Reasoning About Periodicity on Infinite Words

Wanwei Liu[1(✉)], Fu Song[2], and Ge Zhou[1]

[1] College of Computer Science, National University of Defense Technology, Changsha, China
wwliu@nudt.edu.cn

[2] School of Information Science and Technology, ShanghaiTech University, Pudong, China

Abstract. Characterization of temporal properties is the original purpose of inventing of temporal logics. In this paper, we show that the property like "some event holds periodically" is not omega-regular. Such property is called "periodicity", which plays an important role in task scheduling and system design. To give a characterization of periodicity, we present the logic QPLTL, which is an extension of LTL via adding quantified step variables. Based on the decomposition theorem, we show that the SATISFIABILITY problem of QPLTL is **PSPACE**-complete.

1 Introduction

In some sense, concurrency and interleaving are the main course of complexity in distributed and parallel programs. For such a program, we are in general concerned about its interactive behaviors, rather than the final output. Hence, various temporal logics are designed to characterize such issues of parallel programs. Characterization of temporal properties is always one of the central topics in the research on temporal logics.

Linear-time temporal logic (LTL), is one of the most frequently used logics, which is obtained from propositional logic via merely adding two temporal connectives — X (next) and U (until). Simple as such logic is, LTL could express a majority of properties which are usually concerned about — for example, *responsibility* — such as the formula $\mathsf{G}(req \to \mathsf{F}\ ack)$, which depicts the assertion that "each *req*uest will be eventually *ack*nowledged".

In [Wol83], Wolper pointed out some properties, like "p holds at least in every even moment" (we refer to it as $\mathsf{G}^2 p$ in this paper), cannot be expressed by any LTL formula. As a consequence, numerous extensions are presented to enhance the expressive power. To mention a few: In [VW94], ω-automata are employed as extended temporal connectives. In [BB87], Banieqbal and Barringer yield the linear-time version of modal μ-calculus. In [LS07], Leucker and Sánchez propose to use regular-expressions as prefixes of formulae. All of these logics are shown to be as expressive as the full ω-regular languages, or the whole set of (nondeterministic) Büchi automata [Büc62]. As a result, properties like $\mathsf{G}^2 p$ can be described by these logics. In the view of formal languages, LTL formulae precisely correspond to *star-free* ω-regular languages [Tho79], in which

K.G. Larsen et al. (Eds.): SETTA 2017, LNCS 10606, pp. 200–215, 2017.
https://doi.org/10.1007/978-3-319-69483-2_12

the Kleene-star ($*$) and the omega-power (ω) operators can only be applied to Σ (the alphabet). Meanwhile, in the automata-theoretic perspective, LTL formulae are inter-convertable with Büchi automata of special forms, e.g., *counter-free automata* or *aperiodic automata* (cf. [DG08] for a comprehensive survey).

We say that an event f happens *periodically*, if there exists some number $n > 0$ (called the *period*, which is not pre-given) such that f holds at least in every moment which is a multiple of n. Such kind of property is called *periodicity*, which plays an important role in task scheduling and system design. For example, when designing a synchronous circuit, we need to guarantee the clock interrupt generates infinitely and equidistantly. An aerospace control system has a diagnosis module which periodically checks whether the system works properly. If it was not, then the system immediately enters the recovery state.

Enlightened by the fact that "$\mathsf{G}^2 p$ cannot be expressed by any LTL formula", we conjecture that "the periodicity property $\exists k.\mathsf{G}^k p$ is not expressible by any ω-regular language". We give an affirmative proof of this conjecture in this paper, and hence deduce that ω-regular properties are not closed under infinite unions and/or intersections. To express periodicity within linear-time framework, we tentatively suggest to add quantified *step variables* into logics. For example, if we use μTL [BB87] as the base logic, the aforementioned property could be described by the formula $\exists k.\nu Z.(p \wedge \mathsf{X}^k Z)$. However, such mechanism gives rise to an expressive power beyond expectations — we show that, if we allow nesting of step variables, all formulae of Peano arithmetic can be encoded — provided that the base logic involves the X-operator and two distinct propositions. Hence, it leads to undecidability for the SATISFIABILITY problem of the logic.

Thus, to obtain a decidable extension, we have to impose strong syntactic restriction to the logic. In this paper, using LTL as the base logic, we introduce the logic *Quantified Periodic LTL* (QPLTL, for short). In such logic, formulae are categorized into four groups, the first three are contained within ω-regular languages, and the last group is specially tailored for defining periodicity: Each of such formulae uses at most one step variable, and the occurrence of this variable must be of special form. We show that, for QPLTL, the SATISFIABILITY problem is also **PSPACE**-complete. Our proof approach is mainly based on the *decomposition theorem* [AS85], namely, each property could be decomposed into an intersection of a liveness property and a safety property, with which, we give a normal form of star-free liveness properties. Instead of deciding the satisfiability of the formulae, we give a decision procedure to decide their validity.

The rest part of this paper is organized as follows: Sect. 2 introduces some basic notations and definitions. In Sect. 3, we show that periodicity properties are not ω-regular. Thus, we suggest to add quantified step variables into linear-time logics to gain such an enhancement. However, we show that an unrestricted use of such extension will result in undecidability of SATISFIABILITY, provided that the base logic involves the X-operator and at least two propositions. We then present the logic QPLTL in Sect. 4 via imposing strict syntactic constraints to the use of step variables, by revealing a normal form of star-free liveness properties, we show that the SATISFIABILITY problem of the proposed logic is **PSPACE**-complete. We summarize this paper and discuss future work in Sect. 5.

2 Preliminaries

2.1 Finite and Infinite Words

Fix an alphabet Σ, whose elements are called *letters*. We call w an *ω-word* (or *infinite*-word) if $w \in \Sigma^\omega$, and we call w a *finite-word* if $w \in \Sigma^*$.

We use $|w|$ to denote the *length* of w, and definitely $|w| = \infty$ if w is an ω-word. For each $i < |w|$, let $w(i)$ be the ith letter of w. Remind the first letter should be $w(0)$.

Given $w_1 \in \Sigma^*$ and $w_2 \in \Sigma^* \cup \Sigma^\omega$, we denote by $w_1 \cdot w_2$ the *concatenation* of w_1 and w_2. Namely, we have: $(w_1 \cdot w_2)(i) = w_1(i)$ whenever $i < |w_1|$; and $(w_1 \cdot w_2)(i) = w_2(i - |w_1|)$ whenever $i \geq |w_1|$. In this case, we say that w_1 and w_2 are respectively a *prefix* and a *postfix* (or *suffix*) of $w_1 \cdot w_2$.

In the rest of this paper, we fix a (potentially infinite) set $\mathcal{P}$ of *propositions*, the elements range over p, p_1, p_2, etc., and when mentioning about "words", without explicit declaration, we let $\Sigma = 2^{\mathcal{P}}$.

2.2 Linear-Time Temporal Logic

Syntax. Formulae of linear-time temporal logic (LTL, [Pnu77]), ranging over $f, g, f_1, f_2, \ldots$, can be described by the following abstract grammar:

$$f ::= \bot \mid p \mid f \rightarrow f \mid \mathsf{X}f \mid f \,\mathsf{U}\, f.$$

We usually use the following derived operators as syntactic sugars:

$$\begin{array}{rcl} \neg f \overset{\text{def}}{=} f \rightarrow \bot & \top \overset{\text{def}}{=} \neg\bot & f_1 \vee f_2 \overset{\text{def}}{=} \neg f_1 \rightarrow f_2 \\ f_1 \wedge f_2 \overset{\text{def}}{=} \neg(\neg f_1 \vee \neg f_2) & \mathsf{F}f \overset{\text{def}}{=} \top \,\mathsf{U}\, f & f_1 \leftrightarrow f_2 \overset{\text{def}}{=} (f_1 \rightarrow f_2) \wedge (f_2 \rightarrow f_1) \\ f_1 \mathsf{R} f_2 \overset{\text{def}}{=} \neg(\neg f_1 \,\mathsf{U}\, \neg f_2) & \mathsf{G}f \overset{\text{def}}{=} \neg\mathsf{F}\neg f & f_1 \,\mathsf{W}\, f_2 \overset{\text{def}}{=} (f_1 \,\mathsf{U}\, f_2) \vee \mathsf{G}f_2 \end{array}$$

Semantics. The *satisfaction* relation ($\models$) can be defined w.r.t. an ω-word $w \in (2^{\mathcal{P}})^\omega$ and a position $i \in \mathbb{N}$. Inductively:

- $w, i \not\models \bot$ for each w and i.
- $w, i \models p$ iff $p \in w(i)$.
- $w, i \models f_1 \rightarrow f_2$ iff either $w, i \not\models f_1$ or $w, i \models f_2$.
- $w, i \models \mathsf{X}f$ iff $w, i+1 \models f$.
- $w, i \models f_1 \,\mathsf{U}\, f_2$ iff $w, j \models f_2$ for some $j \geq i$ and $w, k \models f_1$ for each $k \in [i, j)$.

We may abbreviate $w, 0 \models f$ as $w \models f$. The *language* of f, denoted by $\mathscr{L}(f)$, consists of all ω-words initially satisfying f, i.e., $\mathscr{L}(f) = \{w \in (2^{\mathcal{P}})^\omega \mid w \models f\}$.

X, U and W are respectively the "next", "until" and "weak until" operators. According to the definition, $\mathsf{G}f$ holds if f holds at each position, and $\mathsf{F}f$ is true if f eventually holds at some position. In addition, $f_1 \,\mathsf{W}\, f_2$ holds if at every position, either f_1 holds, or at some previous position f_2 holds.

2.3 Automata on Finite/Infinite-Words

An *automaton* is a tuple $A = (Q, \Sigma, \delta, Q_0, F)$ where: Q is a finite set of *states*; Σ is a finite *alphabet*; $\delta : Q \times \Sigma \to 2^Q$, is the *transition function*; $Q_0 \subseteq Q$ is a set of *initial states*; and $F \subseteq Q$ is a set of *accepting states.*

The automaton A is *deterministic*, if $\#Q_0 = 1$, and $\#\delta(q, a) = 1$ for each $q \in Q$ and $a \in \Sigma$.

In this paper, we are both concerned about automata on finite and infinite words. For an automaton A on infinite (resp. finite) words, a *run* of A over a word $w \in \Sigma^\omega$ (resp. $w \in \Sigma^*$) is a sequence $q_0, q_1, \ldots \in Q^\omega$ (resp. $q_0, q_1, \ldots, q_{|w|} \in Q^*$), where $q_0 \in Q_0$ and $q_{i+1} \in \delta(q_i, w(i))$.

If A is an automaton on finite words, the run $q_0, q_1, \ldots, q_m$ is *accepting* if $q_m \in F$. Otherwise, if A is on infinite word, the run $q_0, q_1, \ldots$ is accepting if there are infinitely many i's having $q_i \in F$ — such kind of acceptance is called *Büchi* [Büc62] acceptance condition. For this reason, in this paper, automata on infinite words are also called Büchi automata.

For convenience, we in what follows use a three-letter-acronym to designate the type of an automaton: The first letter can be either "N" (non-deterministic) or "D" (deterministic). The second letter can either be "B" or "F", referring to Büchi and finite acceptance condition, respectively. The last letter is always "A", the acronym of automaton. For example, NBA stands for nondeterministic Büchi automaton, and DFA means deterministic automaton on finite words.

A word w is *accepted* by A, if A has an accepting run on it. We denote by $\mathscr{L}(A)$ the set of words accepted by A, call it the *language* of A. In the case of $L = \mathscr{L}(A)$, we say that L is *recognized* by A.

It is well known that NFAs and DFAs are of equivalent expressive power, because each NFA can be transformed into a DFA via the power-set construction. However, it is not the case for Büchi automata. NBAs are strictly more expressive than DBAs. As an example, let $\Sigma = \{a, b\}$, then the language consisting of "ω-words involving finitely many a's" can only be recognized by NBAs. For Büchi automata, determinization requires more complex acceptance conditions, such as *Rabin* or *parity* (cf. [Rab69, McN66]).

Languages recognized by NBAs are called *ω-regular* ones. Meanwhile, transformation from LTL formulae to Büchi automata has been well studied ever since the logic was presented (cf. [TH02, ST03] etc.).

Theorem 1. *Given an LTL formula f, there is an NBA A_f such that $\mathscr{L}(A_f) = \mathscr{L}(f)$.*

Theorem 2. *Given an automaton $A = (Q, \Sigma, \delta, Q_0, F)$, then there exists an automaton $A' = (Q', \Sigma, \delta', Q'_0, F')$ such that $\mathscr{L}(A') = \mathscr{L}(A)$ and $\#Q'_0 = 1$.*

Proof. Let q be some new state not belonging to Q, then: let $Q' = Q \cup \{q\}$; let the function δ' be the extension of δ by defining $\delta'(q, a) = \bigcup_{q_0 \in Q_0} \delta(q_0, a)$ for each $a \in \Sigma$; let $Q'_0 = \{q\}$; and, if A is an NFA/DFA and $Q_0 \cap F \neq \emptyset$ then we let $F' = F \cup \{q\}$, otherwise, let $F' = F$. $\square$

2.4 Safety and Liveness

Generally speaking, each set $L \subseteq (2^{\mathcal{P}})^{\omega}$ defines a *property*, and a formula f *corresponds* to L if $\mathscr{L}(f) = L$. Remind that in this definition, f is not necessary to be an LTL formula. Actually, it could be a formula of more (or less) powerful logic within linear-time framework.

There are two kinds of fundamental properties relating to temporal logics, called *safety* and *liveness*. Informally, safety and liveness are described as "the bad thing never happens" and "the good thing eventually happens", respectively. Below we give a formal characterization, which is introduced in [AS85].

Safety: A formula f corresponds to a safety property if: For every ω-word w, $w \not\models f$ implies there is some finite prefix (called "*bad-prefix*") w' of w, such that $w' \cdot w'' \not\models f$ for each ω-word w''.

Liveness: A formula f corresponds to a liveness property if: For every finite-word w, there is some ω-word w' having $w \cdot w' \models f$.

Theorem 3 ([AS85, CS01]). *For safety and liveness properties, we have:*

1. *Safety properties are closed under finite unions and arbitrary intersections.*
2. *Liveness properties are closed under arbitrary unions, but not under intersections.*
3. $\top$ *is the only property which is both a safety and a liveness property.*
4. *For any property f, there exists a liveness property g and a safety property h such that $f = g \wedge h$.*

The last proposition is the so-called *decomposition theorem*. As an example, the LTL formula $f \,\mathsf{U}\, g$ can be decomposed as $(f \mathsf{W} g) \wedge \mathsf{F}g$, where $f \,\mathsf{W}\, g$ corresponds to a safety property, whereas $\mathsf{F}g$ corresponds to a liveness property.

Lemma 1. *If $f = \bigwedge_i f_i$ corresponds to a liveness property, then so does each f_i.*

Proof. Otherwise, according to the decomposition theorem, f_i can be written as the conjunction of some g_i and h_i, which respectively correspond to a liveness property and a safety property. If $h_i \not\leftrightarrow \top$, there is some ω-word w violating h_i. Just let $w' \in (2^{\mathcal{P}})^*$ be the bad-prefix of w w.r.t. h_i, we thus have $w' \cdot w'' \not\models f$ for every $w'' \in (2^{\mathcal{P}})^{\omega}$, contradiction! $\square$

3 Periodicity and Step Variables

3.1 Periodicity: Beyond Omega-Regular

In [Wol83], Wolper pointed out that: The property $\mathsf{G}^2 p$, namely "p holds at every even moment", is not expressible in LTL. Indeed, $\mathscr{L}(\mathsf{G}^2 p) = \bigcap_{k \in \mathbb{N}} \mathscr{L}(\mathsf{X}^{2k} p)$, where X^n is the shorthand of n successive X-operators. This implies that languages captured by LTL formulae are not closed under infinite intersections.

This naturally enlightens us to make one step ahead — we would like to know: "Are ω-regular languages closed under infinite intersections/unions?" Now, let us consider the language $\bigcup_{k>0} \mathscr{L}(\mathsf{G}^k p)$, which consists of all ω-words along which p holds periodically. Remind that $w \models \mathsf{G}^k p$ if $w, i \times k \models p$ for each $i \in \mathbb{N}$.

Theorem 4. *The language $\bigcup_{k>0} \mathscr{L}(\mathsf{G}^k p)$ is not ω-regular.*

Proof. Assume by contradiction that this language is ω-regular, then there is an NBA A precisely recognizing it. Therefore, each ω-word being of the form $(\{p\} \cdot \emptyset^k)^\omega$ is accepted by A.

Suppose that A has n states, and let us fix some $k > n$. Suppose that the corresponding accepting run of A on $w_0 = (\{p\} \cdot \emptyset^k)^\omega$ is $\sigma_0 = s_0, s_1, s_2, \ldots$, and we denote s_i by $\sigma_0(i)$. Since we have totally n states, for each $t \in \mathbb{N}$, there exists a pair (i_t, j_t), s.t. $0 < i_t < j_t \leq k$ and $\sigma_0(t \times (k+1) + i_t) = \sigma_0(t \times (k+1) + j_t)$.

Since $\sigma_0(i_0) = \sigma_0(j_0)$ in the case of $t = 0$, for any $\ell_0 > 0$, from Pumping lemma, A has the run

$$\sigma_1 = \sigma_0(0), \sigma_0(1), \ldots, \sigma_0(i_0), [\sigma_0(i_0+1), \ldots, \sigma_0(j_0)]^{\ell_0}, \sigma_0(j_0+1), \sigma_0(j_0+2) \ldots$$

on the word

$$w_1 = (\{p\} \cdot \emptyset^{i_0 - 1} \cdot \emptyset^{(j_0 - i_0) \times \ell_0} \cdot (\emptyset)^{k - j_0 + 1}) \cdot (\{p\} \cdot \emptyset^k)^\omega.$$

σ_1 is definitely an accepting run because states occurring infinitely often in σ_0 are the same as that in σ_1. Let $L_0 = k + (j_0 - i_0) \times (\ell_0 - 1)$, then we have $w_1 = (\{p\} \cdot \emptyset^{L_0}) \cdot (\{p\} \cdot \emptyset^k)^\omega$. The above process is depicted by Fig. 1.

Also note that $\sigma_0(\ell) = \sigma_1(\ell + L_0 - k)$ for each $\ell > k$, and hence $\sigma_0(k+1+i_1) = \sigma_0(k+1+j_1)$ implies $\sigma_1(L_0+1+i_1) = \sigma_1(L_0+1+j_1)$. Then, for any $\ell_1 > 0$, using Pumping lemma again, A also has an accepting run on the word

$$w_2 = (\{p\} \cdot \emptyset^{L_0}) \cdot (\{p\} \cdot \emptyset^{i_1 - 1} \cdot \emptyset^{(j_1 - i_1) \times \ell_1} \cdot (\emptyset)^{k - j_1 + 1}) \cdot (\{p\} \cdot \emptyset^k)^\omega.$$

Now, let $L_1 = k + (j_1 - i_1) \times (\ell_1 - 1)$, then $w_2 = (\{p\} \cdot \emptyset^{L_0}) \cdot (\{p\} \cdot \emptyset^{L_1}) \cdot (\{p\} \cdot \emptyset^k)^\omega$.

Likewise and stepwise, we may obtain a sequence of ω-words accepted by A:

- $w_0 = (\{p\} \cdot \emptyset^k)^\omega$.
- $w_1 = (\{p\} \cdot \emptyset^{L_0}) \cdot (\{p\} \cdot \emptyset^k)^\omega$.
- $w_2 = (\{p\} \cdot \emptyset^{L_0}) \cdot (\{p\} \cdot \emptyset^{L_1}) \cdot (\{p\} \cdot \emptyset^k)^\omega$.
- $\cdots$

Since that w_{i+1} is constructed based on w_i, we can always choose a proper ℓ_{i+1} to guarantee that $L_{i+1} > L_i$ for each i. Then, consider the limit

$$w_\infty = (\{p\} \cdot \emptyset^{L_0}) \cdot (\{p\} \cdot \emptyset^{L_1}) \cdot (\{p\} \cdot \emptyset^{L_2}) \cdot \ldots$$

of all such w_is: On one hand, we have $w_\infty \in \mathscr{L}(A)$ because the corresponding run σ_∞ is accepting — observe that each occurrence of an accepting state in σ_0 will be "postponed" to at most finitely many steps in σ_∞. On the other hand, the "distance" between two adjacent "occurrences" of p monotonically increases, and it would be eventually larger than any fixed number — this implies that p cannot have a period w.r.t. w_∞. We thus get a contradiction because A accepts some word not belonging to $\bigcup_{k>0} \mathscr{L}(\mathsf{G}^k p)$. $\square$

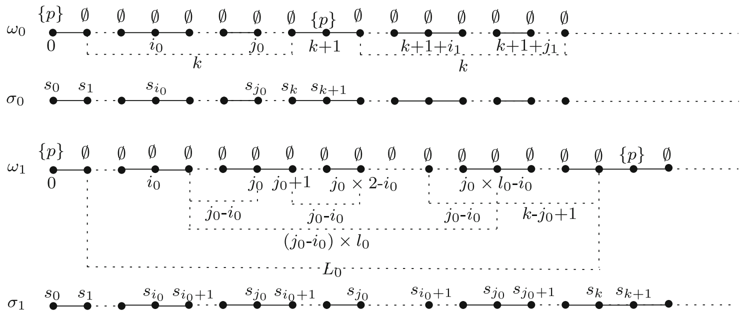

Fig. 1. The construction of w_1 from w_0.

Note that for each fixed number $k > 0$, the language $\mathscr{L}(\mathsf{G}^k p)$ is ω-regular, and it is not the case for $\bigcup_{k>0} \mathscr{L}(\mathsf{G}^k p)$, and we thus have the following corollary.

Corollary 1. *ω-regular languages are not closed under infinite unions and/or intersections.*

Remark 1. We say that ω-regular languages are not closed under infinite intersections because such languages are closed under complement. $\square$

3.2 Logics with Step Variables

To express periodicity, we tentatively propose to add (quantified) *step variables* into logics. As an example, if we choose μTL as the base logic, then the aforementioned property can be described as[1] $\exists k.\nu Z.(p \wedge \mathsf{X}^k\mathsf{X}Z)$ and here k is a step variable. As another example, one can see that $\forall k.\mathsf{X}^k\mathsf{X}^k p$ is precisely the description of $\mathsf{G}^2 p$.

From now on, we fix a set $\mathcal{K}$ of step variables, whose elements range over k, k_1, k_2, etc., and each step variable is interpreted as a natural number. In addition, we require that the base logic must involve the X-operator to designate "distance between events". Therefore, at first thought, each such extension involves the following logic (called *core logic*)

$$f ::= \bot \mid p \mid f \to f \mid \mathsf{X}f \mid \mathsf{X}^k f \mid \exists k.f$$

as fragment[2]. For convenience, we define $\forall k.f$ as the shorthand of $\neg\exists k.\neg f$ (recall that $\neg g$ stands for $g \to \bot$).

Since we have step variables, when giving semantics of a formula, besides an ω-word and a position, an evaluation $v : \mathcal{K} \to \mathbb{N}$ is also required. For the core logic, we define the semantics as follows.

[1] Note that we here have an extra X in the formula, because k can be assigned to 0.

[2] Remind that step variables cannot be instantiated as concrete numbers in such logic, hence we have both $\mathsf{X}f$ and $\mathsf{X}^k f$ in the grammar.

- $w, i \not\models_v \perp$ for every $w \in (2^{\mathcal{P}})^\omega$, $i \in \mathbb{N}$.
- $w, i \models_v p$ iff $p \in w(i)$.
- $w, i \models_v f_1 \to f_2$ iff either $w, i \not\models_v f_1$ or $w, i \models_v f_2$.
- $w, i \models_v \mathsf{X} f$ iff $w, i+1 \models_v f$.
- $w, i \models_v \mathsf{X}^k f$ iff $w, i + v(k) \models_v f$.
- $w, i \models_v \exists k.f$ iff there is some $n \in \mathbb{N}$ such that $w, i \models_{v[k \leftarrow n]} f$.

Here, $v[k \leftarrow n]$ is also an evaluation which is almost identical to v, except that it assigns n to k. To simplify notations, when f is a closed formula[3], we often omit v from the subscript; meanwhile, we can also omit i whenever $i = 0$.

Though such kind of extensions seems to be natural and succinct, however, we will show that the SATISFIABILITY problem, even if for the core logic, is not decidable. But before that, we first introduce the following notations:

- We abbreviate $\underbrace{\mathsf{X} \ldots \mathsf{X}}_{n \text{ times}} f$ and $\underbrace{\mathsf{X}^k \ldots \mathsf{X}^k}_{n \text{ times}} f$ as $\mathsf{X}^n f$ and $\mathsf{X}^{n \times k} f$, respectively, where $n \in \mathbb{N}$ and $k \in \mathcal{K}$.
- We sometimes directly write $\mathsf{X}^{t_1} \mathsf{X}^{t_2} f$ as $\mathsf{X}^{t_1 + t_2} f$.

Note that in this setting, both the addition $(+)$ and the multiplication $(\times)$ are communicative and associative. Meanwhile, "$\times$" is distributive w.r.t. "$+$", namely, $t_1 \times t_2 + t_1 \times t_3$ can be rewritten as $t_1 \times (t_2 + t_3)$.

Theorem 5. *The* SATISFIABILITY *problem of the core logic is undecidable.*

Proof. Our goal is to show that "each formula of Peano arithmeticcan be encoded with the core logic". To this end, we need to build the following predicates:

1. Fix a proposition $p \in \mathcal{P}$, let f_p be $\forall k_1.\forall k_2.\exists k_3.(\mathsf{X}^{k_1+k_3} p \not\leftrightarrow \mathsf{X}^{k_1+k_2+k_3+1} p)$. Actually, f_p just depicts the *non-shifting* property of p. i.e., $w \models f_p$ only if for each $i, j \in \mathbb{N}$ with $i < j$, there is some t having: either $(w, i+t \models p$ and $w, j+t \not\models p)$ or $(w, i+t \not\models p$ and $w, j+t \models p)$ — indeed, one can just view k_1 as i and view $k_1 + k_2 + 1$ as j. Note that f_p is satisfied by any ω-word along which p occurs infinitely often and the distance between two adjacent occurrences of p monotonically increases.
2. Let $P_=(k_1, k_2) \overset{\text{def}}{=} f_p \wedge \forall k.(\mathsf{X}^{k_1+k} p \leftrightarrow \mathsf{X}^{k_2+k} p)$. Hence, $w, i \models_v P_=(k_1, k_2)$ iff $v(k_1) = v(k_2)$. Because, if $v(k_1) \neq v(k_2)$, according to the definition of f_p, there must exist some $n \in \mathbb{N}$, such that p differs from $w(v(k_1) + n)$ and $w(v(k_2) + n)$.
3. Let $P_<(k_1, k_2) \overset{\text{def}}{=} \exists k.P_=(k_1 + k + 1, k_2)$. Then, $w, i \models_v P_<(k_1, k_2)$ iff $v(k_1) < v(k_2)$.
4. Subsequently, we use $P_+(k_1, k_2, k_3)$ to denote $P_=(k_1 + k_2, k_3)$. According to the definition, $w, i \models_v P_+(k_1, k_2, k_3)$ iff $v(k_3) = v(k_1 + k_2) = v(k_1) + v(k_2)$.

[3] That is, f involves no free variable.

5. Now, let us fix another proposition $q \in \mathcal{P}$ and define

$$\begin{aligned} f_q \overset{\text{def}}{=} & \; q \;\wedge\; \mathsf{X}q \;\wedge\; \forall k_1.\exists k_2.\mathsf{X}^{k_1+k_2}q \;\wedge \\ & \forall k_1.\forall k_2.\forall k_3.(\mathsf{X}^{k_1}q \wedge \mathsf{X}^{k_2}q \wedge \mathsf{X}^{k_3}q \wedge \\ & P_<(k_1,k_2) \wedge P_<(k_2,k_3) \wedge \\ & \forall k_4.((P_<(k_1,k_4) \wedge P_<(k_4,k_2)) \vee (P_<(k_2,k_4) \wedge P_<(k_4,k_3)) \to \neg \mathsf{X}^{k_4}q) \\ & \to \exists k_5.\exists k_6.(P_+(k_5,k_1,k_2) \wedge P_+(k_6,k_2,k_3) \wedge P_+(2,k_5,k_6))) \end{aligned}$$

We may assert that $w, i \models f_q$ iff i is a complete square number (i.e., $i = j^2$ for some j). Let us explain: The first line indicates that q holds infinitely often, and it holds at the positions of 0 and 1. For every three adjacent positions k_1, k_2, k_3 at which q holds (hence, q does not hold between k_1 and k_2, nor between k_2 and k_3), we have $(k_3 - k_2) = 2 + (k_2 - k_1)$. Inductively, we can show that q becomes true precisely at positions 0, 1, 4, ..., $(n-1)^2$, n^2, $(n+1)^2$, ...[4].

6. We let $P_2(k_1, k_2)$ be

$$f_q \wedge \mathsf{X}^{k_2}q \wedge \mathsf{X}^{k_2+2\times k_1+1}q \wedge \neg\exists k_3.(P_<(k_2,k_3) \wedge P_<(k_3, 2\times k_1 + k_2 + 1) \wedge \mathsf{X}^{k_3}q)$$

then $w \models_v P_2(k_1, k_2)$ iff $v(k_2) = (v(k_1))^2$.

7. Lastly, we define

$$\begin{aligned} P_\times(k_1,k_2,k_3) \overset{\text{def}}{=} & \; \exists k_4.\exists k_5.\exists k_6.(P_2(k_1,k_4) \wedge P_2(k_2,k_5) \\ & \wedge P_2(k_1+k_2,k_6) \wedge P_=(2\times k_3 + k_4 + k_5, k_6)) \end{aligned}$$

Then, once $w \models_v P_\times(k_1,k_2,k_3)$ holds, we can infer that there is some evaluation v' which agrees with v on k_1, k_2 and k_3 (i.e., $v'(k_i) = v(k_i)$ for $i = 1, 2, 3$) having

$$\begin{cases} v'(k_4) = (v'(k_1))^2 \\ v'(k_5) = (v'(k_2))^2 \\ v'(k_6) = (v'(k_1) + v'(k_2))^2 \\ v'(k_6) = v'(k_4) + v'(k_5) + 2 \times v'(k_3) \end{cases}$$

and we thus subsequently have $v(k_3) = v(k_1) \times v(k_2)$.

Therefore, "addition", "multiplication", and the "less than" relation over natural numbers can be encoded in terms of the core logic. Since quantifiers are also involved here, the SATISFIABILITY problem of Peano arithmetic, which is known to be undecidable (cf. [Göd31, Chu36]), is now reduced to that of the core logic. □

4 The Logic QPLTL

Theorem 5 indicates that the unrestricted use of step variables leads to undecidability for the SATISFIABILITY problem. To obtain a decidable extension, we need to impose strong syntactic constraints to the logic. In this paper, we investigate the logic *Quantified Periodic LTL* (QPLTL for short), based on LTL.

[4] The encoding of f_q is enlightened by [Sch10].

4.1 Syntax and Semantics

A QPLTL formula can be one of the following forms:

(I) An LTL formula.
(II) A formula like $f_1 \mathsf{U}^n f_2$ or $f_1 \mathsf{W}^n f_2$, where f_1 and f_2 are LTL formulae, n is a positive natural number.
(III) Boolean or temporal combinations of (I) and (II).
(IV) A formula being of the form $\exists\!\!\!/\, k.(f_1 \mathsf{U}^k f_2)$ or $\exists\!\!\!/\, k.(f_1 \mathsf{W}^k f_2)$, where $\exists\!\!\!/ \in \{\exists, \forall\}$, $k \in \mathcal{K}$, f_1 and f_2 are two LTL formulae.

Given an ω-word $w \in (2^{\mathcal{P}})^\omega$, and a position $i \in \mathbb{N}$, we define the *satisfaction* relation as follows.

– Satisfaction of an LTL formula is defined in the way same as before.
– $w, i \models f_1 \mathsf{U}^n f_2$ iff there is some $t \in \mathbb{N}$ such that $w, i + t \times n \models f_2$ and $w, i + j \times n \models f_1$ for every $j < t$.
– $w, i \models f_1 \mathsf{W}^n f_2$ iff either $w, i \models f_1 \mathsf{U}^n f_2$ or $w, i + t \times n \models f_1$ for each $t \in \mathbb{N}$.
– Boolean and temporal combinations are defined accordingly with the corresponding operators.
– $w, i \models \exists k.(f_1 \mathsf{U}^k f_2)$ (resp. $w, i \models \exists k.(f_1 \mathsf{W}^k f_2)$) iff there is some $n > 0$ such that $w, i \models f_1 \mathsf{U}^n f_2$ (resp. $w, i \models f_1 \mathsf{W}^n f_2$).
– $w, i \models \forall k.(f_1 \mathsf{U}^k f_2)$ (resp. $w, i \models \forall k.(f_1 \mathsf{W}^k f_2)$) iff for each $n > 0$ we have $w, i \models f_1 \mathsf{U}^n f_2$ (resp. $w, i \models f_1 \mathsf{W}^n f_2$).

As usual, we directly write $w, 0 \models f$ as $w \models f$, and we also define the following derived notations[5].

$$\mathsf{F}^n f \overset{\text{def}}{=} \top \mathsf{U}^n f \qquad\qquad \mathsf{G}^n f \overset{\text{def}}{=} f \mathsf{W}^n \bot$$
$$\exists\!\!\!/\, k.\mathsf{F}^k f \overset{\text{def}}{=} \exists\!\!\!/\, k.\top \mathsf{U}^k f \qquad\qquad \exists\!\!\!/\, k.\mathsf{G}^k f \overset{\text{def}}{=} \exists\!\!\!/\, k.f \mathsf{W}^n \bot$$

Indeed, from the proof of Theorem 5, one can see that nested use of quantifiers leads to undecidability of satisfaction decision. Thus, in QPLTL, we use at most one step variable in a formula.

Remark 2. $f_1 \mathsf{U}^1 f_2$ and $f_1 \mathsf{W}^1 f_2$ can be just written as $f_1 \mathsf{U} f_2$ and $f_1 \mathsf{W} f_2$, which coincide with the definitions given in LTL.

Also remind that a step variable must be interpreted as a positive number — beware that this is different from that in Sect. 3.2. This is just for the following consideration: If we allow assigning 0 to a step variable, the formula $\exists k.\mathsf{G}^k p$ is no longer the description of "p holds periodically", because this formula is weaker than $\mathsf{G}^0 p$ (which is equivalent to p). Note that in such setting, the "core logic" remains undecidable — the proof can be obtained via doing a simple adaptation from that of Theorem 5.

[5] We do not consider the "releases" (R) operator here, which is the duality of U. Indeed, $f_1 \mathsf{R} f_2$ is equivalent to $f_2 \mathsf{W}(f_1 \wedge f_2)$.

To make things compatible, for the formulae like $f_1 \, \mathsf{U}^n \, f_2$ or $f_1 \, \mathsf{W}^n \, f_2$, we don't allow $n = 0$. Actually, according to the definition, we can see that $f_1 \, \mathsf{U}^0 f_2$ and $f_1 \, \mathsf{W}^0 \, f_2$ are just merely f_2 and $f_1 \vee f_2$, and such operators are redundant.

Note that X^n is still the abbreviation of n successive Xs. We don't explicitly add the formulae like $\exists k.\mathsf{X}^k f$ and $\forall k.\mathsf{X}^k f$ into the logic, because they are essentially $\mathsf{XF}f$ and $\mathsf{XG}f$, respectively. □

4.2 The Decision Problem

In this section, we will show that the SATISFIABILITY problem of QPLTL is decidable. Indeed, this proof also reveals the close connection among liveness, safety and periodicity.

We can equivalently transform each LTL formula f to make it involve only literals (i.e., $\top$, $\bot$, formulae like p or $\neg p$), $\vee$, $\wedge$, X, U and W. We in what follows call it the *positive normal form* (PNF, for short) of f. An LTL formula is U-*free* if its PNF involves no U-operator.

Lemma 2 ([CS01]). *Each* U*-free LTL formula corresponds to a safety property.*

Below we give a characterization of LTL formulae corresponding to liveness properties. In some sense, it could be considered as a normal form of such kind of star-free properties.

Theorem 6. *If the LTL formula f corresponds to a liveness property, then it can be equivalently written as $\bigwedge_i (f'_i \vee \mathsf{F} f''_i)$, where each f'_i corresponds to a safety property.*

Proof. Suppose that f is already in its PNF, then we conduct a series of transformations on f.

First of all, we use the pattern $f_1 \, \mathsf{U} \, f_2 \leftrightarrow (f_1 \, \mathsf{W} \, f_2) \wedge \mathsf{F} f_2$ to replace each occurrence of U operator with that of W and F.

Then, use the following rules

$$\begin{array}{cc} \mathsf{X}(f_1 \wedge f_2) \leftrightarrow \mathsf{X}f_1 \wedge \mathsf{X}f_2 & \mathsf{X}(f_1 \vee f_2) \leftrightarrow \mathsf{X}f_1 \vee \mathsf{X}f_2 \\ \mathsf{X}(f_1 \, \mathsf{W} \, f_2) \leftrightarrow (\mathsf{X}f_1)\mathsf{W}(\mathsf{X}f_2) & \mathsf{XF}f' \leftrightarrow \mathsf{FX}f' \end{array}$$

to push X inward, until X occurrs only before a literal or another X.

Subsequently, repeatedly use the following schemas[6]

$$\begin{array}{rl} (f_1 \wedge f_2)\mathsf{W} \, f_3 & \leftrightarrow (f_1 \, \mathsf{W} \, f_3) \wedge (f_2 \, \mathsf{W} \, f_3) \\ f_1 \, \mathsf{W}(f_2 \vee f_3) & \leftrightarrow (f_1 \, \mathsf{W} \, f_2) \vee (f_1 \, \mathsf{W} \, f_3) \\ (f_1 \vee \mathsf{F}f_2)\mathsf{W} \, f_3 & \leftrightarrow \mathsf{F}(f_2 \wedge \mathsf{X}(f_1 \, \mathsf{W} \, f_3)) \vee (f_1 \, \mathsf{W} \, f_3) \vee \mathsf{F}(f_3 \wedge \mathsf{F}f_2) \vee \mathsf{FGF}f_2 \\ f_1 \, \mathsf{W}(f_2 \wedge \mathsf{F}f_3) & \leftrightarrow (f_1 \, \mathsf{W} \, f_2) \wedge (\mathsf{F}(f_2 \wedge \mathsf{F}f_3) \vee \mathsf{G}f_1) \\ (\mathsf{F}f_1)\mathsf{W} \, f_2 & \leftrightarrow f_2 \vee \mathsf{F}(f_1 \wedge \mathsf{X}f_2) \vee \mathsf{F}(f_2 \wedge \mathsf{F}f_1) \vee \mathsf{FGF}f_1 \\ f_1 \, \mathsf{W}(\mathsf{F}f_2) & \leftrightarrow \mathsf{G}f_1 \vee \mathsf{F}f_2 \end{array}$$

[6] For example, when dealing with $(p_1 \vee p_2 \wedge \mathsf{F}p_3)\mathsf{W} \, p_4$, we need first transform the first operand into disjunctive normal form — i.e., rewrite it as $((p_1 \vee \mathsf{F}p_3) \wedge (p_2 \vee \mathsf{F}p_3))\mathsf{W} \, p_4$, and then conduct the transformation with the first rule and the third rule. Similarly, for the second operand, we need first transform it into conjunctive normal form.

until the following holds: For each subformula $f' = f_1 \mathsf{W} f_2$, if f_1 or f_2 involves F, then f' must be contained in the scope of some other F. Remind that we equivalently write $\mathsf{GF}f_i$ as $\mathsf{FGF}f_i$ in the second rule and the fifth rule to fulfill such requirement[7]. Note that when the third or the fifth schema is applied, one need further push Xs inward using the previous group of rules.

Lastly, we write the resulting formula into the conjunctive normal form $\bigwedge_i f_i$ where each $f_i = \bigvee_j f_{i,j}$. We now show that it must be of the desired form.

- First of all, since f corresponds to a liveness property, so does each f_i (cf. Lemma 1).
- Second, for each $f_i = \bigvee_j f_{i,j}$, if there is no such $f_{i,j}$ whose outermost operator is F, then f_i is U-free[8]. From Lemma 2, f_i also corresponds to a safety property. Hence, such f_i is essentially equivalent to $\top$ (cf. Theorem 3), and we can simply remove this conjunct.
- Then, for the other f_is: By applying the scheme $\mathsf{F}g_1 \vee \ldots \vee \mathsf{F}g_n \leftrightarrow \mathsf{F}(g_1 \vee \ldots \vee g_n)$, we can just preserve one disjunct having F as the outermost operator, denote it by $\mathsf{F}f''_i$. Since other disjuncts are U-free, the disjunction of them, denoted by f'_i, corresponds to a safety property (cf. Theorem 3).

Then, the above discussion concludes the proof. □

Lemma 3 **([AS87]).** *Given an LTL formula f, the question "whether f corresponds to a liveness property" is decidable. In addition, given a safety LTL formula, then there is an automaton on finite words recognizing all its bad-prefixes.*

Theorem 7. *Given two LTL formulae f_1 and f_2, where f_2 corresponds to a safety property, then the question "whether exists some ω-word w such that $w \models \mathsf{G}f_1$ and $w \models \exists k.\mathsf{G}^k \neg f_2$" is decidable.*

Proof. Let $A_1 = (Q_1, 2^{\mathcal{P}}, \delta_1, \{q_1\}, F_1)$ be the NBA recognizing $\mathscr{L}(\mathsf{G}f_1)$, and $A_2 = (Q_2, 2^{\mathcal{P}}, \delta_2, \{q_2\}, F_2)$ be the NFA[9] recognizing the bad-prefixes of f_2.

For each $q \in Q_1$, we can define a *product* $A_1^q \otimes A_2 \overset{\text{def}}{=} (Q_1 \times Q_2, 2^{\mathcal{P}}, \delta', \{(q, q_2)\})$, where

$$\delta'((q'_1, q'_2), a) = \begin{cases} \{(q''_1, q''_2) \mid q''_1 \in \delta_1(q'_1, a), q''_2 \in \delta_2(q'_2, a)\} & q'_2 \notin F \\ \{(q''_1, q''_2) \mid q''_1 \in \delta_1(q'_1, a), q''_2 \in Q_2\} & q'_2 \in F \end{cases}.$$

The product is almost an automaton, but we are not concerned about its run over words, hence the accepting state set is not given here. Instead, we will define the notion of *accepting loops*: An accepting loop is a finite sequence $(q_{1,0}, q_{2,0}), (q_{1,1}, q_{2,1}), \ldots, (q_{1,m}, q_{2,m}) \in (Q_1 \times Q_2)^*$ such that:

[7] Note that the operator G is derived from W.

[8] Because, the previous transformations could guarantee that: If the outermost operator of $f_{i,j}$ is not F, then no F occurs in $f_{i,j}$.

[9] Remind that each automaton can be equivalently transformed into another one having a unique initial state, and the transformation is linear (cf. Theorem 2). Indeed, to obtain a finite alphabet, here we may temporarily take $\mathcal{P}$ as the set constituted with propositions occurring in f_1 or f_2.

1. $q_{1,0} = q_{1,m}$ and $q_{2,0} = q_{2,m}$.
2. For each $0 \leq i < m$, there is some a_i having $(q_{1,i+1}, q_{2,i+1}) \in \delta'((q_{1,i}, q_{2,i}), a_i)$.
3. There exists some $0 \leq i < m$ such that $q_{1,i} \in F_1$.
4. There also exists some $0 \leq j < m$ such that $q_{2,j} \in F_2$.

We will then show the following claim:

> *There is some ω-word $w \in (2^{\mathcal{P}})^\omega$ making $w \models \mathsf{G} f_1$ and $w \models \exists k.\mathsf{G}^k \neg f_2$ iff there is some $q \in Q_1$ such that $A_1^q \otimes A_2$ involves an accepting loop starting from (q, q_2).*

$\Longrightarrow$: Suppose that $w \models \mathsf{G} f_1$ and $w \models \exists k.\mathsf{G}^k \neg f_2$ with the period n, namely $w \models \mathsf{G}^n \neg f_2$ — which implies $w, i \times n \not\models f_2$ for every $i \in \mathbb{N}$.

Let $\sigma = \sigma(0), \sigma(1), \sigma(2), \ldots \in Q_1^\omega$ be an accepting run of A_1 on w. Then, there exists some $q_f \in F_1$ such that there are infinitely many i's having $\sigma(i) = q_f$. Because the run is infinite, there must exist some $q \in Q_1$ fulfilling: there are infinitely many i's having $\sigma(i \times n) = q$. W.l.o.g., suppose that $\sigma(i_0 \times n) = q$.

Because $w, i_0 \times n \not\models f_2$, there exists some bad-prefix w' of f_2, which starts from $w(i_0 \times n)$. Let i_1 be the number fulfilling that: $\sigma(i_1 \times n) = q$, and $(i_1 - i_0) \times n > |w'|$, and there exists some $\ell \in [i_0 \times n, i_1 \times n)$ having $\sigma(\ell) = q_f \in F_1$.

Since w' is a bad-prefix of f_2, there is a finite accepting run $\sigma' = \sigma'(0), \sigma'(1), \ldots, \sigma'(|w'|)$ of A_2 on w', where $\sigma'(0) = q_2$ and $\sigma'(|w'|) \in F_2$.

Let $t = (i_1 - i_0) \times n$, since $t > |w|'$, according to the construction, we can prolong σ' by defining $\sigma'(j) = \sigma'(|w'|)$ for each $j \in (|w'|, t)$ and $\sigma'(t) = q_2$.

For each $j \in [0, t]$, we let $\sigma''(j) = \sigma(i_0 \times n + j)$. Thus, we get the accepting loop $(\sigma''(0), \sigma'(0)), \ldots, (\sigma''(t), \sigma'(t))$ in the product. Indeed, according to the construction, we can see that both $(\sigma''(0), \sigma'(0))$ and $(\sigma''(t), \sigma'(t))$ are (q, q_2), and the loop is accepting.

$\Longleftarrow$: Conversely, suppose $(q_{1,0}, q_{2,0}), \ldots, (q_{1,n}, q_{2,n})$ to be an accepting loop of $A_1^q \otimes A_2$ where $q_{1,0} = q_{1,n} = q$ and $q_{2,0} = q_{2,n} = q_2$. We can, of course, assume that q is reachable from q_1 in A_1 — if not so, such state can be safely removed.

Suppose that $(q_{1,i+1}, q_{2,i+1}) \in \delta'((q_{1,i}, q_{2,i}), a_i)$, we let $w = (a_0 \cdot a_1 \cdot \ldots \cdot a_{n-1})^\omega$. Also let $m \in [0, n)$ be the minimal index having $q_{2,m} \in F_2$, then $a_0 \cdot a_1 \cdot \ldots \cdot a_{m-1}$ is a bad-prefix of f_2, hence w violates f_2 with the period n, and thus $w \models \exists k.\mathsf{G}^k \neg f_2$.

What left is to ensure that $w \models \mathsf{G} f_1$ also holds. Suppose that q_1 reaches q via reading the finite word w_0, then $w_0 \cdot w \in \mathscr{L}(A_1)$, and thus $w_0 \cdot w \models \mathsf{G} f_1$. Consequently, we have $w \models \mathsf{G} f_1$. □

Theorem 8. *Given an LTL formula f, the question "whether $\exists k.\mathsf{G}^k \neg f$ is satisfiable" is decidable.*

Proof. First of all, we may decompose f as $g \wedge h$ where g corresponds to a liveness property and h corresponds to a safety property.

If $h \not\leftrightarrow \top$, then there is a finite-word w_0 acting as the bad-prefix of h, and hence a bad-prefix of f. Therefore, the ω-word $w_0^\omega = w_0 \cdot w_0 \cdot w_0 \cdot \ldots$ violates f periodically, and thus $\exists k.\mathsf{G}^k \neg f$ must be satisfiable.

In what follows, we just consider the case that $h \leftrightarrow \top$, and hence f corresponds to a liveness property. From Theorem 6, let us further assume the normal form of f is $\bigwedge_{i=1}^{m} f_i$, where $f_i = f'_i \vee \mathsf{F} f''_i$, and f'_i corresponds to a safety property.

Thus, f is periodically violated, if and only if: there is some ω-word w, a set of indices $J \subseteq \{1, 2, \dots, m\}$, and a positive number n, such that for each $i \in \mathbb{N}$ we have $w, i \times n \not\models f_j$ for some $j \in J$.

Then, the obligation is to detect the existence of such word w, index set J and period n. First, we may choose the set J, and the number of such choices are finite. We can further assume that $w \models \mathsf{G}\neg f''_j$ for each $j \in J$, because:

- If there is some $j \in J$ having $w \models \mathsf{GF} f''_j$, then the disjunct f_j is never violated by w. In this case, we may choose some $J' \subseteq J \setminus \{j\}$ to be the index set.
- If there are finitely many i's having $w, i \models f''_j$, then we may choose some postfix of w to be the word.

Now, given J, the problem becomes: Find some ω-word w, which fulfills $w \models \bigwedge_{j \in J} \mathsf{G}\neg f''_j$ and $w \models \exists k.\mathsf{G}^k \neg \bigwedge_{j \in J} f'_j$. Since that $\bigwedge_{j \in J} \mathsf{G}\neg f''_j$ is equivalent to $\mathsf{G} \bigwedge_{j \in J} \neg f''_j$, and $\bigwedge_{j \in J} f'_j$ corresponds to a safety property, from Theorem 7, we know that this problem is decidable. □

Theorem 9. *The* SATISFIABILITY *problem of QPLTL is decidable.*

Proof. Since formulae of Type (I)–(III) can be expressed by some logics equal to ω-regular languages, such as μTL or ETL, we here just consider formulae of Type (IV).

First, we lift the negation operator ($\neg$) to that group of formulae by defining

$$\neg \exists\!\!\!\forall k.(f_1 \,\mathsf{U}^k f_2) \overset{\text{def}}{=} \overline{\exists\!\!\!\forall} k.(\neg f_2 \,\mathsf{W}^k (\neg f_1 \wedge \neg f_2))$$
$$\neg \exists\!\!\!\forall k.(f_1 \,\mathsf{W}^k f_2) \overset{\text{def}}{=} \overline{\exists\!\!\!\forall} k.(\neg f_2 \,\mathsf{U}^k (\neg f_1 \wedge \neg f_2))$$

where $\overline{\exists\!\!\!\forall}$ is $\forall$(resp. $\exists$) if $\exists\!\!\!\forall$ is $\exists$(resp. $\forall$). Indeed, for each such formula f, we can examine that $w \models f$ iff $w \not\models \neg f$ for every ω-word w. Hence, such lifting is admissible. This also implies that formulae of Type (IV) are closed under negation.

Then, for such a formula, instead of deciding its *satisfiability*, we would rather decide its *validity*, because f is satisfiable iff $\neg f$ is not valid. Below gives the decision approach.

(1) $\exists k.(f_1 \,\mathsf{W}^k f_2)$ is valid iff $f_1 \vee f_2$ is valid.
(2) $\forall k.(f_1 \,\mathsf{W}^k f_2)$ is valid iff $f_1 \vee f_2$ is valid.
(3) $\exists k.(f_1 \,\mathsf{U}^k f_2)$ is valid iff $f_2 \vee (f_1 \wedge \mathsf{F} f_2)$ is valid.
(4) $\forall k.(f_1 \,\mathsf{U}^k f_2)$ is valid iff $f_1 \vee f_2$ is valid and $\exists k.\mathsf{G}^k \neg f_2$ is not satisfiable.

For (1) and (2): If $\exists\!\!\!\forall k.(f_1 \,\mathsf{W}^k f_2)$ is valid, then $w \not\models f_2$ implies $w \models f_1$ for each ω-word w, hence $f_1 \vee f_2$ must be valid. Conversely, if $f_1 \vee f_2$ is valid, then $w \models f_1 \,\mathsf{W}^n f_2$ holds for any positive natural number n.

For (3): The "only if" direction is also trivial, we just show the "if" direction. Indeed, if $f_2 \vee (f_1 \wedge \mathsf{F} f_2)$ is valid, then for each ω-word w, according to the definition, $w \models f_2$ implies $w \models \exists k.(f_1 \, \mathsf{U}^k \, f_2)$ holds; otherwise, if $w \models f_1 \wedge \mathsf{F} f_2$, w.l.o.g., assume that $w, 0 \models f_1$ and $w, n \models f_2$, this also guarantees $w \models \exists k.(f_1 \, \mathsf{U}^k \, f_2)$ because $w \models f_1 \, \mathsf{U}^n \, f_2$.

As for the "only if" direction of (4), in the same way as before, we can infer that $f_1 \vee f_2$ should be valid if $\forall k.(f_1 \, \mathsf{U}^k \, f_2)$ is. Meanwhile, since for every ω-word w and every $n > 0$ there exists some $i \in \mathbb{N}$ having $w, i \times n \models f_2$, hence $\exists k.\mathsf{G}^k \neg f_2$ should not be satisfiable. Conversely, $\exists k.\mathsf{G}^k \neg f_2$ is not satisfiable implies that $\forall k.\mathsf{F}^k f_2$ is valid. Then, for each ω-word w and each $n > 0$, there exists a minimal number i making $w, i \times n \models f_2$, and since $f_1 \vee f_2$ is valid, we have $w, j \times n \models f_1$ for every $j < i$, hence $w \models f_1 \, \mathsf{U}^n \, f_2$. □

From the decision procedure, one can examine that each step could be accomplished with polynomial (in the size of the formula) space. Since **PSPACE** is closed under relativization, namely, $\mathbf{PSPACE}^{\mathbf{PSPACE}} = \mathbf{PSPACE}$, we can thus conclude that SATISFIABILITY of QPLTL is in **PSPACE**. On the other hand, the SATISFIABILITY problem of LTL is also **PSPACE**-hard. Thus, we have the following conclusion.

Corollary 2. *The* SATISFIABILITY *problem of QPLTL is **PSPACE**-complete.*

5 Discussion and Future Work

In this paper, we suggest to use quantified step variables to describe periodicity. As an attempt, using LTL as the base logic, we can obtain one of its decidable periodic extension — QPLTL.

Indeed, formulae of Type (I)–(III) constitutes a proper super logic of LTL, whereas a subset of whole ω-regular properties. It is interesting to study the relation between this set and ω-regular languages.

For formulae of Type (IV), actually, we may make a bit relaxation on that part. For example, consider the formula $f = \exists k.\mathsf{F}\mathsf{G}^k p$, which gives the assertion "p eventually holds periodically", and call such property *soft periodicity*. We can see that f is satisfiable if and only if $\exists k.\mathsf{G}^k p$ is satisfiable.

To make the logic more flexible in syntax and more expressive, as a future work, we need carefully study some extensions of QPLTL. For example, Boolean combinations of formulae of Type (IV), or combinations of Type (III) and (IV). Further, we are also wonder about the decidability of the extension built up from more expressive logics, such as linear-time μTL. The key issue is also to establish the corresponding normal form of general liveness properties.

Acknowledgement. The first author would thank Normann Decker, Daniel Thoma and Martin Leucker for the fruitful discussion on this problem. We would also thank the anonymous reviewers for their valuable comments on an earlier version of this paper. Wanwei Liu is supported by Natural Science Foundation of China (No. 61103012, No. 61379054, and No. 61532007). Fu Song is supported by Natural Science Foundation of China (No. 61402179 and No. 61532019).

References

[AS85] Alpern, B., Schneider, F.B.: Defining liveness. Inf. Process. Lett. **21**, 181–185 (1985)

[AS87] Alpern, B., Schneider, F.B.: Recognizing safety and liveness. Distrib. Comput. **2**(3), 117–126 (1987)

[BB87] Banieqbal, B., Barringer, H.: Temporal logic with fixed points. In: Banieqbal, B., Barringer, H., Pnueli, A. (eds.) Temporal Logic in Specification. LNCS, vol. 398, pp. 62–74. Springer, Heidelberg (1989). doi:10.1007/3-540-51803-7_22

[Büc62] Büchi, J.R.: On a decision method in restricted second order arithmetic. In: Proceedings of the International Congress on Logic, Method and Philosophy of Science 1960, pp. 1–12, Palo Alto. Stanford University Press (1962)

[Chu36] Church, A.: A note on the Entscheidungsproblem. J. Symb. Log. **1**, 101–102 (1936)

[CS01] Clarke, E.M., Schlingloff, B.: Model checking. In: Robinson, A., Voronkov, A. (eds.) Handbook of Automated Reasoning. vol. 2, Chapt. 24, pp. 1369–1522. MIT and Elsevier Science Publishers (2001)

[DG08] Diekert, V., Gastin, P.: First-order definable languages. In: Flum, J., Grädel, E., Wilke, T. (eds.) Logic and Automata: History and Perspectives [in Honor of Wolfgang Thomas], vol. 2, pp. 261–306. Amsterdam University Press (2008)

[Göd31] Gödel, K.: Über formal unentscheidbare sätze der principia mathematica und verwandter system I. Monatshefte für Methematik und Physik **38**, 173–198 (1931)

[LS07] Leucker, M., Sánchez, C.: Regular linear temporal logic. In: Jones, C.B., Liu, Z., Woodcock, J. (eds.) ICTAC 2007. LNCS, vol. 4711, pp. 291–305. Springer, Heidelberg (2007). doi:10.1007/978-3-540-75292-9_20

[McN66] McNaughton, R.: Testing and generating infinite sequences by a finite automaton. Inf. Comput. **9**, 521–530 (1966)

[Pnu77] Pnueli, A.: The temporal logic of programs. In: Proceedings of 18th IEEE Symposium on Foundation of Computer Science (FOCS 1977), pp. 46–57. IEEE Computer Society (1977)

[Rab69] Rabin, M.O.: Decidability of second order theories and automata on infinite trees. Trans. AMS **141**, 1–35 (1969)

[Sch10] Schweikardt, N.: On the expressive power of monadic least fixed point logic. Theor. Comput. Sci. **350**(2–3), 1123–1135 (2010)

[ST03] Sebastiani, R., Tonetta, S.: "More deterministic" vs. "smaller" Büchi automata for efficient LTL model checking. In: Geist, D., Tronci, E. (eds.) CHARME 2003. LNCS, vol. 2860, pp. 126–140. Springer, Heidelberg (2003). doi:10.1007/978-3-540-39724-3_12

[TH02] Taurainen, H., Heljanko, K.: Testing LTL formula translation into Büchi automata. STTT **4**, 57–70 (2002)

[Tho79] Thomas, W.: Star-free regular sets of omega-sequences. Inf. Control **42**, 148–156 (1979)

[VW94] Vardi, M.Y., Wolper, P.: Reasoning about infinite computations. Inf. Comput. **115**(1), 1–37 (1994)

[Wol83] Wolper, P.: Temporal logic can be more expressive. Inf. Control **56**(1–2), 72–99 (1983)