MTL Satisfiability over the Integers

Carlo A. Furia and Paola Spoletini

February 2008

Abstract

We investigate the satisfiability problem for metric temporal logic (MTL) with both past and future operators over linear discrete bi-infinite time models — where time is unbounded both in the future and in the past — isomorphic to the integer numbers. We provide a technique to reduce satisfiability over the integers to satisfiability over the well-known mono-infinite time model of natural numbers, and we show how to implement the technique through an automata-theoretic approach. We also prove that MTL satisfiability over the integers is EXPSPACE-complete, hence the given algorithm is optimal in the worst case.

Contents

Tomorrow, and tomorrow, and tomorrow
Creeps in this petty pace from day to day
To the last syllable of recorded time;
And all our yesterdays have lighted fools
The way to dusty death.

W. Shakespeare, *Macbeth* (Act 5 Scene 5)

1 Introduction

Temporal logic has become a very widespread notation for the formal specification of systems, temporal properties, and requirements. Its popularity is significantly due to the fact that it provides highly effective conceptual tools to model, specify, and reason about systems [Eme90], and it is amenable to fully automated verification techniques, the most notable being model-checking [CGP00].

In temporal logic frameworks it is customary to model time as infinite in the future and finite in the past, i.e., with an origin; in other words, time is mono-infinite. On the contrary, models where time is infinite both in the future and in the past — i.e., it is *bi-infinite* [PP04] — have been routinely neglected. The reasons for this strong preference are mainly historical, as it has been pointed out by various authors [Eme90, Koy92, PMS07]. Namely, temporal logic has been originally introduced for the purpose of reasoning about the behavior of "ongoing concurrent programs" [Eme90, Sec. 3.1], [Pnu77], hence a model of time with an origin is appropriate since "computation begins at an initial state" [Eme90].

However, there are various motivations in favor of the adoption of bi-infinite time models [PMS07] as well, and they go beyond the obvious theoretical interest.

The first of such reasons has to do with the usage of temporal logics with operators that reference to the past — as well as the future — of the current instant, i.e., the instant at which the operator is evaluated. If past is bounded, we may have to deal with past operators referring to instants that are before the origin of time: this gives rise to so-called *border effects* [CPPS98, MMG92]. For instance, consider *yesterday* operator Y of LTL-with-past: $\mathsf{Y}p$ evaluates to true at some instant t if and only if its argument p holds at the previous instant $t-1$. Then, consider formula `Yalarm` which models an alarm being raised at the previous instant. If we evaluate the formula at the origin, the reference to the "previous" instant of time is moot as there is no such instant, and whether the evaluation should default to true or to false depends on the role the formula plays in the whole specification. A possible solution to these problems is to introduce two variants of every past operator, one defaulting to true and the other to false [CPPS98]; however, this is often complicated and cumbersome, especially in practical applications. On the contrary, the adoption of bi-infinite time gets rid of such border effects single-handedly, in a very uniform and natural manner, because there are simply no "inaccessible" instants of time.

The second main motivation for considering bi-infinite time models [PMS07] is derived from a reason for adopting mono-infinite time models: the fact that

ongoing non-terminating processes are considered. Similarly, when modeling processes that are "time invariant" (i.e., whose behavior does not depend on *absolute* time values) and where initialization can be abstracted away, a time model which is infinite both in the past and in the future is the most natural and terse assumption.

Contribution. This paper investigates temporal logic over bi-infinite discrete-time models. More precisely, we consider a linear-time model which is isomorphic to the integer numbers. Correspondingly, Metric Temporal Logic (MTL) with past operators [AH93, Koy90] is taken as temporal logic notation. It will be clear that, over the adopted discrete-time model, MTL boils down to LTL (with past operators) with a succinct encoding of constants in formulas. Hence, our results will be easily stateable in terms of LTL as well. The main contributions are as follows. First, we present a general technique to reduce the satisfiability problem for MTL over the integers to the same problem over the more familiar mono-infinite time model isomorphic to the natural numbers. Second, we show how the technique can be implemented with an automata-theoretic approach — derived from previous work of ours [MPSS03, PSSM03, Spo05] — which can work on top of the Spin model-checker [Hol03]. Third, the complexity of the MTL satisfiability problem over the integer is assessed, and it is shown that it matches the well-known upper and lower bounds for the same problem over mono-infinite discrete time domain [AH93]. To the best of our knowledge, this is the first work which analyzes the complexity of MTL (and LTL) satisfiability over bi-infinite time and provides a practical algorithm for it.

Structure of the paper. Section 2 introduces the definitions that will be used in the rest of the paper, and in particular the time model, MTL and its semantics, its relation with LTL, and the various automata used in the verification technique. Section 3 recalls a few results from [Spo05, MPSS03, PSSM03] about automata-theoretic MTL verification over mono-infinite discrete time, in order to make the paper self-contained. Then, Section 4 presents the technique for bi-infinite satisfiability checking, shows a few details about its practical implementation, and analyzes its computational complexity. Section 5 compares briefly the results of the previous sections with those from the most relevant related literature and sketches some informal considerations about the practical performances of our technique. Finally, Section 6 concludes with a summary of directions for future work.

2 Definitions and Preliminaries

This section introduces the notation and definitions that will be used in the rest of the paper.

The symbols $\mathbb{Z}$ and $\mathbb{N}$ denote respectively the set of integer numbers and the set of nonnegative integers (i.e., the natural numbers). We extend the sum over integers to the set $\mathbb{Z} \cup \{\infty\}$, by defining $k \pm \infty = \pm\infty$ for all $k \in \mathbb{Z}$. For

greater clarity, connectives and quantifiers of the meta-language are typeset in a bold underlined font.

2.1 Metric Temporal Logic

We define Metric Temporal Logic (MTL) [AH93, Koy90] over mono-infinite and bi-infinite linear discrete time. We always consider the variant with both past and future operators (called MTL^{P} by some authors [AH93]).

2.1.1 Syntax

Let $\Pi = \{p, q, \ldots\}$ be a finite set of propositions. MTL formulas are given by:

$$\phi ::= p \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \phi_1 \,\mathsf{U}_I\, \phi_2 \mid \phi_1 \,\mathsf{S}_I\, \phi_2$$

where $p \in \Pi$, I is an interval of the naturals (possibly unbounded to the right), and the symbols U_I, S_I denote the bounded *until* and *since* operator, respectively.

Standard abbreviations are assumed such as $\top, \bot, \vee, \Rightarrow, \Leftrightarrow$. In addition, we introduce some useful derived temporal operators: the (bounded) *eventually* $\mathsf{F}_I\phi = \top \,\mathsf{U}_I\, \phi$; the (bounded) *always* $\mathsf{G}_I\phi = \neg\mathsf{F}_I\neg\phi$; the (bounded) *next* $\mathsf{X}_k\phi = \mathsf{F}_{[k,k]}\phi$; the (bounded) *release* $\phi_1 \mathsf{R}_I\, \phi_2 = \neg(\neg\phi_1 \mathsf{U}_I\, \neg\phi_2)$. Each of these operators has its past counterpart; that is, respectively: $\mathsf{P}_I\phi = \top \,\mathsf{S}_I\, \phi$ (eventually in the *past*); $\mathsf{H}_I\phi = \neg\mathsf{P}_I\neg\phi$ (*historically*); $\mathsf{Y}_k\phi = \mathsf{P}_{[k,k]}\phi$ (*previous* or *yesterday*); $\phi_1 \,\mathsf{T}_I\, \phi_2 = \neg(\neg\phi_1 \,\mathsf{S}_I\, \neg\phi_2)$ (*trigger*). Note that, whenever no interval is specified, $I = (0, \infty)$ is assumed for all operators except X where the interval $[1, 1]$ is assumed instead; also, the singleton interval $[k, k]$ is abbreviated by $= k$.

Precedence of operators is defined as follows: $\neg$ has the highest binding power, then we have the temporal modalities U_I, S_I and derived ones, then $\wedge$ and $\vee$, $\Rightarrow$, and finally $\Leftrightarrow$.

For a set S, $\mathbb{B}(S)$ denotes the set of all Boolean combinations of elements in S, and $\mathbb{B}^+(S)$ the set of all *positive* Boolean combinations (i.e., negation-free combinations). We also introduce the shorthand $\text{Alw}(\phi)$ to denote that ϕ holds always over the time domain, i.e., $\text{Alw}(\phi) = \mathsf{G}\phi \wedge \phi \wedge \mathsf{H}\phi$. $\widetilde{\phi}$ denotes the formula obtained from ϕ by switching every future operator with its past counterpart, and *vice versa*. For instance $\widetilde{\theta} = p \wedge q \,\mathsf{S}\, \neg p \vee \mathsf{F}r$ for $\theta = p \wedge q \,\mathsf{U}\, \neg p \vee \mathsf{P}r$. Clearly $\widetilde{\widetilde{\phi}} = \phi$ holds for all ϕ.

Classes of formulas. The size $|\phi|$ of a formula ϕ is given by the product of its number of connectives $|\phi|_{\#}$ times the size $|\phi|_{\text{M}}$ of the largest constant used in its formulas, succinctly encoded in binary.

A *future* formula ϕ is a formula which does not use any past operator; conversely, a *past* formula π is a formula which does not use any future operator.

A formula ψ is *flat* if it does not nest temporal operators, i.e., it is definable by:

$$\psi \quad ::= \quad p \mid \neg\psi \mid \psi_1 \wedge \psi_2 \mid \beta_1 \,\mathsf{U}_I\, \beta_2 \mid \beta_1 \,\mathsf{S}_I\, \beta_2$$

where $p \in \Pi$ and $\beta_1, \beta_2 \in \mathbb{B}(\Pi)$. A flat formula is *propositional* if it does not use temporal operators at all.

2.1.2 Words and operations on them

For a finite alphabet Σ, we introduce the sets of right-infinite words (called ω-words, read "omega words"), of left-infinite words (called $\widetilde{\omega}$-words, read "omega-reverse words"), and of bi-infinite words (called $\mathbb{Z}$-words, read "zee words"[1]) over Σ, and we denote them as Σ^ω, ${}^\omega\Sigma$, and $\Sigma^{\mathbb{Z}}$, respectively. Correspondingly, an ω-language (resp. $\widetilde{\omega}$-language, $\mathbb{Z}$-language) is a subset of Σ^ω (resp. ${}^\omega\Sigma$, $\Sigma^{\mathbb{Z}}$).

Given an ω-word $w = w_0 w_1 w_2 \cdots$, $\widetilde{w}$ denotes the $\widetilde{\omega}$-word $\cdots w_{-2} w_{-1} w_0$ defined by the bijection $w_{-k} = w_k$ for $k \in \mathbb{N}$. The same notation is used for the inverse mapping from $\widetilde{\omega}$-words to ω-words. The mapping is also extended to languages as obvious, with the same notation. Given a $\mathbb{Z}$-word $x = \cdots x_{-2} x_{-1} x_0 x_1 x_2 \cdots$ and $k \in \mathbb{Z}$, x^k denotes the ω-word obtained by truncating x at x_k on the left, i.e., $x^k = x_k x_{k+1} x_{k+2} \cdots$; similarly, ${}^k x$ denotes the $\widetilde{\omega}$-word obtained by truncating x at x_k on the right, i.e., ${}^k x = \cdots x_{k-2} x_{k-1} x_k$.

The operations of intersection ($\cap$), union ($\cup$), and concatenation ($.$) for words and languages are defined as usual. Let w and $\overline{w}$ be an ω- and an $\widetilde{\omega}$-word, respectively. The $\mathbb{Z}$-word $\overline{w} \triangleright w$ (*right join*) is defined as ${}^{-1}\overline{w}.w$, and the $\mathbb{Z}$-word $\overline{w} \triangleleft w$ (*left join*) is defined as $\overline{w}.w^1$. The join operations are extended to languages as obvious, with the same notation.

MTL formulas using a set of propositions Σ will be interpreted on infinite words over the alphabet 2^Σ. Correspondingly, $\downarrow^\Sigma$ denotes the projection homomorphism over Σ: for a word $w = w_0 w_1 w_2 \cdots$ over 2^Σ, $\downarrow^\Sigma w$ is the word $w'_0 w'_1 w'_2 \cdots$ obtained by removing all elements which are not in Σ from the w_i's; that is, $w'_i = w_i \cap \Sigma$ for all i's. The $\downarrow^\Sigma$ operator is extended to languages as obvious, with the same notation.

2.1.3 Semantics

We define the semantics of MTL formulas for infinite words over 2^Π, where Π is a finite set of atomic propositions. As it is standard, every letter $y_k \in 2^\Pi$ in such words represents the set of atomic propositions that are true at integer time instant k (also called *position*). We introduce the predicate $\mathbf{valid}(y, i)$ which holds iff i is a valid position in the infinite word y, i.e., iff y is a $\mathbb{Z}$-word and $i \in \mathbb{Z}$, or y is an ω-word and $i \in \mathbb{N}$, or y is an $\widetilde{\omega}$-word and $-i \in \mathbb{N}$.

Let ϕ be an MTL formula, y be a generic infinite word over 2^Π, and i be an integer such that $\mathbf{valid}(y, i)$. The satisfaction relation $\models$ is defined inductively as:

[1] Or "zed words", if your prefer.

$$
\begin{array}{lcl}
y,i \models p & \underline{\Leftrightarrow} & p \in y_i \\
y,i \models \neg\phi & \underline{\Leftrightarrow} & y,i \not\models \phi \\
y,i \models \phi_1 \wedge \phi_2 & \underline{\Leftrightarrow} & y,i \models \phi_1 \ \underline{\wedge}\ y,i \models \phi_2 \\
y,i \models \phi_1 \,\mathsf{U}_I\, \phi_2 & \underline{\Leftrightarrow} & \underline{\exists} d \in I\colon (\mathbf{valid}(y,i+d) \ \underline{\wedge} \\
 & & \quad y,i+d \models \psi_2 \underline{\wedge} \underline{\forall} 0 < u < d : y,i+u \models \psi_1) \\
y,i \models \phi_1 \,\mathsf{S}_I\, \phi_2 & \underline{\Leftrightarrow} & \underline{\exists} d \in I\colon (\mathbf{valid}(y,i-d) \ \underline{\wedge} \\
 & & \quad y,i-d \models \psi_2 \underline{\wedge} \underline{\forall} 0 < u < d : y,i-u \models \psi_1) \\
y \models \phi & \underline{\Leftrightarrow} & \underline{\forall} i \in \mathbb{Z} : (\mathbf{valid}(y,i) \ \underline{\Rightarrow}\ y,i \models \phi)
\end{array}
$$

Remark 2.1. For $k \in \mathbb{N}$, formula $\mathsf{H}_{=k}\bot$ holds over an ω-word w exactly at all positions $j < k$. In fact, $w,j \models \mathsf{H}_{=k}\bot$ is the case only if $\mathbf{valid}(w, j-k)$ is false, that is $j-k<0$. Similarly, formula $\mathsf{P}_{=k}\top \wedge \mathsf{H}_{=k+1}\bot$ holds over an ω-word w exactly at position k, and nowhere else.

Example 2.2 (Border effects)**.** Consider formula $\nu = \mathsf{H}_{[0,3]}p$ and its interpretation over ω-word w^+ in Figure 1. According to the semantics defined above, ν is true at 1 because p holds for all *valid* positions between 1 and $1-3=-2$. However, there may be justifications in favor of evaluating ν false at 1: there is no complete interval of size 4 where p holds continuously. This is an example of so-called *border effect*: what is a "reasonable" evaluation of formulas near the origin is influenced by the role the formulas play in a specification.

There is an interesting relation between the reverse $\widetilde{\phi}$ of a formula ϕ and the reverse $\widetilde{w}$ of ω-words w that are models of ϕ, as the following example shows.

Example 2.3. Consider formula $\theta = \mathsf{H}_{[0,3]}p \Rightarrow \mathsf{F}q$ and its reverse $\widetilde{\theta} = \mathsf{G}_{[0,3]}p \Rightarrow \mathsf{P}q$. θ asserts that whenever p held continuously for 4 time units, q must hold somewhere in the future (excluding the current instant), hence θ is true at position 4 and false at position 11 over ω-word w^+ in Figure 1. If we consider $\widetilde{\omega}$-word w^- obtained by reversing w^+ (also in Figure 1), we see that $\widetilde{\theta}$ is true at position -4 and false at position -11 over w^-.

By generalizing the example, we have the following proposition.

Proposition 2.4. *Let $w^+ \in \left(2^\Sigma\right)^\omega$ and $w^- \in {}^\omega\left(2^\Sigma\right)$ be an ω-word and an $\widetilde{\omega}$-word, respectively, ϕ be an MTL formula, and $i \in \mathbb{N}$. Then: $w^+, i \models \phi$ iff $\widetilde{w^+}, -i \models \widetilde{\phi}$; and $w^-, -i \models \phi$ iff $\widetilde{w^-}, i \models \widetilde{\phi}$*

Proof. The proof is a straightforward application of the definitions above, and it is therefore omitted. □

Satisfiability and language of a formula. Satisfiability is the following problem: "given a formula ϕ is there some word y such that $y \models \phi$?". It is the verification problem we will consider in this paper.

Note that we defined $y \models \phi$ to denote "global satisfiability", i.e., the fact that ϕ holds at all valid positions of y. This definition is especially natural over bi-infinite words, where there is no initial instant at which to evaluate

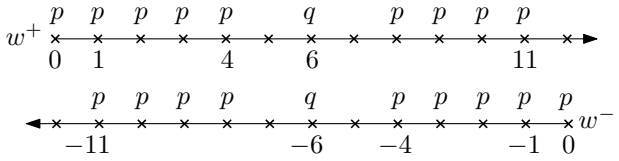

Figure 1: ω-word w^+ (above) and its reverse $\widetilde{\omega}$-word w^- (below).

formulas. On the contrary, "initial satisfiability" is more common over mono-infinite words where an origin is unambiguously fixed. However, the global satisfiability problem is easily reducible to the initial satisfiability problem, as $\underline{\forall} i : y, i \models \phi$ iff $y, 0 \models \mathrm{Alw}(\phi)$.

For an MTL formula ϕ, let $\mathcal{L}_0^\omega(\phi)$ denote the set of ω-words w such that $w, 0 \models \phi$, let $\mathcal{L}^\omega(\phi)$ denote the set of ω-words w such that $w \models \phi$, and let $\mathcal{L}^{\mathbb{Z}}(\phi)$ denote the set of $\mathbb{Z}$-words x such that $x \models \phi$. Then, the satisfiability problem for a formula ϕ is equivalent to the emptiness problem for the corresponding language.

LTL and expressiveness. LTL is a well-known linear temporal logic based on the unique modality U (note that X can be derived from it as $\mathsf{X}\phi \equiv \bot \,\mathsf{U}\, \phi$). Originally LTL did not include past modalities (i.e., S and Y which is derivable from the former), mainly because they do not add expressive power [GHR94]. However, it has been acknowledged that past operators are very useful in formalizing certain properties naturally [LPZ85] and concisely [LMS02]. In this paper, we will consider the past-enhanced variant of the logic, and call it simply "LTL".

For the time models we consider in this paper, MTL is simply LTL with an exponentially succinct encoding. In fact, the following equivalences hold, for $0 < l \leq u < \infty$, $1 < d < \infty$, $0 \leq e < \infty$, $0 \leq f \leq \infty$, and I a bounded interval of the naturals:[2]

$$
\begin{array}{lcl}
\mathsf{X}_{=e}\phi & \equiv & \underbrace{\mathsf{X}\mathsf{X}\cdots\mathsf{X}}_{e \text{ times}}\phi \\
\mathsf{G}_I\phi & \equiv & \bigwedge_{k\in I} \mathsf{X}_{=k}\phi_1 \\
\phi_1 \,\mathsf{U}_{\emptyset}\, \phi_2 & \equiv & \bot \\
\phi_1 \,\mathsf{U}_{=e}\, \phi_2 & \equiv & \mathsf{G}_{(0,e)}\phi_1 \wedge \mathsf{X}_{=e}\phi_2 \\
\phi_1 \,\mathsf{U}_{(0,d)}\, \phi_2 & \equiv & \mathsf{X}\left(\phi_2 \vee \left(\phi_1 \;\wedge\; \phi_1 \,\mathsf{U}_{(0,d-1)}\, \phi_2\right)\right) \\
\phi_1 \,\mathsf{U}_{(l,u)}\, \phi_2 & \equiv & \mathsf{G}_{(0,l]}\phi_1 \wedge \mathsf{X}_{=l}\left(\phi_1 \,\mathsf{U}_{(0,u-l)}\, \phi_2\right) \\
\phi_1 \,\mathsf{U}_{(e,\infty)}\, \phi_2 & \equiv & \mathsf{G}_{(0,e]}\phi_1 \wedge \mathsf{X}_{=e}\left(\phi_1 \,\mathsf{U}\, \phi_2\right) \\
\phi_1 \,\mathsf{U}_{[0,f)}\, \phi_2 & \equiv & \phi_2 \;\vee\; \phi_1 \,\mathsf{U}_{(0,f)}\, \phi_2 \\
\phi_1 \,\mathsf{U}_{[l,u]}\, \phi_2 & \equiv & \phi_1 \,\mathsf{U}_{(l,u)}\, \phi_2 \;\vee\; \phi_1 \,\mathsf{U}_{=l}\, \phi_2 \;\vee\; \phi_1 \,\mathsf{U}_{=u}\, \phi_2
\end{array}
$$

[2]Notice that the given equivalences are sufficient to translate any occurrence of the bounded *until* operator, because $[l, u] = (l - 1, u + 1)$ over the naturals. Also, we omit the encoding of the *since* modality which is similarly derivable.

Hence, every MTL formula μ can be translated into an LTL formula λ such that $|\lambda| = |\lambda|_{\#} = \exp \mathrm{O}(|\mu|_{\#} |\mu|_{\mathrm{M}})$, due to the succinct encoding assumption for MTL formulas.

Example 2.5 (From MTL to LTL)**.** According to the rules presented above, formula $\theta = \mathsf{H}_{[0,3]} p \Rightarrow \mathsf{F} q$ is equivalently expressible in LTL as $\theta' = p \wedge \mathsf{Y} p \wedge \mathsf{YY} p \wedge \mathsf{YYY} p \Rightarrow \mathsf{F} q$.

Timed words and the integers. MTL is commonly interpreted over *timed words* with integer timestamps [AH93]. Usually, timed ω-words are considered, although timed $\widetilde{\omega}$-words or timed $\mathbb{Z}$-words could be defined in a natural way. Timed ω-words are ω-words over $2^{\Pi} \times \mathbb{N}$: every element $w_k = \langle \sigma_k, \delta_k \rangle$ in a timed ω-word $w = w_0 w_1 \cdots$ records a set σ_k of atomic proposition that are true at absolute integer time $t_k = \sum_{i=0}^{k} \delta_k$, with the condition that $\delta_k > 0$ for all $k > 0$. The satisfaction relation $\models^{\tau}$ for MTL over timed ω-words is defined as follows, where z is a timed ω-word and $i \in \mathbb{N}$:

$$
\begin{array}{lcl}
z, i \models^{\tau} p & \Leftrightarrow & p \in \sigma_i \\
z, i \models^{\tau} \neg \phi & \Leftrightarrow & z, i \not\models^{\tau} \phi \\
z, i \models^{\tau} \phi_1 \wedge \phi_2 & \Leftrightarrow & z, i \models^{\tau} \phi_1 \ \underline{\wedge} \ z, i \models^{\tau} \phi_2 \\
z, i \models^{\tau} \phi_1 \, \mathsf{U}_I \, \phi_2 & \Leftrightarrow & \exists k > i : (t_k - t_i \in I \ \ \underline{\wedge} \\
 & & \qquad z, k \models \psi_2 \ \ \underline{\wedge} \, \underline{\forall} \, i < j < k : z, j \models \psi_1) \\
z, i \models^{\tau} \phi_1 \, \mathsf{S}_I \, \phi_2 & \Leftrightarrow & \exists 0 \leq k < i : (t_i - t_k \in I \ \ \underline{\wedge} \\
 & & \qquad z, k \models \psi_2 \ \ \underline{\wedge} \, \underline{\forall} \, k < j < i : z, j \models \psi_1)
\end{array}
$$

Hence, in timed words there are two "times": one is given by integer position $i \in \mathbb{N}$ in a timed word, and the other is given by timestamp t_i. Despite the consequent subtle semantic differences that arise between the timed word interpretation and the integer interpretation we introduced above, it is not difficult to show that the two models (timed ω-words and ω-words) are reconcilable, as far as satisfiability is concerned. In fact, let ϕ be an MTL formula and $i \in \mathbb{N}$. If $w, i \models \phi$ for some ω-word $w = w_0 w_1 \cdots$ then $z, i \models^{\tau} \phi$ for the timed ω-word $z = z_0 z_1 \cdots$ defined as: $z_0 = \langle w_0, 0 \rangle$ and $z_k = \langle w_k, 1 \rangle$ for $k > 0$. Conversely, let $z = z_0 z_1 \cdots$ be some timed ω-word. Then, let us extend the alphabet Π with the fresh proposition a (for "action") which is true exactly when some events are recorded in z. Let us define an ω-word $w = w_0 w_1 \cdots$ over $\Pi \cup \{a\}$ as follows: $w_{t_k} = \sigma_k \cup \{a\}$ whenever t_k is defined, and $w_k = \emptyset$ otherwise. Let ϕ' be the MTL formula obtained from ϕ by substituting every occurrence of every atomic proposition $p \in \Pi$ with the formula $p \wedge a$. It should be clear that $z, i \models^{\tau} \phi$ iff $w, i \models \phi'$. All in all, the satisfiability problems for MTL over the naturals and over timed words are inter-reducible.

2.2 Automata over Infinite Words

Languages definable in MTL can also be described as languages accepted by finite state automata such as Büchi automata [Tho90].

Definition 2.6 (Büchi automaton (BA))**.** A Büchi automaton $\mathcal{A}$ is a tuple $\langle \Sigma, Q, q_0, \delta, F \rangle$ where:

- Σ is a finite set of input symbols,
- Q is a finite set of states,
- $q_0 \in Q$ is the initial state,
- $\delta : Q \times \Sigma \to 2^Q$ is the transition relation,
- $F \subseteq Q$ is a set of accepting states.

Differently than finite state automata on *finite* words, BA accept infinite ω-words according to the Büchi condition: an ω-word $w = w_0 w_1 \cdots$ is accepted by a BA $\mathcal{A}$ iff there exists an infinite sequence of states $\mathbf{s} = q_0 q_1 q_2 \cdots$ such that q_0 is the initial state, $q_{i+1} \in \delta(q_i, w_i)$ for all $i \in \mathbb{N}$, and there exists a $\overline{q} \in F$ which appears in $\mathbf{s}$ infinitely many times.

The size $|\mathcal{A}|$ of a BA $\mathcal{A}$ is defined as $|Q|$. It is well-known that every LTL formula ϕ can be translated into a (nondeterministic) Büchi automaton $\mathcal{A}_\phi$ such that $|\mathcal{A}_\phi| = \exp \mathrm{O}(|\phi|)$ [Var06].

Alternating automata (AA, [CKS81, Var06]) are an equally expressive but possibly more concise version of BA. Formally, the transition relation δ of AA has a signature $Q \times \Sigma \to \mathbb{B}^+(Q)$. This means that AA have two kinds of transitions: nondeterministic transitions (also called existential, corresponding to $\vee$) just like vanilla BA, and parallel transitions (also called universal, corresponding to $\wedge$). Nondeterminism allows the automaton to choose, for a given input symbol, among more than one next state; a word is accepted iff at least one of the existential choices leads to an accepting run. Dually, parallelism lets the automaton move, for a given input symbol, to more than one next state in parallel; this can be seen as the creation of many parallel copies of the automaton, one for each of the possible next states. A word is accepted iff all the parallel runs are accepting.

Alternation can represent concisely the structure of an LTL formula [Var06], avoiding the exponential blow-up. In [Spo05] we introduced an enriched variant of AA which makes use of (bounded) counters; this new feature can represent succinctly MTL formulas as well, i.e., it can encode succinctly constants used in MTL modalities. Let us recall the definition of such Alternating Modulo-Counting Automata.

Definition 2.7 (Alternating Modulo Counting Automaton (AMCA) [Spo05])**.** An Alternating Modulo Counting Automaton is a tuple $\langle \Sigma, Q, \mu, q_0, \delta, F \rangle$ where:

- Σ is a finite alphabet,
- Q is a set of states,
- $\mu \in \mathbb{N}_{\geq 1}$ such that $C = [0..\mu]$ denotes a modulo-μ finite counter,
- $q_0 \in Q$ is the initial state,

- $\delta : Q \times C \times \Sigma \to \mathbb{B}^+(Q \times C)$ is the transition relation,
- $F \subseteq Q$ is a set of accepting states.

For the sake of readability when indicating the elements in $\mathbb{B}^+(Q \times C)$ we will use the symbol / to separate the component in Q from the component in C. Note that the size $|\mathcal{A}|$ of an AMCA $\mathcal{A}$ can be defined as the product of $|Q|$ times the size of the counter, succinctly encoded in binary, i.e., $|\mathcal{A}| = \mathrm{O}(|Q| \log \mu)$.

A run of an AMCA is defined as follows.

Definition 2.8 (Run of an AMCA)**.** A run (T, ρ) of an AMCA $\mathcal{A}$ on the ω-word $w = w_0 w_1 \cdots \in \Sigma^\omega$ is a $(Q \times C \times \mathbb{N})$-labeled tree, where ρ is the labeling function defined as: $\rho(\epsilon) = (q_0/0, 0)$; for all $x \in T$, $\rho(x) = (q/k, n)$; and the set $\{(q'/h, 1) \mid c \in \mathbb{N}, x.c \in T, h \in C, \rho(x.c) = (q'/h, n+1)\}$ satisfies the formula $\delta(q/k, w_n)$.

The acceptance condition for AMCA is defined similarly as for regular BA: a path is accepting iff it passes infinitely many times on at least one state in F. Formally, for a sequence $P \in \mathbb{N}^\omega$ and a labeling function ρ, let $\inf(\rho, P) = \{s \mid \rho(n) \in \{s\} \times \mathbb{N} \text{ for infinitely many } n \in P\}$. A run (T, ρ) of an AMCA is *accepting* iff for all paths P of T it is $\inf(\rho, P) \cap F \neq \emptyset$.

With the usual notation, $\mathcal{L}^\omega(\mathcal{A})$ denotes the set of all ω-words accepted by the automaton $\mathcal{A}$.

3 Automata-Based MTL Satisfiability over the Naturals

A widespread approach to testing the satisfiability of an MTL (or LTL) formula over mono-infinite time models isomorphic to the natural numbers relies on the well-known tight relationship between LTL and finite state automata. In order to test the satisfiability of an MTL formula μ, one translates it into an LTL formula λ_μ, and then builds a nondeterministic BA $\mathcal{A}_{\lambda_\mu}$ that accepts precisely the models of λ_μ, hence of μ. Correspondingly, an emptiness test on $\mathcal{A}_{\lambda_\mu}$ is equivalent to a satisfiability check of μ. This procedure, very informally presented, relies on the following two well-known results.

Proposition 3.1 ([VW94, GO03, Fri05])**.** *Given an LTL formula ϕ, one can build a (nondeterministic) BA $\mathcal{A}_\phi$ with $|\mathcal{A}_\phi| = \exp \mathrm{O}(|\phi|)$ such that $\mathcal{L}^\omega(\mathcal{A}_\phi) = \mathcal{L}^\omega(\phi)$ and $\mathcal{L}^\omega_0(\mathcal{A}_\phi) = \mathcal{L}^\omega_0(\phi)$.*

Proposition 3.2 ([Var06, VW94, EL85a, EL85b])**.** *The emptiness problem for (nondeterministic) BA of size n is decidable in time $\mathrm{O}(n)$ and space $\mathrm{O}(\log^2 n)$.*

In practice, however, this unoptimized approach is inconvenient, because the BA representing an MTL formula is in general doubly-exponential in the size of the formula, hence algorithmically very inefficient. On the contrary, we would like to exploit more concise classes of automata (such as AMCA) to represent

MTL formulas more efficiently in practice. With this aim, in [Spo05, MPSS03, PSSM03] we proposed a novel approach to model-checking and satisfiability checking over discrete mono-infinite time domains for a propositional subset of TRIO [GMM90], a metric temporal logic with both past and future modalities. It is clear that the subset of TRIO considered in [Spo05] corresponds to MTL as we defined it in this paper. Hence, in the following we briefly recall the method of [Spo05] with reference to MTL formulas. In the remainder of the paper we will show how to exploit such satisfiability checking procedures over the naturals to perform satisfiability checking over the integers.

The approach of [Spo05] considers MTL formulas in the form $\mathrm{Alw}(\pi \vee \varphi)$ where π is a past formula and φ is a future formula. The past component can be translated into a deterministic BA, whereas the future component can be translated into an AMCA. A suitable composition of the two automata is a then an acceptor for the language $\mathcal{L}^{\omega}(\mathrm{Alw}(\pi \vee \varphi))$.

Let us consider past formulas first. Since the past is bounded over ω-words, at each time instant the prefix of a word — i.e., the only part of the word needed to evaluate the past formula — is finite. From this consideration one proves the following proposition. Intuitively, it asserts that for every past formula π it is possible to build a deterministic BA such that the ω-language accepted by the automaton is equivalent to the language of π.

Proposition 3.3 (Past automaton [Spo05])**.** *Given a past MTL formula π, one can build two deterministic BA $\mathcal{A}_{\mathrm{Alw}(\pi)}$ and $\mathcal{A}_{\pi}$, called past automaton of* $\mathrm{Alw}(\pi)$ *and π, respectively, such that:*

- $\mathcal{L}^{\omega}(\mathcal{A}_{\pi}) = \mathcal{L}_0^{\omega}(\pi)$*;*
- $\mathcal{L}^{\omega}\big(\mathcal{A}_{\mathrm{Alw}(\pi)}\big) = \mathcal{L}_0^{\omega}(\mathrm{Alw}(\pi)) = \mathcal{L}^{\omega}(\pi)$*;*
- *the size of both $\mathcal{A}_{\mathrm{Alw}(\pi)}$ and $\mathcal{A}_{\pi}$ is* $\exp \mathrm{O}(|\pi|)$.

On the contrary, the evaluation of a future formula depends in general on the whole infinite future of the current instant. Correspondingly, future formulas are translated into AMCA according to the schema we sketch in the following. The AMCA for a future formula φ over alphabet Π is $\mathcal{A}_{\varphi} = \langle \Sigma, Q, \mu, q_0, \delta, F \rangle$ where:

- $\Sigma = 2^{\Pi}$,
- $Q = \{\nu \mid \nu \text{ is a subformula of } \varphi\} \cup \{\neg\nu \mid \nu \text{ is a subformula of } \varphi\}$,
- $\mu = |\varphi|_{\mathrm{M}}$,
- $q_0 = \varphi$,
- the transition relation δ is defined as follows:
  - $\delta(\chi/0, p) = \top/0$ for $\chi \in \Pi$ and $\chi = p$,
  - $\delta(\chi/0, p) = \bot/0$ for $\chi \in \Pi$ and $\chi \neq p$,

- $\delta(\psi \wedge \upsilon/0, p) = \delta(\psi/0, p) \wedge \delta(\upsilon/0, p)$,
- $\delta(\neg\psi/0, p) = \mathsf{dual}(\delta(\psi/0, p))$, where $\mathsf{dual}(\phi)$ is a formula obtained from ϕ by switching $\top$ and $\bot$, $\wedge$ and $\vee$, and by complementing all subformulas of ϕ,
- $\delta(\psi \mathsf{U}_{[a,b]}\, \upsilon/k, p) = \begin{cases} \psi \,\mathsf{U}_{[a,b]}\, \upsilon/k+1 & k = 0 \\ \delta(\psi/0, p) \wedge \left(\psi \,\mathsf{U}_{[a,b]}\, \upsilon/k+1\right) & 0 < k < a \\ \delta(\upsilon/0, p) \vee \left(\delta(\psi/0, p) \wedge \left(\psi \,\mathsf{U}_{[a,b]}\, \upsilon/k+1\right)\right) & a \leq k \leq b \\ \bot & k > b \end{cases}$
  for $a \leq b < \infty$,
- $\delta(\psi \,\mathsf{U}\, \upsilon/k, p) = \delta(\upsilon/0, p) \vee (\delta(\psi/0, p) \wedge (\psi \,\mathsf{U}\, \upsilon/0))$,

- $F = \{\xi \mid \xi \in Q \text{ and } \xi \text{ has the form } \neg(\psi \,\mathsf{U}\, \upsilon)\}$

Correspondingly, we have the following proposition:

Proposition 3.4 (Future automaton [Spo05])**.** *Given a future MTL formula φ, one can build two AMCA $\mathcal{A}_{\mathrm{Alw}(\varphi)}$ and $\mathcal{A}_\pi$, called future automaton of* $\mathrm{Alw}(\varphi)$ *and φ, respectively, such that:*

- $\mathcal{L}^\omega(\mathcal{A}_\varphi) = \mathcal{L}_0^\omega(\varphi)$*;*
- $\mathcal{L}^\omega\big(\mathcal{A}_{\mathrm{Alw}(\varphi)}\big) = \mathcal{L}_0^\omega(\mathrm{Alw}(\varphi)) = \mathcal{L}^\omega(\varphi)$*;*
- *the size of both $\mathcal{A}_{\mathrm{Alw}(\varphi)}$ and $\mathcal{A}_\varphi$ is* $\mathrm{O}(|\varphi|)$.

4 Automata-Based MTL Satisfiability over the Integers

This section presents the main contribution of the paper: a technique to reduce the satisfiability problem for MTL formulas over the integers to the same problem over the naturals, and how to actually implement the technique with an automata-based model checker. To this end, Sub-Section 4.1 shows how any MTL formula can be translated into an equi-satisfiable formula in a canonical normal form which allows for a straightforward presentation of our technique. Then, Sub-Section 4.2 shows how satisfiability of MTL formulas in normal form over the integers can be reduced to satisfiability of related formulas over the naturals, by taking into account what happens about the origin. Sub-Section 4.3 shows how to implement the satisfiability check over the naturals, introduced in the previous sub-sections, through a suitable automata-based technique. Finally, Sub-Section 4.4 summarizes the satisfiability algorithm and analyzes its worst-case complexity.

4.1 Flat Normal Form

We introduce a suitable normal form where each application of temporal operators can be analyzed in isolation, and we show that any MTL formula can be translated into this normal form by introducing auxiliary atomic proposition but without changing the asymptotic size of the formula.

Definition 4.1 (Flat normal form)**.** An MTL formula ϕ is in *flat normal form* when it is written as:

$$\beta \wedge \bigwedge_{k=1}^{n} \mathrm{Alw}(p_k \Leftrightarrow \psi_k) \tag{1}$$

where $\beta \in \mathbb{B}(\Sigma)$ and ψ_k is a flat formula, for all $k = 1, \ldots, n$. In addition, if every ψ_k is a pure past formula or a pure future formula, Formula 1 is named flat *separated* normal form.

For a generic MTL formula ϕ over an alphabet Σ, let us show how to build an MTL formula ϕ' over an alphabet $\Sigma' \supseteq \Sigma$ such that ϕ' is in flat separated normal form and ϕ and ϕ' are such that $\mathcal{L}(\phi) =\downarrow^{\Sigma} \mathcal{L}(\phi')$. The idea is straightforward: for every temporal subformula μ in ϕ we add a new propositions p_μ to Σ'. p_μ is an alias for μ and thus it is defined by $\mathrm{Alw}(p_\mu \Leftrightarrow \mu)$. By applying this idea recursively we get to the desired form.

More formally, the set of temporal subformulas of ϕ, denoted by $\mathbf{tsf}(\phi)$, is defined inductively as:

$$\begin{array}{lcl}
\mathbf{tsf}(\beta) & = & \emptyset \\
\mathbf{tsf}(\neg\phi) & = & \mathbf{tsf}(\phi) \\
\mathbf{tsf}(\phi_1 \wedge \phi_2) & = & \mathbf{tsf}(\phi_1) \cup \mathbf{tsf}(\phi_2) \\
\mathbf{tsf}(\phi_1 \,\mathsf{U}_I\, \phi_2) & = & \{\phi_1 \,\mathsf{U}_I\, \phi_2\} \cup \mathbf{tsf}(\phi_1) \cup \mathbf{tsf}(\phi_2) \\
\mathbf{tsf}(\phi_1 \,\mathsf{S}_I\, \phi_2) & = & \{\phi_1 \,\mathsf{S}_I\, \phi_2\} \cup \mathbf{tsf}(\phi_1) \cup \mathbf{tsf}(\phi_2)
\end{array}$$

where β is a propositional formula. Notice that $|\mathbf{tsf}(\phi)| \leq |\phi|_{\#}$.

We build a an extended alphabet Σ', a new set of formulas Ξ, and a formula φ from Σ, $\mathbf{tsf}(\phi)$, and ϕ, respectively, as follows. Let initially $\Sigma' := \Sigma$, $\Xi := \mathbf{tsf}(\phi)$, and $\varphi := \phi$. We repeatedly pick a flat formula ψ from Ξ and recursively: (1) add the element p_ψ to Σ'; (2) replace every occurrence of ψ in all (non-flat) elements of Ξ with p_ψ; and (3) replace every occurrence of ψ in φ with p_ψ. Note that, when no more substitutions can be made, all formulas in Ξ are a flat application of a single temporal operator and φ is a propositional formula. Also, the number of elements in Ξ does not change during the process. Finally, we build ϕ' as:

$$\phi' \quad = \quad \varphi \wedge \bigwedge_{\mu \in \Xi} \mathrm{Alw}(p_\mu \Leftrightarrow \mu)$$

It is clear that the following theorem holds by construction.

Theorem 4.2. *Let ϕ be an MTL formula over Σ, and let ϕ' be the formula in flat separated normal form built as above. Then $\mathcal{L}^{\mathbb{Z}}(\phi) =\downarrow^{\Sigma} \mathcal{L}^{\mathbb{Z}}(\phi')$, $|\phi'|_{\mathrm{M}} = |\phi|_{\mathrm{M}}$, and $|\phi'|_{\#} = \mathrm{O}(|\phi|_{\#})$.*

Example 4.3. Considering formula $\theta = \mathsf{H}_{[0,3]} p \Rightarrow \mathsf{F} q$, we can build θ' by replacing $\mathsf{H}_{[0,3]} p$ and $\mathsf{F} q$ with two new Boolean literals p' and q' respectively. Hence, $\theta' = (p' \Rightarrow q') \wedge (p' \Leftrightarrow \mathsf{H}_{[0,3]} p) \wedge (q' \Leftrightarrow \mathsf{F} q)$.

4.2 Splitting the Evaluation about the Origin

The overall goal of this sub-section is providing a means to check the emptiness of the language $\mathcal{L}^{\mathbb{Z}}(\phi)$, for any MTL formula ϕ. Following Theorem 4.2, we consider instead a formula ϕ' — computed from ϕ — in flat separated normal form:

$$\phi' \quad = \quad \beta \wedge \bigwedge_{k=1}^{n} (p_k \Leftrightarrow \psi_k) \tag{2}$$

Notice that the satisfiability of ϕ' can be analyzed by considering each of the $n+1$ subformulas $\beta, p_k \Leftrightarrow \psi_k|_{1 \leq k \leq n}$ separately. In fact, $x \models \phi'$ iff $x \models \beta$ and $\underline{\forall} k = 1, \ldots, n : x \models p_k \Leftrightarrow \psi_k$. Hence, without loss of generality, we focus on studying the satisfiability of formulas in the form $p \Leftrightarrow \psi^+$ and $p \Leftrightarrow \psi^-$, where ψ^+ and ψ^- are flat *until* and *since* formulas, respectively.

More precisely, let us start with the future formula:

$$\psi \quad = \quad f \Leftrightarrow p \,\mathsf{U}_I\, q \;\equiv\; (\neg f \vee p \,\mathsf{U}_I\, q) \wedge (f \vee \neg p \,\mathsf{R}_I\, \neg q) \tag{3}$$

In turn, $x \models \psi$ iff $x \models \neg f \vee p \,\mathsf{U}_I\, q$ and $x \models f \vee \neg p \,\mathsf{R}_I\, \neg q$. Correspondingly, we now focus on studying the satisfiability of the simple formula $\neg f \vee p \,\mathsf{U}_I\, q$ over the integers. Once we have this fundamental characterization, we will see that it is straightforward to extend it to handle the other formula $f \vee \neg p \,\mathsf{R}_I\, \neg q$ by duality, as well as the corresponding past formula $f \Leftrightarrow p \,\mathsf{S}_I\, q$.

4.2.1 Behavior about the origin

Let us consider a $\mathbb{Z}$-word x such that $x \models p \,\mathsf{U}_{[l,u]}\, q$ for some $0 \leq l \leq u < \infty$. We would like to split the evaluation of $x \models p \,\mathsf{U}_{[l,u]}\, q$ into the evaluation of other — suitably built — formulas over the two mono-infinite words x^0 and ${}^0x$.

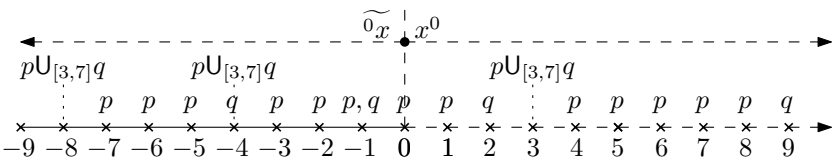

Figure 2: Splitting the evaluation of $p \,\mathsf{U}_{[3,7]}\, q$ about the origin.

Before introducing and proving the formal results, let us provide some intuition about our technique, and let $l = 3, u = 7$. First of all, $x \models p \mathsf{U}_{[3,7]}\, q$ requires in particular that $x^0 \models p \,\mathsf{U}_{[3,7]}\, q$: *until* is a future operator, thus its evaluation

over x^0 is independent of all instant before the origin, hence $x^0 \models p \,\mathsf{U}_{[3,7]}\, q$ iff $\underline{\forall} k \geq 0 : x, k \models p \,\mathsf{U}_{[3,7]}\, q$. For instance this is the case of instant 3 in Figure 2.

Similarly, let us consider any position k of x such that the interval $(k, k+7] \subset (-\infty, 0]$ is contained completely to the left of the origin, such as position -8 in Figure 2. The evaluation of $p \,\mathsf{U}_{[3,7]}\, q$ at k is independent of all instants after the origin, hence $x, k \models p \,\mathsf{U}_{[3,7]}\, q$ iff ${}^0x, k \models p \,\mathsf{U}_{[3,7]}\, q$, for all $k + 7 \leq 0$, i.e., $k \leq -7$.

Finally, let us consider what happens to the evaluation of $p \mathsf{U}_{[3,7]}\, q$ at instants k such that the interval $(k, k+7] \ni 0$ contains the origin; for instance let $k = -4$ and consider again Figure 2. Hence, there exists a $h \in [-1, 3]$ such that $x, h \models q$ and for all $-4 < j < h$ it is $x, j \models p$. Here, we have to distinguish two cases and handle them differently. If $h \leq 0$ such as for $h = -1$ in Figure 2, the evaluation of $p \,\mathsf{U}_{[3,7]}\, q$ at -4 is still independent of instants after the origin, hence $x, k \models p \,\mathsf{U}_{[3,7]}\, q$ iff ${}^0x, k \models p \,\mathsf{U}_{[3,7]}\, q$. Otherwise, if $h > 0$ such as for $h = 2$ in Figure 2, we consider separately the adjacent intervals $(k, 0]$ and $(0, k + 7]$. The fact that p holds throughout $(k, 0]$ is independent of instants after the origin, so $x, k \models \mathsf{G}_{(0,-k]} p$ iff ${}^0x, k \models \mathsf{G} p$. Moreover, $p \,\mathsf{U}_I\, q$ holds at the origin for the "residual" interval $(0, 3]$, thus $x, 0 \models p \,\mathsf{U}_{[1,3]}\, q$ iff $x^0, 0 \models p \,\mathsf{U}_{[1,3]}\, q$.

By generalizing the above informal reasoning, we get the following.

Lemma 4.4. *For any bi-infinite word x, $0 \leq l \leq u < \infty$ such that $u \neq 0$,*[3] *for all $1 - u \leq i \leq -1$:*

$$x, i \models p \,\mathsf{U}_{[l,u]}\, q \quad \Leftrightarrow \quad \begin{array}{c} x, i \models p \,\mathsf{U}_{[l,-i]}\, q \\ \veebar \\ \Big(x, i \models \mathsf{G}_{[1,-i]} p \mathbin{\triangle} x, 0 \models p \,\mathsf{U}_{[\max(1,i+l),i+u]}\, q\Big) \end{array} \tag{4}$$

Proof. Let us start with the $\Rightarrow$ direction: assume $x, i \models p \,\mathsf{U}_{[l,u]}\, q$. Hence, there exists a $d \in [l, u]$ such that $x, i + d \models q$ and for all $i < j < i + d$ it is $x, j \models p$. If $i + d \leq 0$ then $0 \leq d \leq -i$, hence $x, i \models p \,\mathsf{U}_{[l,-i]}\, q$ holds. Otherwise, $i + d > 0$; in this case, p holds throughout $(i, 0]$ and thus $x, i \models \mathsf{G}_{[1,-i]} p$ holds. In addition, let $d' = i + d$; note that $1 \leq d' \leq i + u$ and also $i + l \leq d'$, so $x, 0 \models p \mathsf{U}_{[\max(1,i+l),i+u]}\, q$ holds.

Let us now consider the $\Leftarrow$ direction. If $x, i \models p \mathsf{U}_{[l,-i]}\, q$, from $1 - u \leq i \leq -1$ we get $1 \leq -i \leq u - 1$, thus $[l, -i] \subseteq [l, u]$ which entails $x, i \models p \,\mathsf{U}_{[l,u]}\, q$. Otherwise, let $x, i \models \mathsf{G}_{[1,-i]} p$ and $x, 0 \models p \,\mathsf{U}_{[\max(1,i+l),i+u]}\, q$. That is, p holds throughout $(i, 0]$, and there exists a $k \in [\max(1, i + l), i + u]$ such that $x, k \models q$ and p holds throughout $(0, k)$. Let $d = -i + k$; from $k \in [\max(1, i + l), i + u]$ we get $d \in [l, u]$, which establishes $x, i \models p \,\mathsf{U}_{[l,u]}\, q$. □

Lemma 4.4 showed how to "split" the evaluation of an *until* formula into the evaluation of two derived formulas, one to be evaluated to the left of the origin, and one to its right. Next, we use that result to express the satisfiability of a formula of the form $\neg f \vee p \,\mathsf{U}_{[l,u]}\, q$ over a bi-infinite word x as the satisfiability of

[3] This restriction is clearly without loss of generality, as $\phi_1 \,\mathsf{U}_{[0,0]}\, \phi_2 \equiv \phi_2$.

several different formulas, each evaluated separately either on the whole mono-infinite word x^0 or on the whole mono-infinite word $\widetilde{{}^0x}$. Precisely, we have the following.

Lemma 4.5. *Let x be a bi-infinite word, and $0 \leq l \leq u < \infty$ such that $u \neq 0$; then:*

$$x \models \neg f \vee p \,\mathsf{U}_{[l,u]}\, q \quad \Leftrightarrow \quad \begin{array}{c} x^0 \models \neg f \vee p \,\mathsf{U}_{[l,u]}\, q \;\underline{\wedge}\; \widetilde{{}^0x} \models \neg f \vee p \,\mathsf{S}_{[l,u]}\, q \vee (\mathsf{H}p \wedge \mathsf{H}_{=u}\bot) \\ \underline{\wedge} \\ \forall 1 \leq i \leq u-1 : \left(\begin{array}{c} \widetilde{{}^0x} \models \mathsf{P}_{=i}\top \wedge \mathsf{H}_{=i+1}\bot \Rightarrow \neg f \vee p \,\mathsf{S}_{[l,u]}\, q \\ \underline{\vee} \\ x^0, 0 \models p \,\mathsf{U}_{[\max(1,-i+l),-i+u]}\, q \end{array} \right) \end{array} \tag{5}$$

Proof. Let us start with the $\Rightarrow$ direction where we assume $x \models \neg f \vee p \,\mathsf{U}_{[l,u]}\, q$. For all $k \geq 0$, $x, k \models \neg f \vee p \,\mathsf{U}_{[l,u]}\, q$ implies $x^0, k \models \neg f \vee p \,\mathsf{U}_{[l,u]}\, q$, hence $x^0 \models \neg f \vee p \,\mathsf{U}_{[l,u]}\, q$ is established. Notice also that ${}^0x, 0 \models \mathsf{G}p \wedge \mathsf{G}_{=u}\bot$ holds trivially because the interval $(0, 0]$ is empty, thus also $\widetilde{{}^0x}, 0 \models \mathsf{H}p \wedge \mathsf{H}_{=u}\bot$ (see Remark 2.1).

Then, let $k < 0$ and show ${}^0x, k \models \neg f \vee p \,\mathsf{U}_{[l,u]}\, q \vee (\mathsf{G}p \wedge \mathsf{G}_{=u}\bot)$, hence $\widetilde{{}^0x}, -k \models \neg f \vee p \,\mathsf{S}_{[l,u]}\, q \vee (\mathsf{H}p \wedge \mathsf{H}_{=u}\bot)$. From the current hypothesis, $x, k \models \neg f \vee p \,\mathsf{U}_{[l,q]}\, q$. In particular, if $x, k \models \neg f$ we are done. Otherwise, $x, k \models p \,\mathsf{U}_{[l,u]}\, q$, that is there exists a $d \in [k + l, k + u]$ such that $x, d \models q$ and for all $k < j < d$ it is $x, j \models p$. If $d \leq 0$, the truth of the *until* at k does not depend on any instant beyond 0, hence also ${}^0x, k \models p \,\mathsf{U}_{[l,u]}\, q$. Otherwise, let $d > 0$. In this case, $(k, 0] \subseteq (k, d)$, therefore ${}^0x, k \models \mathsf{G}p$ is established. Moreover, $0 < d \leq k + u$ implies $k > -u$. Hence ${}^0x, k \models \mathsf{G}_{=u}\bot$ as required.

Finally, let $1 \leq i \leq u - 1$ and establish that either $\widetilde{{}^0x} \models \mathsf{P}_{=i}\top \wedge \mathsf{H}_{=i+1}\bot \Rightarrow \neg f \vee p \,\mathsf{S}_{[l,u]}\, q$ or $x^0, 0 \models p \,\mathsf{U}_{[\max(1,-i+l),-i+u]}\, q$. Note that $\mathsf{P}_{=i}\top \wedge \mathsf{H}_{=i+1}\bot$ holds at position h in an ω-word iff $h = i$ (see Remark 2.1). So, if $\widetilde{{}^0x}, i \models \neg f \vee p \,\mathsf{S}_{[l,u]}\, q$ then we are done. Otherwise, assume that $\widetilde{{}^0x}, i \not\models \neg f \vee p \,\mathsf{S}_{[l,u]}\, q$. Since it is $x, -i \models \neg f \vee p \,\mathsf{U}_{[l,u]}\, q$, we infer that, in this case, there exists a $d \in [-i+l, -i+u]$ such that $d > 0$, $x, d \models q$, and p holds throughout $(-i, d)$ over x. Notice that $d > 0$ and $d \in [-i + l, -i + u]$ means that $d \geq \max(1, -i + l)$ and $d \leq -i + u$. Therefore, $x, 0 \models p \,\mathsf{U}_{[\max(1,-i+l),-i+u]}\, q$ and thus $x^0, 0 \models p \,\mathsf{U}_{[\max(1,-i+l),-i+u]}\, q$ *a fortiori*, which concludes this direction of the proof.

Let us now tackle the $\Leftarrow$ direction.

First of all, from $x^0 \models \neg f \vee p \,\mathsf{U}_{[l,u]}\, q$ it follows that we are left with proving $x, k \models \neg f \vee p \,\mathsf{U}_{[l,u]}\, q$ for all $k < 0$.

If $\widetilde{{}^0x}, -k \models \neg f \vee p \,\mathsf{S}_{[l,u]}\, q$, then ${}^0x, k \models \neg f \vee p \,\mathsf{U}_{[l,u]}\, q$ follows from Proposition 2.4, hence $x, k \models \neg f \vee p \,\mathsf{U}_{[l,u]}\, q$ *a fortiori* because the truth of it does not depend on instants beyond 0 in this case.

Otherwise, let $\widetilde{{}^0x}, -k \not\models \neg f \vee p \,\mathsf{S}_{[l,u]}\, q$ and assume $\widetilde{{}^0x}, -k \models \mathsf{H}p \wedge \mathsf{H}_{=u}\bot$, that is p holds over $(k, 0]$ on ${}^0x$ (and x) and $0 < -k < u$ (see Remark 2.1). From the

right-hand side of (5) for $i = -k$, it must be $x^0, 0 \models p \,\mathsf{U}_{[\max(1,k+l),k+u]}\, q$, that is there exists a $d \in [\max(1, k+l), k+u]$ such that $x^0, d \models q$ and p holds over $(0, d)$. Then, p holds over the whole (k, d). Also, from $d \leq k+u$ it follows that $-k+d \leq u$, and from $d \geq k+l$ it follows that $-k+d \geq l$. Hence $-k+d \in [l, u]$ and $x, k \models p \,\mathsf{U}_{[l,u]}\, q$ is established. □

Remark 4.6. Let ϕ_{L} and ϕ_{R} be the left- and right-hand side of Formula (5), respectively. Note that $u = \exp \mathrm{O}(|\phi_{\mathrm{L}}|_{\mathrm{M}})$, due to the succinct encoding of constants assumption. Then, $|\phi_{\mathrm{R}}|_{\mathrm{M}} = \mathrm{O}(|\phi_{\mathrm{L}}|_{\mathrm{M}})$ and $|\phi_{\mathrm{R}}|_{\#} = \mathrm{O}(u \cdot |\phi_{\mathrm{L}}|_{\#}) = |\phi_{\mathrm{L}}|_{\#} \exp \mathrm{O}(|\phi_{\mathrm{L}}|_{\mathrm{M}})$.

4.2.2 From formulas to languages

It is not difficult to show that the equivalence of Formula (5) can be exploited to derive an equivalent formulation of the bi-infinite language $\mathcal{L}^{\mathbb{Z}}\Big(\neg f \vee p \,\mathsf{U}_{[l,u]}\, q\Big)$ in terms of mono-infinite ω-languages and composition operations on them.

Theorem 4.7. *Let $0 \leq l \leq u < \infty$ and $u \neq 0$; then:*

$$\mathcal{L}^{\mathbb{Z}}\Big(\neg f \vee p \,\mathsf{U}_{[l,u]}\, q\Big) = \begin{array}{c} \widetilde{\mathcal{L}^{\omega}}\Big(\neg f \vee p \,\mathsf{S}_{[l,u]}\, q \vee (\mathsf{H}p \wedge \mathsf{H}_{=u}\bot)\Big) \triangleright \mathcal{L}^{\omega}\Big(\neg f \vee p \,\mathsf{U}_{[l,u]}\, q\Big) \\ \cap \\ \bigcap_{i=1}^{u-1} \left(\begin{array}{c} \widetilde{\mathcal{L}^{\omega}}\Big(\mathsf{P}_{=i}\top \wedge \mathsf{H}_{=i+1}\bot \Rightarrow \neg f \vee p \,\mathsf{S}_{[l,u]}\, q\Big) \triangleright \big(2^{\Pi}\big)^{\omega} \\ \cup \\ {}^{\omega}\big(2^{\Pi}\big) \triangleright \mathcal{L}_0^{\omega}\Big(p \,\mathsf{U}_{[\max(1,-i+l),-i+u]}\, q\Big) \end{array} \right) \end{array} \tag{6}$$

Proof. Let us simplify the presentation with the abbreviations: $\phi_{\mathrm{L}} = \neg f \vee p \,\mathsf{U}_{[l,u]}\, q$, $\phi_{\mathrm{R}_1} = \neg f \vee p \,\mathsf{S}_{[l,u]}\, q \vee (\mathsf{H}p \wedge \mathsf{H}_{=u}\bot)$, $\phi^i_{\mathrm{R}_2} = \mathsf{P}_{=i}\top \wedge \mathsf{H}_{=i+1}\bot \Rightarrow \neg f \vee p \,\mathsf{S}_{[l,u]}\, q$, $\phi^i_{\mathrm{R}_3} = p \,\mathsf{U}_{[\max(1,-i+l),-i+u]}\, q$.

Let $x \in \mathcal{L}^{\mathbb{Z}}(\phi_{\mathrm{L}})$ so $x \models \phi_{\mathrm{L}}$. From Lemma 4.5, it is $\widetilde{{}^0x} \models \phi_{\mathrm{R}_1}$ and $x^0 \models \phi_{\mathrm{L}}$, so $\widetilde{{}^0x} \in \mathcal{L}^{\omega}(\phi_{\mathrm{R}_1})$ — or equivalently ${}^0x \in \widetilde{\mathcal{L}^{\omega}}(\phi_{\mathrm{R}_1})$ — and $x^0 \in \mathcal{L}^{\omega}(\phi_{\mathrm{L}})$. Also, note that $w, 0 \models \phi_{\mathrm{R}_1}$ is the case for any ω-word w, because $\mathsf{H}p \wedge \mathsf{H}_{=u}\bot$ holds trivially at 0. Hence, $x \in \widetilde{\mathcal{L}^{\omega}}(\phi_{\mathrm{R}_1}) \triangleright \mathcal{L}^{\omega}(\phi_{\mathrm{L}})$. Let now pick a generic $1 \leq i \leq u-1$. From Lemma 4.5, it is: (1) $\widetilde{{}^0x} \models \phi^i_{\mathrm{R}_2}$; or (2) $x^0, 0 \models \phi^i_{\mathrm{R}_3}$. Note that $w, 0 \models \phi^i_{\mathrm{R}_2}$ is the case for any ω-word w, because the antecedent $\mathsf{P}_{=i}\top \wedge \mathsf{H}_{=i+1}$ holds trivially at 0 for $i \geq 1$. So, if (1) is the case, $\widetilde{{}^0x} \in \mathcal{L}^{\omega}\big(\phi^i_{\mathrm{R}_2}\big)$, that is ${}^0x \in \widetilde{\mathcal{L}^{\omega}}\big(\phi^i_{\mathrm{R}_3}\big)$; hence, $x \in \widetilde{\mathcal{L}^{\omega}}\big(\phi^i_{\mathrm{R}_3}\big) \triangleright \big(2^{\Pi}\big)^{\omega}$. If (2) is the case, $x^0 \in \mathcal{L}_0^{\omega}\big(\phi^i_{\mathrm{R}_3}\big)$; hence, $x \in {}^{\omega}\big(2^{\Pi}\big) \triangleright \mathcal{L}_0^{\omega}\big(\phi^i_{\mathrm{R}_3}\big)$.

For the converse, let us consider any $\mathbb{Z}$-word x such that: ${}^{-1}x \in \widetilde{\mathcal{L}^{\omega}}(\phi_{\mathrm{R}_1})$, $x^0 \in \mathcal{L}^{\omega}(\phi_{\mathrm{L}})$, and for any $1 \leq i \leq u-1$: either ${}^{-1}x \in \widetilde{\mathcal{L}^{\omega}}\big(\phi^i_{\mathrm{R}_2}\big)$ or $x^0 \in \mathcal{L}^{\omega}\big(\phi^i_{\mathrm{R}_3}\big)$. Correspondingly, we have that $\widetilde{{}^0x} \models \phi_{\mathrm{R}_1}$, $x^0 \models \phi_{\mathrm{L}}$, and for any $1 \leq i \leq u-1$: either $\widetilde{{}^0x} \models \phi^i_{\mathrm{R}_2}$, or $x^0, 0 \models \phi^i_{\mathrm{R}_3}$. From Lemma 4.5 we immediately infer that $x \models \phi_{\mathrm{L}}$, hence $x \in \mathcal{L}^{\mathbb{Z}}(\phi_{\mathrm{L}})$. □

4.2.3 Other operators

So far, we have provided a characterization of flat formulas only in the form $\neg f \vee p \,\mathsf{U}_{[l,u]}\, q$, for finite $l \leq u$. In order to handle every possible subformula in the form (1), we have to present similar characterizations for the subformulas:

1. $\neg f \vee p \,\mathsf{U}_{[l,\infty)}\, q$;
2. $f \vee p \,\mathsf{R}_I\, q$, for any interval I;
3. $f \Leftrightarrow p \,\mathsf{S}_I\, q \;\equiv\; (\neg f \vee p \,\mathsf{S}_I\, q) \wedge (f \vee \neg p \,\mathsf{T}_I\, \neg q)$, for any interval I;
4. $\beta \in \mathbb{B}(\Sigma')$.

We devote the remainder of this sub-section to the presentation of such characterizations. We omit most of the proofs, as they can be easily derived from the corresponding ones for the *until* presented above.

***Until* with unbounded interval.** Notice that $\phi_1 \,\mathsf{U}_{[l,\infty)}\, \phi_2$ is equivalent to $\mathsf{G}_{(0,l]}\phi_1 \wedge \mathsf{F}_{=l}\,(\phi_1 \,\mathsf{U}\, \phi_2)$ for all $l \geq 1$, and $\phi_1 \,\mathsf{U}_{[0,\infty)}\, \phi_2$ is equivalent to $\phi_2 \vee \phi_1 \,\mathsf{U}\, \phi_2$. Hence, without loss of generality we just consider the case $p \,\mathsf{U}\, q$; we have the following results.

Lemma 4.8. *For any bi-infinite word x, for all $i \leq -1$:*

$$x, i \models p \,\mathsf{U}\, q \quad \Leftrightarrow \quad \begin{array}{c} x, i \models p \,\mathsf{U}_{[1,-i]}\, q \\ \veebar \\ \Big(x, i \models \mathsf{G}_{[1,-i]} p \;\barwedge\; x, 0 \models p \,\mathsf{U}\, q\Big) \end{array} \tag{7}$$

Proof. Let us start with the $\Rightarrow$ direction: assume $x, i \models p \,\mathsf{U}\, q$. Hence, there exists a $d > 0$ such that $x, i + d \models q$ and for all $i < j < i + d$ it is $x, j \models p$. If $i + d \leq 0$ then $0 \leq d \leq -i$, hence $x, i \models p \,\mathsf{U}_{[1,-i]}\, q$ holds. Otherwise, $i + d > 0$; in this case, p holds throughout $(i, 0]$ and thus $x, i \models \mathsf{G}_{[1,-i]} p$ holds. In addition, let $d' = i + d$; note that $d' \geq 1$, so $x, 0 \models p \,\mathsf{U}\, q$ holds.

Let us now consider the $\Leftarrow$ direction. If $x, i \models p \,\mathsf{U}_{[1,-i]}\, q$, from $i \leq -1$ we get $-i \geq 1$, thus $[1, -i] \subseteq (0, \infty)$ which entails $x, i \models p \,\mathsf{U}\, q$. Otherwise, let $x, i \models \mathsf{G}_{[1,-i]} p$ and $x, 0 \models p \,\mathsf{U}\, q$. That is, p holds throughout $(i, 0]$, and there exists a $k > 0$ such that $x, k \models q$ and p holds throughout $(0, k)$. Let $d = -i + k$; note that $d > k > 0$, so $x, i \models p \,\mathsf{U}\, q$ is established. □

Lemma 4.9. *Let x be a bi-infinite word; then:*

$$x \models \neg f \vee p \,\mathsf{U}\, q \quad \Leftrightarrow \quad \begin{array}{c} x^0 \models \neg f \vee p \,\mathsf{U}\, q \;\barwedge\; \widetilde{{}^0 x} \models \neg f \vee p \,\mathsf{S}\, q \vee \mathsf{H} p \\ \barwedge \\ \Big(\widetilde{{}^0 x} \models \neg f \vee p \,\mathsf{S}\, q \;\veebar\; x^0, 0 \models p \,\mathsf{U}\, q\Big) \end{array} \tag{8}$$

Proof. Let us start with the $\Rightarrow$ direction: assume $x \models \neg f \vee p \,\mathsf{U}\, q$. Clearly, the assumption entails $x^0 \models \neg f \vee p \,\mathsf{U}\, q$ because we are considering a future formula. Then, let $k > 0$ be any positive integer and prove $\widetilde{{}^0x}, k \models \neg f \vee p \,\mathsf{S}\, q \vee \mathsf{H}p$. Since $x, -k \models \neg f \vee p \,\mathsf{U}\, q$, let us assume $x, -k \models p \,\mathsf{U}\, q$: there exists a $d > 0$ such that $x, -k + d \models q$ and p holds over $(-k, -k + d)$. If $-k + d \leq 0$, then ${}^0x, -k + d \models q$, hence $\widetilde{{}^0x}, k \models p \,\mathsf{S}\, q$ by Proposition 2.4. Otherwise, $-k + d > 0$, hence $(-k, -k + d) \supset (-k, 0]$; so ${}^0x, -k \models \mathsf{G}p$ and $\widetilde{{}^0x}, k \models \mathsf{H}p$ by Proposition 2.4 and Lemma 4.9. Finally, let $k > 0$ be the least position such that $x, -k \models \neg f \vee p \,\mathsf{U}\, q$ but $\widetilde{{}^0x}, k \models f \wedge \neg(p \,\mathsf{S}\, q)$. If such k does not exist, we conclude that $\widetilde{{}^0x} \models \neg f \vee p \,\mathsf{S}\, q$. Otherwise, there exists a $d > -k$ such that $x, d \models q$ and for all $-k < j < d$ it is $x, j \models p$. It should be clear that it cannot be $d \leq 0$, otherwise it would also be ${}^0x, -k \models p \,\mathsf{U}\, q$, in contradiction with $\widetilde{{}^0x}, k \models \neg(p \,\mathsf{S}\, q)$ because of Proposition 2.4. Hence $d > 0$, which implies $x^0, 0 \models p \,\mathsf{U}\, q$.

For the $\Leftarrow$ direction, let k be any integer: we show that $x, k \models \neg f \vee p \,\mathsf{U}\, q$. If $k \geq 0$, $x^0 \models \neg f \vee p \,\mathsf{U}\, q$ entails the goal, because we are considering a future formula. So, let $k < 0$. If $\widetilde{{}^0x}, -k \models \neg f \vee p \,\mathsf{S}\, q$, then ${}^0x, k \models \neg f \vee p \,\mathsf{U}\, q$ by Proposition 2.4, and $x, k \models \neg f \vee p \,\mathsf{U}\, q$ holds because of Lemma 4.9. Otherwise, $\widetilde{{}^0x}, -k \models f \wedge \neg(p \,\mathsf{S}\, q) \wedge \mathsf{H}p$. Notice that in this case $\widetilde{{}^0x} \not\models \neg f \vee p \,\mathsf{S}\, q$, thus it must be $x^0, 0 \models p \,\mathsf{U}\, q$, that is there exists a $d > 0$ such that $x^0, x \models q$ and p holds over $(0, d)$. Moreover, p holds over $(k, 0]$ because of $\widetilde{{}^0x}, -k \models \mathsf{H}p$. All in all, p holds throughout (k, d) and hence $x, k \models p \,\mathsf{U}\, q$ is the case from Lemma 4.9. $\square$

Theorem 4.10.

$$\mathcal{L}^{\mathbb{Z}}(\neg f \vee p \,\mathsf{U}\, q) \quad = \quad \begin{array}{c} \widetilde{\mathcal{L}^{\omega}}(\neg f \vee p \,\mathsf{S}\, q \vee \mathsf{H}p) \triangleright \mathcal{L}^{\omega}(\neg f \vee p \,\mathsf{U}\, q) \\ \cap \\ \left(\widetilde{\mathcal{L}^{\omega}}(\neg f \vee p \,\mathsf{S}\, q) \triangleleft \left(2^{\Pi}\right)^{\omega} \;\cup\; {}^{\omega}\!\left(2^{\Pi}\right) \triangleright \mathcal{L}_0^{\omega}(p \,\mathsf{U}\, q)\right) \end{array} \tag{9}$$

Release operator. The characterization of the subformula $f \vee p \,\mathsf{R}_I\, q$ in terms of mono-infinite languages can be derived from the corresponding characterization of the *until* subformula $\neg f \vee p \,\mathsf{U}_I\, q$ by duality. In fact, we have the following straightforward results.

Corollary 4.11. *For any bi-infinite word x, $l \leq u < \infty$ such that $u \neq 0$, for all $1 - u \leq i \leq -1$:*

$$x, i \models p \,\mathsf{R}_{[l,u]}\, q \quad \Leftrightarrow \quad \begin{array}{c} x, i \models p \,\mathsf{R}_{[l,-i]}\, q \\ \wedge \\ \left(x, i \models \mathsf{F}_{[1,-i]} p \vee x, 0 \models p \,\mathsf{R}_{[\max(1,i+l),i+u]}\, q\right) \end{array} \tag{10}$$

$$x \models f \vee p \,\mathsf{R}_{[l,u]}\, q \quad \Leftrightarrow \quad \begin{array}{c} x^0 \models f \vee p \,\mathsf{R}_{[l,u]}\, q \wedge \widetilde{{}^0x} \models f \vee p \,\mathsf{T}_{[l,u]}\, q \\ \wedge \\ \forall 1 \leq i \leq u - 1 : \left(\begin{array}{c} \widetilde{{}^0x} \models \mathsf{P}_{=i} \top \wedge \mathsf{H}_{=i+1} \bot \Rightarrow \mathsf{P}p \\ \vee \\ x^0, 0 \models p \,\mathsf{R}_{[\max(1,-i+l),-i+u]}\, q \end{array} \right) \end{array} \tag{11}$$

$$\mathcal{L}^{\mathbb{Z}}\Big(f \vee p\,\mathsf{R}_{[l,u]}\,q\Big) = \begin{array}{c} \widetilde{\mathcal{L}^{\omega}}\Big(f \vee p\,\mathsf{T}_{[l,u]}\,q\Big) \triangleright \mathcal{L}^{\omega}\Big(f \vee p\,\mathsf{R}_{[l,u]}\,q\Big) \\ \cap \\ \bigcap_{i=1}^{u-1} \left(\begin{array}{c} \widetilde{\mathcal{L}^{\omega}}(\mathsf{P}_{=i}\top \wedge \mathsf{H}_{=i+1}\bot \Rightarrow \mathsf{P}p) \triangleright \Big(2^{\Pi}\Big)^{\omega} \\ \cup \\ {}^{\omega}\Big(2^{\Pi}\Big) \triangleright \mathcal{L}_0^{\omega}\Big(p\,\mathsf{R}_{[\max(1,-i+l),-i+u]}\,q\Big) \end{array} \right) \end{array} \tag{12}$$

Corollary 4.12. *For any bi-infinite word x, for all $i \leq -1$:*

$$x,i \models p\,\mathsf{R}\,q \quad \underline{\Leftrightarrow} \quad \begin{array}{c} x,i \models p\,\mathsf{R}_{[1,-i]}\,q \\ \underline{\wedge} \\ \Big(x,i \models \mathsf{F}_{[1,-i]}p \,\underline{\vee}\, x,0 \models p\,\mathsf{R}\,q\Big) \end{array} \tag{13}$$

$$x \models f \vee p\mathsf{R}\,q \,\underline{\Leftrightarrow}\, \begin{array}{c} x^0 \models f \vee p\,\mathsf{R}\,q \\ \underline{\wedge} \\ \Big({}^{\widetilde{0}}x \models f \vee (p\,\mathsf{T}\,q \wedge \mathsf{P}p) \,\underline{\vee}\, \Big({}^{\widetilde{0}}x \models f \vee p\,\mathsf{T}\,q \;\; \underline{\wedge} \;\; x^0,0 \models p\,\mathsf{R}\,q\Big)\Big) \end{array} \tag{14}$$

$$\mathcal{L}^{\mathbb{Z}}(f \vee p\,\mathsf{R}\,q) \quad = \quad \begin{array}{c} \widetilde{\mathcal{L}^{\omega}}(f \vee (p\,\mathsf{T}\,q \wedge \mathsf{P}p)) \triangleright \mathcal{L}^{\omega}(f \vee p\,\mathsf{R}\,q) \\ \cup \\ \widetilde{\mathcal{L}^{\omega}}(f \vee p\,\mathsf{T}\,q) \triangleright (\mathcal{L}_0^{\omega}(p\,\mathsf{R}\,q) \cap \mathcal{L}^{\omega}(f \vee p\,\mathsf{R}\,q)) \end{array} \tag{15}$$

Past operators. Clearly, equivalences for past operators (namely, *since* and *trigger*) can be derived mechanically from the corresponding equivalences for its future counterpart. Namely, every reference to future is switched to a reference to past, and *vice versa*. For instance, the equivalents of Lemmas 4.4–4.5 and of Theorem 4.7 for the *since* operator are, respectively:

Lemma 4.13. *For any bi-infinite word x, $l \leq u < \infty$ such that $u \neq 0$, for all $1 \leq i \leq u-1$:*

$$x,i \models p\,\mathsf{S}_{[l,u]}\,q \quad \underline{\Leftrightarrow} \quad \begin{array}{c} x,i \models p\,\mathsf{S}_{[l,i]}\,q \\ \underline{\vee} \\ \Big(x,i \models \mathsf{H}_{[1,i]}p \,\underline{\wedge}\, x,0 \models p\,\mathsf{S}_{[\max(1,-i+l),-i+u]}\,q\Big) \end{array} \tag{16}$$

Lemma 4.14. *Let x be a bi-infinite word, and $l \leq u < \infty$ such that $u \neq 0$; then:*

$$x \models \neg f \vee p\,\mathsf{S}_{[l,u]}\,q \quad \underline{\Leftrightarrow} \quad \begin{array}{c} \widetilde{x^0} \models \neg f \vee p\,\mathsf{U}_{[l,u]}\,q \,\underline{\wedge}\, x^0 \models \neg f \vee p\,\mathsf{S}_{[l,u]}\,q \vee (\mathsf{H}p \wedge \mathsf{H}_{=u}\bot) \\ \underline{\forall} 1 \leq i \leq u-1 : \left(\begin{array}{c} x^0 \models \mathsf{P}_{=i}\top \wedge \mathsf{H}_{=i+1}\bot \Rightarrow \neg f \vee p\,\mathsf{S}_{[l,u]}\,q \\ \underline{\vee} \\ {}^{\widetilde{0}}x,0 \models p\,\mathsf{U}_{[\max(1,-i+l),-i+u]}\,q \end{array} \right) \end{array} \tag{17}$$

Theorem 4.15. *Let $l \leq u < \infty$ and $u \neq 0$; then:*

$$\mathcal{L}^{\mathbb{Z}}\Big(\neg f \vee p\,\mathsf{S}_{[l,u]}\,q\Big) = \begin{array}{c} \widetilde{\mathcal{L}^{\omega}}\Big(\neg f \vee p\,\mathsf{U}_{[l,u]}\,q\Big) \triangleleft \mathcal{L}^{\omega}\Big(\neg f \vee p\,\mathsf{S}_{[l,u]}\,q \vee (\mathsf{H}p \wedge \mathsf{H}_{=u}\bot)\Big) \\ \cap \\ \bigcap_{i=1}^{u-1}\left(\begin{array}{c} {}^{\omega}\Big(2^{\Pi}\Big) \triangleleft \mathcal{L}^{\omega}\Big(\mathsf{P}_{=i}\top \wedge \mathsf{H}_{=i+1}\bot \Rightarrow \neg f \vee p\,\mathsf{S}_{[l,u]}\,q\Big) \\ \cup \\ \widetilde{\mathcal{L}_0^{\omega}}\Big(p\,\mathsf{U}_{[\max(1,-i+l),-i+u]}\,q\Big) \triangleleft \Big(2^{\Pi}\Big)^{\omega} \end{array}\right) \end{array} \tag{18}$$

For brevity, we omit the full presentation, and the proofs, of all the other easily derivable equivalences.

Propositional formulas. The case for propositional formulas $\beta \in \mathbb{B}(\Sigma')$ is trivial, as the evaluation of atomic proposition at some position is independent of any other (future or past) position, hence $x \models \beta$ iff $x^0 \models \beta$ and $\widetilde{{}^{0}x} \models \beta$.

Example 4.16. Let us consider the running example of formula $\theta = \mathsf{H}_{[0,3]}p \Rightarrow \mathsf{F}q$. θ in separated normal form becomes $\theta' = (p' \Rightarrow q') \wedge (p' \Leftrightarrow \mathsf{H}_{[0,3]}p) \wedge (q' \Leftrightarrow \mathsf{F}q)$. Then, subformula $\lambda = p' \vee \neg\mathsf{H}_{[0,3]}p = p' \vee \mathsf{P}_{[0,3]}\neg p = p' \vee \top\,\mathsf{S}_{[0,3]}\,\neg p$ can be directly decomposed according to (17):

$$x \models p' \vee \mathsf{P}_{[0,3]}\neg p \quad \underline{\Leftrightarrow} \quad \begin{array}{c} \widetilde{x^0} \models p' \vee \mathsf{F}_{[0,3]}\neg p \triangleq x^0 \models p' \vee \mathsf{P}_{[0,3]}\neg p \vee \mathsf{H}_{=u}\bot \\ \underline{\wedge} \\ \underline{\forall} 1 \leq i \leq 2 : \left(\begin{array}{c} x^0 \models \mathsf{P}_{=i}\top \wedge \mathsf{H}_{=i+1}\bot \Rightarrow p' \vee \mathsf{P}_{[0,3]}\neg p \\ \underline{\vee} \\ \widetilde{{}^{0}x}, 0 \models \mathsf{F}_{[1,-i+3]}\neg p \end{array}\right) \end{array} \tag{19}$$

4.3 From Languages to Automata (to ProMeLa)

In Section 3 we showed how to build an automaton that accepts any given MTL ω-language. On the other hand, in the previous section we showed how to reduce MTL satisfiability over $\mathbb{Z}$-languages to MTL satisfiability over ω-languages composed through the operations of right $\triangleright$ and left $\triangleleft$ join, union $\cup$, intersection $\cap$, and projection $\downarrow^{\Sigma}$. In this section we show that the reduction can be fully implemented, by showing that the automata we consider are closed under the operations of union and intersection, and by showing how to deal with join and projection.

4.3.1 Union and intersection

Notice that BA can be regarded as a special case of AMCA, one where counters and universal transitions are not used at all. Hence, let us just show how to build intersection and union of of AMCA.

Proposition 4.17 (AMCA union)**.** *Consider two AMCA $\mathcal{A}_i = \langle \Sigma, Q_i, \mu_i, q_0^i, \delta_i, F_i \rangle$ for $i = 1, 2$. Let $\mathcal{A}_1 \cup \mathcal{A}_2$ be the AMCA defined as:*

$$\mathcal{A}_1 \cup \mathcal{A}_2 = \langle \Sigma, Q_1 \cup Q_2 \cup \{q_0\}, \max(\mu_1, \mu_2), q_0, \delta, F_1 \cup F_2 \rangle$$

where:

- $q_0 \notin Q_1 \cup Q_2$,
- $\delta = \delta_1 \cup \delta_2 \cup \{(q_0/0, \sigma) \mapsto \delta(q_0^1/0, \sigma) \vee \delta(q_0^2/0, \sigma)\}$
  for all σ *such that* $\delta(q_0^1/0, \sigma)$ *or* $\delta(q_0^2/0, \sigma)$ *is defined.*

Then, $\mathcal{L}^\omega(\mathcal{A}_1 \cup \mathcal{A}_2) = \mathcal{L}^\omega(\mathcal{A}_1) \cup \mathcal{L}^\omega(\mathcal{A}_2)$.

Proof. Omitted for brevity. □

Proposition 4.18 (AMCA intersection)**.** *Consider two AMCA* $\mathcal{A}_i = \langle \Sigma, Q_i, \mu_i, \Lambda_i, q_0^i, \delta_i, F_i \rangle$ *for* $i = 1, 2$*. Let* $\mathcal{A}_1 \cap \mathcal{A}_2$ *be the AMCA defined as:*

$$\mathcal{A}_1 \cap \mathcal{A}_2 = \langle \Sigma, Q_1 \times Q_2 \times \{0,1,2\}, \mu_1 \times \mu_2, \{q_0^1\} \times \{q_0^2\} \times \{0\}, \delta, F_1 \times F_2 \times \{2\} \rangle$$

where $\mu_1 \times \mu_2$ *means that we have a pair of parallel counters* $C_1 = [0..\mu_1]$ *and* $C_2 = [0..\mu_2]$*, and where* $\delta((q_i, q_j, k)/(m_1, m_2), \sigma) = (q_i', q_j', h)/(m_1', m_2')$ *if and only if:*

- $\delta_1(q_i/m_1, \sigma) = q_i'/m_1'$,
- $\delta_2(q_j/m_2, \sigma) = q_j'/m_2'$,
- k *and* h *are correlated ad follows:*
  - *if* $k = 0$ *and* $q_i' \in F_1$ *then* $h = 1$,
  - *if* $k = 1$ *and* $q_j' \in F_2$ *then* $h = 2$,
  - *if* $h = 2$*, then* $h = 0$,
  - $k = h$*, otherwise.*

Then, $\mathcal{L}^\omega(\mathcal{A}_1 \cap \mathcal{A}_2) = \mathcal{L}^\omega(\mathcal{A}_1) \cap \mathcal{L}^\omega(\mathcal{A}_2)$.

Proof. Omitted for brevity. □

Remark 4.19. Let $\mathcal{A}_1$ and $\mathcal{A}_2$ be two automata (either BA or AMCA). Then $|\mathcal{A}_1 \cup \mathcal{A}_2| = \mathrm{O}(|\mathcal{A}_1| + |\mathcal{A}_2|)$ and $|\mathcal{A}_1 \cap \mathcal{A}_2| = \mathrm{O}(|\mathcal{A}_1| \cdot |\mathcal{A}_2|)$.

4.3.2 Join and projection

Let L_1, L_2 be two ω-languages. Let us consider a $\mathbb{Z}$-language L defined as $\widetilde{L_1 \triangleright L_2}$. Then a $\mathbb{Z}$-word x is in L iff $\widetilde{^{-1}x} \in L_1$ and $x^0 \in L_2$. Similarly, for the language L' defined as $\widetilde{L_1 \triangleleft L_2}$, x is in L' iff $\widetilde{^{0}x} \in L_1$ and $x^1 \in L_2$. Hence, if we have two automata $\mathcal{A}_1, \mathcal{A}_2$ such that $\mathcal{L}^\omega(\mathcal{A}_1) = L_1$ and $\mathcal{L}^\omega(\mathcal{A}_2) = L_2$ the emptiness of L and L' can be checked noting that $L = \emptyset$ iff $\mathcal{L}^\omega(\mathcal{A}_1) = \emptyset$ or $\mathcal{L}^\omega(\mathcal{A}_2) = \emptyset$, and the same for $L' = \emptyset$ iff $\mathcal{L}^\omega(\mathcal{A}_1) = \emptyset$ or $\mathcal{L}^\omega(\mathcal{A}_2) = \emptyset$.

For the projection, for any MTL formula ϕ over Π let ϕ' be an equi-satisfiable MTL formula over $\Pi' \supseteq \Pi$. Then, $\downarrow^\Pi \mathcal{L}^{\mathbb{Z}}(\phi') = \mathcal{L}^{\mathbb{Z}}(\phi)$. Hence, $\mathcal{L}^{\mathbb{Z}}(\phi) = \emptyset$ iff $\mathcal{L}^{\mathbb{Z}}(\phi') = \emptyset$. Correspondingly, the technique to check the satisfiability of the formula over the extended alphabet suffices to complete the satisfiability check on the original formula.

4.3.3 Implementing automata

In [BMP+07, BSM+07] we presented TRIO2ProMeLa, a tool that translates TRIO formulas (or, equivalently, MTL formulas) into a ProMeLa representation of the automata presented in Section 3. ProMeLa is the input language to the Spin model-checker [Hol03], hence the tool allows one to check the satisfiability of an MTL formula on top of Spin. This approach is very efficient in practice, since it translates directly AMCA, BA, and compositions thereof (i.e., unions and intersections) to ProMeLa, obtaining a code of the same size as the original automata. When Spin is run on the automata described in ProMeLa, it unfolds them on-the-fly. This unfolding may lead to a blow-up in the dimension of the automata but it is performed by the model-checker only when needed. This approach is convenient, since in many practical cases — when the original formulas are large — the direct translation to BA and then to ProMeLa is simply unfeasible.

In a nutshell, every state of an AMCA is implemented with a ProMeLa process, existential transitions are implemented as nondeterministic choices, and universal transitions as the parallel run of concurrent processes. The tool also introduces some useful optimizations, such as merging processes when possible. We refer the reader to [BMP+07, BSM+07, Spo05] for a detailed description of the translation from AMCA and BA to ProMeLa code.

For our purposes, TRIO2ProMeLa can be reused to provide an implementation of our satisfiability checking procedure over the integers. Once a formula is decomposed as explained in the previous sections, each component is translated into the ProMeLa process that represents the equivalent automaton. All the obtained processes are then suitably composed and coordinated by starting them together at time 0. The results of the various emptiness checks are then combined to have a response about the satisfiability of the original formula.

4.4 Summary and Complexity

Let us briefly summarize the satisfiability checking technique we presented in this section and let us analyze its worst-case asymptotic complexity.

4.4.1 Summary of the satisfiability checking algorithm

Given an MTL formula ϕ over alphabet Σ, the satisfiability over $\mathbb{Z}$-words is checked according to the following steps.

1. From ϕ, build a formula ϕ' in flat separated normal form such that $\mathcal{L}^{\mathbb{Z}}(\phi) =\downarrow^{\Sigma} \mathcal{L}^{\mathbb{Z}}(\phi')$.

2. For each subformula ϕ'_i of ϕ', build a set of formulas $\{\phi'_{i,j}\}_j$, whose combined satisfiability over ω-words is equivalent to the satisfiability of ϕ'_i over $\mathbb{Z}$-words (e.g., according to (6) for the bounded *until*). Let $\phi''_i = \bigcup_j \{\phi'_{i,j}\}$.

3. Translate each subformula $\phi'_{i,j}$ into an automaton $\mathcal{A}_{i,j}$ according to what described in Section 3.

4. For each i, compose the various automata $\mathcal{A}_{i,j}$ according to the structure of the corresponding language equivalence theorems (e.g., according to (6) for the bounded *until*). In practice, for every i we can assume to have two automata $\mathcal{A}_i^+, \mathcal{A}_i^-$ such that $\widetilde{\mathcal{L}^\omega}(\mathcal{A}_i^-) \sim \mathcal{L}^\omega(\mathcal{A}_i^+) = \mathcal{L}^{\mathbb{Z}}(\phi_i')$, where $\sim$ is $\triangleright$ or $\triangleleft$.

5. Let $\mathcal{A}^+, \mathcal{A}^-$ be the automata resulting from the intersection of the various $\mathcal{A}_i^\pm$'s according to the structure of $\mathcal{L}^{\mathbb{Z}}(\phi')$.

6. Since the equivalence $\downarrow^\Sigma \mathcal{L}^{\mathbb{Z}}(\phi') = \mathcal{L}^{\mathbb{Z}}(\phi)$ holds by construction, the emptiness test on $\mathcal{L}^\omega(\mathcal{A}^+)$ and on $\mathcal{L}^\omega(\mathcal{A}^-)$ is equivalent to the satisfiability check of ϕ over $\mathbb{Z}$-words.

Example 4.20. Let us go back to our running example of $\theta = \mathsf{H}_{[0,3]}p \Rightarrow \mathsf{F}q$. In Example 4.16 we showed how to decompose the subformula $\lambda = p' \vee \mathsf{P}_{[0,3]}p$ (see (19)). Correspondingly, we would build the following automata:

- $\mathcal{A}_1$ for $p' \vee \mathsf{P}_{[0,3]}\neg p$;
- $\mathcal{A}_2$ for $p' \vee \mathsf{P}_{[0,3]}\neg p \vee \mathsf{H}_{=u}\bot$;
- $\mathcal{A}_3^j$ for $\mathsf{P}_{=j}\top \wedge \mathsf{H}_{=j+1}\bot \Rightarrow p' \vee \mathsf{P}_{[0,3]}\neg p$, $j = 1, 2$;
- $\mathcal{A}_4^j$ for $\mathsf{F}_{[1,-j+3]}\neg p$, $j = 1, 2$.

The automata would then be composed into:

- $\mathcal{A}_\lambda^- = \mathcal{A}_1$;
- $\mathcal{A}_\lambda^+ = \mathcal{A}_2 \cap \bigcap_{j=1}^2 \left(\mathcal{A}_3^j \cup \mathcal{A}_4^j\right)$.

Overall, we build two such automata $\mathcal{A}_i^-$ and $\mathcal{A}_i^+$ for each of the 5 subformulas θ' can be decomposed into. Let $\mathcal{A}^+ = \bigcap_{i=1}^5 \mathcal{A}_i^+$ and $\mathcal{A}^- = \bigcap_{i=1}^5 \mathcal{A}_i^-$. Finally, we conclude that θ is satisfiable iff $\mathcal{L}^\omega(\mathcal{A}^+)$ is non-empty and $\mathcal{L}^\omega(\mathcal{A}^-)$ is non-empty.

4.4.2 Upper-bound complexity of satisfiability checking over the integers

Let us now evaluate an upper bound on the complexity of the above procedure. The worst-case occurs when overall automata $\mathcal{A}^\pm$ are expanded entirely into nondeterministic BA, thus losing entirely the conciseness of AMCA and the implicit representation of intersections.

First of all, let us estimate the size of every $\mathcal{A}_{i,j}$ with respect to the size of ϕ_i'. In Proposition 3.1 we recalled that the size $|\mathcal{B}|$ of a Büchi automaton $\mathcal{B}$ encoding an LTL formula θ of size $|\theta|$ is $\exp \mathrm{O}(|\theta|)$. Also, every MTL formula η can be translated into an equivalent LTL formula of size $\exp \mathrm{O}(|\eta|_\# \, |\eta|_\mathrm{M})$. In our case, every formula $\phi_{i,j}'$ is translated into an automaton of size:

$$|\mathcal{A}_{i,j}| = \exp\exp \mathrm{O}\left(\left|\phi_{i,j}'\right|_\# \left|\phi_{i,j}'\right|_\mathrm{M}\right) = \exp\exp \mathrm{O}\left(\left|\phi_{i,j}'\right|_\mathrm{M}\right)$$

because every subformula $\phi'_{i,j}$ has a constant (i.e., independent of $|\phi|$) number of connectives. Also, in Remark 4.6 we noted that $\left|\phi'_{i,j}\right|_{\mathrm{M}} = |\phi'_i|_{\mathrm{M}}$, so:

$$|\mathcal{A}_{i,j}| = \exp\exp \mathrm{O}\left(|\phi'_i|_{\mathrm{M}}\right)$$

Next, let us estimate the size of $\mathcal{A}_i^{\pm}$. Roughly, $\mathcal{A}_i^{\pm}$ is the intersection $\bigcap_j \mathcal{A}_{i,j}$, hence its size is upper-bounded by the product of the sizes $|\mathcal{A}_{i,j}|$:

$$\left|\mathcal{A}_i^{\pm}\right| = \prod_j |\mathcal{A}_{i,j}| \leq \left(\max_j |\mathcal{A}_{i,j}|\right)^{|\phi''_i|} = \left(\exp\exp \mathrm{O}(|\phi'_i|_{\mathrm{M}})\right)^{\exp \mathrm{O}(|\phi'_i|_{\mathrm{M}})}$$

where the equivalence between $|\phi''_i|$ and $\exp \mathrm{O}(|\phi'_i|_{\mathrm{M}})$ was highlighted in Remark 4.6. After some manipulation, we get:

$$\left|\mathcal{A}_i^{\pm}\right| = \exp\Big(\left(\exp \mathrm{O}(|\phi'_i|_{\mathrm{M}})\right)\left(\exp \mathrm{O}(|\phi'_i|_{\mathrm{M}})\right)\Big) = \exp\exp \mathrm{O}\left(|\phi'_i|_{\mathrm{M}}\right)$$

Then, the overall size of $\mathcal{A}^+$ and $\mathcal{A}^-$ can be computed as:

$$\left|\mathcal{A}^+\right| + \left|\mathcal{A}^-\right| = \mathrm{O}(\left|\mathcal{A}^{\pm}\right|) = \prod_i \left|\mathcal{A}_i^{\pm}\right| \leq \Big(\max_i |\mathcal{A}_i|\Big)^{\left|\phi'\right|_{\#}} = \exp\big(\left|\phi'\right|_{\#} \exp \mathrm{O}(|\phi'|_{\mathrm{M}})\big)$$

thanks to the equivalence between $|\phi'_i|_{\mathrm{M}}$ and $\mathrm{O}(|\phi'|_{\mathrm{M}})$ stated in Remark 4.6.

Finally, Theorem 4.2 relates the size of ϕ' to that of the original formula ϕ, so we have:

$$\left|\mathcal{A}^+\right| + \left|\mathcal{A}^-\right| = \exp\big(\left|\phi\right|_{\#} \exp \mathrm{O}(|\phi|_{\mathrm{M}})\big)$$

From the well-known result that emptiness check of a Büchi automaton takes time polynomial (actually, linear) in the size of the automaton (see Proposition 3.2), we have established the following.

Theorem 4.21 (Upper-bound complexity)**.** *The verification algorithm of this paper can check the satisfiability of an MTL formula ϕ over $\mathbb{Z}$-words in time doubly-exponential in the size $|\phi|$ of ϕ.*

4.4.3 Complexity of MTL over the integers

Let us now show that the satisfiability problem for MTL over the integers is an EXPSPACE-complete problem, just like it is over the naturals [AH93].

Theorem 4.22 (Complexity of MTL over the integers)**.** *The satisfiability problem for MTL over the integers is* EXPSPACE*-complete.*

Proof. From Proposition 3.2 and the analysis of the previous section, it follows that the satisfiability problem for MTL over the integers is decidable in nondeterministic (singly) exponential space, hence it is in EXPSPACE. (In particular, [VW94] shows how to check emptiness without building the whole (doubly-exponential in size) Büchi automaton.)

For the lower bound, we reduce the satisfiability problem for future-MTL over the naturals to MTL satisfiability over the integers. Let ϕ be a future MTL formula, and let s be a fresh atomic proposition. Let ι be the formula:

$$\iota \quad = \quad \mathrm{Alw}(\mathsf{F}s \vee \mathsf{P}s \vee s) \wedge \mathrm{Alw}(s \Rightarrow \mathsf{G}\neg s \wedge \mathsf{H}\neg s)$$

Basically, ι asserts that s is a "unique event": occurs exactly once over the whole temporal axis. Hence, it can be used to simulate the origin of the mono-infinite case. Let:

$$\phi' \quad = \quad \iota \wedge (s \Rightarrow \phi)$$

Let ϕ' be satisfiable over $\mathbb{Z}$-word x and let $h \in \mathbb{Z}$ be the unique instant where s holds, so $x^h, h \models \phi$ because ϕ is a future formula. Then, let us consider the ω-word w obtained from x^h as $w_k = x^h_{h-k} \setminus \{s\}$. It is clear that $w \models \phi$.

Conversely, let ϕ be satisfiable over the ω-word w. We build a $\mathbb{Z}$-word x as follows: $x^1 = w^1$, $x_0 = w_0 \cup \{s\}$, ${}^{-1}x = {}^{\omega}\emptyset$. It is cleat that $x \models \phi'$.

So, future-MTL satisfiability over the naturals is reducible to MTL satisfiability over the integers. Alur and Henzinger [AH93] showed that the satisfiability problem for future-MTL over the naturals[4] is EXPSPACE-complete, hence the theorem follows. □

5 Discussion

As we discussed in the Introduction, bi-infinite time models for temporal logic have been studied very rarely. Let us briefly consider a few noticeable exceptions.

On the more practical side, Pradella et al. [PMS07] recently developed a tool-supported technique for bounded model-checking of temporal logic specifications over the integers. Bounded model-checking [BHJ+06] is a verification technique based on reduction to the propositional satisfiability (SAT) problem, for which very efficient off-the-shelf tools exist. The technique is however incomplete, as it only looks for words of length up to a given bound k, where k is a parameter of the verification problem instance. [PMS07] describes a direct encoding of MTL bounded satisfiability as a SAT instance and reports on some interesting experimental results with an implementation. [PMS07] also discusses the appeal of bi-infinite time from a system modeling perspective; some of its considerations are also discussed in the Introduction of the present paper.

In the area of automata theory and formal languages, there exist a few works considering bi-infinite time models. For instance Perrin, Pin, et al. [PP04, Chap. 9],[NP82, NP86, GN91] introduce bi-infinite words and automata on them, and extend some classical results for mono-infinite words to these new models. In the same vein, Muller et al. [MSS92] establish the decidability of LTL over the integers. However, to the best of our knowledge the complexity

[4] Actually, [AH93] uses the different semantic model of timed ω-words with natural timestamps, but we discussed in Section 2 how this difference does not impact the problem of satisfiability and in particular its lower-bound complexity.

of temporal logic over bi-infinite time has never been investigated in previous work.

On the contrary, temporal logic over mono-infinite time models has been extensively studied, and it has been the object of an impressive amount of both practical and theoretical research (e.g., [Eme90, CGP00, MP92, GHR94, Var06, AH93, Hen98, AH92, FMMR07]). Satisfiability of both LTL [SC85, DS02] and MTL [AH93] — also with past operators — over mono-infinite discrete time models has been thoroughly investigated. Sistla and Clarke [SC85] proved that LTL satisfiability over the naturals is PSPACE-complete, with a (singly) exponential time algorithm. Correspondingly, Alur and Henzinger [AH93] proved that MTL satisfiability over mono-infinite integer timed words is EXPSPACE-complete, and provided a doubly-exponential time algorithm. In this paper we established that MTL satisfiability remains EXPSPACE-complete over the integers, and we provided an algorithm which matches the worst-case time complexity of MTL satisfiability over mono-infinite time.

We notice, however, that the algorithm we presented in the paper has not only theoretical interest in determining the worst-case complexity of MTL satisfiability over the integers. In fact, it allows for various practical improvements over the naive approach of translating the MTL formulas into LTL, and then directly into Büchi automata. Here it is an informal summary of such improvements:

- MTL future formulas are succinctly encoded as AMCA. With respect to vanilla Büchi automata, alternation can bring an exponential succinctness gain in representing the Boolean structure of the formula, whereas the use of counters can bring an exponential succinctness gain in encoding the constants used in the formula.

- MTL past formulas are conveniently encoded as *deterministic* Büchi automata.

- Then, both AMCA and deterministic Büchi automata can be translated straightforwardly into ProMeLa code of comparable size (i.e., without exponential blow-up in the description). Also, the various operations of intersection and union among automata can be described directly in ProMeLa, without building the intersection or union automata beforehand.

- When the Spin model-checker is run on the generated ProMeLa code it explores the overall automaton on-the-fly. Of course, in the worst case it will end up expanding the whole underlying Büchi automaton, with a doubly-exponential blow-up in size. However, as it is usually the case with on-the-fly algorithms, we expect that the average-case behavior will be much better than the worst case, thus achieving a significant improvement in several cases of interest. Indeed, this guess is supported by our past experience with the mono-infinite tool based on the Spin model checker [BMP+07, BSM+07].

6 Conclusion

We investigated the satisfiability problem for MTL (with both past and future operators) over bi-infinite time models isomorphic to the integer numbers. We provided a technique to reduce such a satisfiability problem to the same problem over mono-infinite time models isomorphic to the natural numbers. We showed how to implement the technique with an automata-theoretic approach which can be implemented on top of the Spin model checker. Also, we investigated the complexity of the integer-time MTL satisfiability problem, and showed that it is EXPSPACE-complete.

In the future, we plan to work on the implementation of an automated translator from integer-time MTL specifications to Spin models, and to experiment with it to assess the practical feasibility of the approach, also in comparison with similar tools for mono-infinite time models. Also, the related MTL model-checking problem over integer time will be investigated.

References

[AH92] Rajeev Alur and Thomas A. Henzinger. Logics and models of real time: A survey. In J. W. de Bakker, Cornelis Huizing, and Willem P. de Roever, editors, *Proc. of the Real-Time: Theory in Practice, REX Workshop*, volume 600 of *Lecture Notes in Computer Science*, pages 74–106. Springer-Verlag, 1992.

[AH93] R. Alur and T. A. Henzinger. Real-time logics: Complexity and expressiveness. *Information and Computation*, 104(1):35–77, 1993.

[BHJ+06] A. Biere, K. Heljanko, T. Junttila, T. Latvala, and V. Schuppan. Linear encodings of bounded LTL model checking. *Logical Methods in Computer Science*, 2(5:5):1–64, 2006.

[BMP+07] Domenico Bianculli, Angelo Morzenti, Matteo Pradella, Pierluigi San Pietro, and Paola Spoletini. Trio2Promela: A model checker for temporal metric specifications. In *ICSE Companion*, pages 61–62, 2007.

[BSM+07] Domenico Bianculli, Paola Spoletini, Angelo Morzenti, Matteo Pradella, and Pierluigi San Pietro. Model checking temporal metric specifications with Trio2Promela. In *Proceedings of the International Symposium on Fundamentals of Software Engineering (FSEN'07)*, Lecture Notes in Computer Science, pages 388–395. Springer-Verlag, 2007.

[CGP00] Edmund M. Clarke, Orna Grumberg, and Doron A. Peled. *Model Checking*. MIT Press, 2000.

[CKS81] Ashok K. Chandra, Dexter Kozen, and Larry J. Stockmeyer. Alternation. *Journal of the ACM*, 28(1):114–133, 1981.

[CPPS98] Alberto Coen-Porisini, Matteo Pradella, and Pierluigi San Pietro. A finite-domain semantics for testing temporal logic specifications. In Anders P. Ravn and Hans Rischel, editors, *Proceedings of the 5th International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems (FTRTFT'98)*, volume 1486 of *Lecture Notes in Computer Science*, pages 41–54. Springer-Verlag, 1998.

[DS02] Stéphane Demri and Philippe Schnoebelen. The complexity of propositional linear temporal logics in simple cases. *Information and Computation*, 174(1):84–103, 2002.

[EL85a] E. Allen Emerson and Chin-Laung Lei. Modalities for model checking: Branching time logic strikes back. In *Proceedings of the 20th ACM Symposium on Principles of Programming Languages (POPL'85)*, pages 84–96. ACM Press, 1985.

[EL85b] E. Allen Emerson and Chin-Laung Lei. Temporal model checking under generalized fairness constraints. In *Proceedings of the 18th Hawaii International Conference on System Sciences*. Western Periodicals Company, 1985.

[Eme90] E. Allen Emerson. Temporal and modal logic. In Jan van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 996–1072. Elsevier Science, 1990.

[FMMR07] C. A. Furia, D. Mandrioli, A. Morzenti, and M. Rossi. Modeling time in computing: A taxonomy and a comparative survey. Technical Report 2007.22, DEI, Politecnico di Milano, 2007.

[Fri05] Carsten Fritz. Concepts of automata construction from LTL. In Geoff Sutcliffe and Andrei Voronkov, editors, *Proceedings of the 12th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR'05)*, volume 3835 of *Lecture Notes in Computer Science*, pages 728–742. Springer-Verlag, 2005.

[GHR94] Dov M. Gabbay, Ian Hodkinson, and Mark Reynolds. *Temporal Logic (vol. 1): mathematical foundations and computational aspects*, volume 28 of *Oxford Logic Guides*. Oxford University Press, 1994.

[GMM90] Carlo Ghezzi, Dino Mandrioli, and Angelo Morzenti. TRIO: A logic language for executable specifications of real-time systems. *The Journal of Systems and Software*, 12(2):107–123, 1990.

[GN91] Françoise Gire and Maurice Nivat. Langages algébriques de mots biinfinis. *Theoretical Computer Science*, 86(2):277–323, 1991. In French.

[GO03] Paul Gastin and Denis Oddoux. LTL with past and two-way very-weak alternating automata. In Branislav Rovan and Peter Vojtás,

editors, *Proceedings of the 28th International Symposium on Mathematical Foundations of Computer Science (MFCS'03)*, volume 2747 of *Lecture Notes in Computer Science*, pages 439–448. Springer-Verlag, 2003.

[Hen98] T. A. Henzinger. It's about time: Real-time logics reviewed. In *Proc. of CONCUR'98*, volume 1466 of *LNCS*, pages 439–454, 1998.

[Hol03] Gerard J. Holzmann. *The SPIN Model Checker: Primer and Reference Manual*. Addison-Wesley, 2003.

[Koy90] Ron Koymans. Specifying real-time properties with metric temporal logic. *Real-Time Systems*, 2(4):255–299, 1990.

[Koy92] Ron Koymans. (real) time: A philosophical perspective. In J. W. de Bakker, Cornelis Huizing, Willem P. de Roever, and Grzegorz Rozenberg, editors, *Proceedings of the REX Workshop: "Real-Time: Theory in Practice"*, volume 600 of *Lecture Notes in Computer Science*, pages 353–370. Springer-Verlag, 1992.

[LMS02] François Laroussinie, Nicolas Markey, and Ph. Schnoebelen. Temporal logic with forgettable past. In *Proceedings of LICS'02*, pages 383–392. IEEE Computer Society, 2002.

[LPZ85] Orna Lichtenstein, Amir Pnueli, and Lenore D. Zuck. The glory of the past. In Rohit Parikh, editor, *Logics of Programs*, volume 193 of *Lecture Notes in Computer Science*, pages 196–218. Springer-Verlag, 1985.

[MMG92] Angelo Morzenti, Dino Mandrioli, and Carlo Ghezzi. A model parametric real-time logic. *ACM Transactions on Programming Languages and Systems*, 14(4):521–573, 1992.

[MP92] Zohar Manna and Amir Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification*. Springer-Verlag, 1992.

[MPSS03] Angelo Morzenti, Matteo Pradella, Pierluigi San Pietro, and Paola Spoletini. Model-checking TRIO specifications in SPIN. In Keijro Araki, Stefania Gnesi, and Dino Mandrioli, editors, *FME 2003: Formal Methods, International Symposium of Formal Methods Europe*, volume 2805 of *Lecture Notes in Computer Science*, pages 542–561, Pisa, Italy, September 2003. Springer-Verlag.

[MSS92] David E. Muller, Paul E. Schupp, and Ahmed Saoudi. On the decidability of the linear Z-temporal logic and the monadic second order theory. In Waldemar W. Koczkodaj, Peter E. Lauer, and Anestis A. Toptsis, editors, *Proceedings of the 4th International Conference on Computing and Information (ICCI'92)*, pages 2–5. IEEE Computer Society, 1992.

[NP82] Maurice Nivat and Dominique Perrin. Ensembles reconnaissables de mots biinfinis. In *Proceedings of the 14th Annual ACM Symposium on Theory of Computing (STOC'82)*, pages 47–59. ACM Press, 1982. In French, abstract in English.

[NP86] Maurice Nivat and Dominique Perrin. Ensembles reconnaissables de mots biinfinis. *Canadian Journal of Mathematics*, 38:513–537, 1986. In French.

[PMS07] Matteo Pradella, Angelo Morzenti, and Pierluigi San Pietro. The symmetry of the past and of the future: Bi-infinite time in the verification of temporal properties. In Ivica Crnkovic and Antonia Bertolino, editors, *Proceedings of the 6th joint meeting of the European Software Engineering Conference and the ACM SIGSOFT International Symposium on Foundations of Software Engineering (ESEC/FSE'07)*, pages 312–320, 2007.

[Pnu77] Amir Pnueli. The temporal logic of programs. In *Proceedings of the 18th IEEE Symposium on Fundations of Computer Science (FOCS'77)*, pages 46–67, 1977.

[PP04] Dominique Perrin and Jean-Éric Pin. *Infinite Words*, volume 141 of *Pure and Applied Mathematics*. Elsevier, 2004.

[PSSM03] Matteo Pradella, Pierluigi San Pietro, Paola Spoletini, and Angelo Morzenti. Practical model checking of LTL with past. In Farn Wang and Insup Lee, editors, *Proceedings of 1st International Workshop on Automated Technology for Verification and Analysis (ATVA'03)*, pages 135–146, Taipei, Taiwan, R.O.C., December 2003.

[SC85] A. Prasad Sistla and Edmund M. Clarke. The complexity of propositional linear temporal logics. *Journal of the ACM*, 32(3):733–749, 1985.

[Spo05] Paola Spoletini. *Verification of Temporal Logic Specification via Model Checking*. PhD thesis, Dipartimento di Elettronica e Informazione, Politecnico di Milano, May 2005.

[Tho90] Wolfgang Thomas. Automata on infinite objects. In Jan van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 133–164. Elsevier Science, 1990.

[Var06] Moshe Y. Vardi. *Handbook of Modal Logic*, volume 3 of *Studies in Logic and Practical Reasoning*, chapter Automata-Theoretic Techniques for Temporal Reasoning, pages 971–990. Elsevier, 2006.

[VW94] Moshe Y. Vardi and Pierre Wolper. Reasoning about infinite computations. *Information and Computation*, 115(1):1–37, 1994.