Constrained Properties, Semilinear Systems, and Petri Nets

Ahmed Bouajjani Peter Habermehl

VERIMAG Miniparc-Zirst, Rue Lavoisier, 38330 Montbonnot St-Martin, France.
email: Ahmed.Bouajjani@imag.fr, Peter.Habermehl@imag.fr

Abstract

We investigate the verification problem of two classes of infinite state systems w.r.t. nonregular properties (i.e., nondefinable by finite-state ω-automata). The systems we consider are Petri nets as well as *semilinear systems* including pushdown systems and PA processes. On the other hand, we consider properties expressible in the logic CLTL which is an extension of the linear-time temporal logic LTL allowing two kinds of constraints: *pattern constraints* using finite-state automata and *counting constraints* using Presburger arithmetics formulas. While the verification problem of CLTL is undecidable even for finite-state systems, we identify a fragment called $\mathrm{CLTL}_{\Box}$ for which the verification problem is decidable for pushdown systems as well as for Petri nets. This fragment is strictly more expressive than finite-state ω-automata. We show that, however, the verification problem of semilinear systems (PA processes in particular) is undecidable even w.r.t. LTL formulas. Therefore, we identify another fragment (a restriction of LTL extended with counting constraints) covering a significant class of properties and for which the verification problem is decidable for all PA processes.

1 Introduction

Reasoning about infinite state systems is an important and intensively studied topic in automatic verification and concurrency theory [1, 8, 6, 12, 4, 9, 11]. Several classes of infinite state systems are investigated corresponding to different description formalisms that are, mainly, either process algebras like BPA (context-free processes) [2] and BPP (basic parallel processes) [7], or "*extended*" automata like pushdown systems, Petri nets (vector addition systems with states), and lossy channel systems. Important results have been established on the verification problem of such systems. These results concern behavioural equivalences/preorders testing [8] as well as model checking [1, 6, 12, 4, 11]. Our work follows the latter verification approach. Its originality consists of the consideration of *nonregular* properties. Indeed, as far as we know, all the other works on the subject address the verification of properties expressed in the usual specification logics like propositional temporal logics and μ-calculi, or by means of finite-state ω-automata. However, these formalisms cannot capture some important aspects of the behaviours of infinite state systems. In particular, it is impossible to express in these formalisms properties involving *counting constraints*, i.e., properties comparing *numbers of occurrences of events*. These properties are essential to characterize behaviours of systems involving counting mechanisms. To express such properties, we have proposed in [5, 4] new specification logics combining temporal logics with Presburger arithmetics. In these previous works, we have considered the verification problem of nonregular properties for infinite state systems described in the process algebra PA [3] (subsuming BPA and BPP). We have identified classes of nonregular properties for which the verification problem is decidable for PA or BPA using different specific reductions to the satisfiability problem in Presburger arithmetics.

In this work, we pursue our investigations by considering a more general framework and establishing decidability results for the verification problem concerning larger classes of systems and properties. We propose classes of nonregular properties for which a uniform approach can be applied to reason about the verification problem for different classes of infinite state systems. Roughly speaking, the basic idea is to define properties that can be *decomposed* into ω-regular properties and elementary nonregular properties, e.g., counting constraints on the set of prefixes. Depending on the nature of the system, the ω-regular property and the constraints, this *constrained emptiness problem* is decidable or not, and when it is decidable, different techniques can be applied to establish this fact.

The properties we consider are expressed in the logic CLTL (Constrained Linear Temporal Logic) introduced in [4]. This logic is an extension of the linear-time propositional temporal logic LTL [16] with the ability of expressing *pattern constraints* and *counting constraints* on computations. Pattern constraints are expressed using finite-state automata and allow to say that the computation since some given point in the past corresponds to some pattern (specified as a regular language). Counting constraints are expressed using Presburger arithmetics formulas and allow to say that the numbers of occurrences of events since some designated points in the past fulfil some arithmetical constraints. CLTL has two sublogics ALTL and PLTL corresponding respectively to the extensions of LTL with either pattern or counting constraints only.

The satisfiability problem of CLTL (as well as PLTL) is highly undecidable (Σ_1^1-complete), and already the verification problem of PLTL is undecidable even for finite-state systems [4]. Therefore, we define a syntactical fragment of CLTL called $\mathrm{CLTL}_\Box$ and its corresponding fragment of PLTL called $\mathrm{PLTL}_\Box$. These fragments are not closed under negation. So, we characterize syntactically the complements of $\mathrm{CLTL}_\Box$ and $\mathrm{PLTL}_\Box$ properties by introducing two other fragments $\mathrm{CLTL}_\Diamond$ and $\mathrm{PLTL}_\Diamond$. Both fragments $\mathrm{CLTL}_\Box$ and $\mathrm{CLTL}_\Diamond$ subsume the logic ALTL which expresses exactly the ω-regular properties. As for $\mathrm{PLTL}_\Box$ and $\mathrm{PLTL}_\Diamond$, they subsume the logic LTL, which means that they can express all the ω-star-free properties [17].

Then, the key result we prove is that $\mathrm{CLTL}_\Diamond$ properties can be decomposed into ω-regular properties and eventuality properties with counting constraints. The same holds for $\mathrm{PLTL}_\Diamond$ with ω-star-free properties instead of ω-regular ones. By this decomposition, the satisfiability problem of a $\mathrm{CLTL}_\Diamond$ formula φ relatively to a system $\mathcal{S}$ reduces to a *constrained emptiness problem* as mentioned above. Then, we discuss the decidability of this problem and the techniques that can be applied to solve it, depending on the considered systems and formulas.

We consider two incomparable and fairly general classes of systems that are *semilinear systems* and Petri nets. Semilinear systems are those generating sets of finite traces whose Parikh images are semilinear and effectively constructible (examples are pushdown systems and PA processes). On the other hand, BPP processes, that are semilinear, can also be encoded as Petri nets.

First, we investigate the case of semilinear systems and consider the decision method based on reduction to the nonemptiness problem of semilinear sets. This reduction is possible if the intersection of the ω-language of the considered system with the ω-regular part of the property is semilinear (via Parikh image of its set of prefixes). This is the case for pushdown systems, and hence, we deduce that their verification problem w.r.t. $\mathrm{CLTL}_\square$ is decidable. This result generalizes the one we have established in [4] for BPA processes and a subset of $\mathrm{CLTL}_\square$. This reduction is, however, not possible for all semilinear systems. Indeed, we can show that for PA processes the verification problem is undecidable even for LTL properties. Therefore, we consider another pair of dual fragments, called simple-$\mathrm{PLTL}_\square$ and simple-$\mathrm{PLTL}_\diamond$, and prove that the verification problem of all PA processes w.r.t. simple-$\mathrm{PLTL}_\square$ is decidable. This result generalizes the one we established in [4] for a subset of simple-$\mathrm{PLTL}_\square$.

Then, we consider the case of Petri nets. We show that the constrained nonemptiness problem stated above can be reduced in this case to the reachability problem in Petri nets. Consequently, the verification problem of Petri nets w.r.t. $\mathrm{CLTL}_\square$ is decidable. In particular this fact holds for BPP processes, and thus, answers the question we left open in [4]. Our result extends the one shown in [12] for Petri nets and the linear-time μ-calculus, since we consider a strictly more expressive logic allowing nonregular properties. Moreover, $\mathrm{CLTL}_\square$ allows to express constraints on places of Petri nets by counting their ingoing and outgoing transitions. Then, our result can be considered as a decidability result for a logic on Petri nets markings. In this context, our result is incomparable with the existing results [14, 13].

The paper is organized as follows. In Section 2, we introduce notations and give some preliminary results. In Section 3, we define CLTL. In Section 4, we introduce the fragments of CLTL. In Section 5, we show the decomposition of $\mathrm{CLTL}_\diamond$ properties. In Section 6, we consider the verification problem of semilinear systems and its reducibility to the emptiness problem of semilinear sets. In Section 7, we consider the verification problem of Petri nets. We conclude in Section 8.

2 Preliminaries

2.1 Sequences, languages, projection, and cylindrification

Let Σ be a finite alphabet. We denote by Σ^* (resp. Σ^ω) the set of finite (resp. infinite) sequences over Σ. Let $\Sigma^\infty = \Sigma^* \cup \Sigma^\omega$. A *language* (resp. *$\omega$-language*) is a subset of Σ^* (resp. Σ^ω).

An infinite sequence $\sigma \in \Sigma^\omega$ can be seen as a mapping from $\mathbb{N}$ to Σ. Hence, σ is equal to $\sigma(0)\sigma(1)\cdots$. Given i and j with $i \leq j$, we denote by $\sigma(i,j)$ the finite sequence $\sigma(i)\cdots\sigma(j)$ (with $\sigma(i,i) = \sigma(i)$). We denote by $Pref(\sigma)$ the set of finite prefixes of σ, i.e., $Pref(\sigma) = \{\sigma(0,i) \ : \ i \geq 0\}$. This notation can be extended to sets of sequences in the obvious way.

Let $\mathcal{P}$ be a finite set of *atomic propositions*. Then, we consider a set of *transition labels* $\Sigma = 2^{\mathcal{P}}$. Let $\Sigma' \supseteq \Sigma$. Then, given a sequence $\sigma' \in (\Sigma')^\omega$, the *projection* of σ' on Σ, denoted $\sigma'|_\Sigma$, is the sequence $\sigma \in \Sigma^\omega$ such that, for every $i \geq 0$, $\sigma(i) = \sigma'(i) \cap \mathcal{P}$. Conversely, given a sequence $\sigma \in \Sigma^\omega$, the *cylindrification* of σ to Σ', denoted $\widetilde{\sigma}$, is the set of sequences $\sigma' \in (\Sigma')^\omega$ such that $\sigma = \sigma'|_\Sigma$. These definitions are generalized to sets of sequences.

Notice that for every $S \subseteq (\Sigma')^\omega$, $S = \emptyset$ iff $S|_\Sigma = \emptyset$. Moreover, it is clear that projection distributes w.r.t. union (i.e., $(S \cup S')|_\Sigma = S|_\Sigma \cup S'|_\Sigma$). However, projection does not distribute w.r.t. conjunction. Indeed, given $S, S' \in (\Sigma')^\omega$, we have $(S \cap S')|_\Sigma \subseteq S|_\Sigma \cap S'|_\Sigma$, but the converse does not hold in general. Nevertheless, we can show the following fact:

Lemma 2.1 *Let $S, T \subseteq \Sigma^\omega$, and $T' \subseteq (\Sigma')^\omega$ such that $T = T'|_\Sigma$. Then, $S \cap T = (\widetilde{S} \cap T')|_\Sigma$.*

2.2 Finite-state ω-automata

A finite-state Büchi ω-automaton is a tuple $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ where (Q, Σ, q_0, δ) is a finite-state labelled transition system (LTS), let us call it $\mathcal{S}_\mathcal{A}$, and $F \subseteq Q$ is the set of repeating locations. Given a sequence $\sigma \in \Sigma^\omega$, a run of $\mathcal{S}_\mathcal{A}$ over σ is a sequence $\rho \in Q^\omega$ such that $\rho(0) = q_0$, and $\forall i \geq 0$, $(\rho(i), \sigma(i), \rho(i+1)) \in \delta$. Let $\rho \in Q^\omega$ be a run of $\mathcal{S}_\mathcal{A}$. We denote by $Inf(\rho)$ the set of locations q such that $\exists^\infty i \in I\!N$ with $\rho(i) = q$. Then, a run ρ is *accepting* if $Inf(\rho) \cap F \neq \emptyset$. The *$\omega$-language* of $\mathcal{A}$, denoted by $L(\mathcal{A})$, is the set of sequences $\sigma \in \Sigma^\omega$ such that $\mathcal{S}_\mathcal{A}$ has an accepting run over σ. Subsets of Σ^ω that are recognizable by finite-state Büchi ω-automata are called *ω-regular* languages. The class of ω-regular languages is closed under all boolean operations.

A *simple ω-regular language* is an ω-language definable by a finite-state Büchi ω-automaton such that every loop in its transition system is a self-loop. The class of simple ω-regular language is closed under union and intersection but not under complementation.

2.3 Pushdown systems, and PA processes

The definitions of finite-state labelled transition systems and ω-automata can be extended in the usual way to *pushdown systems* and *ω-pushdown automata* (the Büchi acceptance condition is defined, as in the finite-state case, by means of a set of repeating control locations). Subsets of Σ^ω that are recognizable by pushdown Büchi ω-automata are called ω-context-free languages [10].

A PA process [3] is defined by a finite set of well-guarded recursive equations of the form $X = t$ where X is a process variable, and t is a term constructed from transition labels (actions), process variables, and binary operators: nondeterministic choice "+", sequential composition "$\cdot$", and merge (or asynchronous parallel) composition "$\|$". BPA processes are PA processes without merge composition.

They generate the same ω-languages as pushdown systems. BPP processes are PA processes with prefixing ("$a \cdot t$") instead of general sequential composition "$\cdot$". They correspond to a subclass of Petri nets and generate a subclass of ω-context sensitive languages incomparable with BPA ω-languages [7]. We mention finally that the classes of PA and Petri nets ω-languages are incomparable.

2.4 Petri nets

A *Petri net* is a tuple $\mathcal{N} = (\mathbf{P}, \mathbf{T}, \mathbf{F}, M_{\mathcal{N}}, \Lambda)$ where $\mathbf{P}$ is a finite set of *places*, $\mathbf{T}$ is a finite set of *transitions* such that $\mathbf{P}$ and $\mathbf{T}$ are disjoint, $\mathbf{F} : (\mathbf{P} \times \mathbf{T}) \cup (\mathbf{T} \times \mathbf{P}) \rightarrow \mathbb{N}$ is the *flow function*, $M_{\mathcal{N}} : \mathbf{P} \rightarrow \mathbb{N}$ is an *initial marking*, and $\Lambda : \mathbf{T} \rightarrow \Sigma$ is a labelling function. A *marking* M associates a natural number (number of *tokens*) to each place. A marking is also considered as a vector in $\mathbb{N}^{|\mathbf{P}|}$. We write $M[t\rangle$ if, $\forall p \in P$, $M(p) \geq \mathbf{F}(p, t)$, and we write $M[t\rangle M'$ if $M[t\rangle$ and, $\forall p \in P$, $M'(p) = M(p) - \mathbf{F}(p, t) + \mathbf{F}(t, p)$. Given $a \in \Sigma$, we write $M \xrightarrow{a}$ (resp. $M \xrightarrow{a} M'$) if $\exists t \in \mathbf{T}$ such that $M[t\rangle$ (resp. $M[t\rangle M'$) and $\Lambda(t) = a$. These definitions can be extended to sequences of transitions $\tau \in \mathbf{T}^{\infty}$ and sequences of transition labels $\sigma \in \Sigma^{\infty}$. The *reachability set* of $\mathcal{N}$, denoted by $\mathcal{R}(\mathcal{N})$, is the set of markings M such that $\exists \tau \in \mathbf{T}^*$, $M_{\mathcal{N}}[\tau\rangle M$. The *$\omega$-language* of $\mathcal{N}$, denoted by $L(\mathcal{N})$, is the set of infinite sequences $\sigma \in \Sigma^{\omega}$ such that $M_{\mathcal{N}} \xrightarrow{\sigma}$. Finally, given a transition $t \in \mathbf{T}$, we denote by $\mathcal{M}_{\infty}(\mathcal{N}, t)$ the set of markings M for which $\exists \tau \in \mathbf{T}^{\omega}$ such that $M[\tau\rangle$ and $\exists^{\infty} i \in \mathbb{N}$ with $\tau(i) = t$.

2.5 Semilinear sets, semilinear languages, and semilinear systems

A *linear set* is a subset of $\mathbb{N}^n$ of the form $\{\vec{v} + k_1\vec{u}_1 + \cdots + k_m\vec{u}_m \; : \; k_1, \cdots, k_m \in \mathbb{N}\}$ where $n > 0$ and $\vec{v}, \vec{u}_1, \cdots, \vec{u}_m \in \mathbb{N}^n$. A *semilinear set* is a finite union of linear sets. Let $\sigma \in \Sigma^*$. For every $a \in \Sigma$, $|\sigma|_a$ is the number of occurrences of a in σ. Let $\Sigma = \{a_1, \cdots, a_n\}$. We denote by $[\sigma]$ the Parikh image of σ, i.e., the vector $(|\sigma|_{a_1}, \cdots, |\sigma|_{a_n})$ of $\mathbb{N}^n$. This notation is generalized to sets of sequences. A set $S \subseteq \Sigma^*$ (resp. $S \subseteq \Sigma^{\omega}$) is a *semilinear language* (resp. *semilinear ω-language*) if the set of vectors $[S]$ (resp. $[Pref(S)]$) is semilinear. A *semilinear system* is any system whose ω-language S is semilinear and such that (a representation of) $[Pref(S)]$ is effectively constructible from the representation of the system.

Lemma 2.2 *ω-context-free languages as well as PA ω-languages are semilinear. Petri nets ω-languages are, however, not semilinear.*

For PA processes, the proof uses Parikh's theorem (concerning context-free languages) and the fact that permutation of symbols preserves Parikh image. As for Petri nets, this is a direct consequence of the well known fact that sets of reachable markings are not semilinear in general.

Remark 2.1 The class of semilinear sets are closed under all boolean operations (they correspond exactly to the Presburger arithmetics definable sets). The class of semilinear languages is, however, not closed under intersection (see Section 6).

3 Constrained Linear Temporal Logic

3.1 Syntax and semantics

Recall that $\mathcal{P}$ is a finite set of atomic propositions and that $\Sigma = 2^{\mathcal{P}}$. We use letters $P, Q, \ldots$ to range over elements of $\mathcal{P}$. Let $\mathcal{V}$ be a set of integer valued variables. We use letters $x, y, \ldots$ to range over variables in $\mathcal{V}$. We use also letters $f, g, \ldots$ to range over Presburger arithmetics formulas (the first order logic of natural numbers with addition, subtraction, and the usual ordering). We introduce a set $\mathcal{W}$ of *position variables*, and use letters $u, v, \ldots$ to range over $\mathcal{W}$. We use letters $A, B, \ldots$ to range over deterministic finite-state (Rabin-Scott) automata over Σ. We denote by $L(A)$ the set of sequences in Σ^* accepted by A, by $\overline{A}$ an automaton recognizing $\Sigma^* - L(A)$, and by $A \times B$ an automaton recognizing $L(A) \cap L(B)$. Finally, let π range over the set of *propositional formulas* that are boolean combinations of atomic propositions. Then, the set of formulas of CLTL is defined by:

$$\varphi ::= P \mid \neg\varphi \mid \varphi \vee \varphi \mid \bigcirc\varphi \mid \varphi\,\mathcal{U}\varphi \mid \widetilde{\exists} x.\varphi \mid [x : \pi].\varphi \mid f \mid u.\varphi \mid A^u$$

We define also two sublogics of CLTL obtained by extending LTL by either pattern or counting constraints only. The first logic, called ALTL, corresponds to the set of formulas

$$\psi \ ::= \ P \mid \neg\psi \mid \psi \vee \psi \mid \bigcirc\psi \mid \psi\,\mathcal{U}\psi \mid u.\psi \mid A^u$$

whereas the second logic, called PLTL, corresponds to the set of formulas

$$\psi \ ::= \ P \mid \neg\psi \mid \psi \vee \psi \mid \bigcirc\psi \mid \psi\,\mathcal{U}\psi \mid \widetilde{\exists} x.\varphi \mid [x : \pi].\psi \mid f$$

We consider abbreviations as the boolean connectives $\wedge$, $\Rightarrow$, the universal quantification $\widetilde{\forall}$, $\Diamond\varphi = true\,\mathcal{U}\varphi$, $\Box\varphi = \neg\Diamond\neg\varphi$, and $\varphi_1\overline{\mathcal{U}}\varphi_2 = \varphi_1\mathcal{U}(\varphi_1 \wedge \varphi_2)$. We write $[\vec{x} : \vec{\pi}].\varphi$ or $[x_1, \ldots, x_n : \pi_1, \ldots, \pi_n].\varphi$ or $[x_i : \pi_i]_{i=1}^{n}.\varphi$ for $[x_1 : \pi_1].\cdots[x_n : \pi_n].\varphi$.

CLTL formulas are interpreted on infinite sequences over Σ. The operators $\bigcirc$, $\mathcal{U}$, $\Diamond$, and $\Box$, are the *next*, *until*, *eventually*, and *always* operators of LTL; $\overline{\mathcal{U}}$ is a *right-closed until* operator.

The operator $\widetilde{\exists}$ is the (rigid) quantification over natural numbers. We distinguish between $\widetilde{\exists}$ and the Presburger arithmetics quantifier $\exists$ since they do not have the same scope. The construction "$[x : \pi]$." introduce a *counting variable* x which is associated with the propositional formula π. The variable x counts from the current position the number of occurrences of transition labels satisfying π on the sequence. Then, x can be used in Presburger formulas f to express *counting constraints* (that may involve several counting variables). For instance, the formula $\phi_1 = [x, y : \pi_1, \pi_2].\Box(P \Rightarrow (x \leq y))$ expresses the fact that from now on, whenever P holds, the number of transitions satisfying π_2 is greater than the number of transitions satisfying π_1.

The construction "u." associates the position variable u with the current position on the sequence. The variable u is used as a label allowing to refer to the position associated with it. Then, u can be used to express *pattern constraints* A^u saying that the subsequence since the position u is accepted by the automaton A. For instance, the formula $\phi_2 = u.[x, y : \pi_1, \pi_2].\Box(A^u \Rightarrow (x \leq y))$ expresses the fact that from now on, in every finite subsequence accepted by A, the number of transitions satisfying π_2 is greater or equal than the number of transitions satisfying π_1.

In the formula ϕ_2, the construction "$[x : \pi_1]$." (resp. "u.") binds the variable x (resp. u) in the subformula $\Box(A^u \Rightarrow (x \leq y))$. So, a variable $x \in \mathcal{V}$ may be bound by either $\widetilde{\exists}$, or by Presburger quantification, or by the construction "$[x : \pi]$.". A position variable $u \in \mathcal{W}$ can be bound by the construction "u.". We call "$[x : \pi]$." (resp. "u.") the *reset quantification* (resp. *position quantification*). We suppose without loss of generality that each variable is bound at most once. Then, every variable appearing in some formula is either *bound* or *free*. A formula φ is *closed* if all the variables occurring in it are bound, otherwise φ is *open*.

The formal semantics of CLTL is defined using a satisfaction relation $\models$ between sequences in Σ^ω, positions (positive integers), and formulas. Since formulas may be open, the relation is parameterized by a valuation E of the variables in $\mathcal{V}$ (we write $E \models f$ when the evaluation of f under E is true), a *position association* θ that associates with each counting or position variable the position where it has been introduced, and a *propositional formula association* η that records for each counting variable the propositional formula which is associated with it. Let ξ stands for E, θ, or η. Then, $\mathcal{D}(\xi)$ denotes the domain of ξ; the function ξ such that $\mathcal{D}(\xi) = \emptyset$ is denoted by $\emptyset$. We denote by $\xi[z \leftarrow \kappa]$ the function ξ' such that $\mathcal{D}(\xi') = \mathcal{D}(\xi) \cup \{z\}$, and which associates the value κ with z and coincides with ξ on all the other variables.

Now, let $\sigma \in \Sigma^\omega$. Then, for every $i \geq 0$, every valuation E, every position association θ (such that $\forall z \in \mathcal{D}(\theta)$, $0 \leq \theta(z) \leq i$), every propositional formula association η, and every CLTL formula φ, we define the meaning of $\langle\sigma, i\rangle \models_{(E,\theta,\eta)} \varphi$ inductively on the structure of φ; the definition is given in Table 1. Let φ be a closed formula. It is clear that $\langle\sigma, i\rangle \models_{(E,\theta,\eta)} \varphi$ iff $\langle\sigma, i\rangle \models_{(\emptyset,\emptyset,\emptyset)} \varphi$, and hence, we write simply $\langle\sigma, i\rangle \models \varphi$. We write also $\sigma \models \varphi$, and say that σ *satisfies* φ, if $\langle\sigma, 0\rangle \models \varphi$. Let $[\![\varphi]\!]$ be the set of sequences $\sigma \in \Sigma^\omega$ such that $\sigma \models \varphi$. For every $S \subseteq \Sigma^\omega$, φ is *satisfiable* (resp. *valid*) relatively to S iff $S \cap [\![\varphi]\!] \neq \emptyset$ (resp. $S \subseteq [\![\varphi]\!]$), and φ is *satisfiable* (resp. *valid*) iff it is satisfiable (resp. valid) relatively to Σ^ω. The relative satisfiability (resp. validity) problem is whether a given formula is satisfiable (resp. valid) relatively to a given set of sequences. The relative validity problem is also called verification problem.

3.2 Expressiveness

We can show that ALTL is as expressive as finite-state ω-automata. Indeed, by McNaughton's theorem, every ω-regular language can be defined by an ALTL formula of the form:

$$u.\ \bigvee_{i=1}^{n} (\Box\Diamond A_i^u \wedge \Diamond\Box B_i^u)$$

On the other hand, for any given closed ALTL formula ψ, we can construct a Büchi ω-automaton which recognizes precisely $[\![\psi]\!]$. This construction generalizes the one given in [19] for LTL formulas by dealing with position quantification

$$
\begin{array}{ll}
\langle\sigma,i\rangle \models_{(E,\theta,\eta)} P & \text{iff } P\in\sigma(i)\\
\langle\sigma,i\rangle \models_{(E,\theta,\eta)} \neg\varphi & \text{iff } \langle\sigma,i\rangle \not\models_{(E,\theta,\eta)} \varphi\\
\langle\sigma,i\rangle \models_{(E,\theta,\eta)} \varphi_1\vee\varphi_2 & \text{iff } \langle\sigma,i\rangle \models_{(E,\theta,\eta)} \varphi_1 \text{ or } \langle\sigma,i\rangle \models_{(E,\theta,\eta)} \varphi_2\\
\langle\sigma,i\rangle \models_{(E,\theta,\eta)} \bigcirc\varphi & \text{iff } \langle\sigma,i+1\rangle \models_{(E,\theta,\eta)} \varphi\\
\langle\sigma,i\rangle \models_{(E,\theta,\eta)} \varphi_1\mathcal{U}\varphi_2 & \text{iff } \exists j.\ i\le j.\ \langle\sigma,j\rangle \models_{(E,\theta,\eta)} \varphi_2 \text{ and}\\
& \quad \forall k.\ i\le k<j.\ \langle\sigma,k\rangle \models_{(E,\theta,\eta)} \varphi_1\\
\langle\sigma,i\rangle \models_{(E,\theta,\eta)} \widetilde{\exists}x.\varphi & \text{iff } \exists k\in I\!N.\ \langle\sigma,i\rangle \models_{(E',\theta,\eta)} \varphi \ \text{ where } E'=E[x\leftarrow k]\\
\langle\sigma,i\rangle \models_{(E,\theta,\eta)} [x:\pi].\varphi & \text{iff } \langle\sigma,i\rangle \models_{(E,\theta',\eta')} \varphi \text{ where}\\
& \quad \theta'=\theta[x\leftarrow i] \text{ and } \eta'=\eta[x\leftarrow\pi]\\
\langle\sigma,i\rangle \models_{(E,\theta,\eta)} f & \text{iff } E'\models f \ \text{ where}\\
& \quad E'=E[x\leftarrow |\{j\in[\theta(x),i]\ :\ \langle\sigma,j\rangle\models\eta(x)\}|]_{x\in\mathcal{D}(\eta)}\\
\langle\sigma,i\rangle \models_{(E,\theta,\eta)} u.\varphi & \text{iff } \langle\sigma,i\rangle \models_{(E,\theta',\eta)} \varphi \ \text{ where } \theta'=\theta[u\leftarrow i]\\
\langle\sigma,i\rangle \models_{(E,\theta,\eta)} A^u & \text{iff } \sigma(\theta(u),i)\in L(A)
\end{array}
$$

Table 1. Definition of the satisfaction relation

and pattern constraints. It uses mainly the fact that every regular language (pattern constraint) has a finite number of derivatives (left-quotients) w.r.t. finite sequences over Σ.

Using counting constraints, we can express *nonregular* properties, i.e., properties that cannot be expressed by ω-regular automata. For instance, consider the property saying: given an infinite sequence of transitions (events), every a is followed by a b, at each position between an a and the next b, the number of c's is greater or equal than the number of d's, and at b, the numbers of c's and d's are equal. Formally, the property imposes that the sequences between two successive a and b are in the language $\{\sigma\in\Sigma^*\ :\ |\sigma|_c=|\sigma|_d$, and $\forall i\le|\sigma|$, $|\sigma(0,i)|_c\ge|\sigma(0,i)|_d\}$. This property can be expressed in PLTL by:

$$\Box\,(a\Rightarrow[x,y,z:c,d,b].\,((x\ge y\,\mathcal{U}\,b)\wedge\Box\,((b\wedge z=1)\Rightarrow x=y))) \tag{1}$$

The introduction of counting constraints allows to characterize nonregular languages that can be context-free as in (1), but also context-sensitive when constraints relating more than two counting variables are considered.

The use of pattern constraints allows to constrain the order of appearance of events. Suppose for instance that we want to strengthen the property above by imposing that between two successive a and b, all the c's appear before all the d's. The new property can be expressed by the conjunction of the LTL formula $\Box(a\Rightarrow\Diamond b)$ (every a is followed by a b) with the CLTL formula:

$$u.\,[x,y:c,d].\,\Box\,(A^u\Rightarrow(B^u\wedge x=y)) \tag{2}$$

where A and B are finite-state automata such that $L(A)=\Sigma^*a(\Sigma-\{a,b\})^*b$, and $L(B)=\Sigma^*a(\Sigma-\{d\})^*(\Sigma-\{c\})^*b$.

Finally, let us illustrate the use of the $\widetilde{\forall}$ quantification. It allows to relate counting constraints at different positions on the sequence, and express counting constraints on the numbers of occurrences of events in different subsequences. For instance, consider the property saying: every a is followed by a b, and between successive a's and b's the subsequences are in the language $\{\sigma s \sigma' \ : \ \sigma, \sigma' \in (\Sigma - \{a,b,s\})^*, |\sigma|_c = |\sigma'|_d\}$. This property can be expressed by the conjunction of the LTL formula $\Box\,(a \Rightarrow \bigcirc\,(\neg s\,\mathcal{U}\,(s \wedge \bigcirc\,(\neg s\,\mathcal{U}\,b))))$ with the PLTL formula:

$$\widetilde{\forall} n.\, \Box\,(a \Rightarrow [x,y,z : c,d,b].\, \Box\,((s \wedge x = n) \Rightarrow \Box\,((b \wedge z = 1) \Rightarrow y = n))) \qquad (3)$$

The variable n in (3) is used to memorize the number of c's between a and s, and then, this number can be compared with the number of d's between s and b.

3.3 Undecidability results

Theorem 3.1 *([4]) The satisfiability problems of PLTL and CLTL are Σ_1^1-complete. Consequently, the validity problems of PLTL and CLTL, as well as their verification problem for finite-state LTS's, are Π_1^1-complete.*

Actually, we can prove the undecidability of the verification problem for even a very simple class of PLTL formulas corresponding to *counting constraints eventuality properties*, i.e., formulas of the form $[\vec{x} : \vec{\pi}].\, \Diamond f(\vec{x})$. For this kind of formulas, the verification problem for finite-state LTS's is Σ_1^0-complete.

4 The fragments CLTL$_\Box$ and CLTL$_\Diamond$

We introduce hereafter several fragments of CLTL. These fragments are defined so that they do not contain formulas that cause the undecidability of the verification problem, namely the counting constraints eventuality formulas. These fragments are not closed under negation. So, for each of these fragments we introduce another one such that the negation of every formula in the first one is equivalent to a formula in the second one, and vice versa. We discuss the expressiveness of these fragments and consider their satisfiability and validity problems.

4.1 Definitions

We start by defining the most expressive fragments, called CLTL$_\Box$ and CLTL$_\Diamond$. To describe simply the syntactical restrictions corresponding to these fragments, we introduce the *positive form* of CLTL formulas given by:

$$\varphi \ ::= \ \widetilde{\exists} x.\varphi \mid \widetilde{\forall} x.\varphi \mid [x:\pi].\varphi \mid f \mid u.\varphi \mid A^u \mid P \mid \neg P \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \bigcirc \varphi \mid \Box \varphi \mid \varphi\,\mathcal{U}\,\varphi \qquad (4)$$

We can prove that every CLTL formula has an equivalent formula in positive normal form. Notice that the positive form of ALTL (resp. PLTL) formulas corresponds to (4) without reset (resp. position) quantification and counting (resp. pattern) constraints.

Now, recall that the verification problem is undecidable as soon as eventuality formulas with counting constraints are considered. Hence, to avoid such formulas, we define the fragment CLTL$_\Box$ obtained by imposing in (4) that the formulas in the right-hand side of $\mathcal{U}$ must be in ALTL (i.e., counting constraints free). We admit also in this fragment $\overline{\mathcal{U}}$-formulas under the same condition. We also forbid in $\mathrm{CLTL}_\square$ the operator $\widetilde{\exists}$. The fragment $\mathrm{CLTL}_\diamond$ is defined in such a manner that it characterizes exactly the complements of $\mathrm{CLTL}_\square$ properties. So, $\mathrm{CLTL}_\square$ consists of the set of formulas φ defined by:

$$\varphi \ ::= \ \widetilde{\forall} x.\varphi \mid [x:\pi].\varphi \mid f \mid u.\varphi \mid A^u \mid P \mid \neg P \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \bigcirc \varphi \mid \square \varphi \mid \varphi \mathcal{U} \psi \mid \varphi \overline{\mathcal{U}} \psi$$

whereas $\mathrm{CLTL}_\diamond$ consists of the set of formulas φ defined by:

$$\varphi \ ::= \ \widetilde{\exists} x.\varphi \mid [x:\pi].\varphi \mid f \mid u.\varphi \mid A^u \mid P \mid \neg P \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \bigcirc \varphi \mid \square \psi \mid \psi \mathcal{U} \varphi \mid \psi \overline{\mathcal{U}} \varphi$$

where, in both definitions, ψ stands for any ALTL formula.

We define in a similar way the fragments of PLTL called $\mathrm{PLTL}_\square$ and $\mathrm{PLTL}_\diamond$ (in this case, the ψ's are required to be LTL formulas). Moreover, we consider the fragments *simple*-$\mathrm{PLTL}_\square$ and *simple*-$\mathrm{PLTL}_\diamond$ obtained from the previous ones by imposing that the ψ's are propositional formulas instead of any LTL formulas.

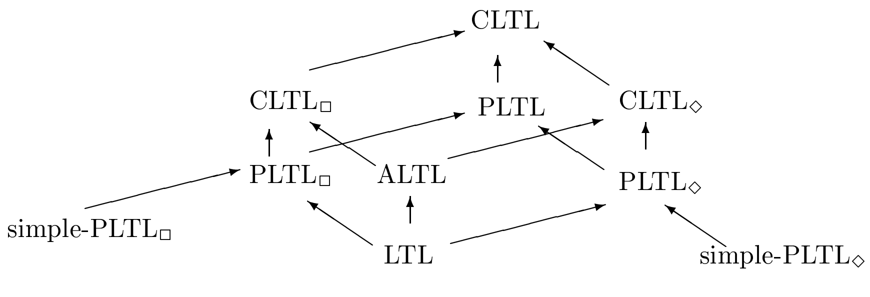
Proposition 4.1 *For every $\mathrm{CLTL}_\square$ (resp. $\mathrm{PLTL}_\square$, simple-$\mathrm{PLTL}_\square$) closed formula φ, there exists a $\mathrm{CLTL}_\diamond$ (resp. $\mathrm{PLTL}_\diamond$, simple-$\mathrm{PLTL}_\diamond$) closed formula φ' such that $[\![\neg\varphi]\!] = [\![\varphi']\!]$, and conversely.*

4.2 Expressiveness

It is easy to see that the set of ALTL formulas in positive form is a subset of both $\mathrm{CLTL}_\square$ and $\mathrm{CLTL}_\diamond$. Hence, both fragments $\mathrm{CLTL}_\square$ and $\mathrm{CLTL}_\diamond$ can express all the ω-regular properties. However, these fragments obviously do not express the same classes of nonregular properties. For instance, $\mathrm{CLTL}_\square$ allow to express *constrained safety properties* (see (2) for instance), but cannot express their negations (*constrained eventuality properties*) that are $\mathrm{CLTL}_\diamond$ properties.

Similarly, it can be observed that $\mathrm{PLTL}_\square$ and $\mathrm{PLTL}_\diamond$ subsume the logic LTL, and hence, they can express all the ω-*star-free* properties [17]. Moreover, we can show that simple-$\mathrm{PLTL}_\diamond$ expresses all the *simple* ω-regular languages (the definition is given in Section 2.2) whereas simple-$\mathrm{PLTL}_\square$ expresses obviously all their complements (recall that simple ω-regular languages are not closed under complementation). The same duality existing between $\mathrm{CLTL}_\square$ and $\mathrm{CLTL}_\diamond$ concerning nonregular properties exists also between their subfragments. Even restricted, these subfragments allow to capture significant classes of nonregular properties. For instance, the formulas (1) and (3) are in simple-$\mathrm{PLTL}_\square$.

The following picture shows the inclusions between the different logics we consider.



4.3 Satisfiability problem

We can prove in the same manner as Theorem 3.1 the following undecidability result concerning the satisfiability and validity problems of the fragments introduced above.

Theorem 4.1 *The satisfiability (resp. validity) problems of simple-PLTL$_\Box$, PLTL$_\Box$, and CLTL$_\Box$ (resp. simple-PLTL$_\Diamond$, PLTL$_\Diamond$, and CLTL$_\Diamond$) are Σ^1_1-complete (resp. Π^1_1-complete).*

We prove in Section 6 that the validity (resp. satisfiability) problem of CLTL$_\Box$ (resp. CLTL$_\Diamond$) is actually decidable (see Corollary 6.1).

5 Decomposing CLTL$_\Diamond$ properties

We show in this section that every CLTL$_\Diamond$ property can be decomposed (modulo projection) into an ω-regular property and a counting constraints eventuality property (more precisely, every CLTL$_\Diamond$ property is a finite union of projections of sets that are intersections of ω-regular properties with counting constraints eventuality properties).

This decomposition is helpful for reasoning about the (relative) satisfiability problem of CLTL$_\Diamond$, and hence, on the verification problem of CLTL$_\Box$. Indeed, it allows to reduce the satisfiability problem of a CLTL$_\Diamond$ formula φ relatively to a set of sequences S, to a *constrained emptiness problem*: whether there exists a sequence in a set of finite sequences which satisfies some counting constraints. This set of finite sequences is the set of prefixes of sequences that are in the intersection of S (a cylindrification of S actually) with, roughly speaking, the "ω-regular part" of the property expressed by φ. Hence, when the set S is the ω-language of a system, the problem reduces to a *constrained reachability problem* in the product of the considered system with an ω-automaton characterizing the "ω-regular part" of φ. The decidability of this problem is discussed in the next sections depending on the considered classes of systems and properties.

To establish the decomposition property mentioned above, we proceed in several steps. The first one is to put CLTL$_\Diamond$ formulas into a *normal form* which is defined below. Let $\Sigma = \{a_1, \cdots, a_n\}$, and for every $i \in \{1, \cdots, n\}$, let $\pi_i = (\bigwedge_{P \in a_i} P) \wedge (\bigwedge_{P \notin a_i} \neg P)$. We say that a CLTL$_\Diamond$ formula is in *normal form* if it is a disjunction of formulas of the form

$$\begin{aligned}\widetilde{\exists}\vec{y}.\, u_0.\, [x^0_i : \pi_i]^n_{i=1}.\, (\psi_0 \wedge (\psi'_0\, \overline{\mathcal{U}}\, (f_0 \wedge \bigcirc u_1.\, [x^1_i : \pi_i]^n_{i=1}.\, (\psi_1 \wedge (\psi'_1\, \overline{\mathcal{U}}\, (f_1 \wedge \cdots \\ \cdots \bigcirc u_m.\, [x^m_i : \pi_i]^n_{i=1}.\, (\psi_m \wedge (\psi'_m\, \overline{\mathcal{U}}\, (f_m \wedge \bigcirc u_{m+1}.\, \psi_{m+1}))) \cdots)))))) \quad (5)\end{aligned}$$

where $\vec{y}$ is a vector of variables, and the ψ_j's and ψ'_j's are ALTL formulas. Then:

Proposition 5.1 *For every CLTL$_\Diamond$ closed formula φ, we can construct a closed formula in normal form φ' such that $[\![\varphi']\!] = [\![\varphi]\!]$.*

Now, observe that every formula ϕ of the form (5) imposes that it must exist a *finite* number of positions $u_0, \cdots, u_{m+1}$ where counting variables can be introduced. Moreover, there exists also a finite number of positions, situated just before the u_j's, where counting constraints (on counting variables previously introduced) must be satisfied. Actually, we can construct from ϕ another formula where all the counting variables are introduced at the first position (u_0), and all the counting constraints are checked at the last position (u_{m+1}). For that, we introduce new propositional formulas to distinguish between the different subsequences delimited by successive positions u_j and u_{j+1}. So, for every $j \in \{0, \cdots, m+1\}$, we consider a new atomic proposition at_j, and we define the formula $\lambda_j = \mathrm{at}_j \wedge \bigwedge_{k \neq j} \neg\, \mathrm{at}_k$. Let $\mathcal{P}'$ be the union of $\mathcal{P}$ with the set of the new atomic propositions, and let $\Sigma' = 2^{\mathcal{P}'}$. Then, given a formula ϕ of the form (5), let $\widehat{\phi}$ denote the formula:

$$\begin{aligned}&\widetilde{\exists}\vec{y}.\, u_0.\, [z_i^j : \pi_i \wedge \lambda_j]_{i=1..n}^{j=0..m}.\, (\psi_0 \wedge ((\psi_0' \wedge \lambda_0)\, \overline{\mathcal{U}} \cdots \\ &\quad \cdots \bigcirc u_m.\, (\psi_m \wedge ((\psi_m' \wedge \lambda_m)\, \overline{\mathcal{U}}\, ((\textstyle\bigwedge_{j=0}^{m} f_j[\sum_{\ell=k}^{j} z_i^\ell / x_i^k]_{i=1..n}^{k=0..j}) \wedge \\ &\qquad\qquad\qquad\qquad \bigcirc u_{m+1}.\, (\psi_{m+1} \wedge \Box\lambda_{m+1})))) \cdots))\end{aligned}$$

It can be seen that $[\![\phi]\!] = [\![\widehat{\phi}]\!]|_\Sigma$. Indeed, notice that the λ_j's are mutually exclusive propositional formulas, and that the counting variables z_i^j are associated with $\pi_i \wedge \lambda_j$. Hence, each variable z_i^j counts the number of occurrences of π_i exactly in the subsequence between u_j (included) and u_{j+1} (excluded). So, we can move every counting constraint f_j to the position u_{m+1} provided each counting variable x_i^k appearing in f_j, for every $k \leq j$, is substituted by the sum $\sum_{\ell=k}^{j} z_i^\ell$.

Now, it can be observed that $\widehat{\phi}$ is equivalent to the conjunction of two formulas, one of them is in ALTL, the other one is a *counting constraints eventuality* formula. Let $\widehat{\phi}^*$ be the ALTL formula:

$$u_0.\, (\psi_0 \wedge ((\psi_0' \wedge \lambda_0)\, \overline{\mathcal{U}} \cdots \bigcirc u_m.\, (\psi_m \wedge ((\psi_m' \wedge \lambda_m)\, \overline{\mathcal{U}} \bigcirc u_{m+1}.\, (\psi_{m+1} \wedge \Box\lambda_{m+1}))) \cdots)) \tag{6}$$

and $\widehat{\phi}^{\#}$ the eventuality formula:

$$[z_i^j : \pi_i \wedge \lambda_j]_{i=1..n}^{j=0..m}.\, [z : \lambda_{m+1}].\, \Diamond(\overbrace{z = 1 \wedge \exists\vec{y}.\, \bigwedge_{j=0}^{m} f_j[\sum_{\ell=k}^{j} z_i^\ell / x_i^k]_{i=1..n}^{k=0..j}}^{g}) \tag{7}$$

Notice that the global quantification $\widetilde{\exists}\vec{y}$ in $\widehat{\phi}$ has been replaced by a quantification $\exists\vec{y}$ in the Presburger formula g above.

It can be seen that $[\![\widehat{\phi}]\!] = [\![\widehat{\phi}^* \wedge \widehat{\phi}^{\#}]\!]$. This fact holds since, given a sequence that satisfies $\widehat{\phi}^*$, the positions $u_0, \ldots, u_{m+1}$ are uniquely determined by the truth of the λ_j's. Moreover, the constraint $z = 1$ ensures that the counting constraints of $\widehat{\phi}^{\#}$ are checked at u_{m+1}. Actually, we could replace the constraint $z = 1$ by $z \geq 1$ since after u_{m+1}, all the counting variables, except z, are frozen due to the fact that λ_{m+1} is continuously true.

Then, since $[\![\phi]\!] = [\![\widehat{\phi}]\!]|_\Sigma$ and $[\![\widehat{\phi}]\!] = [\![\widehat{\phi}^* \wedge \widehat{\phi}^{\#}]\!]$, we obtain the following fact:

Theorem 5.1 *Let φ be a CLTL$_\Diamond$ closed formula and let $\bigvee_{i=1}^{\ell} \phi_i$ be a formula in normal form which is equivalent to φ. Then, we have $[\![\varphi]\!] = \bigcup_{i=1}^{\ell}([\![\widehat{\phi}_i^*]\!] \cap [\![\widehat{\phi}_i^{\#}]\!])|_\Sigma$.*

By Theorem 5.1, Lemma 2.1, and the fact that projection preserves nonemptiness:

Corollary 5.1 *Let $S \subseteq \Sigma^\omega$, φ a CLTL$_\diamond$ closed formula, and $\bigvee_{i=1}^{\ell} \phi_i$ a formula in normal form equivalent to φ. Then, φ is satisfiable relatively to S if and only if $\bigcup_{i=1}^{\ell}(\widetilde{S} \cap [\![\widehat{\phi}_i^*]\!] \cap [\![\widehat{\phi}_i^\#]\!]) \neq \emptyset$.*

6 Reasoning about semilinear systems

We address the relative satisfiability problem of CLTL$_\diamond$ formulas and show the conditions of its reducibility to the emptiness problem of semilinear sets. Then we exhibit classes of semilinear systems and properties whose verification problem is (i) decidable by reduction to emptiness of semilinear sets, (ii) not reducible to emptiness of semilinear sets but still decidable, or (iii) undecidable.

First we need to introduce some notations. Let f be a Presburger formula with n free variables. We denote by $\langle\!| f |\!\rangle$ the set of valuations (vectors in $\mathbb{N}^n$) satisfying f. It is well known that for every Presburger formula f, the set $\langle\!| f |\!\rangle$ is semilinear.

Let $S \subseteq \Sigma^\omega$, φ a CLTL$_\diamond$ closed formula, and $\bigvee_{i=1}^{\ell} \phi_i$ a formula in normal form equivalent to φ (by Proposition 5.1). Then, by Corollary 5.1, we have $S \cap [\![\varphi]\!] \neq \emptyset$ iff $\exists i \in \{1, \ldots, \ell\}$, $\widetilde{S} \cap [\![\widehat{\phi}_i^*]\!] \cap [\![\widehat{\phi}_i^\#]\!] \neq \emptyset$, i.e., $\widehat{\phi}_i^\#$ is satisfiable relatively to $\widetilde{S} \cap [\![\widehat{\phi}_i^*]\!]$. Let g_i be the Presburger formula expressing the counting constraints in $\widehat{\phi}_i^\#$ (see 7). Then, $\widetilde{S} \cap [\![\widehat{\phi}_i^*]\!] \cap [\![\widehat{\phi}_i^\#]\!] \neq \emptyset$ iff there exists a finite prefix of some sequence σ in $\widetilde{S} \cap [\![\widehat{\phi}_i^*]\!]$, say $\sigma(0, j)$, whose Parikh image satisfies g_i, i.e., $[\sigma(0, j)] \in \langle\!| g_i |\!\rangle$. Hence:

$$S \cap [\![\varphi]\!] \neq \emptyset \text{ iff } \exists i \in \{1, \ldots, \ell\}.\ [Pref(\widetilde{S} \cap [\![\widehat{\phi}_i^*]\!])] \cap \langle\!| g_i |\!\rangle \neq \emptyset. \qquad (8)$$

This fact allows to reduce relative satisfiability of CLTL$_\diamond$ formulas to nonemptiness of semilinear sets provided the considered set S and formula φ are such that all the $(\widetilde{S} \cap [\![\widehat{\phi}_i^*]\!])$'s are semilinear ω-languages (semilinear sets being closed under intersection).

Now, recall that the $\widehat{\phi}_i^*$'s are ALTL formulas. Thus, all the $[\![\widehat{\phi}_i^*]\!]$'s are ω-regular languages. Moreover, it can be seen from (6) that, if φ is a PLTL$_\diamond$, the $\widehat{\phi}_i^*$'s are actually LTL formulas, and thus, the $[\![\widehat{\phi}_i^*]\!]$'s are in this case ω-star-free languages. Finally, it is easy to show that when φ is in simple-PLTL$_\diamond$, the $[\![\widehat{\phi}_i^*]\!]$'s are simple ω-regular languages. Indeed, if the ψ_i's and the ψ_i''s in (6) are propositional formulas, then roughly speaking, each set $[\![\widehat{\phi}_i^*]\!]$ corresponds to a union of ω-languages of the form $a_0 b_0^* a_1 b_1^* \cdots a_n b_n^\omega$, that are clearly simple ω-regular.

Then, since all the ω-regular languages are semilinear, and since the emptiness problem of semilinear sets is decidable, we obtain the following result.

Theorem 6.1 *Let $\mathcal{C}$ be a class of ω-languages over Σ such that: for every $\Sigma' \supseteq \Sigma$, for every $S \in \mathcal{C}$, and for every ω-regular (resp. ω-star-free, simple ω-regular) language R over Σ', $\widetilde{S} \cap R$ is semilinear. Then, the satisfiability problem of CLTL$_\diamond$ (resp. PLTL$_\diamond$, simple-PLTL$_\diamond$) closed formulas relatively to ω-languages in $\mathcal{C}$ is decidable, and consequently, the validity problem of CLTL$_\Box$ (resp. PLTL$_\Box$, simple-PLTL$_\Box$) relatively to $\mathcal{C}$ is decidable.*

We deduce from Theorem 6.1 several decidability results depending on which class of ω-languages we consider. First of all, let us address the problem of *satisfiability* and *validity* of, respectively, $\mathrm{CLTL}_\diamond$ and $\mathrm{CLTL}_\square$. Then, by taking $\mathcal{C} = \{\Sigma^\omega\}$ we obtain by Theorem 6.1:

Corollary 6.1 *The satisfiability (resp. validity) problem of $CLTL_\diamond$ (resp. $CLTL_\square$) closed formulas is decidable.*

Now, let us address the *verification problem* of semilinear systems. We start by considering the case of pushdown systems. These systems generate ω-context-free languages that are semilinear (see Lemma 2.2). The class of ω-context-free languages is clearly closed under cylindrification, and it is also closed under intersection with ω-regular languages [10]. Then, by Theorem 6.1:

Corollary 6.2 *The satisfiability (resp. validity) problem of $CLTL_\diamond$ (resp. $CLTL_\square$) closed formulas relatively to ω-context-free languages is decidable. In particular, the verification problem of pushdown systems w.r.t. $CLTL_\square$ closed formulas is decidable.*

This decidability result, however, does not hold for the whole class of semilinear systems. Indeed, we can encode the halting of a 2-counter machine as the nonemptiness of the intersection of an ω-star-free language R with a semilinear (actually a PA) ω-language S.

Proposition 6.1 *([4]) The satisfiability problem of LTL formulas relatively to PA ω-languages is undecidable. Consequently, the verification problem of PA w.r.t. LTL is undecidable.*

As a consequence of Theorem 6.1 and Proposition 6.1, the class of semilinear ω-languages is not closed under intersection with ω-star-free languages. Actually, we can give a direct proof of this fact by showing that even the intersection of a BPP ω-language with an ω-regular one can be nonsemilinear, and this holds as soon as we consider *nonsimple* ω-regular languages definable by automata having loops with two control locations [4]. This means that the verification problem of BPP processes w.r.t. $\mathrm{CLTL}_\square$ formulas cannot be reduced to the emptiness problem of semilinear sets. We show in the next section that, nevertheless, this problem is decidable (since we will show that it is decidable for Petri nets and BPP corresponds to a subclass of Petri nets). Now, if we restrict ourselves to *simple* ω-regular languages, we can prove that:

Proposition 6.2 *The class of PA ω-languages is closed under intersection with simple ω-regular languages.*

Then, using Theorem 6.1, Proposition 6.2, and the fact that the class of semilinear ω-languages is closed under cylindrification, we obtain the following result:

Corollary 6.3 *The satisfiability (resp. validity) problem of simple-$PLTL_\diamond$ (resp. simple-$PLTL_\square$) closed formulas relatively to PA ω-languages is decidable. Consequently, the verification problem of PA processes w.r.t. simple-$PLTL_\square$ closed formulas is decidable.*

7 Reasoning about Petri nets

We consider now the satisfiability problem of $\mathrm{CLTL}_{\diamond}$ closed formulas relatively to Petri nets. Recall that these systems are not semilinear (see Lemma 2.2), and thus, the problem we consider cannot be tackled as in the previous section by reduction to the emptiness problem of semilinear sets. Nevertheless, we show that it is decidable by reduction to the reachability problem in Petri nets.

Let $\mathcal{N}$ be a Petri net, φ a $\mathrm{CLTL}_{\diamond}$ closed formula, and $\bigvee_{i=1}^{\ell} \phi_i$ a formula in normal form equivalent to φ. By Corollary 5.1, we have $L(\mathcal{N}) \cap [\![\varphi]\!] \neq \emptyset$ iff $\exists i \in \{1, \ldots, \ell\}, \widetilde{L(\mathcal{N})} \cap [\![\widehat{\phi}_i^*]\!] \cap [\![\widehat{\phi}_i^{\#}]\!] \neq \emptyset$, where $\widetilde{L(\mathcal{N})}$ as well as the $\widehat{\phi}_i^*$'s and the $\widehat{\phi}_i^{\#}$ are defined over a new alphabet Σ'. Let us fix $i \in \{1, \ldots, \ell\}$ and focus on the problem $\widetilde{L(\mathcal{N})} \cap [\![\widehat{\phi}_i^*]\!] \cap [\![\widehat{\phi}_i^{\#}]\!] \neq \emptyset$.

The net $\mathcal{N}$ can be transformed straightforwardly into a net $\widetilde{\mathcal{N}}$ over Σ' such that $L(\widetilde{\mathcal{N}}) = \widetilde{L(\mathcal{N})}$. We denote by g_i the counting constraint involved in $[\![\widehat{\phi}_i^{\#}]\!]$. Then, by definition of $[\![\widehat{\phi}_i^{\#}]\!]$, we have

$$\widetilde{L(\mathcal{N})} \cap [\![\widehat{\phi}_i^*]\!] \cap [\![\widehat{\phi}_i^{\#}]\!] \neq \emptyset \text{ iff } \exists \sigma \in Pref(L(\widetilde{\mathcal{N}}) \cap [\![\widehat{\phi}_i^*]\!]).\ [\sigma] \models g_i \tag{9}$$

Recall that $\widehat{\phi}_i^*$ is an ALTL formula, and that we can effectively construct a finite-state Büchi ω-automaton $\mathcal{A} = (\mathcal{S}_{\mathcal{A}}, F)$ such that $L(\mathcal{A}) = [\![\widehat{\phi}_i^*]\!]$. Let $\widetilde{\mathcal{N}} \times \mathcal{S}_{\mathcal{A}}$ be the product net of $\widetilde{\mathcal{N}}$ and the net obtained from $\mathcal{S}_{\mathcal{A}}$ by considering each control location as a place. Moreover, let $\mathbf{T}_{\infty}$ be the set of transitions in $\widetilde{\mathcal{N}} \times \mathcal{S}_{\mathcal{A}}$ that involve a transition of $\mathcal{S}_{\mathcal{A}}$ having as target some repeating location in F. Then, $L(\widetilde{\mathcal{N}}) \cap [\![\widehat{\phi}_i^*]\!]$ is the set of infinite sequences over Σ' generated by sequences of transitions in $\widetilde{\mathcal{N}} \times \mathcal{S}_{\mathcal{A}}$ including infinitely often transitions in $\mathbf{T}_{\infty}$. Thus, $Pref(L(\widetilde{\mathcal{N}}) \cap [\![\widehat{\phi}_i^*]\!])$ is the set of finite sequences over Σ' generated by sequences of transitions in $\widetilde{\mathcal{N}} \times \mathcal{S}_{\mathcal{A}}$ reaching markings from which there are infinite sequences of transitions including infinitely often transitions in $\mathbf{T}_{\infty}$. The set of such markings is actually semilinear and effectively constructible using the result proved in [18]:

Lemma 7.1 *Let $\mathcal{N}$ be a Petri net, and t one of its transitions. Then, the set $\mathcal{M}_{\infty}(\mathcal{N}, t)$ is semilinear and can be effectively constructed.*

Let us denote by $\mathcal{L}_{\infty}$ the semilinear set $\bigcup_{t \in \mathbf{T}_{\infty}} \mathcal{M}_{\infty}(\widetilde{\mathcal{N}} \times \mathcal{S}_{\mathcal{A}}, t)$. Then, by (9):

$$\widetilde{L(\mathcal{N})} \cap [\![\widehat{\phi}_i^*]\!] \cap [\![\widehat{\phi}_i^{\#}]\!] \neq \emptyset \text{ iff } \exists \sigma \in (\Sigma')^*.\ \exists M \in \mathcal{L}_{\infty}.\ M_{\widetilde{\mathcal{N}} \times \mathcal{S}_{\mathcal{A}}} \xrightarrow{\sigma} M \text{ and } [\sigma] \models g_i \tag{10}$$

To deal with counting constraints, we extend the net $\widetilde{\mathcal{N}} \times \mathcal{S}_{\mathcal{A}}$ by new places encoding the counting variables in g_i. Each such a place is associated with some label in Σ', say a, and counts the number of times the transitions of $\widetilde{\mathcal{N}} \times \mathcal{S}_{\mathcal{A}}$ labelled by a are fired. Let $\mathbf{P}^{\#}$ be the set of the new places, and $\widetilde{\mathcal{N}}_{\mathcal{A}}^{\#}$ the net resulting from this extension. Clearly, $\langle\!| g_i |\!\rangle$ is a semilinear subset of $\mathbb{N}^d$ where $d = |\mathbf{P}^{\#}|$. Let $\langle\!| g_i |\!\rangle'$ (resp. $\mathcal{L}'_{\infty}$) be the set of all markings of $\widetilde{\mathcal{N}}_{\mathcal{A}}^{\#}$ whose projections on $\mathbf{P}^{\#}$ (resp. the places of $\widetilde{\mathcal{N}} \times \mathcal{S}_{\mathcal{A}}$) are in $\langle\!| g_i |\!\rangle$ (resp. $\mathcal{L}_{\infty}$). The sets $\langle\!| g_i |\!\rangle'$ and $\mathcal{L}'_{\infty}$

are semilinear and can be constructed easily from the original ones. Moreover, the class of semilinear sets being closed under intersection, the set $\mathcal{L}'_\infty \cap \langle\!| g_i |\!\rangle'$ is also semilinear. Then, we obtain from (10) the following fact:

$$\widetilde{L(\mathcal{N})} \cap [\![\widehat{\phi}_i^*]\!] \cap [\![\widehat{\phi}_i^\#]\!] \neq \emptyset \text{ iff } \exists \sigma \in (\Sigma')^*.\ \exists M \in (\mathcal{L}'_\infty \cap \langle\!| g_i |\!\rangle').\ M_{\widetilde{\mathcal{N}}_{\mathcal{A}}^\#} \xrightarrow{\sigma} M \quad (11)$$

We have reduced the nonemptiness problem we are interested in to the reachability problem of a semilinear set in Petri nets, i.e., whether there is a reachable marking in some given semilinear set of markings. We can prove that this problem is reducible to the reachability problem in Petri nets (i.e., whether a given marking is reachable), which is decidable [15]: First of all, notice that given two nets $\mathcal{N}_1$ and $\mathcal{N}_2$ with the same number of places, the problem whether $\mathcal{R}(\mathcal{N}_1) \cap \mathcal{R}(\mathcal{N}_2) \neq \emptyset$ is reducible to the reachability problem in Petri nets. Indeed, we can add to $\mathcal{N}_1$ and $\mathcal{N}_2$ transitions clearing simultaneously, token by token, places in both nets with the same index, and then, $\mathcal{R}(\mathcal{N}_1) \cap \mathcal{R}(\mathcal{N}_2) \neq \emptyset$ iff the empty marking is reachable. Now, given a *linear* set $\mathcal{L}$, we can construct easily a net $\mathcal{N}_\mathcal{L}$ such that $\mathcal{R}(\mathcal{N}_\mathcal{L}) = \mathcal{L}$. Then, assuming that $\mathcal{L}$ is a set of markings of some net $\mathcal{N}$, by the remark above, the problem $\mathcal{R}(\mathcal{N}) \cap \mathcal{L} \neq \emptyset$ is reducible to the reachability problem in Petri nets. Obviously, this can be generalized to semilinear sets.

Lemma 7.2 *Let $\mathcal{N}$ be a Petri net, and $\mathcal{L}$ a semilinear set of markings of $\mathcal{N}$. Then, the problem $\mathcal{R}(\mathcal{N}) \cap \mathcal{L} \neq \emptyset$ is decidable.*

Then, by (11) and Lemma 7.2, we deduce the following result:

Theorem 7.1 *The satisfiability problem of CLTL$_\Diamond$ closed formulas relatively to Petri nets is decidable. Consequently, the verification problem of Petri nets w.r.t. CLTL$_\Box$ closed formulas is decidable.*

8 Conclusion

We have addressed the verification problem of nonregular properties for two incomparable classes of infinite state systems: semilinear systems including pushdown systems and PA processes, and Petri nets.

Our main results are that the verification problems of pushdown systems as well as Petri nets w.r.t. CLTL$_\Box$ formulas are decidable. This logic is strictly more expressive than the linear-time μ-calculus and allows the expression of nonregular properties like constrained safety properties.

$\models$	LTL	ALTL	PLTL	CLTL	simple-PLTL$_\Box$	PLTL$_\Box$	CLTL$_\Box$
Finite-state LTS	yes	yes	no	no	yes	yes	yes
Pushdown syst.	yes	yes	no	no	yes	yes	yes
PA processes	no	no	no	no	yes	no	no
Petri nets	yes	yes	no	no	yes	yes	yes

Table 2. Decidability of the verification problem

To establish these results, we have reduced the relative satisfiability problem of CLTL$_\Diamond$ (the dual fragment of CLTL$_\Box$) to a constrained emptiness problem.

Then, we have tackled the latter problem either by reduction to the emptiness problem of semilinear sets or by reduction to the reachability problem of Petri nets. We have shown also that the verification problem of semilinear systems is in general undecidable for LTL (ω-star-free properties). Nevertheless, we have shown that this problem is decidable for all PA processes and simple-PLTL$_{\Box}$ which allows to express significant nonregular properties (like (1) and (3)). We summarize these results in Table 2.

References

1. P. Abdulla and B. Jonsson. Verifying Programs with Unreliable Channels. In *LICS'93*. IEEE, 1993.
2. J. Baeten, J.A. Bergstra, and J.W. Klop. Decidability of Bisimulation Equivalence for Processes Generating Context-Free Languages. T.R. CS-R8632, 1987. CWI.
3. J.A. Bergstra and J.W. Klop. Process Theory based on Bisimulation Semantics. In *REX School/Workshop on Linear Time, Branching Time and Partial Order in Logics and Models for Concurrency*, 1988. LNCS 354.
4. A. Bouajjani, R. Echahed, and P. Habermehl. On the Verification Problem of Nonregular Properties for Nonregular Processes. In *LICS'95*. IEEE, 1995.
5. A. Bouajjani, R. Echahed, and P. Habermehl. Verifying Infinite State Processes with Sequential and Parallel Composition. In *POPL'95*. ACM, 1995.
6. O. Burkart and B. Steffen. Pushdown Processes: Parallel Composition and Model Checking. In *CONCUR'94*, 1994. LNCS 836.
7. S. Christensen. *Decidability and Decomposition in Process Algebra.* PhD thesis, University of Edinburgh, 1993.
8. S. Christensen and H. Hüttel. Decidability Issues for Infinite State Processes - A Survey. *Bull. of the EATCS*, 51, 1993.
9. S. Christensen, H. Hüttel, and C. Stirling. Bisimulation Equivalence is Decidable for all Context-Free Processes. *Information and Computation*, 121, 1995.
10. R.S. Cohen and A.Y. Gold. Theory of ω-Languages. I: Characterizations of ω-Context-Free Languages. *J.C.S.S.*, 15, 1977.
11. J. Esparza and A. Kiehn. On the Model Checking Problem for Branching Time Logics and Basic Parallel Processes. In *CAV'95*. LNCS 939, 1995.
12. Javier Esparza. On the Decidability of Model-Checking for Several Mu-calculi and Petri Nets. In *CAAP'94*. LNCS 787, 1994.
13. R. Howell, L. Rosier, and H.C. Yen. A Taxonomy of Fairness and Temporal Logic Problems for Petri Nets. *T.C.S.*, 82, 1991.
14. P. Jancar. Decidability of a Temp. Logic Problem for Petri Nets. *T.C.S.*, 74, 1990.
15. E. Mayr. An Algorithm for the General Petri Net Reachability Problem. *SIAM J. on Comput.*, 13, 1984.
16. A. Pnueli. The Temporal Logic of Programs. In *FOCS'77*. IEEE, 1977.
17. W. Thomas. Star-Free Regular Sets of ω-Sequences. *Inform. and Cont.*, 42, 1979.
18. R. Valk and M. Jantzen. The Residue of Vector Sets with Applications to Decidability Problems in Petri Nets. *Acta Informatica*, 21, 1985.
19. M.Y. Vardi and P. Wolper. An Automata-Theoretic Approach to Automatic Program Verification. In *LICS'86*. IEEE, 1986.

This article was processed using the LaTeX macro package with LLNCS style