, \ldots\}$ be the temporal domain, which we assumed to be isomorphic to $\mathbb{N}$ or to a prefix of it. The model building process begins from the time point d_1 by considering the initial interval $[d_0, d_1]$. It associates with $[d_0, d_1]$ the set $A_{[d_0,d_1]}$ of all and only the formulae which hold over it.

Next, it moves from d_1 to its immediate successor d_2 and it takes into consideration the two intervals ending in d_2, namely, $[d_0, d_2]$ and $[d_1, d_2]$. As before, it associates with $[d_1, d_2]$ (resp. $[d_0, d_2]$) the set $A_{[d_1,d_2]}$ (resp. $A_{[d_0,d_2]}$) of all and only the formulae which hold over $[d_1, d_2]$ (resp. $[d_0, d_2]$). Since $[d_1, d_2]$ is a right neighbor of $[d_0, d_1]$, if $[A]\psi$ holds over $[d_0, d_1]$, then ψ must hold over $[d_1, d_2]$. Hence, for every formula $[A]\psi$ in $A_{[d_0,d_1]}$, it puts ψ in $A_{[d_1,d_2]}$. Moreover, since every interval which is a right neighbor of $[d_0, d_2]$ is also a right neighbor of $[d_1, d_2]$, and vice versa, for every formula ψ of the form $\langle A\rangle\xi$ or $[A]\xi$, ψ holds over $[d_0, d_2]$ if and only if it holds over $[d_1, d_2]$. Accordingly, it requires that $\psi \in A_{[d_0,d_2]}$ if and only if $\psi \in A_{[d_1,d_2]}$. Let us denote by $\mathrm{REQ}(d_2)$ the set of formulae of the form $\langle A\rangle\psi$ or $[A]\psi$ which hold over an interval ending in d_2 (by analogy, let $\mathrm{REQ}(d_1)$ be the set of formulae of the form $\langle A\rangle\psi$ or $[A]\psi$ which hold over an interval ending in d_1, that is, the formulae $\langle A\rangle\psi$ or $[A]\psi$ which hold over $[d_0, d_1]$).

Next, the process moves from d_2 to its immediate successor d_3 and it takes into consideration the three intervals ending in d_3, namely, $[d_0, d_3]$, $[d_1, d_3]$, and $[d_2, d_3]$. As at the previous steps, for $i = 0, 1, 2$, it associates the set $A_{[d_i,d_3]}$ with $[d_i, d_3]$. Since $[d_1, d_3]$ is a right neighbor of $[d_0, d_1]$, for every formula $[A]\psi \in \mathrm{REQ}(d_1)$, $\psi \in A_{[d_1,d_3]}$. Moreover, $[d_2, d_3]$ is a right neighbor of both $[d_0, d_2]$ and $[d_1, d_2]$, and thus for every formula $[A]\psi \in \mathrm{REQ}(d_2)$, $\psi \in A_{[d_2,d_3]}$. Finally, for every formula ψ of the form $\langle A\rangle\xi$ or $[A]\xi$, we have that $\psi \in A_{[d_0,d_3]}$ if and only if $\psi \in A_{[d_1,d_3]}$ if and only if $\psi \in A_{[d_2,d_3]}$.

Next, the process moves from d_3 to its successor d_4 and it repeats the same operations, and so on.

The layered structure generated by the process is graphically depicted in Figure 1. The first layer correspond to time point d_1, and for all $i > 1$, the i-th layer corresponds to time point d_i. If we associate with each node $A_{[d_i,d_j]}$ the corresponding interval $[d_i, d_j]$, we can interpret the set of edges as the neighborhood relation between pairs of intervals. As a general rule, given a time point $d_j \in D$, for every $d_i < d_j$, the set $A_{[d_i,d_j]}$ of all and only the formulae which hold over $[d_i, d_j]$ satisfies the following conditions:

- since $[d_i, d_j]$ is a right neighbor of every interval ending in d_i, for every formula $[A]\psi \in \mathrm{REQ}(d_i)$, $\psi \in A_{[d_i,d_j]}$;

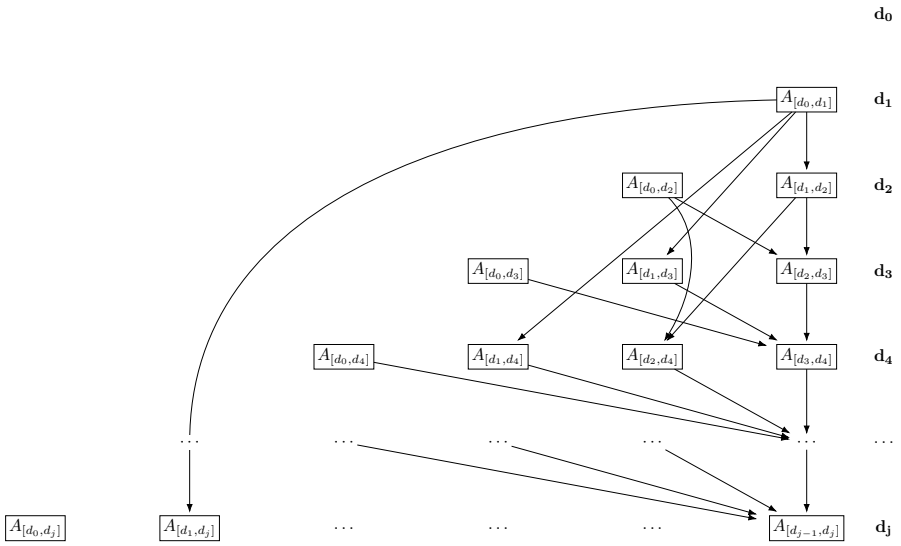


Figure 1. The layered structure

– since every right neighbor of $[d_i, d_j]$ is also a right neighbor of all intervals $[d_k, d_j]$ belonging to layer d_j, for every formula ψ of the form $\langle A\rangle\xi$ or $[A]\xi$, $\psi \in A_{[d_i,d_j]}$ if and only if it belongs to all sets $A_{[d_k,d_j]}$ belonging to the layer.

In [4], Bresolin and Montanari turn such a model building process into an effective tableau-based decision procedure for RPNL$^-$. Given an RPNL$^-$ formula φ, the procedure builds a tableau for φ whose (macro)nodes correspond to the layers of the structure in Figure 1 and whose edges connect pairs of nodes that correspond to consecutive layers. Unlike other tableau methods for interval temporal logics, where each node corresponds to a single interval [8, 11], such a method associates any *set* of intervals $[d_i, d_j]$ ending at the same point d_j with a single node, whose label consists of a set of sets of formulae $A_{[d_i,d_j]}$ (one for every interval ending in d_j). Moreover, two nodes are connected by an edge (only) if their labels satisfy suitable constraints encoding the neighborhood relation among the associated intervals. Formulae devoid of temporal operators as well as formulae of the form $[A]\psi$ are satisfied by construction. Establishing the satisfiability of φ thus reduces to finding a (possibly infinite) path of nodes on which formulae of the form $\langle A\rangle\psi$ are satisfied as well (*fulfilling path*). To find such a path, the decision procedure first generates the whole (finite) tableau for φ; then it progressively removes parts of the tableau that cannot participate in a fulfilling path. It can be proved that φ is satisfiable if and only if the final tableau obtained by this pruning process is not empty.

As for the computational complexity, we have that the number of nodes of the tableau is $2^{2^{O(|\varphi|)}}$ and that, to determine the existence of a fulfilling path, the algorithm may take time polynomial in the number of nodes. Hence, the algorithm has a time complexity that is doubly exponential in the size of φ. Its performance can be improved by exploiting nondeterminism to guess a fulfilling path for the formula φ. In such a case, the fulfilling path can be built one node at a time: at each step, the procedure guesses the next node in the path and it moves from the current node to such a node. Since every (macro)node maintains the set of existential temporal formulae which have not been satisfied yet, at any time the algorithm basically needs to store only a pair of consecutive nodes in the path, namely, the current and the next ones, rather than the entire path. Hence, such a nondeterministic variant of the algorithm needs an amount of space which is exponential in the size of the formula, thus providing an EXPSPACE decision procedure for RPNL$^-$.

In the following, we shall develop an alternative NEXPTIME decision procedure for RPNL$^-$, interpreted over natural numbers, and we shall prove its optimality. Such a procedure follows the above-described approach, but its nodes are the single sets $A_{[d_i,d_j]}$, instead of layers, of the structure depicted in Figure 1. In such a way, the procedure avoids the double exponential blow-up of the method given in [4].

4. Labelled Interval Structures and satisfiability

In this section we introduce some preliminary notions and we establish some basic results on which our tableau method for RPNL$^-$ relies.

Let φ be an RPNL$^-$ formula to be checked for satisfiability and let AP be the set of its propositional letters. For the sake of brevity, we use $(A)\psi$ as a shorthand for both $\langle A\rangle\psi$ and $[A]\psi$.

Definition 4.1. The *closure* $\mathrm{CL}(\varphi)$ of φ is the set of all subformulae of φ and of their negations (we identify $\neg\neg\psi$ with ψ).

Definition 4.2. The set of *temporal requests* of φ is the set $\mathrm{TF}(\varphi)$ of all temporal formulae in $\mathrm{CL}(\varphi)$, that is, $\mathrm{TF}(\varphi) = \{(A)\psi \in \mathrm{CL}(\varphi)\}$.

By induction on the structure of φ, we can easily prove the following proposition.

Proposition 4.3. *For every formula* φ, $|\,\mathrm{CL}(\varphi)|$ *is less than or equal to* $2 \cdot |\varphi|$, *while* $|\,\mathrm{TF}(\varphi)|$ *is less than or equal to* $2 \cdot (|\varphi| - 1)$.

The notion of φ-atom is defined in the standard way.

Definition 4.4. A φ-*atom* is a set $A \subseteq \mathrm{CL}(\varphi)$ such that:

- for every $\psi \in \mathrm{CL}(\varphi)$, $\psi \in A$ iff $\neg\psi \notin A$;
- for every $\psi_1 \vee \psi_2 \in \mathrm{CL}(\varphi)$, $\psi_1 \vee \psi_2 \in A$ iff $\psi_1 \in A$ or $\psi_2 \in A$.

We denote the set of all φ-atoms by A_φ. We have that $|A_\varphi| \leq 2^{|\varphi|}$. Atoms are connected by the following binary relation.

Definition 4.5. Let R_φ be a binary relation over A_φ such that, for every pair of atoms $A, A' \in A_\varphi$, $A \; R_\varphi \; A'$ if and only if, for every $[A]\psi \in \mathrm{CL}(\varphi)$, if $[A]\psi \in A$, then $\psi \in A'$.

We now introduce a suitable labelling of interval structures based on φ-atoms.

Definition 4.6. A φ-*labelled interval structure* (LIS for short) is a pair $\mathbf{L} = \langle \langle \mathbb{D}, \mathbb{I}(\mathbb{D})^- \rangle, \mathcal{L} \rangle$, where $\langle \mathbb{D}, \mathbb{I}(\mathbb{D})^- \rangle$ is an interval structure and $\mathcal{L} : \mathbb{I}(\mathbb{D})^- \to A_\varphi$ is a *labelling function* such that, for every pair of neighboring intervals $[d_i, d_j], [d_j, d_k] \in \mathbb{I}(\mathbb{D})^-$, $\mathcal{L}([d_i, d_j]) \; R_\varphi \; \mathcal{L}([d_j, d_k])$.

If we interpret the labelling function as a valuation function, LISs represent *candidate models* for φ. The truth of formulae devoid of temporal operators and that of $[A]$-formulae indeed follow from the definition of φ-atom and the definition of R_φ, respectively. However, to obtain a model for φ we must also guarantee the truth of $\langle A \rangle$-formulae. To this end, we introduce the notion of fulfilling LIS.

Definition 4.7. A φ-labelled interval structure $\mathbf{L} = \langle \langle \mathbb{D}, \mathbb{I}(\mathbb{D})^- \rangle, \mathcal{L} \rangle$ is *fulfilling* if and only if, for every temporal formula $\langle A \rangle \psi \in \mathrm{TF}(\varphi)$ and every interval $[d_i, d_j] \in \mathbb{I}(\mathbb{D})^-$, if $\langle A \rangle \psi \in \mathcal{L}([d_i, d_j])$, then there exists $d_k > d_j$ such that $\psi \in \mathcal{L}([d_j, d_k])$.

The following theorem proves that for any given formula φ, the satisfiability of φ is equivalent to the existence of a fulfilling LIS with the initial interval labelled by φ. The implication from left to right is straightforward; the opposite implication is proved by induction on the structure of the formula.

Theorem 4.8. *A formula φ is satisfiable if and only if there exists a fulfilling* LIS $\mathbf{L} = \langle \langle \mathbb{D}, \mathbb{I}(\mathbb{D})^- \rangle, \mathcal{L} \rangle$ *with* $\varphi \in \mathcal{L}([d_0, d_1])$.

Proof. Let φ be a satisfiable formula and let $\mathbf{M} = \langle \langle \mathbb{D}, \mathbb{I}(\mathbb{D})^- \rangle, \mathcal{V} \rangle$ be a model for it. We define a LIS $\mathbf{L_M} = \langle \langle \mathbb{D}, \mathbb{I}(\mathbb{D})^- \rangle, \mathcal{L}_\mathbf{M} \rangle$ such that for every interval $[d_i, d_j] \in \mathbb{I}(\mathbb{D})^-$, $\mathcal{L}_\mathbf{M}([d_i, d_j]) = \{\psi \in \mathrm{CL}(\varphi) : \mathbf{M}, [d_i, d_j] \Vdash \psi\}$. It is immediate to prove that $\mathbf{L_M}$ is a fulfilling LIS and $\varphi \in \mathcal{L}_\mathbf{M}([d_0, d_1])$.

As for the opposite implication, let $\mathbf{L} = \langle \langle \mathbb{D}, \mathbb{I}(\mathbb{D})^- \rangle, \mathcal{L} \rangle$ be a fulfilling LIS with $\varphi \in \mathcal{L}([d_0, d_1])$. We define a model $\mathbf{M_L} = \langle \langle \mathbb{D}, \mathbb{I}(\mathbb{D})^- \rangle, \mathcal{V}_\mathbf{L} \rangle$

such that for every interval $[d_i, d_j] \in \mathbb{I}(\mathbb{D})^-$ and every propositional letter $p \in AP$, $p \in \mathcal{V}_{\mathbf{L}}([d_i, d_j])$ if and only if $p \in \mathcal{L}([d_i, d_j])$. We prove by induction on the structure of φ that for every $\psi \in \mathrm{CL}(\varphi)$ and every interval $[d_i, d_j] \in \mathbb{I}(\mathbb{D})^-$, $\mathbf{M_L}, [d_i, d_j] \Vdash \psi$ if and only if $\psi \in \mathcal{L}([d_i, d_j])$. Since $\varphi \in \mathcal{L}([d_0, d_1])$, we can conclude that $\mathbf{M_L}, [d_0, d_1] \Vdash \varphi$.

- If ψ is the propositional letter p, then $p \in \mathcal{L}([d_i, d_j]) \overset{\mathcal{V}_{\mathbf{L}} \text{ def.}}{\iff} p \in \mathcal{V}_{\mathbf{L}}([d_i, d_j]) \iff \mathbf{M_L}, [d_i, d_j] \Vdash p$.
- If ψ is the formula $\neg\xi$, then $\neg\xi \in \mathcal{L}([d_i, d_j]) \overset{\text{atom def.}}{\iff} \xi \notin \mathcal{L}([d_i, d_j]) \overset{\text{ind. hyp.}}{\iff} \mathbf{M_L}, [d_i, d_j] \nVdash \xi \iff \mathbf{M_L}, [d_i, d_j] \Vdash \neg\xi$.
- If ψ is the formula $\xi_1 \vee \xi_2$, then $\xi_1 \vee \xi_2 \in \mathcal{L}([d_i, d_j]) \overset{\text{atom def.}}{\iff} \xi_1 \in \mathcal{L}([d_i, d_j])$ or $\xi_2 \in \mathcal{L}([d_i, d_j]) \overset{\text{ind. hyp.}}{\iff} \mathbf{M_L}, [d_i, d_j] \Vdash \xi_1$ or $\mathbf{M_L}, [d_i, d_j] \Vdash \xi_2 \iff \mathbf{M_L}, [d_i, d_j] \Vdash \xi_1 \vee \xi_2$.
- Let ψ be the formula $\langle A \rangle \xi$. Suppose that $\langle A \rangle \xi \in \mathcal{L}([d_i, d_j])$. Since $\mathbf{L}$ is fulfilling, there exists an interval $[d_j, d_k] \in \mathbb{I}(\mathbb{D})^-$ such that $\xi \in \mathcal{L}([d_j, d_k])$. By inductive hypothesis, we have that $\mathbf{M_L}, [d_j, d_k] \Vdash \xi$, and hence $\mathbf{M_L}, [d_i, d_j] \Vdash \langle A \rangle \xi$. As for the opposite implication, assume by contradiction that $\mathbf{M_L}, [d_i, d_j] \Vdash \langle A \rangle \xi$, but $\langle A \rangle \xi \notin \mathcal{L}([d_i, d_j])$. By atom definition, this implies that $\neg\langle A \rangle \xi = [A]\neg\xi \in \mathcal{L}([d_i, d_j])$. By definition of LIS, we have that, for every $d_k > d_j$, $\mathcal{L}([d_i, d_j])\ R_\varphi\ \mathcal{L}([d_j, d_k])$, and thus $\neg\xi \in \mathcal{L}([d_j, d_k])$. By inductive hypothesis, this implies that $\mathbf{M_L}, [d_j, d_k] \Vdash \neg\xi$ for every $d_k > d_j$, and hence $\mathbf{M_L}, [d_i, d_j] \Vdash [A]\neg\xi$, which contradicts the hypothesis that $\mathbf{M_L}, [d_i, d_j] \Vdash \langle A \rangle \xi$. □

Theorem 4.8 reduces the satisfiability problem for φ to the problem of finding a fulfilling LIS with the initial interval labelled by φ. From now on, we say that a fulfilling LIS $\mathbf{L} = \langle \langle \mathbb{D}, \mathbb{I}(\mathbb{D})^- \rangle, \mathcal{L} \rangle$ *satisfies* φ if and only if $\varphi \in \mathcal{L}([d_0, d_1])$.

Since fulfilling LISs satisfying φ may be arbitrarily large or even infinite, we must find a way to finitely establish their existence. In the following, we first give a bound on the size of finite fulfilling LISs that must be checked for satisfiability, when searching for finite φ-models; then, we show that we can restrict ourselves to infinite fulfilling LISs with a finite bounded representation, when searching for infinite φ-models. To prove these results, we take advantage of the following two fundamental properties of LISs:

1. *The labellings of a pair of intervals $[d_i, d_j], [d_k, d_j]$ with the same right endpoint must agree on temporal formulae.*
   Since every right neighbor of $[d_i, d_j]$ is also a right neighbor of $[d_k, d_j]$, we have that for every existential formula $\langle A \rangle \psi \in \mathrm{TF}(\varphi)$,

$\langle A\rangle\psi \in \mathcal{L}([d_i, d_j])$ iff $\langle A\rangle\psi \in \mathcal{L}([d_k, d_j])$ (it easily follows from Definitions 4.4, 4.5, and 4.6). The same holds for universal formulae $[A]\psi$.

2. $\frac{|\mathrm{TF}(\varphi)|}{2}$ *right neighboring intervals suffice to fulfill the existential formulae belonging to the labelling of an interval* $[d_i, d_j]$.
The number of right neighboring intervals which are needed to fulfill all existential formulae of $\mathcal{L}([d_i, d_j])$ is bounded by the number of $\langle A\rangle$-formulae in $\mathrm{TF}(\varphi)$, which is equal to $\frac{|\mathrm{TF}(\varphi)|}{2}$ (in the worst case, different existential formulae are satisfied by different right neighboring intervals).

Definition 4.9. Given a LIS $\mathbf{L} = \langle\langle\mathbb{D}, \mathbb{I}(\mathbb{D})^-\rangle, \mathcal{L}\rangle$ and $d \in D$, we denote by $\mathrm{REQ}(d)$ the set of all and only the temporal formulae belonging to the labellings of the intervals ending in d.

We denote by $\mathrm{REQ}(\varphi)$ the set of all possible sets of requests. It is not difficult to show that $|\,\mathrm{REQ}_\varphi\,|$ is equal to $2^{\frac{\mathrm{TF}(\varphi)}{2}}$.

Definition 4.10. Given a LIS $\mathbf{L} = \langle\langle\mathbb{D}, \mathbb{I}(\mathbb{D})^-\rangle, \mathcal{L}\rangle$, a set of points $D' \subseteq D$, and a set of temporal formulae $\mathcal{R} \subseteq \mathrm{TF}(\varphi)$, we say that $\mathcal{R}$ *occurs* n *times in* D' if and only if there exist exactly n distinct points $d_{i_1}, \ldots, d_{i_n} \in D'$ such that $\mathrm{REQ}(d_{i_j}) = \mathcal{R}$, for all $1 \leq j \leq n$.

Theorem 4.11. *Let* $\mathbf{L} = \langle\langle\mathbb{D}, \mathbb{I}(\mathbb{D})^-\rangle, \mathcal{L}\rangle$ *be a* finite *fulfilling* LIS *that satisfies* φ *and let* $m = \frac{|\mathrm{TF}(\varphi)|}{2}$. *Then there exists a* finite *fulfilling* LIS $\overline{\mathbf{L}} = \langle\langle\overline{\mathbb{D}}, \mathbb{I}(\overline{\mathbb{D}})^-\rangle, \overline{\mathcal{L}}\rangle$ *that satisfies* φ *such that, for every* $\overline{d}_i \in \overline{D}$, $\mathrm{REQ}(\overline{d}_i)$ *occurs at most* m *times in* $\overline{D} \setminus \{d_1\}$.

Proof. Let $\mathbf{L} = \langle\langle\mathbb{D}, \mathbb{I}(\mathbb{D})^-\rangle, \mathcal{L}\rangle$ be a finite fulfilling LIS that satisfies φ. If for every $d_i \in D$, $\mathrm{REQ}(d_i)$ occurs at most m times in $D \setminus \{d_1\}$, we are done. If this is not the case, we show how to build a fulfilling LIS with the requested property by progressively removing exceeding points from D.

Let $\mathbf{L}_0 = \mathbf{L}$ and let $\mathcal{R}_0 = \{\mathrm{REQ}_1, \mathrm{REQ}_2, \ldots, \mathrm{REQ}_k\}$ be the (arbitrarily ordered) finite set of all and only the sets of requests that occur more than m times in $D \setminus \{d_1\}$. We show how to turn $\mathbf{L}_0$ into a fulfilling LIS $\mathbf{L}_1 = \langle\langle\mathbb{D}_1, \mathbb{I}(\mathbb{D}_1)^-\rangle, \mathcal{L}_1\rangle$ satisfying φ, which, unlike $\mathbf{L}_0$, contains exactly m points $d \in D_1 \setminus \{d_1\}$ such that $\mathrm{REQ}(d) = \mathrm{REQ}_1$. By iterating such a transformation $k-1$ times, we can turn $\mathbf{L}_1$ into a fulfilling LIS devoid of exceeding points that satisfies φ.

The fulfilling LIS $\mathbf{L}_1 = \langle\langle\mathbb{D}_1, \mathbb{I}(\mathbb{D}_1)^-\rangle, \mathcal{L}_1\rangle$ can be obtained as follows. Let $d_{j_1}, d_{j_2}, \ldots, d_{j_n}$, with $d_1 < d_{j_1} < d_{j_2} < \ldots < d_{j_n}$, be the n points in D, with $n > m$, such that $\mathrm{REQ}(d_{j_i}) = \mathrm{REQ}_1$. We define

$\mathbb{D}'_1 = \langle D \setminus \{d_{j_1}\}, < \rangle$ and $\mathcal{L}'_1 = \mathcal{L}|_{\mathbb{I}(\mathbb{D}'_1)^-}$ (the restriction of $\mathcal{L}$ to the intervals on $\mathbb{D}'_1$). The pair $\mathbf{L}'_1 = \langle \langle \mathbb{D}'_1, \mathbb{I}(\mathbb{D}'_1)^- \rangle, \mathcal{L}'_1 \rangle$ is obviously a finite LIS, but it is not necessarily a fulfilling one. The removal of d_{j_1} causes the removal of all intervals either beginning or ending at d_{j_1}. While the removal of intervals beginning at d_{j_1} is not critical (intervals ending at d_{j_1} are removed as well), there can be some points $d < d_{j_1}$ such that some formulae $\langle A \rangle \psi \in \mathrm{REQ}(d)$ are fulfilled in $\mathbf{L}_0$, but they are not fulfilled in $\mathbf{L}'_1$ anymore. We fix such defects (if any) one-by-one by properly redefining $\mathcal{L}'_1$. Let $d < d_{j_1}$ and $\langle A \rangle \psi \in \mathrm{REQ}(d)$ such that $\psi \in \mathcal{L}([d, d_{j_1}])$ and there exists no $d' \in D \setminus \{d_{j_1}\}$ such that $\psi \in \mathcal{L}'_1([d, d'])$. Since $\mathrm{REQ}(d)$ contains at most m $\langle A \rangle$-formulae, there exists at least one point $d_{j_i} \in \{d_{j_2}, \ldots, d_{j_n}\}$ such that the atom $\mathcal{L}'_1([d, d_{j_i}])$ either fulfills no $\langle A \rangle$-formulae or it fulfills only $\langle A \rangle$-formulae which are also fulfilled by some other φ-atom $\mathcal{L}'_1([d, d'])$. Let d_{j_i} one of such "useless" points. We can redefine $\mathcal{L}'_1([d, d_{j_i}])$ by putting $\mathcal{L}'_1([d, d_{j_i}]) = \mathcal{L}([d, d_{j_1}])$, thus fixing the problem with $\langle A \rangle \psi \in \mathrm{REQ}(d)$. Notice that, since $\mathrm{REQ}(d_{j_1}) = \mathrm{REQ}(d_{j_i}) = \mathrm{REQ}_1$, such a change has no impact on the right neighboring intervals of $[d, d_{j_i}]$. In a similar way, we can fix the other possible defects caused by the removal of d_{j_1}. We repeat such a process until we remain with exactly m distinct points d such that $\mathrm{REQ}(d) = \mathrm{REQ}_1$. Let $\mathbf{L}_1 = \langle \langle \mathbb{D}_1, \mathbb{I}(\mathbb{D}_1) \rangle, \mathcal{L}_1 \rangle$ be the resulting LIS. It is immediate to show that it is fulfilling and that it satisfies φ. □

To deal with the case of infinite (fulfilling) LISs, we introduce the notion of *ultimately periodic* LIS.

Definition 4.12. An infinite LIS $\mathbf{L} = \langle \langle \mathbb{D}, \mathbb{I}(\mathbb{D})^- \rangle, \mathcal{L} \rangle$ is *ultimately periodic*, with prefix l and period $p > 0$, if and only if for all $i > l$, $\mathrm{REQ}(d_i) = \mathrm{REQ}(d_{i+p})$.

The following theorem shows that if there exists an infinite fulfilling LIS that satisfies φ, then there exists an ultimately periodic fulfilling one that satisfies φ. Furthermore, it provides a bound to the prefix and period of such a fulfilling LIS which closely resembles the one that we established for finite fulfilling LISs.

Theorem 4.13. *Let* $\mathbf{L} = \langle \langle \mathbb{D}, \mathbb{I}(\mathbb{D})^- \rangle, \mathcal{L} \rangle$ *be an* infinite *fulfilling* LIS *that satisfies* φ *and let* $m = \frac{|\mathrm{TF}(\varphi)|}{2}$. *Then there exists an ultimately periodic fulfilling* LIS $\overline{\mathbf{L}} = \langle \langle \overline{\mathbb{D}}, \mathbb{I}(\overline{\mathbb{D}})^- \rangle, \overline{\mathcal{L}} \rangle$, *with prefix* l *and period* p, *that satisfies* φ *such that:*

1. *for every pair of points* $\overline{d}_i, \overline{d}_j \in \overline{D}$, *with* $\overline{d}_0 \leq \overline{d}_i \leq \overline{d}_l$ *and* $\overline{d}_j > \overline{d}_l$, $\mathrm{REQ}(\overline{d}_i) \neq \mathrm{REQ}(\overline{d}_j)$, *that is, points belonging to the prefix and points belonging to the period have different sets of requests;*

2. *for every* $\overline{d}_i \in \overline{D}$*, with* $\overline{d}_0 \leq \overline{d}_i \leq \overline{d}_l$, $\mathrm{REQ}(\overline{d}_i)$ *occurs at most* m *times in* $\{\overline{d}_2, \ldots, \overline{d}_l\}$*;*
3. *for every pair of points* $\overline{d}_i, \overline{d}_j \in \overline{D}$*, with* $\overline{d}_{l+1} \leq \overline{d}_i, \overline{d}_j \leq \overline{d}_{l+p}$*, if* $i \neq j$*, then* $\mathrm{REQ}(\overline{d}_i) \neq \mathrm{REQ}(\overline{d}_j)$*.*

Proof. Let φ be a satisfiable formula and let $\mathbf{L} = \langle \langle \mathbb{D}, \mathbb{I}(\mathbb{D})^- \rangle, \mathcal{L} \rangle$ be an infinite fulfilling LIS that satisfies φ. We define the following sets:

- $Fin(\mathbf{L}) = \{\mathrm{REQ}(d_i)$: there exists a *finite* number of points $d \in \mathbb{D}$ such that $\mathrm{REQ}(d) = \mathrm{REQ}(d_i)\}$;
- $Inf(\mathbf{L}) = \{\mathrm{REQ}(d_i)$: there exists an *infinite* number of points $d \in \mathbb{D}$ such that $\mathrm{REQ}(d) = \mathrm{REQ}(d_i)\}$.

We build an infinite ultimately periodic LIS $\overline{\mathbf{L}}$, with prefix $l \leq m \cdot |Fin(\mathbf{L})| + 1$ and period $p = |Inf(\mathbf{L})|$, that satisfies φ (and respects Conditions 1 - 3) as follows.

Let $n \in \mathbb{N}$ be the index of the smallest point in $\mathbb{D}$ such that, for every $i \geq n$, $\mathrm{REQ}(d_i) \in Inf(\mathbf{L})$. We first collect and (arbitrarily) enumerate the elements of $Inf(\mathbf{L})$, that is, let $Inf(\mathbf{L}) = \{\mathrm{REQ}_0, \ldots, \mathrm{REQ}_{p-1}\}$. The cardinality p of $Inf(\mathbf{L})$ is the period of $\overline{\mathbf{L}}$. As a first step, we define an ultimately periodic LIS $\mathbf{L}' = \langle \langle \mathbb{D}, \mathbb{I}(\mathbb{D})^- \rangle, \mathcal{L}' \rangle$ (which respects Condition 3), with prefix $n-1$ and period p, in such a way that:

- for all $0 \leq i < n$, $\mathrm{REQ}(d_i)$ remains unchanged;
- for all $i \geq n$, $\mathrm{REQ}(d_i)$ becomes $\mathrm{REQ}_{(i-n) \bmod p}$.

To make the ultimately periodic LIS $\mathbf{L}'$ a fulfilling LIS satisfying φ which respects the above conditions on $\mathrm{REQ}(d_i)$, for all $i \geq 0$, we must properly define the labelling $\mathcal{L}'$. We distinguish two cases:

(i) $d_1 \leq d_i < d_n$. Since $\mathbf{L}$ is a fulfilling LIS, for all $\langle A \rangle \psi \in \mathrm{REQ}(d_i)$ there exists a point $d_j > d_i$ such that $\psi \in \mathcal{L}([d_i, d_j])$. Two cases may arise:

- if $d_j < d_n$, we put $\mathcal{L}'([d_i, d_j]) = \mathcal{L}([d_i, d_j])$;
- if $d_j \geq d_n$, then $\mathrm{REQ}(d_j)$ in $\mathbf{L}$ belongs to $Inf(\mathbf{L})$. We have that there exist infinitely many points $d_k \geq d_n$ such that $\mathrm{REQ}(d_k)$ in $\mathbf{L}'$ is equal to $\mathrm{REQ}(d_j)$ in $\mathbf{L}$. Let d_k be one of such points for which the labelling $\mathcal{L}'([d_i, d_k])$ has not been defined yet. We fulfill $\langle A \rangle \psi$ in $\mathbf{L}'$ by putting $\mathcal{L}'([d_i, d_k]) = \mathcal{L}([d_i, d_j])$.

We repeat such a construction until we fulfill (in $\mathbf{L}'$) all $\langle A \rangle$-formulae belonging to $\mathrm{REQ}(d_i)$. To complete the labelling $\mathcal{L}'$ on the remaining intervals $[d_i, d_k]$, we (arbitrarily) put $\mathcal{L}'([d_i, d_k]) = \mathcal{L}([d_i, d_j])$, with d_j such that $\mathrm{REQ}(d_k)$ in $\mathbf{L}'$ is equal to $\mathrm{REQ}(d_j)$ in $\mathbf{L}$.

(ii) $d_i \geq d_n$. We have that there exists a point $d_j \geq d_n$ such that $\mathrm{REQ}(d_j)$ in $\mathbf{L}$ is equal to $\mathrm{REQ}(d_i)$ in $\mathbf{L}'$. Let $\langle A\rangle\psi \in \mathrm{REQ}(d_i)$ (in $\mathbf{L}'$). Since $\mathbf{L}$ is fulfilling, there exists a point $d_k > d_j$, with $\mathrm{REQ}(d_k) \in Inf(\mathbf{L})$, such that $\psi \in \mathcal{L}([d_j, d_k])$. From $\mathrm{REQ}(d_k) \in Inf(\mathbf{L})$, it follows that there exist infinitely many points $d_l > d_i$ such that $\mathrm{REQ}(d_l)$ in $\mathbf{L}'$ is equal to $\mathrm{REQ}(d_k)$ in $\mathbf{L}$. Let d_l be one of such points for which the labelling $\mathcal{L}'([d_i, d_l])$ has not been defined yet. We fulfill $\langle A\rangle\psi$ by putting $\mathcal{L}'([d_i, d_l]) = \mathcal{L}([d_j, d_k])$. We can repeat such a construction until we fulfill (in $\mathbf{L}'$) all $\langle A\rangle$-formulae belonging to $\mathrm{REQ}(d_i)$. To complete the labelling of the remaining intervals $[d_i, d_l]$, we arbitrarily put $\mathcal{L}'([d_i, d_l]) = \mathcal{L}([d_j, d_k])$, with d_k such that $\mathrm{REQ}(d_k)$ in $\mathbf{L}$ is equal to $\mathrm{REQ}(d_l)$ in $\mathbf{L}'$.

Condition 3 of the theorem is respected by construction. Conditions 1 and 2 require that the prefix $\{d_0, \ldots, d_{n-1}\}$ does not include points d such that $\mathrm{REQ}(d) \in Inf(\mathbf{L})$ and that for every point d in $\{d_0, \ldots, d_{n-1}\}$, $\mathrm{REQ}(d)$ occurs at most m times in $\{d_2, \ldots, d_{n-1}\}$, respectively. These two conditions are not necessarily guaranteed by $\mathbf{L}'$. We turn $\mathbf{L}'$ into a fulfilling ultimately periodic LIS $\overline{\mathbf{L}}$ satisfying φ that respects conditions 1 and 2 by means of a two-step removal process.

- **Step 1.** We replace the LIS $\mathbf{L}'$ by a LIS $\mathbf{L}'' = \langle\langle \mathbb{D}'', \mathbb{I}(\mathbb{D}'')^- \rangle, \mathcal{L}''\rangle$ which is obtained from $\mathbf{L}'$ by deleting all points $d_1 < d_i < d_n$ such that $\mathrm{REQ}(d_i) \in Inf(\mathbf{L})$. The resulting LIS $\mathbf{L}''$ is still ultimately periodic, but it is not necessarily fulfilling. As an effect of the removal of d_i, there can be some point d in the prefix such that some formula $\langle A\rangle\psi \in \mathrm{REQ}(d)$ was fulfilled in $\mathcal{L}'$, but it is not fulfilled in $\mathcal{L}''$ anymore. We can fix such a defect by redefining $\mathcal{L}''$ in an appropriate way. Since $\mathbf{L}'$ is fulfilling, there exists a point $d_j > d$ such that $\psi \in \mathcal{L}'([d, d_j])$ and d_j does not belong to $\mathbf{L}''$. This happens if (and only if) $\mathrm{REQ}(d_j) \in Inf(\mathbf{L})$ and thus there exist infinitely many points d_k such that $\mathrm{REQ}(d_k)$ in $\mathbf{L}''$ is equal to $\mathrm{REQ}(d_j)$ in $\mathbf{L}'$. Let d_k be one of these points which is "useless" (see the proof of Theorem 4.11). We can fulfill $\langle A\rangle\psi$ by putting $\mathcal{L}''([d, d_k]) = \mathcal{L}'([d, d_j])$. In a similar way, we can fix the other possible defects caused by the removal of d_i. By repeating such a process for every other point that must be removed (if any), we guarantee that $\mathbf{L}''$ is a fulfilling ultimately periodic LIS that satisfies φ.
- **Step 2.** If there exists some point d'' belonging to the prefix $\{d''_0, \ldots, d''_h\}$ of $\mathbf{L}''$ such that $\mathrm{REQ}(d'')$ occurs more than m times in $\{d''_2, \ldots, d''_h\}$, we can proceed as in the proof of Theorem 4.11 to obtain a new fulfilling ultimately periodic LIS $\overline{\mathbf{L}}$ satisfying φ with

prefix $\{\overline{d}_0, \ldots, \overline{d}_l\}$ such that every set of requests of the prefix is repeated at most m times in $\{\overline{d}_2, \ldots, \overline{d}_l\}$. □

5. The complexity of the satisfiability problem for RPNL$^-$

In this section we provide a precise characterization of the computational complexity of the satisfiability problem for RPNL$^-$.

5.1. An upper bound to the computational complexity

A decision procedure for RPNL$^-$ can be derived from the results of Section 4 in a straightforward way. Theorems 4.11 and 4.13 indeed provide a bound on the size of the LISs to be checked:

- by Theorem 4.11, we have that if there exists a finite LIS satisfying φ, then there exists a finite one of size less than or equal to $|\,\mathrm{REQ}_\varphi\,| \cdot \frac{|\,\mathrm{TF}(\varphi)|}{2} + 1$ which satisfies φ;
- by Theorem 4.13, we have that if there exists an infinite LIS satisfying φ, then there exists an ultimately periodic one, with prefix $l \leq |\,\mathrm{REQ}_\varphi\,| \cdot \frac{|\,\mathrm{TF}(\varphi)|}{2} + 1$ and period $p \leq |\,\mathrm{REQ}_\varphi\,|$, which satisfies φ.

A simple decision algorithm to check the satisfiability of an RPNL$^-$ formula φ that nondeterministically guesses a LIS $\mathbf{L}$ satisfying it can be defined as follows.

First, the algorithm guesses the set $Inf(\mathbf{L}) = \{\mathrm{REQ}_1, \ldots, \mathrm{REQ}_p\}$, with $Inf(\mathbf{L}) \subseteq \mathrm{REQ}_\varphi$, of the sets of requests that occur infinitely often in $\mathbf{L}$. If $p = 0$, then it guesses the length $l \leq |\,\mathrm{REQ}_\varphi\,| \cdot \frac{|\,\mathrm{TF}(\varphi)|}{2} + 1$ of a finite LIS. Otherwise, it takes p as the period of an ultimately periodic LIS and it guesses the length $l \leq (|\,\mathrm{REQ}_\varphi\,| - p) \cdot \frac{|\,\mathrm{TF}(\varphi)|}{2} + 1$ of its prefix. Next, the algorithm guesses the labelling of the initial interval $[d_0, d_1]$, taking an atom $A_{[d_0,d_1]}$ that includes φ, and it initializes a counter c to 1. If $c < l$, then it guesses the labelling of the intervals ending in d_2, that is, it associates an atom $A_{[d_0,d_2]}$ with $[d_0, d_2]$ and an atom $A_{[d_1,d_2]}$ with $[d_1, d_2]$ such that $A_{[d_0,d_1]}\ R_\varphi\ A_{[d_1,d_2]}$, with $\mathrm{REQ}(d_2) \in \mathrm{REQ}_\varphi \setminus Inf(\mathbf{L})$, and it increments c by one. The algorithm proceeds in this way, incrementing c by one for every point d_j it considers and checking that, for every pair of atoms $A_{[d_k,d_i]}$ and $A_{[d_i,d_j]}$, $A_{[d_k,d_i]}\ R_\varphi\ A_{[d_i,d_j]}$. For each point d_j, it must guarantee that $\mathrm{REQ}(d_j) \in \mathrm{REQ}_\varphi \setminus Inf(\mathbf{L})$ and that $\mathrm{REQ}(d_j)$ occurs at most $\frac{|\,\mathrm{TF}(\varphi)|}{2}$ times in $\{d_2, \ldots, d_j\}$.

When c reaches the value l, two cases are possible. If $p = 0$, then d_l is the last point of the finite LIS $\mathbf{L}$, and the algorithm checks whether it

is fulfilling. If $p > 0$, it checks if the guessed prefix and period represent a fulfilling LIS by proceeding as follows:

- for every atom $A_{[d_i,d_j]}$ in the prefix and for every formula $\langle A\rangle\psi \in A_{[d_i,d_j]}$, it checks if either there exists an atom $A_{[d_j,d_k]}$ in the prefix that contains ψ or there exists an atom A' containing ψ and a set $\mathrm{REQ}_h \in Inf(\mathbf{L})$ such that $\mathrm{REQ}_h = A' \cap \mathrm{TF}(\varphi)$ and $A_{[d_i,d_j]}\ R_\varphi\ A'$;
- for every atom $A_{[d_i,d_j]}$ in the prefix and for every set $\mathrm{REQ}_h \in Inf(\mathbf{L})$, it checks if there exists an atom A' such that $\mathrm{REQ}_h = A' \cap \mathrm{TF}(\varphi)$ and $A_{[d_i,d_j]}\ R_\varphi\ A'$;
- for every set $\mathrm{REQ}_h \in Inf(\mathbf{L})$ and for every formula $\langle A\rangle\psi \in \mathrm{REQ}_h$, it checks if there exists an atom A' containing ψ and a set $\mathrm{REQ}_k \in Inf(\mathbf{L})$ such that $\mathrm{REQ}_k = A' \cap \mathrm{TF}(\varphi)$ and $\mathrm{REQ}_h\ R_\varphi\ A'$;
- for every pair of sets $\mathrm{REQ}_h, \mathrm{REQ}_k \in Inf(\mathbf{L})$, it checks if there exists an atom A' such that $\mathrm{REQ}_k = A' \cap \mathrm{TF}(\varphi)$ and $\mathrm{REQ}_h\ R_\varphi\ A'$.

By Theorems 4.11 and 4.13, it follows that the algorithm returns *true* if and only if φ is satisfiable. As for the computational complexity of the algorithm, we observe that:

1. l is less than or equal to $|\,\mathrm{REQ}_\varphi\,| \cdot \frac{|\,\mathrm{TF}(\varphi)|}{2} + 1$, while p is less than or equal to $|\,\mathrm{REQ}_\varphi\,|$;
2. for every point $d_1 \leq d_j \leq d_l$, the algorithm guesses exactly j atoms $A_{[d_i,d_j]}$;
3. checking for the fulfillness of the guessed LIS takes time polynomial in p and in the number of guessed atoms;
4. $|\,\mathrm{TF}(\varphi)|$ is linear in the length of φ, while $|\,\mathrm{REQ}_\varphi\,|$ is exponential in it.

Hence, if $|\varphi| = n$, the number of guessed sets in $Inf(\mathbf{L})$ plus number of guessed atoms in the prefix is bounded by

$$
\begin{aligned}
|\,\mathrm{REQ}_\varphi\,| + \sum_{i=1}^{|\,\mathrm{REQ}_\varphi\,| \cdot \frac{|\,\mathrm{TF}(\varphi)|}{2}+1} i \;&=\; O\left(|\,\mathrm{REQ}_\varphi\,| \cdot \frac{|\,\mathrm{TF}(\varphi)|}{2}\right)^2 \\
&=\; \left(2^{O(n)} \cdot O(n)\right)^2 \\
&=\; 2^{2\cdot O(n)} \cdot O(n^2) \\
&=\; 2^{O(n)} \cdot O(n^2) \\
&=\; 2^{O(n)},
\end{aligned}
$$

that is, it is exponential in the length of φ. This implies that the satisfiability problem for RPNL$^-$ can be solved by the above nondeterministic algorithm in nondeterministic exponential time.

Theorem 5.1. *The satisfiability problem for* RPNL$^-$*, over natural numbers, is in NEXPTIME.*

5.2. A lower bound to the computational complexity

We now provide a NEXPTIME lower bound for the complexity of the satisfiability problem for RPNL$^-$ by reducing to it the exponential tiling problem, which is known to be NEXPTIME-complete [1].

Let us denote by $\mathbb{N}_m$ the set of natural numbers less than m and by $N(m)$ the grid $\mathbb{N}_m \times \mathbb{N}_m$. A *domino system* is a triple $\mathcal{D} = \langle C, H, V \rangle$, where C is a finite set of *colors* and $H, V \subseteq C \times C$ are the horizontal and vertical *adjacency relations*. We say that $\mathcal{D}$ *tiles* $N(m)$ if there exists a mapping $\tau : N(m) \to C$ such that, for all $(x, y) \in N(m)$:

1. if $\tau(x, y) = c$ and $\tau(x + 1, y) = c'$, then $(c, c') \in H$;
2. if $\tau(x, y) = c$ and $\tau(x, y + 1) = c'$, then $(c, c') \in V$.

The exponential tiling problem consists in determining, given a natural number n and a domino system $\mathcal{D}$, whether $\mathcal{D}$ tiles $N(2^n)$ or not. Proving that the satisfiability problem for RPNL$^-$ is NEXPTIME-hard can be done by encoding the exponential tiling problem with a formula $\varphi(\mathcal{D})$, of length polynomial in n, which uses propositional letters to represent positions in the grid and colors, and by showing that $\varphi(\mathcal{D})$ is satisfiable if and only if $\mathcal{D}$ tiles $N(2^n)$. Such a formula consists of three main parts. The first part imposes a sort of locality principle; the second part encodes the $N(2^n)$ grid; the third part imposes that every point of the grid is tiled by exactly one color and that the colors respect the adjacency conditions. Intervals are exploited to express relations between pairs of points.

Theorem 5.2. *The satisfiability problem for* RPNL$^-$*, over natural numbers, is NEXPTIME-hard.*

Proof. Given a domino system $\mathcal{D} = \langle C, H, V \rangle$, we build an RPNL$^-$ formula φ, of length polynomial in n, that is satisfiable if and only if $\mathcal{D}$ tiles $N(2^n)$.

The models for φ encode a tiling $\tau : N(2^n) \to C$ in the following way. First, we associate with every point $z = (x, y) \in N(2^n)$ a $2n$-bit word $(z_{2n-1} z_{2n-2} \dots z_1 z_0) \in \{0, 1\}^{2n}$ such that $x = \sum_{i=0}^{n-1} z_i 2^i$ and $y = \sum_{i=n}^{2n-1} y_i 2^{i-n}$. Pairs of points $[z, t]$ of $N(2^n)$ are represented as intervals by means of the propositional letters Z_i, T_i, with $0 \leq i \leq 2n - 1$, as follows:

$$Z_i : z_i = 1; \; T_i : t_i = 1.$$

Moreover, the colors of $z = (x, y)$ and $t = (x', y')$ are expressed by means of the propositional letters Z_c, T_c, with $c \in C$, as follows:

$$Z_c : \tau(x, y) = c;\ T_c : \tau(x', y') = c.$$

To ease the writing of the formula φ encoding the tiling problem, we use the auxiliary propositional letters Z_i^* (for $0 \leq i \leq 2n-1$) and ZH_i^* (for $n \leq i \leq 2n-1$), with the following intended meaning:

$$Z_i^* :\ \text{for all } 0 \leq j < i, z_j = 1;\ ZH_i^* :\ \text{for all } n \leq j < i, z_j = 1.$$

To properly encode the tiling problem, we must constrain the relationships among these propositional letters.

Definition of auxiliary propositional letters. As a preliminary step, we define the auxiliary propositional letters Z_i^*, with $0 \leq i \leq 2n-1$, and ZH_i^*, with $n \leq i \leq 2n-1$, as follows:

$$[A][A]\left(Z_0^* \wedge \bigwedge_{i=1}^{2n-1} (Z_i^* \ \leftrightarrow\ (Z_{i-1}^* \wedge Z_{i-1}))\right)$$

$$[A][A]\left(ZH_n^* \wedge \bigwedge_{i=n+1}^{2n-1} (ZH_i^* \ \leftrightarrow\ (ZH_{i-1}^* \wedge Z_{i-1}))\right).$$

Let us call α the conjunction of the above two formulae.

Locality conditions. Then, we impose a sort of "locality principle" on the interpretation of the propositional letters. Given an interval $[z, t]$, we encode the position $z = (x, y)$ (resp., $t = (x', y')$) and its color $\tau(x, y)$ (resp., $\tau(x', y')$) by means of the propositional letters Z_i, Z_i^*, ZH_i^*, and Z_c (resp., T_i and T_c) by imposing the following constraints:

- all intervals $[z, w]$ starting in z must agree on the truth value of Z_i, Z_i^*, ZH_i^*, and Z_c;
- for every pair of neighboring intervals $[z, t], [t, w]$, the truth value of T_i and T_c over $[z, t]$ must agree with the truth value of Z_i and Z_c over $[t, w]$.

From the above constraints, it easily follows that all intervals $[w, t]$ ending in t must agree on the truth value of T_i and T_c.

Such constraints are encoded by the conjunction of the following formulae (let us call it β):

$$\bigwedge_{i=0}^{2n-1}(\langle A\rangle Z_i \rightarrow [A]Z_i) \wedge \bigwedge_{i=0}^{2n-1}[A](\langle A\rangle Z_i \rightarrow [A]Z_i)$$

$$\bigwedge_{i=0}^{2n-1}(\langle A\rangle Z_i^* \rightarrow [A]Z_i^*) \wedge \bigwedge_{i=0}^{2n-1}[A](\langle A\rangle Z_i^* \rightarrow [A]Z_i^*)$$

$$\bigwedge_{i=n}^{2n-1}(\langle A\rangle ZH_i^* \rightarrow [A]ZH_i^*) \wedge \bigwedge_{i=n}^{2n-1}[A](\langle A\rangle ZH_i^* \rightarrow [A]ZH_i^*)$$

$$\bigwedge_{c\in C}(\langle A\rangle Z_c \rightarrow [A]Z_c) \wedge \bigwedge_{c\in C}[A](\langle A\rangle Z_c \rightarrow [A]Z_c)$$

$$\bigwedge_{i=0}^{2n-1}[A](T_i \leftrightarrow [A]Z_i) \wedge \bigwedge_{i=0}^{2n-1}[A][A](T_i \leftrightarrow [A]Z_i)$$

$$\bigwedge_{c\in C}[A](T_c \leftrightarrow [A]Z_c) \wedge \bigwedge_{c\in C}[A][A](T_c \leftrightarrow [A]Z_c)$$

Encoding of the grid. Next, we must guarantee that every point $z = (x, y) \in N(2^n)$, with the exception of the upper-right corner $(2^n - 1, 2^n - 1)$, has a "successor" $t = (x', y')$, that is, if $x \neq 2^n - 1$, then $(x', y') = (x + 1, y)$; otherwise $(x = 2^n - 1)$, $(x', y') = (0, y + 1)$. Note that, thanks to our encoding of z and t, the binary encoding of the successor of z is equal to the binary encoding of z incremented by 1. Such a successor relation can be encoded as follows. Given two $2n$-bit words $z = \sum_{i=0}^{2n-1} z_i 2^i$ and $t = \sum_{i=0}^{2n-1} t_i 2^i$, we have that $t = z + 1$ if and only if there exists some $0 \leq j \leq 2n - 1$ such that:

1. $z_j = 0$ and, for all $i < j$, $z_i = 1$;
2. $t_j = 1$ and, for all $i < j$, $t_i = 0$;
3. for all $j < k \leq 2n - 1$, $z_k = t_k$.

It is easy to show that, for every i, with $0 \leq i \leq 2n - 1$, we can write t_i as $z_i \oplus \bigwedge_{k<i} z_k$, where $\oplus$ denotes the exclusive or. Taking advantage of this fact, the successor relation can be expressed by the following formula (let us call it γ):

$$[A]\left(\langle A\rangle\neg(Z_{2n-1}^* \wedge Z_{2n-1}) \rightarrow \langle A\rangle \bigwedge_{i=0}^{2n-1}(T_i \leftrightarrow (Z_i \oplus Z_i^*))\right).$$

Furthermore, the left conjunct of the following formula (let us call it δ) encodes the initial point $(0, 0)$ of the grid, while the right one encodes

the final point $(2^n - 1, 2^n - 1)$:

$$\langle A\rangle\langle A\rangle \bigwedge_{i=0}^{2n-1} \neg Z_i \wedge \langle A\rangle\langle A\rangle \bigwedge_{i=0}^{2n-1} Z_i.$$

Grid coloring. To complete the reduction, we must properly define the tiling of the grid. To this end, we preliminary need to express the relations of right (horizontal) neighborhood and upper (vertical) neighborhood over the grid. We have that the following formula ψ_H (resp., ψ_V) holds over any interval $[z, t]$ such that t is the right (resp. upper) neighbor of z in $N(2^n)$:

$$\psi_H := \bigwedge_{i=n}^{2n-1} (Z_i \leftrightarrow T_i) \wedge \bigwedge_{i=0}^{n-1} (T_i \leftrightarrow (Z_i \oplus Z_i^*))$$

$$\psi_V := \bigwedge_{i=0}^{n-1} (Z_i \leftrightarrow T_i) \wedge \bigwedge_{i=n}^{2n-1} (T_i \leftrightarrow (Z_i \oplus ZH_i^*))$$

By using ψ_H and ψ_V, we can impose the *adjacency conditions* by means of the following formula (let us call ϵ):

$$[A][A]\left((\psi_H \rightarrow \bigvee_{(c,c')\in H} Z_c \wedge T_{c'}) \wedge (\psi_V \rightarrow \bigvee_{(c,c')\in V} Z_c \wedge T_{c'})\right).$$

The fact that every point is tiled by exactly one color can be forced by the following formula (let us call it ζ):

$$[A][A]\left(\dot{\bigvee_{c\in C}} Z_c \wedge \dot{\bigvee_{c\in C}} T_c\right),$$

where $\dot{\bigvee}$ is a *generalized exclusive or* which is true if and only if exactly one of its arguments is true.

Let us define φ as the conjunction $\alpha\wedge\beta\wedge\gamma\wedge\delta\wedge\epsilon\wedge\zeta$. The length of φ is polynomial in n as requested. It remains to show that φ is satisfiable if and only if $\mathcal{D}$ tiles $N(2^n)$. As for the implication from left to right, if a correct tiling exists, then let $\mathbb{D} = \langle D, <\rangle$ be a linear ordering such that:

- $D = \{d_0, d_1\} \cup N(2^n) \cup \{d_\top\}$;
- $d_0 < d_1 < (x, y) < d_\top$, for every $(x, y) \in N(2^n)$;
- given two points (x, y) and (x', y') of $N(2^n)$, $(x, y) < (x, y')$ iff $y < y' \vee (y = y' \wedge x < x')$.

Notice that we take as the domain of the interval structure the set of elements of the grid extended with the elements d_0, d_1, and $d_\top$. The elements d_0, d_1 define the initial interval $[d_0, d_1]$ over which our formula will be interpreted. The element $d_\top$ is the right endpoint of the only interval having the last point of the grid as its left endpoint, namely, $[(2^n - 1, 2^n - 1), d_\top]$.
As for the valuation $\mathcal{V}$, for any interval $[z, t]$, with $z = (x, y), t = (x', y')$, and $z, t \in N(2^n)$, $Z_i \in \mathcal{V}([z, t])$ if and only $z_i = 1$ and $T_i \in \mathcal{V}([z, t])$ if and only $t_i = 1$. Moreover, $Z_c \in \mathcal{V}([z, t])$ (resp., $T_c \in \mathcal{V}([z, t])$) if and only if $\tau(x, y) = c$ (resp., $\tau(x', y') = c$). Whenever, the left (resp. right) endpoint of an interval does not belong to $N(2^n)$, the valuation of the propositional letters Z_i and Z_c (resp. T_i and T_c) over the interval is arbitrary. It is not difficult to prove that $\mathbf{M} = \langle\langle \mathbb{D}, \mathbb{I}(\mathbb{D})^- \rangle, \mathcal{V}\rangle$ is a model of φ, that is, $\mathbf{M}, [d_0, d_1] \Vdash \varphi$.

Conversely, let $\mathbf{M} = \langle\langle \mathbb{D}, \mathbb{I}(\mathbb{D})^- \rangle, \mathcal{V}\rangle$ be a model for φ, that is, $\mathbf{M}, [d_0, d_1] \Vdash \varphi$. To provide a tiling of $N(2^n)$, we first define a function $f : N(2^n) \to D$ that associates a point $d \in D$ with every point $(x, y) \in N(2^n)$ in such a way that:

1. the binary representation of (x, y) coincides with the sequence of truth values of the propositional letters $Z_{2n-1}, Z_{2n-2}, \ldots, Z_0$ over the intervals $[d, d'] \in \mathbb{I}(\mathbb{D})^-$;
2. for every $(x, y), (x', y') \in N(2^n)$, $(x, y) < (x', y')$ iff $f(x, y) < f(x', y')$.

The formula φ ensures that such a function exists. Note that the definition of f guarantees commutativity: by moving first one step right and then one step up on the grid one reaches the same point that can be reached by moving first one step up and then one step right. On the basis of such a function, we define the tiling $\tau(x, y) = c$, where c is the unique element of C such that $\mathbf{M}, [f(x, y), d'] \Vdash Z_c$, for every $d' > f(x, y)$. It is not difficult to prove that τ defines a tiling of $N(2^n)$. □

From Theorems 5.1 and 5.2, we have the following corollary.

Corollary 5.3. *The satisfiability problem for* RPNL$^-$*, over natural numbers, is NEXPTIME-complete.*

6. A tableau-based decision procedure for RPNL$^-$

In this section, we define a tableau-based decision procedure for RPNL$^-$, whose behavior is illustrated by means of a simple example, and we

analyze its computational complexity. Then, we prove its soundness and completeness. The procedure is based on two expansion rules, respectively called step rule and fill-in rule, and a blocking condition, that guarantees the termination of the method. Unlike the naïve procedure described in the previous section, it does not need to differentiate the search for a finite model from that for an infinite one.

6.1. The tableau method

We first define the structure of a tableau for an RPNL$^-$ formula and then we show how to construct it. A tableau for RPNL$^-$ is a suitable *decorated tree* $\mathcal{T}$. Each branch B of a tableau is associated with a finite prefix of the natural numbers $\mathbb{D}_B = \langle D_B, < \rangle$. The *decoration* of each node n in $\mathcal{T}$, denoted by $\nu(n)$, is a pair $\langle [d_i, d_j], A \rangle$, where d_i, d_j, with $d_i < d_j$, belong to D_B (for all branches B containing n) and A is an atom. The root r of $\mathcal{T}$ is labelled by the *empty decoration* $\langle \emptyset, \emptyset \rangle$. Given a node n, we denote by $A(n)$ the atom component of $\nu(n)$.

Given a branch B, we define a function REQ : $D_B \rightarrow 2^{\mathrm{TF}(\varphi)}$ as follows. For every $d_i \in D_B$, $\mathrm{REQ}(d_i) = (\bigcap_j A_j) \cap \mathrm{TF}(\varphi)$, where n_j is a node such that $\nu(n_j) = \langle [d_j, d_i], A_j \rangle$ and $d_0 \leq d_j < d_i$. Moreover, given a node $n \in B$, with decoration $\langle [d_i, d_j], A \rangle$, and an existential formula $\langle A \rangle \psi \in A$, we say that $\langle A \rangle \psi$ is *fulfilled on* B if and only if there exists a node $n' \in B$ such that $\nu(n') = \langle [d_j, d_k], A' \rangle$ and $\psi \in A'$. A node n is said to be *active on* B if and only if $A(n)$ contains at least one existential formula that is not fulfilled on B.

Expansion rules. The construction of a tableau is based on the following expansion rules. Let B be a branch of a decorated tree $\mathcal{T}$ and let d_k be the greatest point in D_B. The following *expansion rules* can be possibly applied to extend B:

1. *Step rule*: if there exists at least one active node $n \in B$, with $\nu(n) = \langle [d_i, d_j], A \rangle$, then take an atom A' such that $A \; R_\varphi \; A'$ and expand B to $B \cdot n'$, with $\nu(n') = \langle [d_j, d_{k+1}], A' \rangle$.
2. *Fill-in rule*: if there exists a node $n \in B$, with decoration $\langle [d_i, d_j], A \rangle$ and $d_j < d_k$, such that there are no nodes n' in B with decoration $\langle [d_j, d_k], A' \rangle$, for some $A' \in A_\varphi$, then take any atom $A'' \in A_\varphi$ such that $A \; R_\varphi \; A''$ and $\mathrm{REQ}(d_k) = A'' \cap \mathrm{TF}(\varphi)$, and expand B to $B \cdot n''$, with $\nu(n'') = \langle [d_j, d_k], A'' \rangle$.

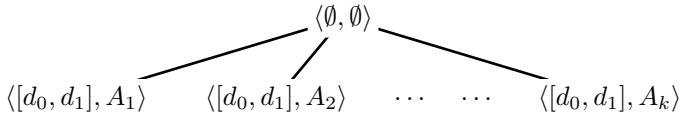
Both rules add a new node to the branch B. However, while the step rule decorates such a node with a new interval ending at a new point d_{k+1}, the fill-in rule decorates it with a new interval whose endpoints were already in D_B.

Blocking condition. To guarantee the termination of the method, we need a suitable *blocking condition* to avoid the infinite application of the expansion rules in case of infinite models. Given a branch B, with $D_B = \{d_0, d_1, \ldots, d_k\}$, we say that B is *blocked* if $\mathrm{REQ}(d_k)$ occurs $\frac{|\mathrm{TF}(\varphi)|}{2} + 1$ times in D_B.

Expansion strategy. Given a decorated tree $\mathcal{T}$ and a branch B, we say that an expansion rule is *applicable on* B if B is non-blocked and the application of the rule generates a new node. The *branch expansion strategy* for a branch B is the following one:

1. if the fill-in rule is applicable, apply the fill-in rule to B and, for every possible choice for the atom A'', add an immediate successor to the last node in B;
2. if the fill-in rule is not applicable and there exists a node $n \in B$, with decoration $\langle [d_i, d_j], A \rangle$ and $d_j < d_k$, such that there are no nodes in B with decoration $\langle [d_j, d_k], A' \rangle$, for some $A' \in A_\varphi$, *close* the branch;
3. if the fill-in rule is not applicable, B is not closed, and there exists at least one active node in B, then apply the step rule to B and, for every possible choice of the atom A', add an immediate successor to the last node in B.

Tableau. Let φ be the formula to be checked for satisfiability and let $A_1, \ldots, A_k$ be all and only the atoms containing φ. The *initial tableau* for φ is the following:



A *tableau* for φ is any decorated tree $\mathcal{T}$ obtained by expanding the initial tableau for φ through successive applications of the branch-expansion strategy to currently existing branches, until the branch-expansion strategy cannot be applied anymore.

Fulfilling branches. Given a branch B of a tableau $\mathcal{T}$ for φ, we say that B is a *fulfilling branch* if and only if B is not closed and one of the following conditions holds:

1. B is non-blocked and for every node $n \in B$ and existential formula $\langle A \rangle \psi \in A(n)$, there exists a node $n' \in B$ fulfilling $\langle A \rangle \psi$ (finite model case);

2. B is blocked, d_k is the greatest point of D_B, d_i ($\neq d_k$) is the smallest point in D_B such that $\mathrm{REQ}(d_i) = \mathrm{REQ}(d_k)$, and the following conditions hold:
   (a) for every node $n \in B$ and every formula $\langle A\rangle\psi \in A(n)$ not fulfilled on B, there exist a point $d_i \leq d_l \leq d_k$ and an atom A' such that $\psi \in A'$, $A(n)\ R_\varphi\ A'$, and $\mathrm{REQ}(d_l) = A' \cap \mathrm{TF}(\varphi)$;
   (b) for every node $n \in B$ and every point $d_i \leq d_m \leq d_k$, there exists an atom A' such that $A(n)\ R_\varphi\ A'$ and $\mathrm{REQ}(d_m) = A' \cap \mathrm{TF}(\varphi)$.

The decision procedure works as follows: given a formula φ, it constructs a tableau $\mathcal{T}$ for φ and it returns "satisfiable" if and only if there exists at least one fulfilling branch in $\mathcal{T}$.

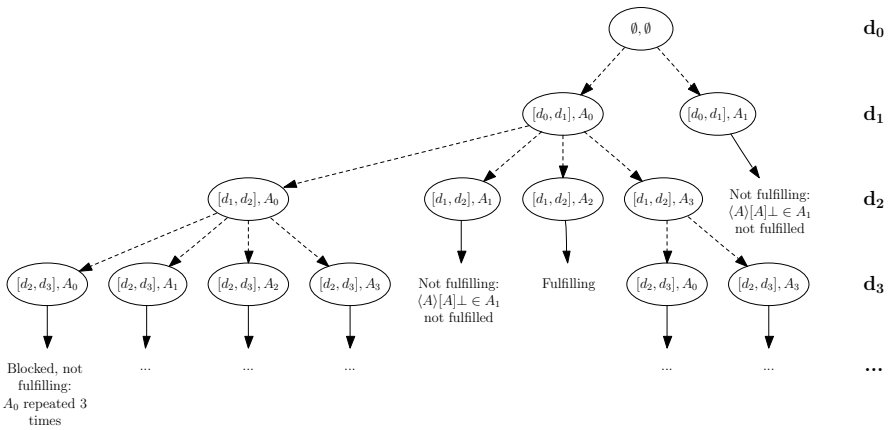


Figure 2. A tableau for $\langle A\rangle[A]\bot$.

We conclude the section by showing how the proposed method works on the simple case of the formula $\varphi = \langle A\rangle[A]\bot$. For sake of simplicity, we treat the logical constants $\top$ and $\bot$ as propositional letters, with the further constraint that, for any atom A, $\bot \notin A$. Hence, the set of φ-atoms is the following one:

$$\begin{aligned} A_0 &= \{\langle A\rangle[A]\bot, \langle A\rangle\top, \top\} \\ A_1 &= \{\langle A\rangle[A]\bot, [A]\bot, \top\} \\ A_2 &= \{[A]\langle A\rangle\top, [A]\bot, \top\} \\ A_3 &= \{[A]\langle A\rangle\top, \langle A\rangle\top, \top\} \end{aligned}$$

As for the relation R_φ between atoms, we have that

$$R_\varphi = \{(A_0, A_0), (A_0, A_1), (A_0, A_2), (A_0, A_3), (A_3, A_0), (A_3, A_3)\}.$$

Note that A_1 and A_2 have no R_φ-successors (since they both contains $[A]\bot$). Figure 2, where dashed arrows represent applications of the

step rule, depicts a portion of a tableau for φ which is sufficiently large to include a fulfilling branch, and thus to prove that φ is satisfiable. Indeed, it is easy to see that, over natural numbers, φ is satisfiable and it admits only *finite models*.

Notice that there are no nodes in the tableau which are labelled by intervals beginning at d_0, except for the nodes associated with the initial interval $[d_0, d_1]$. Since RPNL is not strong enough to force any condition on intervals beginning at d_0 and different from $[d_0, d_1]$, such intervals can be ignored without affecting the soundness and completeness of the method.

6.2. Computational complexity

As a preliminary step, we show that the proposed tableau method terminates; then we analyze its computational complexity.

In order to prove termination of the tableau method, we give a bound on the length of any branch B of any tableau for φ:

1. by the blocking condition, after at most $|\,\mathrm{REQ}_\varphi\,| \cdot \frac{|\,\mathrm{TF}(\varphi)|}{2}$ applications of the step rule, the expansion strategy cannot be applied anymore to a branch;
2. given a branch B, between two successive applications of the step rule, the fill-in rule can be applied at most k times, where k is the current number of elements in D_B (k is exactly the number of applications of the step rule up to that point);
3. $|\,\mathrm{TF}(\varphi)|$ is linear in the length of φ, while $|\,\mathrm{REQ}_\varphi\,|$ is exponential in it.

Hence, if $|\varphi| = n$, the length of any branch B of a tableau $\mathcal{T}$ for φ is bounded by

$$
\begin{aligned}
|\,\mathrm{REQ}_\varphi\,| \cdot \frac{|\,\mathrm{TF}(\varphi)|}{2} + \sum_{i=1}^{|\,\mathrm{REQ}_\varphi\,| \cdot \frac{|\,\mathrm{TF}(\varphi)|}{2}} i \;&\leq\; \left(|\,\mathrm{REQ}_\varphi\,| \cdot \frac{|\,\mathrm{TF}(\varphi)|}{2}\right)^2 + 1 \\
&=\; \left(2^{O(n)} \cdot O(n)\right)^2 \\
&=\; 2^{2 \cdot O(n)} \cdot O(n^2) \\
&=\; 2^{O(n)} \cdot O(n^2) \\
&=\; 2^{O(n)},
\end{aligned}
$$

that is, the length of a branch is (at most) exponential in $|\varphi|$.

Theorem 6.1 (Termination). *The tableau method for* RPNL$^-$ *terminates.*

Proof. Given a formula φ, let $\mathcal{T}$ be a tableau for φ. Since, by construction, every node of $\mathcal{T}$ has a finite outgoing degree and every branch of it is of finite length, by König's Lemma, $\mathcal{T}$ is finite. □

The computational complexity of the tableau-based decision procedure depends on the strategy used to search for a fulfilling branch in the tableau. The strategy that first builds the entire tableau and then looks for a fulfilling branch requires an amount of time and space that can be doubly exponential in the length of φ. However, by exploiting nondeterminism, the existence of a fulfilling branch can be determined without visiting the entire tableau, by exploiting the following alternative strategy. First, select one of the nodes decorated with $\langle [d_0, d_1], A\rangle$ of the initial tableau and expand it as follows. Instead of generating all successors nodes, nondeterministically select one of them and expand it. Iterate such a revised expansion strategy until it cannot be applied anymore. Finally, return "satisfiable" if and only if the guessed branch is a fulfilling one.

Such a procedure has a nondeterministic time complexity which is polynomial in the length of the branch, and thus exponential in the size of φ. Giving the NEXPTIME-completeness of the satisfiability problem for RPNL$^-$, this allows us to conclude that the proposed tableau-based decision procedure is optimal.

6.3. Soundness and completeness

The soundness and completeness of the proposed method can be proved as follows. Soundness is proved by showing how it is possible to construct a fulfilling LIS satisfying φ from a fulfilling branch B in a tableau $\mathcal{T}$ for φ (by Theorem 4.8, it follows that φ has a model). The proof must encompass both the case of blocked branches and that of non-blocked ones. Proving completeness consists in showing, by induction on the height of $\mathcal{T}$, that for any satisfiable formula φ, there exists a fulfilling branch B in any tableau $\mathcal{T}$ for φ. To this end, we take a model for φ and the corresponding fulfilling LIS $\mathbf{L}$, and we prove the existence of a fulfilling branch in $\mathcal{T}$ by exploiting Theorems 4.11 and 4.13.

Theorem 6.2 (Soundness). *Given a formula φ and a tableau $\mathcal{T}$ for φ, if there exists a fulfilling branch in $\mathcal{T}$, then φ is satisfiable.*

Proof. Let $\mathcal{T}$ be a tableau for φ and B a fulfilling branch in $\mathcal{T}$. We show that, starting from B, we can build up a fulfilling LIS $\mathbf{L}$ satisfying φ. By the definition of fulfilling branch, two cases may arise.

B is non-blocked (*finite model case*). We define $\mathbf{L} = \langle\langle \mathbb{D}, \mathbb{I}(\mathbb{D})^-\rangle, \mathcal{L}\rangle$ as follows:

- $\mathbb{D} = \mathbb{D}_B$;
- $\mathcal{L}([d_0, d_1]) = A$, where $A = A(n_1)$ and n_1 is the unique node of B such that $\nu(n_1) = \langle [d_0, d_1], A \rangle$;
- for every $[d_i, d_j] \in \mathbb{I}(\mathbb{D}_B)^-$, with $d_i > d_0$, we put $\mathcal{L}([d_i, d_j]) = A$, where $A = A(n)$ and n is the node in B such that $\nu(n) = \langle [d_i, d_j], A \rangle$. Since B is not closed, such a node n exists; its uniqueness follows from tableau rules;
- finally, for every interval $[d_0, d_j] \in \mathbb{I}(\mathbb{D}_B)^-$, with $d_j > d_1$, there are no nodes in B labelled with $\langle [d_0, d_j], A \rangle$. We complete the definition of $\mathcal{L}$ by putting, for every $d_j > d_1$, $\mathcal{L}([d_0, d_j]) = A(n)$, where n is an arbitrary node in B such that $\nu(n) = \langle [d_i, d_j], A \rangle$, for some $d_i < d_j$.

Clearly, $\mathbf{L}$ is a LIS. Since B is fulfilling, for every node $n \in B$ and every existential formula $\langle A \rangle \psi \in A(n)$, there exists a node n' fulfilling $\langle A \rangle \psi$. Hence, by the above construction, $\mathbf{L}$ is fulfilling.

B is blocked (*infinite model case*). Let d_k be the last point of D_B and $d_i \neq d_k$ be the smallest point in D_B such that $\mathrm{REQ}(d_i) = \mathrm{REQ}(d_k)$. We build an ultimately periodic LIS $\mathbf{L} = \langle \langle \mathbb{D}', \mathbb{I}(\mathbb{D}')^- \rangle, \mathcal{L} \rangle$ with prefix $l = i - 1$ and period $p = k - i$, as follows:

1. let $D' = \{d'_0 (= d_0), d'_1 (= d_1), \ldots, d'_k (= d_k), d'_{k+1}, \ldots\}$ be any set isomorphic to $\mathbb{N}$;
2. for every $d'_1 \leq d'_j \leq d'_k$, put $\mathrm{REQ}(d'_j) = \mathrm{REQ}(d_j)$;
3. for every $d'_j > d'_k$, put $\mathrm{REQ}(d'_j) = \mathrm{REQ}(d_{i+(j-l) \bmod p})$;
4. put $\mathcal{L}([d'_0, d'_1]) = A(n_1)$, where n_1 is the unique node in B such that $\nu(n_1) = \langle [d_0, d_1], A \rangle$;
5. for every pair of points d'_j, d'_m such that $d_0 < d'_j < d'_m < d'_k$, take the node n in B such that $\nu(n) = \langle [d_j, d_m], A \rangle$ and put $\mathcal{L}([d'_j, d'_m]) = A$;
6. for every point $d'_j \in D'$ and every $\langle A \rangle \psi \in \mathrm{REQ}(d'_j)$ which has not been fulfilled yet, proceed as follows. Let n be a node in B decorated with $\langle [d, d'], A \rangle$ such that $\mathrm{REQ}(d') = \mathrm{REQ}(d'_j)$. Since B is fulfilling, by condition (a) for fulfilling branches, there exist a point $d_i \leq d_m \leq d_k$ and an atom A' such that $\psi \in A'$, $A \; R_\varphi \; A'$ and $\mathrm{REQ}(d_m) = A' \cap \mathrm{TF}(\varphi)$. By the definition of $\mathbf{L}$, we have that there exist infinitely many points $d'_n \geq d'_k$ in D' such that $\mathrm{REQ}(d'_n) = \mathrm{REQ}(d_m)$. We can take one of such points d'_n such that $\mathcal{L}([d'_j, d'_n])$ has not been defined yet and put $\mathcal{L}([d'_j, d'_n]) = A'$;

7. once we have fulfilled all $\langle A\rangle$-formulae in $\mathrm{REQ}(d')$, for all $d' \in \mathbb{D}'$, we arbitrarily define the labelling of the remaining intervals $[d', d'']$. Since B is fulfilling, we can always define $\mathcal{L}([d', d''])$ by exploiting condition (b) for fulfilling branches;

8. as in the finite model case, there are no nodes in B labelled with $\langle [d_0, d_j], A\rangle$, for all $d_j > d_1$. For every $d'_j > d'_1$, take an arbitrary node n labelled with $\langle [d_i, d_l], A'\rangle$ such that $\mathrm{REQ}(d_l) = \mathrm{REQ}(d'_j)$ and put $\mathcal{L}([d'_0, d'_j]) = A'$.

Since $\varphi \in \mathcal{L}([d'_0, d'_1])$, $\mathbf{L}$ is a fulfilling LIS satisfying φ. □

Theorem 6.3 (Completeness). *Given a satisfiable formula φ, there exists a fulfilling branch in every tableau $\mathcal{T}$ for φ.*

Proof. Let φ be a satisfiable formula and let $\mathbf{L} = \langle\langle \mathbb{D}, \mathbb{I}(\mathbb{D})^-\rangle, \mathcal{L}\rangle$ be a fulfilling LIS satisfying φ, whose existence is guaranteed by Theorem 4.8. Without loss of generality, we may assume that $\mathbf{L}$ respects the constraints of Theorem 4.11, if it is finite, and of Theorem 4.13, if it is infinite. We prove there exists a fulfilling branch B in $\mathcal{T}$ which corresponds to $\mathbf{L}$. To this end, we prove the following property: *there exists a non-closed branch B such that, for every node $n \in B$, if n is decorated with $\langle [d_j, d_k], A\rangle$, then $A = \mathcal{L}([d_j, d_k])$*. The proof is by induction on the height $h(\mathcal{T})$ of $\mathcal{T}$.

If $h(\mathcal{T}) = 1$, then $\mathcal{T}$ is the initial tableau for φ and, by construction, it contains a branch

$$B_0 = \langle \emptyset, \emptyset\rangle \cdot \langle [d_0, d_1], A\rangle,$$

with $A = \mathcal{L}([d_0, d_1])$.

Let $h(\mathcal{T}) = i + 1$. By inductive hypothesis, there exists a branch B_i of length i that satisfies the property. Let $D_{B_i} = \{d_0, d_1, \ldots, d_k\}$. We distinguish two cases, depending on the expansion rule that has been applied to B_i in the construction of $\mathcal{T}$.

– **The step rule has been applied.**
Let n be the active node, decorated with $\langle [d_j, d_l], A\rangle$, which the step rule has been applied to. By inductive hypothesis, $A = \mathcal{L}([d_j, d_l])$. Since $\mathbf{L}$ is a LIS, $\mathcal{L}([d_j, d_l])\ R_\varphi\ \mathcal{L}([d_l, d_{k+1}])$. Hence, there must exists in $\mathcal{T}$ a successor n' of the last node of B_i decorated with $\langle [d_l, d_{k+1}], \mathcal{L}([d_l, d_{k+1}])\rangle$. Let $B_{i+1} = B_i \cdot \langle [d_l, d_{k+1}], \mathcal{L}([d_l, d_{k+1}])\rangle$. Since the step rule can be been applied only to non-closed branches (and it does not close any branch), B_{i+1} is non-closed.

– **The fill-in rule has been applied.**
Let n be the node decorated with $\langle [d_j, d_l], A\rangle$ such that there exist no nodes in B_i decorated with $\langle [d_l, d_k], A'\rangle$ for some atom A'. By inductive hypothesis, $A = \mathcal{L}([d_j, d_l])$. Since $\mathbf{L}$ is a LIS, $\mathcal{L}([d_j, d_l])\ R_\varphi\ \mathcal{L}([d_l, d_k])$. Hence, there must exists in $\mathcal{T}$ a successor of the last node of B_i decorated with $\langle [d_l, d_k], \mathcal{L}([d_l, d_k])\rangle$. Let $B_{i+1} = B_i \cdot \langle [d_l, d_k], \mathcal{L}([d_l, d_k])\rangle$. As before, since the fill-in rule can be applied only to non-closed branches (and it does not close any branch), B_{i+1} is not closed.

Now we show that B is the fulfilling branch we are searching for. Since B is not closed, two cases may arise.

– *B is non-blocked and the expansion strategy cannot be applied anymore.* Since B is not closed, this means that there exist no active nodes in B, that is, for every node $n \in B$ and every formula $\langle A\rangle\psi \in A(n)$, there exists a node n' fulfilling $\langle A\rangle\psi$. Hence, B is a fulfilling branch.

– *B is blocked.* This implies that $\mathrm{REQ}(d_k)$ is repeated $\frac{|\mathrm{TF}(\varphi)|}{2} + 1$ times in B. Since B is decorated coherently to $\mathbf{L}$ from d_0 to d_k, by Theorem 4.11, we can assume $\mathbf{L}$ to be infinite. Let d_j be the smallest point in D_B such that $\mathrm{REQ}(d_j) = \mathrm{REQ}(d_k)$. We have that $\mathbf{L}$ is ultimately periodic, with prefix $l = j - 1$, since (by Theorem 4.13) the only set of requests which has been repeated $\frac{|\mathrm{TF}(\varphi)|}{2} + 1$ times in B is the one associated with the first point in the period. Furthermore, we have that, between d_{l+1} and d_{k-1}, there are exactly $\frac{|\mathrm{TF}(\varphi)|}{2}$ repetitions of the period of $\mathbf{L}$. This allows us to exploit the structural properties of $\mathbf{L}$ to prove that B is fulfilling.
For every node $n \in B$ decorated with $\langle [d, d'], A\rangle$ and for every formula $\langle A\rangle\psi \in A$, since $\mathbf{L}$ is fulfilling, there exists a point d'' in D such that $\psi \in \mathcal{L}([d', d''])$. If $d'' \leq d_k$, then $\langle A\rangle\psi$ is fulfilled in B. Otherwise, there exists some point d_m, with $d_i \leq d_m \leq d_k$, such that $\mathrm{REQ}(d'') = \mathrm{REQ}(d_m)$. Hence, the atom $A' = \mathcal{L}([d', d''])$ can be chosen in order to satisfy condition (a) of the definition of fulfilling branch.
For every node $n \in B$ decorated with $\langle [d, d'], A\rangle$ and for every point $d_j \leq d_m \leq d_k$, we have that $\mathrm{REQ}(d_m) \in Inf(\mathbf{L})$. Hence, there exist infinitely many points d_n in $\mathbf{L}$ such that $\mathrm{REQ}(d_m) = \mathrm{REQ}(d_n)$ and $d' < d_n$. Let d_n be one of such points. We can choose the atom $A' = \mathcal{L}([d', d_n])$ to satisfy condition (b) of the definition of fulfilling branch. □

7. A tableau-based decision procedure for RPNL$^+$

In this section we briefly show how to adapt the decision procedure for RPNL$^-$ to RPNL$^+$. First of all, we define the notion of *non-strict* φ-labelled interval structure as follows.

Definition 7.1. A *non-strict φ-labelled interval structure* (non-strict LIS, for short) is a pair $\mathbf{L} = \langle\langle \mathbb{D}, \mathbb{I}(\mathbb{D})^+\rangle, \mathcal{L}\rangle$, where $\langle \mathbb{D}, \mathbb{I}(\mathbb{D})^+\rangle$ is a non-strict interval structure and $\mathcal{L} : \mathbb{I}(\mathbb{D})^+ \to A_\varphi$ is a *labelling function* such that, for every pair of neighboring intervals $[d_i, d_j], [d_j, d_k] \in \mathbb{I}(\mathbb{D})^+$, $\mathcal{L}([d_i, d_j])\ R_\varphi\ \mathcal{L}([d_j, d_k])$.

It is possible to prove that Theorems 4.8, 4.11, and 4.13 hold also for non-strict LISs. Furthermore, we can easily tailor the tableau-based decision method for RPNL$^-$ to RPNL$^+$ by adding the following expansion rule:

3. *Point-intervals rule*: if d_k is the last point of D_B and there exists a node $n \in B$ decorated with $\langle [d_j, d_k], A\rangle$, with $d_j < d_k$, such that there are no nodes in B decorated with $\langle [d_k, d_k], A'\rangle$ for some $A' \in A_\varphi$, then take any atom $A'' \in A_\varphi$ such that $A\ R_\varphi\ A''$ and $\mathrm{REQ}(d_k) = A'' \cap \mathrm{TF}(\varphi)$, and expand B to $B \cdot n'$, with $\nu(n') = \langle [d_k, d_k], A''\rangle$.

The expansion strategy has to be expanded accordingly to take into account this new rule; on the contrary, the blocking condition, the definition of initial tableau, and the definition of fulfilling branch remain unchanged. Termination, soundness, and completeness of the resulting tableau method for RPNL$^+$ can be proved as in the case of RPNL$^-$.

Finally, to prove the optimality of the tableau for RPNL$^+$, we can exploit the reduction given in Section 5, provided that we replace $\langle A\rangle$ by $\Diamond_r$ and $[A]$ by $\Box_r$.

Theorem 7.2. *The satisfiability problem for* RPNL$^+$*, over natural numbers, is NEXPTIME-complete.*

8. Conclusions

In this paper, we focussed our attention on interval logics of temporal neighborhood. We addressed the satisfiability problem for the future fragment of strict Neighborhood Logic (RPNL$^-$), interpreted over natural numbers, and we showed that it is NEXPTIME-complete. In particular, we proved NEXPTIME-hardness by a reduction from

the exponential tiling problem. Then, we developed a sound and complete tableau-based decision procedure for RPNL$^-$ and we proved its optimality. We concluded the paper by briefly showing that such a procedure can be easily adapted to non-strict RPNL (RPNL$^+$).

The proposed decision procedure improves the EXPSPACE tableau-based decision method for checking RPNL$^-$ satisfiability developed by Bresolin and Montanari in [4]. We do not see any relevant problem in adapting our procedure to deal with the satisfiability problem for RPNL$^-$ interpreted over branching temporal structures (where every branch is isomorphic to natural numbers) [5]. Furthermore, we believe it possible to generalize it to cope with Branching Time Neighborhood Logic [3], a branching-time interval neighborhood logic that interleaves operators that quantify over possible branches with operators that quantify over intervals belonging to a given branch. On the contrary, the extension to full PNL turned out to be much more difficult. In particular, there is not a straightforward way of generalizing the basic removal technique exploited by Theorems 4.11 and 4.13 to bound the search space. In the presence of past operators, indeed, the removal of a point may affect both future existential formulae and past existential ones, and there is not an easy way to fix the future and past *defects* (see Section 4) it may introduce.

Acknowledgements

This work has been funded by the bilateral project "Temporal logics in computer and information sciences", supported by the Italian Ministero degli Affari Esteri and the National Research Foundation of South Africa, under the Joint Italy/South Africa Science and Technology Agreement, the European INTAS project on "Algebraic and Deduction Methods in Non Classical Logics and their Applications to Computer Science", and the Italian national project on "Constraints and preferences as a unifying formalism for system analysis and solution of real-life problems".

References

1. Börger, E., E. Grädel, and Y. Gurevich: 1997, *The Classical Decision Problem*, Perspectives of Mathematical Logic. Springer.
2. Bowman, H. and S. Thompson: 2003, 'A decision procedure and complete axiomatization of finite interval temporal logic with projection'. *Journal of Logic and Computation* **13**(2), 195–239.
3. Bresolin, D. and A. Montanari: 2005a, 'A tableau-based decision procedure for a branching-time interval temporal logic'. In: H. Schlingloff (ed.): *Proc. of the 4th Int. Workshop on Methods for Modalities*. Berlin, Germany, pp. 38–53.

4. Bresolin, D. and A. Montanari: 2005b, 'A tableau-based decision procedure for Right Propositional Neighborhood Logic'. In: *Proc. of TABLEAUX 2005*, Vol. 3702 of *LNAI*. Koblenz, Germany, pp. 63–77.
5. Bresolin, D. and A. Montanari: 2005c, 'A tableau-based decision procedure for right propositional neighborhood logic'. Technical Report 02, Dipartimento di Matematica e Informatica, Università di Udine, Italy.
6. Emerson, E. and J. Halpern: 1985, 'Decision procedures and expressiveness in the temporal logic of branching time'. *Journal of Computer and System Sciences* **30**(1), 1–24.
7. Emerson, E. and A. Sistla: 1984, 'Deciding Full Branching Time Logic'. *Information and Control* **61**(3), 175–201.
8. Goranko, V., A. Montanari, and G. Sciavicco: 2003a, 'A general tableau method for propositional interval temporal logic'. In: *Proc. of TABLEAUX 2003*, Vol. 2796 of *LNAI*. Rome, Italy, pp. 102–116.
9. Goranko, V., A. Montanari, and G. Sciavicco: 2003b, 'Propositional interval neighborhood temporal logics'. *Journal of Universal Computer Science* **9**(9), 1137–1167.
10. Goranko, V., A. Montanari, and G. Sciavicco: 2004, 'A road map of interval temporal logics and duration calculi'. *Journal of Applied Non-Classical Logics* **14**(1–2), 9–54.
11. Goranko, V., A. Montanari, G. Sciavicco, and P. Sala: 2006, 'A general tableau method for propositional interval temporal logics: theory and implementation'. *Journal of Applied Logic*. To appear.
12. Halpern, J. and Y. Shoham: 1991, 'A propositional modal logic of time intervals'. *Journal of the ACM* **38**(4), 935–962.
13. Kesten, Y., Z. Manna, H. McGuire, and A. Pnueli: 1993, 'A decision algorithm for full propositional temporal logic'. In: *Proc. of the 5th International Conference on Computer Aided Verification*, Vol. 697 of *LNCS*. pp. 97–109.
14. Lodaya, K.: 2000, 'Sharpening the Undecidability of Interval Temporal Logic'. In: *Proc. of 6th Asian Computing Science Conference*, Vol. 1961 of *LNCS*. pp. 290–298.
15. Montanari, A.: 2005, 'Propositional interval temporal logics: some promising paths'. In: *Proc. of the 12th International Symposium on Temporal Representation and Reasoning (TIME)*. pp. 201–203.
16. Montanari, A. and G. Sciavicco: 2005, 'A Decidability Proof for Propositional Neighborhood Logic (Extended Abstract)'. Contributed Talk, Trends in Logics III Conference, Warsaw - Ruciane Nida (Poland).
17. Montanari, A., G. Sciavicco, and N. Vitacolonna: 2002, 'Decidability of interval temporal logics over split-frames via granularity'. In: *Proc. of the 8th European Conference on Logics in AI*, Vol. 2424 of *LNAI*. pp. 259–270.
18. Moszkowski, B.: 1983, 'Reasoning about digital circuits'. Tech. rep. stan-cs-83-970, Dept. of Computer Science, Stanford University, Stanford, CA.
19. Otto, M.: 2001, 'Two Variable First-Order Logic over Ordered Domains'. *Journal of Symbolic Logic* **66**(2), 685–702.
20. Schmitt, P. and J. Goubault-Larrecq: 1997, 'A Tableau System for Linear-Time Temporal Togic'. In: E. Brinksma (ed.): *3rd Workshop on Tools and Algorithms for the Construction and Analysis of Systems*, Vol. 1217 of *LNCS*. pp. 130–144.
21. Venema, Y.: 1991, 'A modal logic for chopping intervals'. *Journal of Logic and Computation* **1**(4), 453–476.
22. Wolper, P.: 1985, 'The tableau method for temporal logic: An overview'. *Logique et Analyse* **28**, 119–136.