Verification of Nonregular Temporal Properties for Context-Free Processes*

Ahmed Bouajjani** † Rachid Echahed*** † Riadh Robbana** †

Abstract. We address the problem of the specification and the verification of processes with infinite-state spaces. Many relevant properties for such processes involve constraints on numbers of occurrences of events (truth of propositions). These properties are nonregular and hence, they are not expressible neither in the usual logics of processes nor by finite-state ω-automata. We propose a logic called PCTL that allows the description of such properties. PCTL is a combination of the branching-time temporal logic CTL with Presburger arithmetic. Mainly, we study the decidability of the satisfaction relation between *context-free processes* and PCTL formulas. We show that this relation is decidable for a large fragment of PCTL. Furthermore, we study the satisfiability problem for PCTL. We show that this problem is highly undecidable (Σ_1^1-complete), even for the fragment where the satisfaction relation is decidable, and exhibit a nontrivial fragment where the satisfiability problem is decidable.

1 Introduction

The logical framework for the specification and verification of processes has been extensively developed during the last decade. Logics of processes, including temporal logics [Pnu77, GPSS80, Wol83, CES83, EH83], dynamic logics [FL79, Str82] and fixpoint calculi [Pra81, Koz83, Var88], have been proposed as specification formalisms. Important efforts have been devoted to the study of the expressiveness of these logics as well as their decision problems. There are two decision problems concerning logics of processes that are addressed in these works. The first one is the *satisfaction problem* which consists in deciding whether some given process, modeled by a Kripke structure, satisfies some given specification expressed by a formula. The second one is the *satisfiability problem* which consists in deciding whether some given formula is satisfiable, i.e., there exists some Kripke structure that satisfies the considered formula.

The majority of the works done in this area consider propositional logics that express *regular* properties of processes, i.e., properties that correspond to sets of infinite sequences or trees (according to the underlying semantics of the logic) that are definable by finite-state ω-automata [Buc62, Rab69]. Several works have established the links between logics of processes and finite-state automata [Str82, SE84, VW86, Tho87, Niw88, Var88]. These works show in particular that the *regular* logics of processes are expressive enough for the specification of *finite-state* processes (modeled by finite-state Kripke structures). Moreover, the decidability of the satisfiability problem for these logics has been shown

* Partially supported by the ESPRIT-BRA project REACT.
** VERIMAG-SPECTRE, Miniparc-Zirst, Rue Lavoisier, 38330 Montbonnot St-Martin, France.
*** LGI-IMAG, BP53, 38041 Grenoble cedex, France.
† Ahmed.Bouajjani@imag.fr, Rachid.Echahed@imag.fr, Riadh.Robbana@imag.fr

[GPSS80], [Wol83], [Str82], [SE84], and their satisfaction problem has been widely investigated in the case of finite-state processes, leading to automatic verification techniques as model-checking [QS82], [CES83], [VW86], [EL86].

Recently, intensive investigations have been consecrated to the extension of the specification and verification methods successfully used for finite-state processes, in order to deal with processes having infinite state spaces. One of the most important directions of these investigations concerns processes that generate *context-free* sets of computation sequences [BBK87]. Important results have been obtained concerning the comparison between these processes w.r.t. behavioural equivalences [BBK87, GH91, CHS92]. Mainly, it has been shown that bisimulation equivalence is decidable for all context-free processes [CHS92]. However, very few results have been obtained concerning the extension of the logical framework to the specification and verification of context-free processes. As far as we know, the existing results in this topic concern the extension of the model-checking technique of the μ-calculus to context-free processes [BS92]. So, this work, even it is a nice and interesting extension of the existing verification methods, it allows unfortunatly to verify only the regular properties of context-free processes whereas a wide class of the relevant properties of such processes are nonregular. For instance, in the specification of a communication protocol, we may require that

1. *between the beginning and the ending of every session, there are exactly the same numbers of requests and acknowledgements.*
2. *during every session, the number of acknowledgements never exceeds the number of requests.*

Actually, as these examples show, significant properties of context-free processes are essentially temporal properties involving constraints on *numbers of occurrences* of some events (or numbers of states satisfying some state property). For this reason, we propose in this paper a new temporal logic that allows to express such properties. This logic, called PCTL (for Presburger Computation Tree Logic) is a combination of Presburger arithmetic with the branching-time temporal logic CTL [CES83]. In PCTL, we dispose of *occurrence variables* that can be associated with state formulas and then used to express constraints on the number of occurrences of states satisfying these formulas. The constraints are expressed in the language of Presburger arithmetic. For instance, in the formula $[x : \pi].\varphi$, we associate the state formula π with the variable x. Then, x counts the number of occurrences of π along each computation sequence starting from the current state. Using this notation, the properties informally described above can be expressed in PCTL by:

1. $\forall\Box\,(\text{BEGIN} \Rightarrow [x, y : \text{REQ}, \text{ACK}].\, \forall\Box\,(\text{END} \Rightarrow (x = y)))$
2. $\forall\Box\,(\text{BEGIN} \Rightarrow [x, y : \text{REQ}, \text{ACK}].\, (x \geq y)\forall\mathcal{U}\text{END})$

Then, it can be observed that PCTL allows to characterize a large class of nonregular languages. These languages can be context-free as in the examples above, but also non context-free (context-sensitive). The existing logics that can express nonregular properties are extensions of the propositional dynamic logic with nonregular programs [HPS83]. However, concerning these logics, the works that have been done address only the satisfiability problem and never consider the satisfaction problem nor the verification topic. Moreover, most of the presented results for these logics are negative (high undecidability results) [HPS83, HP84] and the positive ones are somewhat restrictive [KP83, HR90].

Our aim in this paper is to present a study of the two decision problems concerning PCTL: mainly, its satisfaction problem for context-free processes in order to provide an automatic verification method for these processes, and also its satisfiability (and dually validity) problem.

First, we show that the satisfaction problem is undecidable for the full PCTL and even non recursively enumerable. Actually, this undecidability result is not due to the fact that we are dealing with context-free processes but rather to the expressive power of PCTL. However, we show that surprisingly, by a slight syntactic restriction, we get a fragment of PCTL where the satisfaction problem for context-free processes becomes decidable. This fragment, called PCTL$^+$, contains the most significant nonregular PCTL properties, as for instance the properties (1) and (2) given above. Our decision procedure is based on a reduction of the satisfaction problem, given a context-free process and a PCTL$^+$ formula, to the validity problem of Presburger formulas. At our knowledge, this is the first result that allows to verify automatically nonregular properties for context-free processes.

On the other hand, we show that the satisfiability problem for PCTL and even for PCTL$^+$ is highly undecidable, more precisely Σ_1^1-complete, and then, the validity problem for these logics is Π_1^1-complete. Nevertheless, we exhibit a nontrivial fragment of PCTL$^+$, containing for instance the properties (1) and (2) above, where the validity problem is decidable.

The remainder of this paper is organized as follows: In Section 2, we recall some basic definitions and results and introduce some notations. In Section 3 we define the context-free processes. The logic PCTL is defined in Section 4. The satisfaction problem for PCTL and the fragment PCTL$^+$ is considered in Section 5, and Section 6 is dedicated to the satisfiability problem for PCTL and its fragments. Finally, concluding remarks are given in Section 7. For lack of space, all the proofs are omitted here and given in the full paper.

2 Preliminaries

We recall in this section some well-known notions that are necessary for the understanding of the paper and introduce some notations.

2.1 Presburger arithmetic

Presburger arithmetic is the first order logic of integers with addition, subtraction and the usual ordering. Let us recall briefly the definition of this logic.

Let $\mathcal{V}$ be a set of variables. We use $x, y, \ldots$ to range over variables in $\mathcal{V}$. Consider the set of terms defined by

$$t ::= 0 \mid 1 \mid x \mid t - t \mid t + t$$

Integer constants ($k \in \mathbb{Z}$) and multiplication by constants (kt) can be introduced as abbreviations. The set of Presburger formulas is defined by

$$f ::= t \leq t \mid \neg f \mid f \vee f \mid \exists x.\, f$$

Classical abbreviations can be used as the boolean connectives as conjunction ($\wedge$), implication ($\Rightarrow$) and equivalence ($\Leftrightarrow$) as well as the universal quantification ($\forall$). The semantics of these formulas is defined in the standard way. Given a formula f with free variables $x_1, \ldots, x_n$, and a valuation $E : \mathcal{V} \rightarrow \mathbb{Z}$, we denote by $||f||(E)$ the truth value of f for the valuation E. We say that a formula is satisfiable if there exists some valuation E such that $||f||(E)$ is true. It is well known that the satisfiability problem for Presburger formulas is decidable (e.g., see [BJ74] for a decision procedure).

2.2 Sequences, languages and grammars

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

Given a sequence $\sigma \in \Sigma^\infty$, $|\sigma| \in \{0, 1, \ldots, \omega\}$ denotes the length of σ. Let ϵ denotes the empty sequence, i.e., the sequence of length 0. Let $\Sigma^+ = \Sigma - \{\epsilon\}$. For every $a \in \Sigma$, $|\sigma|_a$ is the number of occurrences of a in σ. In the sequel, we write $a \in \sigma$ to denote the fact that a appears in the sequence σ. For every $i \in \mathbb{N}$ such that $i \leq |\sigma|$, $\sigma(i)$ is the i^{th} element of σ.

A context-free grammar (CFG) over Σ in Greibach normal form (GNF) is a tuple $G = (\Sigma, N, \mathit{Prod}, Z)$ where N is a set of nonterminals, *Prod* is a set of productions of the form $A \rightarrow a \cdot \alpha$ where $a \in \Sigma$ and $\alpha \in N^*$, and Z is the starting symbol. Elements of Σ are sometimes called terminal symbols. Given a production $p \in \mathit{Prod}$, we denote by $\mathit{lhs}(p)$ the left hand side of p and by $\mathit{rhs}(p)$ its right hand side. We adopt standard notations for the derivation relation ($\Longrightarrow$) and its reflexive-transitive closure ($\stackrel{*}{\Longrightarrow}$). We use subscripts to indicate if necessary the set of productions or the sequences of productions used in the derivation. We denote by $L(G)$ the language generated by the grammar G (i.e., the set of sequences $\sigma \in \Sigma^*$ such that $Z \stackrel{*}{\Longrightarrow} \sigma$). For more details concerning the theory of formal languages, see for instance [Har78].

2.3 Kripke strutures

A Kripke structure over the alphabet Σ (KS for short) is a tuple $K = (\Sigma, S, \Pi, R)$ where S is a countable set of states, $\Pi : S \rightarrow \Sigma$ is a labelling function and $R \subseteq S \times S$ is a transition relation.

We write $s \rightarrow_R s'$ to denote the fact that $(s, s') \in R$. We write $s \not\rightarrow_R$ when there is no state s' such that $(s, s') \in R$. An *infinite computation sequence* of K from a state s is a sequence $s_1 s_2 \cdots$ such that $s = s_1$ and $\forall i \geq 1.\ s_i \rightarrow_R s_{i+1}$. A *finite computation sequence* of K starting from s is a sequence $s_1 \cdots s_n$ such that $s = s_1$, $\forall i.\ 1 \leq i < n.\ s_i \rightarrow_R s_{i+1}$, and $s_n \not\rightarrow_R$. We denote by $\mathcal{C}(K, s)$ the set of finite and infinite computation sequences of K starting from s. We say that K is *finite-branching* if for each state $s \in S$, the set $\{s' : s \rightarrow_R s'\}$ is finite.

3 Context-Free Processes

The definition we adopt for context-free processes is very close to the definition of BPA (Basic Process Algebra) processes [BBK87]. The difference between the two definitions is that the operational semantics of BPA processes is given by means of edge-labelled graphs (i.e., labelled transition systems) whereas the semantics of our context-free processes is defined using state-labelled graphs (i.e., Kripke structures). We choose this semantics because our aim is to consider context-free processes as models for the temporal logic PCTL introduced in the next section which is interpreted on KS's.

Let *Prop* be a finite set of atomic propositions. We consider from now on that the alphabet Σ is 2^{Prop}. We call the elements of Σ *state labels*. We use $P, Q, \ldots$ to denote atomic propositions and we use the letters $a, b, \ldots$ to range over elements of Σ. Let *Var* be a set of variables. We use $A, B, \ldots$ to range over variables in *Var* and greek letters $\alpha, \beta, \ldots$ to range over sequences in Var^*. Then, consider the set of terms $\mathcal{T}$ defined by the following grammar:

$$t ::= a \mid A \mid t + t \mid t \cdot t \mid \epsilon$$

Intuitively, the operator "+" stands for nondeterministic choice whereas "$\cdot$" is the sequential composition operator ; ϵ (the empty sequence) represents the idle process. In the sequel, we identify the terms $\epsilon \cdot t$ and $t \cdot \epsilon$ with the term t, for any term t.

Syntactically, a context-free process (CFP for short) is defined by a finite system of equations $\Delta \stackrel{\text{def}}{=} \{A_i = t_i \ : \ 1 \leq i \leq n\}$ where all the A_i's are distinct variables and all the variables occurring in the terms t_i are in the (finite) set $Var_\Delta = \{A_1, \ldots, A_n\}$.

A term $t \in \mathcal{T}$ is *guarded* if every variable occurrence in t is within the scope of a state label $a \in \Sigma$. A CFP $\Delta = \{A_i = t_i \ : \ 1 \leq i \leq n\}$ is guarded (GCFP) if every term t_i is guarded.

We define the operational semantics of CFP's by associating with each system Δ a Kripke structure $\mathcal{K}_\Delta$ representing its computation graph. This structure is given by $\mathcal{K}_\Delta = (\Sigma, S_\Delta, \Pi_\Delta, R_\Delta)$ where

- $S_\Delta = \Sigma \times \mathcal{T}$,
- $\forall \langle a, t\rangle \in S_\Delta .\ \Pi_\Delta(\langle a, t\rangle) = a$,
- $R_\Delta \subseteq S_\Delta^2$ is the smallest relation such that
  1. $\langle a, a'\rangle \rightarrow_{R_\Delta} \langle a', \epsilon\rangle$,
  2. "$A = t$" $\in \Delta$ and $\langle a, t\rangle \rightarrow_{R_\Delta} \langle a', t'\rangle$ implies $\langle a, A\rangle \rightarrow_{R_\Delta} \langle a', t'\rangle$,
  3. $\langle a, t_1\rangle \rightarrow_{R_\Delta} \langle a', t_1'\rangle$ implies $\langle a, t_1 + t_2\rangle \rightarrow_{R_\Delta} \langle a', t_1'\rangle$,
  4. $\langle a, t_1\rangle \rightarrow_{R_\Delta} \langle a', t_1'\rangle$ implies $\langle a, t_2 + t_1\rangle \rightarrow_{R_\Delta} \langle a', t_1'\rangle$,
  5. $\langle a, t_1\rangle \rightarrow_{R_\Delta} \langle a', t_1'\rangle$ implies $\langle a, t_1 \cdot t_2\rangle \rightarrow_{R_\Delta} \langle a', t_1' \cdot t_2\rangle$.

Clearly, for any variable $A \in Var_\Delta$ and any $a \in \Sigma$, the set of reachable states from $\langle a, A\rangle$ is in general infinite.

Kripke structures such as $\mathcal{K}_\Delta$ (i.e., defined as above from CFP's), are called context-free Kripke structures (CFKS's).

A GCFP $\Delta = \{A_i = t_i \ : \ 1 \leq i \leq n\}$ is in Greibach normal form (GNF), if every term t_i is either ϵ or in the form $\sum_{j=1}^{n_i} a_j^i \alpha_j^i$ where $n_i \geq 1$, the α_j^i's are sequences in Var_Δ^* and, for every A in the α_j^i's, "$A = \epsilon$" $\notin \Delta$. Then, we denote by $\mathcal{S}(\Delta)$ the set of *transition rules* $A_i \mapsto t$ where, either $A_i = \epsilon \in \Delta$ and $t = \epsilon$, or $A_i = \sum_{j=1}^{n_i} a_j^i \alpha_j^i$ and $t = a_j^i \alpha_j^i$ for some $j \in \{1, \cdots, n_i\}$.

It has been shown (see [BBK87]) that every GCFP can be transformed into GNF preserving bisimilarity. Notice that, for every GCFP, the structure $\mathcal{K}_\Delta$ is finite-branching.

4 Presburger CTL

The logic Presburger CTL (PCTL) is an extension of the branching-time temporal logic CTL [CES83] where constraints on numbers of occurrences of state properties can be expressed using Presburger formulas.

Recall that *Prop* is a finite set of atomic proposition and that $\Sigma = 2^{Prop}$. Recall also that we use letters $P, Q, \ldots$ to range over elements of *Prop*, letters $x, y, \ldots$ to range over variables in $\mathcal{V}$ and $f, g, \ldots$ to range over Presburger formulas. First, consider the set of *state formulas* given by:

$$\pi ::= P \mid \neg\pi \mid \pi \vee \pi$$

The set of formulas of PCTL is defined by:

$$\varphi ::= P \mid f \mid \neg\varphi \mid \varphi \vee \varphi \mid \tilde{\exists} x.\varphi \mid [x : \pi].\varphi \mid \varphi \forall\mathcal{U} \varphi \mid \varphi \exists\mathcal{U} \varphi$$

We consider as abbreviations the usual boolean connectives as conjunction ($\wedge$), implication ($\Rightarrow$) and equivalence ($\Leftrightarrow$). In addition, we use the universal quantification $\tilde{\forall} x.\varphi = \neg\tilde{\exists} x.\neg\varphi$ and the following standard abbreviations: $\forall\Diamond\varphi = \mathsf{true}\forall\mathcal{U}\varphi$, $\exists\Diamond\varphi = \mathsf{true}\exists\mathcal{U}\varphi$, $\forall\Box\varphi = \neg\exists\Diamond\neg\varphi$ and $\exists\Box\varphi = \neg\forall\Diamond\neg\varphi$. We write $[x_1, \ldots, x_n : \pi_1, \ldots, \pi_n].\varphi$ or $[x_i : \pi_i]_{i=1}^n.\varphi$ for $[x_1 : \pi_1]. \cdots .[x_n : \pi_n].\varphi$.

The operators $\exists\mathcal{U}$ and $\forall\mathcal{U}$ are the classical CTL *until* operators with existential and universal path quantification. The Presburger formulas f are used to express constraints on the numbers of occurrences of states satisfying some state formulas. Then, we call these formulas *occurrence constraints*. The operator $\tilde{\exists}$ corresponds to the usual existential quantification over integers. We distinguish (even syntactically) between the PCTL operator $\tilde{\exists}$ and the Presburger existential quantifier $\exists$ that may be used locally in some occurrence constraint f. In the formula $[x : \pi].\varphi$, the variable x is associated with the state formula π, and then, starting from the current state, x represents the number of occurrences of states satisfying π. The variable x can be used in the occurrence constraints appearing in φ. For instance, the formula $[x : \pi].\exists\Diamond(P \wedge x \leq 5)$ expresses the fact that from the current state, say s, there exists some reachable state s' where P holds and such that the path relating s to s' contains less than 5 states satisfying π. From now on, we refer to the variables x as *occurrence variables.* The construction "$[x : \pi]$" in the formula above binds the variable x in the subformula $\exists\Diamond(P \wedge x \leq 5)$. So, a variable x may be bound by either the quantifier $\exists$, or by the quantifier $\tilde{\exists}$, or by the construction "$[x : \pi]$". Then, every variable appearing in some formula is either *bound* or *free*. We denote by $\mathcal{F}(\varphi)$ the set of variables occurring free in φ. A formula φ is *closed* if all the variables occurring in it are bound (i.e., $\mathcal{F}(\varphi) = \emptyset$), otherwise φ is *open*. We assume without loss of generality that each variable occurring in any PCTL formula is bound at most once.

The formal semantics of PCTL is defined by a satisfaction relation between the states of a KS over Σ and the formulas. First, let us define a satisfaction relation for state formulas. Let K be a KS over Σ. The satisfaction relation $\models$ for state formulas is defined for any state s and atomic proposition P by $s \models P$ iff $P \in \Pi(s)$, and extended straightforwardly to boolean combinations of atomic propositions. Now, let us consider the general case. Since the formulas may be open, the satisfaction relation is defined w.r.t. a valuation E of the variables. Along a computation sequence, the valuation changes according to the satisfaction, at the visited states, of the state formulas associated with the occurrence variables. We define a *state formulas association* as a function γ that associates state formulas with variables in $\mathcal{V}$.

For any function F from $\mathcal{V}$ to some target set T (F stands for either a valuation E or a state formula γ), we denote by $\mathcal{D}(F)$ the set of variables x such that $F(x)$ is defined. We denote also by $F[x \leftarrow \tau]$ where $\tau \in T$, the function F' such that $\mathcal{D}(F') = \mathcal{D}(F) \cup \{x\}$ and which associates τ with x and coincides with F on all the other variables.

Given a state formulas association γ and a valuation E, we define, for every sequence $\sigma \in S^\infty$ and every two ranks $i, j \in I\!N$ such that $i, j \leq |\sigma|$, the valuation

$$E^{[i,j]}_{(\sigma,\gamma)} = E[x \leftarrow E(x) + |\{k \in \{i, \ldots, j\} \ : \ \sigma(k) \models \gamma(x)\}|]_{x \in \mathcal{D}(\gamma)}.$$

Now, given a state formulas association γ and a valuation E, the satisfaction relation $\models$ for all PCTL formulas is inductively defined for any state s by:

$$\begin{array}{ll}
s \models_{(E,\gamma)} P & \text{iff } P \in \Pi(s) \\
s \models_{(E,\gamma)} f & \text{iff } ||f||(E') = \mathsf{true} \text{ where} \\
 & \quad E' = E[x \leftarrow E(x) + \text{if } s \models \gamma(x) \text{ then } 1 \text{ else } 0]_{x \in \mathcal{D}(\gamma)} \\
s \models_{(E,\gamma)} \neg\varphi & \text{iff } s \not\models_{(E,\gamma)} \varphi \\
s \models_{(E,\gamma)} \varphi_1 \vee \varphi_2 & \text{iff } s \models_{(E,\gamma)} \varphi_1 \text{ or } s \models_{(E,\gamma)} \varphi_2 \\
s \models_{(E,\gamma)} \tilde{\exists} x.\varphi & \text{iff } \exists k \in \mathbb{Z}.\ s \models_{(E',\gamma)} \varphi \text{ where } E' = E[x \leftarrow k] \\
s \models_{(E,\gamma)} [x : \pi].\varphi & \text{iff } s \models_{(E',\gamma')} \varphi \ \text{ where } \ E' = E[x \leftarrow 0] \text{ and } \gamma' = \gamma[x \leftarrow \pi] \\
s \models_{(E,\gamma)} \varphi_1 \forall\mathcal{U} \varphi_2 & \text{iff } \forall \sigma \in \mathcal{C}(K, s). \\
 & \quad \exists i \in I\!N.\ 1 \leq i \leq |\sigma|.\ \sigma(i) \models_{(E'(i),\gamma)} \varphi_2 \text{ and} \\
 & \quad \forall j \in I\!N.\ 1 \leq j < i.\ \sigma(j) \models_{(E'(j),\gamma)} \varphi_1, \ \text{ where } E'(k) = E^{[1,k-1]}_{(\sigma,\gamma)} \\
s \models_{(E,\gamma)} \varphi_1 \exists\mathcal{U} \varphi_2 & \text{iff } \exists \sigma \in \mathcal{C}(K, s). \\
 & \quad \exists i \in I\!N.\ 1 \leq i \leq |\sigma|.\ \sigma(i) \models_{(E'(i),\gamma)} \varphi_2 \text{ and} \\
 & \quad \forall j \in I\!N.\ 1 \leq j < i.\ \sigma(j) \models_{(E'(j),\gamma)} \varphi_1, \ \text{ where } E'(k) = E^{[1,k-1]}_{(\sigma,\gamma)}
\end{array}$$

The CTL operators $\exists\bigcirc$ (there is some successor state) and $\forall\bigcirc$ (all the successors) can be defined in PCTL by: $\exists\bigcirc\varphi = [x : \mathsf{true}].\exists\Diamond((x = 2) \wedge \varphi)$ and $\forall\bigcirc\varphi = \neg\exists\bigcirc\neg\varphi$.

Then, clearly PCTL subsumes the logic CTL. Moreover, it can express properties that can not be expressed in the usual propositional temporal logics [GPSS80, Wol83, EH83], dynamic logics [FL79, Str82] and fixpoint calculi [Koz83, Var88]. Indeed, these logics can express only regular properties, i.e., properties that can be defined by finite-state automata (on infinite trees or sequences) [VW86, Tho87, Niw88] whereas PCTL can express nonregular properties.

For example, we can express the fact that between the beginning and the ending of some communication protocol session, there are exactly the same numbers of requests and acknowledgements. This is done by the formula:

$$\forall\Box\ (\textsc{begin} \Rightarrow [x, y : \textsc{req}, \textsc{ack}].\ \forall\Box\ (\textsc{end} \Rightarrow (x = y))) \tag{1}$$

We can require in addition that during every such session, the number of acknowledgements never exceeds the number of requests. This is done by:

$$\forall\Box\ (\textsc{begin} \Rightarrow [x, y : \textsc{req}, \textsc{ack}].\ (x \geq y)\forall\mathcal{U}\textsc{end}) \tag{2}$$

The conjunction of the two formulas (1) and (2) expresses the fact that, in every computation sequence, the subsequence between any pair of consecutive BEGIN and END is in the language of well-balanced parentheses (semi-Dyck set) with REQ (resp. ACK) as a left (resp. right) parenthesis. Now, we can express the stronger property that every subsequence between two consecutive BEGIN and END is actually in the language

$\{\text{REQ}^n \cdot \text{ACK}^n \ : \ n \geq 1\}$. This is done by the formula:

$$\widetilde{\forall} n.\ \forall\Box\ (\text{BEGIN} \Rightarrow$$
$$[x, y, z : \text{REQ}, \text{ACK}, \text{END}].$$
$$\forall\Box\ ((\text{ACK} \wedge (x = n) \wedge (y = 1)) \Rightarrow \forall\Box\ ((\text{END} \wedge (z = 1)) \Rightarrow (x = n) \wedge (y = n)))) \quad (3)$$

Then, as the examples above show, we can characterize in PCTL a large class of nonregular languages. These languages can be context-free as in (1), (2) and (3), but also context-sensitive using a conjunction of more than two occurrence constraints concerning different sets of occurrence variables. For instance, we can consider languages as $\{\pi_1^n \cdots \pi_k^n \ : \ n \geq 1\}$ or $\{\pi_1^n \cdot \pi_2^m \cdot \pi_1^n \cdot \pi_2^m \ : \ n, m \geq 1\}$.

In the formulas (1), (2) and (3), the constraints concern the numbers of occurrences of some propositions in every fixed computation sequence, independently from the other sequences. Actually, we can express also in PCTL properties involving global constraints on the whole set of computation sequences. For instance, consider the *uniform inevitability* property that says: there exists some rank n such that every computation sequence (of length greater than n) satisfies some proposition P at rank n. This property has been shown in [Eme87] to be non expressible by finite-state infinite-tree automata. We can express this property in PCTL by:

$$\widetilde{\exists} n.\ [x : \mathsf{true}].\ \forall\Box((x = n) \Rightarrow P) \quad (4)$$

Now, we introduce fragments of PCTL that are considered in the next section for decidability issues. The main fragment we consider is called PCTL$^+$ and is obtained by the following definition:

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

as well as the abbreviations introduced previously (hence, the operators $\exists\bigcirc$ and $\forall\bigcirc$ are definable in PCTL$^+$). Thus, the difference with PCTL is that in the formulas of the form $\varphi_1 \forall\mathcal{U} \varphi_2$ (resp. $\varphi_1 \exists\mathcal{U} \varphi_2$), the subformula φ_2 (resp. φ_1) must be a state formula.

Notice that we still can express in PCTL$^+$ significant nonregular properties. For instance, all the formulas (1), (2), (3) and (4) given above are PCTL$^+$ formulas.

Then, we consider two fragments of PCTL$^+$ called PCTL$_\exists^+$ and PCTL$_\forall^+$. The fragment PCTL$_\exists^+$ (resp. PCTL$_\forall^+$) is the positive fragment (i.e., negations appear only in state formulas and occurrence constraints) where only existential (resp. universal) path quantification is used. Formally, PCTL$_\exists^+$ is defined by:

$$\varphi ::= \ \pi \mid f \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \widetilde{\exists} x.\varphi \mid \widetilde{\forall} x.\varphi \mid [x : \pi].\varphi \mid \exists\Box\pi \mid \pi \exists\mathcal{U} \varphi$$

whereas PCTL$_\forall^+$ is defined by:

$$\varphi ::= \ \pi \mid f \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \widetilde{\exists} x.\varphi \mid \widetilde{\forall} x.\varphi \mid [x : \pi].\varphi \mid \forall\Box\varphi \mid \varphi \forall\mathcal{U} \pi$$

Of course, all the abbreviations introduced previously can be used. So, we can enrich the syntax of PCTL$_\exists^+$ (resp. PCTL$_\forall^+$) by the formulas $\exists\Diamond\varphi$ and $\exists\bigcirc\varphi$ (resp. $\forall\Diamond\pi$ and $\forall\bigcirc\varphi$). It can be seen that the negation of every formula in PCTL$_\forall^+$ is a PCTL$_\exists^+$ formula. This is due to the following fact that can easily be shown using PCTL semantics:

$$\varphi \forall\mathcal{U} \pi \Leftrightarrow (\neg\exists\Box\neg\pi) \wedge \neg(\neg\pi \exists\mathcal{U} \neg\pi \wedge \neg\varphi) \quad (5)$$

Notice that all the formulas (1), (2), (3) and (4) given above are in PCTL$_\forall^+$.

5 The Satisfaction Problem

The satisfaction problem corresponds to the question whether some given state in some CFKS satisfies some given formula. We show that the satisfaction problem is undecidable for PCTL. However, it turns out that this problem is actually decidable for PCTL$^+$.

5.1 Undecidability for full PCTL

Let us start with the undecidability results. We adopt the notations of [Rog67] for the elements of the arithmetical hierarchy. The class Σ_1 corresponds to the class of the recursively enumerable sets whereas Π_1 is the class of the complements of Σ_1 sets.

Proposition 5.1 *(Undecidable cases) The problems $s \models_{(E,\gamma)} \phi \exists\mathcal{U}\pi$, and $s \models_{(E,\gamma)} \pi\forall\mathcal{U}\phi$, where $s \in S_\Delta$ for some GCFP Δ and ϕ is a boolean combination of atomic propositions and occurrence constraints, are Σ_1-complete.*

The result above is obtained by a reduction of the halting problem of 2-counter machines. Actually, the proof uses only finite-state KS's. Then, the undecidability of the satisfaction problem for the formulas of the form $\phi\exists\mathcal{U}\pi$ and $\pi\forall\mathcal{U}\phi$ does not come from the fact that we are dealing with CFKS's but rather from the expressive power of PCTL. Now, by Proposition 5.1, and since negation is allowed in PCTL, we obtain

Theorem 5.1 *(Undecidability for PCTL) The problem $s \models_{(E,\gamma)} \varphi$ where $s \in S_\Delta$ for some GCFP Δ and φ is a PCTL formula, is not recursively enumerable.*

5.2 Decidability Results

Now, it remains to consider formulas of the forms $\pi\exists\mathcal{U}\varphi$ and $\varphi\forall\mathcal{U}\pi$. Using the fact (5), it is sufficient to consider the satisfaction problem for the formulas of the forms $\exists\Box\pi$ and $\pi\exists\mathcal{U}\varphi$. We show that the satisfaction problem for this kind of formulas is decidable. More generally, we show that this problem is decidable for all the formulas of PCTL$^+$. Our decision procedure is based on a reduction of the satisfaction problem between GCFP's and PCTL$^+$ formulas to the validity problem of Presburger formulas.

Let Δ be a GCFP. Without loss of generality, we consider the satisfaction problem for states of the form $s = \langle a, A\rangle$ where A is a variable in Var_Δ. To take into account states of the general form $s = \langle a, t\rangle$, it suffices to consider an additional equation $X = t$ where X is a fresh variable and consider the new state $s' = \langle a, X\rangle$. Moreover, we can assume that Δ is in GNF. Indeed, as we have already said in Section 3, every state in any GCFP has a bisimulation equivalent state in some GCFP in GNF and we can prove that bisimilar states satisfy the same PCTL formulas.

Satisfaction of $\exists\Box\pi$ formulas

Let us start with the satisfaction problem in the relatively simple case of the formulas $\exists\Box\pi$. Suppose that we are interested in the problem $s \models_{(E,\gamma)} \exists\Box\pi$ where $s = \langle a, A\rangle$.

Since Δ is in GNF, all the reachable states from s by R_Δ are of the form $\langle b, \alpha\rangle$ where $b \in \Sigma$ and $\alpha \in Var^*_\Delta$. Let us define a Δ-circuit as a sequence $\langle a_1, B_1 \cdot \beta_1\rangle \cdots \langle a_n, B_n \cdot \beta_n\rangle$ where $n \geq 2$, for every $i \in \{1, \ldots, n-1\}$, $\langle a_i, B_i \cdot \beta_i\rangle \rightarrow_{R_\Delta} \langle a_{i+1}, B_{i+1} \cdot \beta_{i+1}\rangle$ and

$B_1 = B_n$. Notice that β_1 and β_n may be different. We say that a Δ-circuit is *elementary* if it does not contain another Δ-circuit.

Now, it is easy to see that $s \models_{(E,\gamma)} \exists\Box\pi$ holds in two cases. The first one is when there exists some finite computation sequence σ (without Δ-circuits) starting from s and satisfying continuously π. The second case corresponds to the existence of some infinite computation sequence starting from s that have some finite prefix $\sigma = \mu\nu$ where μ does not contain any Δ-circuit and ν is an elementary Δ-circuit, such that π is satisfied continuously in σ. Thus, since the structure $\mathcal{K}_\Delta$ is finite-branching (and even it is actually infinite), a finite exploration of this structure allows to decide whether $s \models_{(E,\gamma)} \exists\Box\pi$.

Actually, the problem $s \models_{(E,\gamma)} \exists\Box\pi$ can be reduced to the satisfaction problem for formulas of the form $\pi\exists\mathcal{U}\varphi$. Indeed, let us reconsider the two cases when $s \models_{(E,\gamma)} \exists\Box\pi$ holds. In the first case, we must have $s \models_{(E,\gamma)} \pi\exists\mathcal{U}(\pi \wedge \neg\exists\bigcirc\mathsf{true})$. Concerning the second case, in order to express the existence of a reachable Δ-circuit from s, we need to enrich the set of atomic propositions *Prop* by new propositions P_B for every variable $B \in Var_\Delta$, and replace in Δ each equation $B = \sum_{i=1}^{n} b_i \cdot \beta_i$ by the equation $B = \sum_{i=1}^{n}(b_i \cup \{P_B\}) \cdot \beta_i$. Then, we must have $s \models_{(E,\gamma)} \bigvee_{B \in Var_\Delta}(\pi\exists\mathcal{U}(P_B \wedge (\pi\exists\mathcal{U}(\pi \wedge P_B))))$.

Satisfaction of $\pi\exists\mathcal{U}\varphi$ formulas

Let us consider now the interesting case of formulas of the form $\pi\exists\mathcal{U}\varphi$. In order to present the essence of our technique, we consider at a first step formulas without nesting of the $\exists\mathcal{U}$ operator neither the $\tilde{\exists}$ quantifier nor $\exists\Box\pi$ formulas. The general case is presented later. So, let $\varphi = \pi_1\exists\mathcal{U}(\pi_2 \wedge f)$ and suppose that we are interested in the problem $s \models_{(E,\gamma)} \varphi$ where $s = \langle a, A\rangle$.

First, let us get rid of the case where "$A = \epsilon$" $\in \Delta$. In that case, our problem reduces to the trivial problem of checking whether $\langle a, \epsilon\rangle \models_{(E,\gamma)} \pi_2 \wedge f$.

Now, by definition of the satisfaction relation, the fact that $\langle a, A\rangle \models_{(E,\gamma)} \pi_1\exists\mathcal{U}(\pi_2 \wedge f)$ means that there exists some computation sequence starting from $\langle a, A\rangle$ which has a nonempty *finite prefix* $\sigma = \sigma(1)\cdots\sigma(n)$ such that $\sigma(n) \models \pi_2$ and $\forall j \in \{1,\ldots,n-1\}$. $\sigma(j) \models \pi_1$ and $\|f\|(E^{[1,n]}_{(\sigma,\gamma)}) = \mathsf{true}$. Let $\mathsf{PREF}(\langle a, A\rangle)$ be the set of nonempty finite prefixes of computation sequences starting from $\langle a, A\rangle$.

By abuse of notation, we represent each state s by its label $\Pi_\Delta(s)$, and then, we can consider computation sequences as sequences in Σ^∞ and admit the notation $b \models \pi$ where $b \in \Sigma$ and π is a state formula.

Then, the set $\mathsf{PREF}(\langle a, A\rangle)$ is generated by the CFG such that the set of nonterminal symbols is $Var_\Delta \cup \{[B] : B \in (\{Z\} \cup Var_\Delta)\}$, $[Z]$ is the starting symbol ($Z \notin Var_\Delta$) and the set of productions is $\{B \to b \cdot \beta$: "$B \mapsto b \cdot \beta$" $\in \mathcal{S}(\Delta)\} \cup \{[B] \to b \cdot B_1 \cdots B_{i-1} \cdot [B_i]$: "$B \mapsto b \cdot B_1 \cdots B_n$" $\in (\{Z \mapsto a \cdot A\} \cup \mathcal{S}(\Delta))$ and $0 \leq i \leq n\}$.

Moreover, let $L(\pi_1\mathcal{U}\pi_2)$ be the set of sequences $\sigma(1)\cdots\sigma(n)$ in Σ^+ such that for every $j \in \{1,\ldots,n-1\}$, $\sigma(i) \models \pi_1$ and $\sigma(n) \models \pi_2$. Clearly, this set is regular. Since the intersection of a context-free language with a regular one is context-free, the language $\mathsf{PREF}(\langle a, A\rangle) \cap L(\pi_1\mathcal{U}\pi_2)$ is context-free and then, it is generated by some context-free grammar in GNF $\mathcal{G} = (\Sigma, \mathcal{N}, \mathcal{P}, \mathcal{Z})$. Thus, we obtain

$$\langle a, A\rangle \models_{(E,\gamma)} \varphi \text{ iff } \exists\sigma \in L(\mathcal{G}) \text{ and } \|f\|(E^{[1,|\sigma|]}_{(\sigma,\gamma)}) = \mathsf{true} \tag{6}$$

Now, we construct a Presburger formula Ω which is valid if and only if there exists some sequence $\sigma \in L(\mathcal{G})$ such that $\|f\|(E^{[1,|\sigma|]}_{(\sigma,\gamma)})$ is true. Consider the derivation

$$\omega \Longrightarrow_{p_1} \sigma_1\omega_1 \cdots \Longrightarrow_{p_n} \sigma_n\omega_n \Longrightarrow_{p_{n+1}} \sigma \tag{7}$$

where ω and the ω_i's are nonempty sequences of nonterminals and σ and the σ_i's are sequences in Σ^+.

First, let us define the sets of variables that are involved in Ω. With every state label $b \in \Sigma$ we associate a variable u_b. Let U be the set of such variables. The variable u_b stands for the number of occurrences of b in the sequence σ of (7). Furthermore, let $\Upsilon(f)$ be the set of state formulas π such that there exists some occurrence variable $x \in \mathcal{F}(f)$ such that $\gamma(x) = \pi$. With every $\pi \in \Upsilon(f)$, we associate a variable v_π. Let V be the set of such variables. The variable v_π stands for the number of occurrences of state labels satisfying π in σ. Finally, with every production $p \in \mathcal{P}$, we associate a variable w_p which stands for the number of applications of p in (7). Let W be the set of the w_p's.

Since the grammar $\mathcal{G}$ is in GNF, the number of occurrences of any state label b in σ (represented by u_b) is the addition of all the w_p's such that p is a production applied in (7) generating b (has b in its right-hand side). This fact is expressed, for every $b \in \Sigma$, by the formula $\Theta^b_{\mathcal{P}}(U, W)$:

$$(0 \leq u_b) \wedge u_b = \sum w_p \quad \text{for every } p \in \mathcal{P} \text{ such that } b \in rhs(p).$$

Then, we relate each variable v_π to the variables in U using the formula $\Psi^\pi_{\mathcal{P}}(U, V)$:

$$(0 \leq v_\pi) \wedge v_\pi = \sum u_b \quad \text{for every } b \in \Sigma \text{ such that } b \models \pi.$$

Now, we have to define the constraints on the variables in W. For this aim, we need some additional notations and definitions. We say that a sequence of productions $\delta \in \mathcal{P}^*$ is *elementary* if all its productions apply to different nonterminals, i.e., $\forall p \in \mathcal{P}.\ |\delta|_p \leq 1$. Given a nonterminal B, a sequence $\omega \in \mathcal{N}^+$ and a set of productions $\mathcal{P}' \subseteq \mathcal{P}$, we define $\Pi^B_{(\mathcal{P}',\omega)}$ to be the set of elementary sequences δ on $\mathcal{P}'$ such that $\exists \mu \in (\Sigma \cup \mathcal{N})^*$, $\exists \nu \in \mathcal{N}^*$, $\omega \overset{*}{\Longrightarrow}_\delta \mu B \nu$. Notice that the set $\Pi^B_{(\mathcal{P}',\omega)}$ is finite. We define also $\mathcal{R}(\mathcal{P}',\omega)$ to be the set of the reachable nonterminals from ω using the productions in $\mathcal{P}'$, i.e., $\mathcal{R}(\mathcal{P}',\omega) = \{B \in \mathcal{N} : \Pi^B_{(\mathcal{P}',\omega)} \neq \emptyset\}$.

First of all, the constraints on W must express the fact that any occurrence of a nonterminal appearing along the derivation (7) must be reduced so that only terminal symbols (elements of Σ) remain in the final produced chain σ. Thus, for any nonterminal B, the number of the B-reductions, i.e., applications of some productions p such that $lhs(p) = B$ (B-productions), must be equal to the number of the B-introductions, i.e., the number of the occurrences of B in ω and in the right-hand sides of the applied productions. This fact is expressed by the Presburger formula $\Gamma^B_{(\mathcal{P},\omega)}(W)$:

$$\sum_{p \in \mathcal{P}} |lhs(p)|_B \cdot w_p = |\omega|_B + \sum_{p \in \mathcal{P}} |rhs(p)|_B \cdot w_p$$

However, some valuations validating $\bigwedge_{B \in \mathcal{N}} \Gamma^B_{(\mathcal{P},\omega)}(W)$ may assign to some variable w_p a non null value while p is not necessarily involved in the derivation (7). Indeed, consider some valuation E that validates $\bigwedge_{B \in \mathcal{N}} \Gamma^B_{(\mathcal{P},\omega)}(W)$ and suppose that it corresponds to the derivation (7). Consider also some nonterminal B which does not appear in ω neither in any ω_i in (7). Now, assume that there is some production $p = B \rightarrow b \cdot B$ in $\mathcal{P}$. We can define another valuation E' which assigns to w_p any strictly positive integer and coincides with E on the other variables. Clearly, the new valuation E' validates also $\bigwedge_{B \in \mathcal{N}} \Gamma^B_{\langle \mathcal{P}, \omega \rangle}(W)$. However, this valuation must be discarded since the number of the b's calculated from E' using the formula $\Theta^b_{\mathcal{P}}(U, W)$ does not correspond necessarily to a value that can be obtained from some existing derivation of the grammar $\mathcal{G}$.

Thus, we must express in addition, the fact that for any nonterminal B, there exists some B-production p with $w_p > 0$ if and only if B appears in ω or in the ω_i's. This is done by the formula $\Xi^B_{\langle \mathcal{P}, \omega \rangle}(W)$:

$$\sum_{p \in \mathcal{P}} |lhs(p)|_B \cdot w_p > 0 \Leftrightarrow \bigvee_{\delta \in \Pi^B_{\langle \mathcal{P}, \omega \rangle}} \bigwedge_{p \in \delta} w_p > 0$$

Finally, consider the formula $\Phi_{\langle \mathcal{P}, \omega \rangle}(U, V, W)$ defined by

$$(\bigwedge_{p \in \mathcal{P}} w_p \geq 0) \wedge (\bigwedge_{B \in \mathcal{N}} \Gamma^B_{\langle \mathcal{P}, \omega \rangle} \wedge \Xi^B_{\langle \mathcal{P}, \omega \rangle}) \wedge (\bigwedge_{b \in \Sigma} \Theta^b_{\mathcal{P}}) \wedge (\bigwedge_{\pi \in \Upsilon(f)} \Psi^\pi_{\mathcal{P}})$$

Then, the Presburger formula Ω is

$$\exists U.\, \exists V.\, \exists W.\, \Phi_{\langle \mathcal{P}, \mathcal{Z} \rangle} \wedge f[e(x)/x]_{x \in \mathcal{F}(f)}$$

where $e(x) =$ if $x \in \mathcal{D}(\gamma)$ then $v_{\gamma(x)} + E(x)$ else $E(x)$ and $f[e(x)/x]$ denotes the formula obtained from f by substituting each occurrence of the variable x by the expression $e(x)$. Then, we prove the following result:

Proposition 5.2 *The formula Ω is valid iff $\exists \sigma \in L(\mathcal{G})$ such that $\|f\|(E^{[1,|\sigma|]}_{\langle \sigma, \gamma \rangle}) = true$.*

By (6) and Proposition 5.2. and since the validity problem in Presburger arithmetic is decidable, we can decide whether $\langle a, A \rangle \models_{(E,\gamma)} \pi_1 \exists\mathcal{U}(\pi_2 \wedge f)$. This result can be easily extended to any formula of the form $\pi\exists\mathcal{U}\phi$ where ϕ is a boolean combination of π's (state formulas) and f's (occurrences constraints).

Proposition 5.3 *(Decidable cases) The problems $s \models_{(E,\gamma)} \pi\exists\mathcal{U}\phi$, and $s \models_{(E,\gamma)} \phi\forall\mathcal{U}\pi$, where $s \in S_\Delta$ for some GCFP Δ and ϕ is a boolean combination of state formulas and occurrence constraints, are decidable.*

Satisfaction of PCTL$^+$ formulas

Let us consider now the general case of PCTL$^+$ formulas. Following the same lines as in the previous section, we show that the satisfaction problem for any given state in any CFKS and any given PCTL$^+$ formula is reducible to the validity problem in Presburger arithmetic.

So, consider a PCTL$^+$ formula φ and suppose that we are interested in the problem $\langle a, A \rangle \models_{(E,\gamma)} \varphi$. First of all, we transform the formula φ into a *normal form* defined by:

$\varphi ::= \bigvee \bigwedge (\pi \wedge \phi)$

$\phi ::= \phi \wedge \phi \mid \psi$

$\psi ::= f \mid \neg\psi \mid \tilde{\exists} x.\phi \mid \exists\Box\pi \mid [x : \pi].(\tilde{\pi} \wedge \phi)$ where $\tilde{\pi} \in \{\pi, \neg\pi\} \mid \exists\bigcirc(\pi\exists\mathcal{U}(\pi \wedge \phi))$

Notice that in the case of a formula $[x:\pi].(\tilde{\pi}\wedge\phi)$, the state formula $\tilde{\pi}$ is either *equal* to the formula π which is associated with x, or *equal* to $\neg\pi$.

This normal form is obtained using distributivity laws and facts concerning temporal operators like (5) and the fact that $\pi\exists\mathcal{U}\varphi \Leftrightarrow \varphi \vee (\pi \wedge \exists\bigcirc\,(\pi\exists\mathcal{U}\varphi))$. Then, let us assume that φ is in normal form.

We get rid of the formulas $\exists\Box\pi$ that appears in φ using the fact that their satisfaction is reducible to the satisfaction of formulas in the form $\pi\exists\mathcal{U}\phi$ (see Section 5.2).

So, we can consider a "simplified" normal form which is defined by the syntax given above where the case of $\exists\Box\pi$ formulas is removed.

Let $\varphi = \bigvee_{i=1}^{n}\bigwedge_{j=1}^{m_i}(\pi_i^j \wedge \phi_i^j)$. Clearly, solving the problem $\langle a,A\rangle \models_{(E,\gamma)} \varphi$ reduces to solve the satisfaction problem for each subformula ϕ_i^j.

Then, consider the problem $\langle a,A\rangle \models_{(E,\gamma)} \phi$ where ϕ is one of the subformulas ϕ_i^j of φ. Let *height* be the function that associates with each such a formula its height measured as the number of nested $\exists\mathcal{U}$ operators, and let $h = height(\phi)$. Recall that in the case without nesting of $\exists\mathcal{U}$ operators, (i.e., $h=1$) presented in Section 5.2, to solve the problem $\langle a,A\rangle \models_{(E,\gamma)} \pi_1\exists\mathcal{U}(\pi_2\wedge f)$, we reason on the set of nonempty prefixes of the computation sequences starting from $\langle a,A\rangle$ (see the assertion (6)). Now, in the general case, we have to reason on (at most) h prefixes of each computation sequence starting from $\langle a,A\rangle$. For this, we define the CFG $\mathcal{G}_{(A,\phi)} = (\Sigma,\mathcal{N},\mathcal{P},\mathcal{Z})$ where

- $\mathcal{N} = \{[i,B,j] \;:\; B\in Var_\Delta,\; i,j\in\{1,\ldots,h+1\},\; i\le j\}$,
- $\mathcal{Z} = [1,A,h+1]$,
- $\mathcal{P} = \{[i,B,j]\to b\cdot[i_1,B_1,i_2]\cdots[i_k,B_k,j] \;:\; i_1\le i_2\le\ldots\le i_k\le j$ and $i_1\in\{i,i+1\}$, "$B\mapsto b\cdot B_1\cdots B_k$" $\in\mathcal{S}(\Delta)$ where $B_i\in Var_\Delta$, for $1\le i\le k\}$ $\cup$ $\{[i,B,j]\to b \;:\; j\in\{i,i+1\}$ and "$B\mapsto b$" $\in\mathcal{S}(\Delta)\}$.

For every $k\in\{1,\ldots,h+1\}$, let $\mathcal{N}_k = \{[i,B,j]\in\mathcal{N} \;:\; i=k\}$ and let $\mathcal{P}_k = \{p\in\mathcal{P} \;:\; lhs(p)\in\mathcal{N}_k\}$. It can be seen that every computation sequence σ starting from $\langle a,A\rangle$ can be written $\sigma = a\cdot\mu_1\cdots\mu_h\nu$ such that $\mathcal{Z} \overset{+}{\Longrightarrow}_{\mathcal{P}_1} \mu_1\omega_1 \overset{+}{\Longrightarrow}_{\mathcal{P}_2} \mu_1\mu_2\omega_2\cdots \overset{+}{\Longrightarrow}_{\mathcal{P}_h} \mu_1\cdots\mu_h\omega_h$ where $\forall i.\; 1\le i\le h.\; \omega_i\in(\bigcup_{k=i+1}^{h+1}\mathcal{N}_k)^*$.

We associate each subsequence μ_k with the k^{th} level of $\exists\mathcal{U}$ nesting in the formula ϕ. Notice that the sequences μ_k are nonempty. Indeed, to deal with the satisfaction of a formula $\pi\exists\mathcal{U}\varphi$ at some level k, we distinguish (by taking the normal form of the formula) the case when it is satisfied immediatly at the current state (if φ is) and the case when it is satisfied farther on some outgoing computation sequence (then, the current state must satisfy $\pi\wedge\exists\bigcirc(\pi\exists\mathcal{U}\varphi)$). In the last case, to check the satisfaction of $\exists\bigcirc(\pi\exists\mathcal{U}\varphi)$, we proceed, as in Section 5.2, by reducing this problem to the validity of a Presburger formula that expresses constraints on the derivations of $\mathcal{G}_{(A,\phi)}$ such that, there exists some generated nonempty sequence μ_k where π is continuously satisfied, except in its last state where φ must be satisfied. This last state of μ_k is actually the initial state concerning the level $k+1$ and so on.

So, we construct a Presburger formula Ω which is valid if and only if $\langle a,A\rangle \models_{(E,\gamma)} \phi$. This formula is built by nesting the formulas that constraint the derivations of $\mathcal{G}_{(A,\phi)}$ at each level k. Let us define the set of variables that are involved in Ω. For every $k\in\{1,\ldots,h\}$, and every $b\in\Sigma$, we define a variable u_b^k that stands for the number of occurrences of b in the subsequence μ_k. For every π we define a variable v_π^k standing for the number of states in μ_k which satisfy π. We consider also for every $p\in\mathcal{P}$ a variable

w_p standing for the number of applications of p. Let U_k be the set of the u_b^k's, V_k be the set of the v_π^k's and W_k be the set of the w_p's such that $p \in \mathcal{P}_k$.

Now, for every $k \in \{1, \ldots, h\}$ and $b \in \Sigma$, consider the Presburger formula $\Theta_{\mathcal{P}}^{(b,k)}$:

$$(0 \leq u_b^k) \wedge u_b^k = \sum w_p \text{ for every } p \in \mathcal{P}_k \text{ such that } b \in rhs(p)$$

and let $\Psi_{\mathcal{P}}^{(\pi,k)}$ be the formula

$$(0 \leq v_\pi^k) = \sum u_b^k \text{ for every } b \in \Sigma \text{ such that } b \models \pi.$$

The constraints on the variables W consist in those expressed by the formulas Γ and Ξ defined in Section 5.2 and, in addition, some contraints expressing the fact that the computation sequence must be *consistent* with the state formulas involved in ϕ. Given $k \in \{1, \ldots, h\}$ and a state formula π, let $\mathsf{COND}_{\mathcal{P}}^{(\pi,k)}$ be the formula defined by:

$$\begin{array}{ll}\sum w_p = 0 & \text{for every } p \in \mathcal{P}_k \text{ such that } p = \text{“}[k, B, k] \rightarrow b \cdot \beta\text{”} \\ & \text{for some } B \in Var_\Delta,\ \beta \in \mathcal{N}^*, \text{ and } b \not\models \pi\end{array}$$

and $\mathsf{REACH}_{\mathcal{P}}^{(\pi,k)}$ be the formula defined by:

$$\begin{array}{ll}\sum w_p = 0 & \text{for every } p \in \mathcal{P}_k \text{ such that either } p = \text{“}[k, B, k+1] \rightarrow b\text{”} \\ & \text{or } p = \text{“}[k, B, i] \rightarrow b \cdot [k+1, B', j] \cdot \beta\text{”} \\ & \text{for some } B, B' \in Var_\Delta,\ i, j \geq k+1,\ \beta \in \mathcal{N}^*, \text{ and } b \not\models \pi\end{array}$$

The constraints $\mathsf{COND}_{\mathcal{P}}^{(\pi_1,k)}$ and $\mathsf{REACH}_{\mathcal{P}}^{(\pi_2,k)}$ are used to express the fact that, to satisfy some formula $\exists\bigcirc(\pi_1 \exists\mathcal{U}(\pi_2 \wedge \psi))$ at some level k, necessarily, all the states in the sequence μ_k, except the last one, must satisfy π_1 and its last state must satisfy π_2. Notice that the constraints $\mathsf{COND}_{\mathcal{P}}^{(\pi_1,k)}$ and $\mathsf{REACH}_{\mathcal{P}}^{(\pi_2,k)}$ have been expressed in the case $h = 1$ considered in Section 5.2 by the fact that the set of the (nonempty) prefixes of computation sequences $\mathsf{PREF}(\langle a, A \rangle)$ was restricted by intersection with $L(\pi_1 \mathcal{U} \pi_2)$.

Now, let $E' = E[x \leftarrow E(x) + (\text{if } a \models \gamma(x) \text{ then } 1 \text{ else } 0)]_{x \in \mathcal{F}(\phi) \cap \mathcal{D}(\gamma)}$. Then, we define the formula Ω as $[\![\phi]\!]_1^{(E',\gamma)}$ where for every $k \in \{1, \ldots, h\}$, and for every valuation F, and every state formula association η,

- $[\![\neg\psi]\!]_k^{(F,\eta)} \;=\; \neg[\![\psi]\!]_k^{(F,\eta)}$,
- $[\![\psi_1 \wedge \psi_2]\!]_k^{(F,\eta)} \;=\; [\![\psi_1]\!]_k^{(F,\eta)} \wedge [\![\psi_2]\!]_k^{(F,\eta)}$,
- $[\![f]\!]_k^{(F,\eta)} \;=\; f[F(x)/x]_{x \in \mathcal{F}(f)}$,
- $[\![\tilde{\exists} x.\psi]\!]_k^{(F,\eta)} \;=\; \exists x.[\![\psi]\!]_k^{(F,\eta)}$,
- $[\![[x : \pi].(\pi \wedge \psi)]\!]_k^{(F,\eta)} \;=\; [\![\psi]\!]_k^{(F',\eta')}$ where $F' = F[x \leftarrow 1]$ and $\eta' = \eta[x \leftarrow \pi]$,
- $[\![[x : \pi].(\neg\pi \wedge \psi)]\!]_k^{(F,\eta)} \;=\; [\![\psi]\!]_k^{(F',\eta')}$ where $F' = F[x \leftarrow 0]$ and $\eta' = \eta[x \leftarrow \pi]$,
- $[\![\exists\bigcirc\,(\pi_1 \exists\mathcal{U}(\pi_2 \wedge \psi))]\!]_k^{(F,\eta)} \;=$

$$\begin{array}{l}\exists U_k.\, \exists V_k.\, \exists W_k. \\ \quad (\bigwedge_{p \in \mathcal{P}_k} w_p \geq 0) \wedge (\bigwedge_{X \in \mathcal{N}_k} \Gamma_{(\mathcal{P},\mathcal{Z})}^X \wedge \Xi_{(\mathcal{P},\mathcal{Z})}^X) \wedge \\ \quad (\bigwedge_{b \in \Sigma} \Theta_{\mathcal{P}}^{(b,k)}) \wedge (\bigwedge_{\pi \in \Upsilon(f)} \Psi_{\mathcal{P}}^{(\pi,k)}) \wedge \\ \quad (\mathsf{COND}_{\mathcal{P}}^{(\pi_1,k)} \wedge \mathsf{REACH}_{\mathcal{P}}^{(\pi_2,k)}) \wedge \\ \quad [\![\psi]\!]_{k+1}^{(F',\eta)} \quad \text{where } F' = F[x \leftarrow v_{\eta(x)}^k + F(x)]_{x \in \mathcal{F}(\psi) \cap \mathcal{D}(\eta)}.\end{array}$$

Notice that in the definition above of the function $[\![\cdot]\!]_k^{(F,\eta)}$, we consider that F associates with each variable an *expression* (actually a sum of constants and variables) an not necessarily an integer value. For instance, in the last case, $F' = F[x \leftarrow v_{\eta(x)}^k + F(x)]_{x \in \mathcal{F}(\psi) \cap \mathcal{D}(\eta)}$ associates with each variable in $\mathcal{F}(\psi) \cap \mathcal{D}(\eta)$ the *expression* $v_{\eta(x)}^k + F(x)$. In the case of $[\![f]\!]_k^{(F,\eta)}$, for every variable $x \in \mathcal{F}(f)$, the expression $F(x)$ is substituted to each occurrence of x in f.

Then, we prove that $\langle a, A \rangle \models_{(E,\gamma)} \phi$ if and only if the Presburger formula Ω is valid, and hence, we obtain the following decidability result:

Theorem 5.2 *The problem $s \models_{(E,\gamma)} \varphi$ where $s \in S_\Delta$ for some GCFP Δ and φ is a PCTL$^+$ formula, is decidable.*

6 The Satisfiability Problem

We consider now the satisfiability problem for PCTL, i.e., the problem to know, given some PCTL formula φ, whether there exists some state s in some KS that satisfies φ.

First, we show that when we consider KS's without any restriction, the satisfiability problem is Σ_1^1-complete for PCTL as well as for PCTL$^+$ (see [Rog67] for an exposition of the analytical hierarchy). This makes the validity problem for PCTL, and also for PCTL$^+$, to be highly undecidable (Π_1^1-complete). Furthermore, we consider the satisfiability problem with CFKS's, i.e., KS's that correspond to some context-free process. Indeed, to check that some process specification is consistent (satisfiable), we are more interested by its satisfiability by some KS that corresponds to some process than by its satisfiability by any KS. We show, that when we restrict ourselves to the class of CFKS's, the satisfiability problem for PCTL$^+$ becomes semi-decidable (Σ_1-complete). This is due to the fact that the set of GCFP's over Σ is recursively enumerable and the satisfaction problem for PCTL$^+$ by CFKS's is decidable (see Theorem 5.2). Then, we have the following undecidability results

Theorem 6.1 *(Undecidability results)*

1. *The satisfiability problems for PCTL and PCTL$^+$ are Σ_1^1-complete.*
2. *The satisfiability problem for PCTL$^+$ with CFKS's is Σ_1-complete.*

Finally, we show that the satisfiability problem for PCTL$_\exists^+$ is actually decidable, and hence, the validity problem for PCTL$_\forall^+$ is also decidable.

Theorem 6.2 *The satisfiability problem for PCTL$_\exists^+$ is decidable.*

The proof of Theorem 6.2 is based on Theorem 5.2 and the fact that, since only existential path quantification is allowed in PCTL$_\exists^+$, we can show that a PCTL$_\exists^+$ formula is satisfiable if and only if it is satisfiable by some state in the *finite* Kripke structure $K = (\Sigma, S, \Pi, R)$ such that, for every $a \in \Sigma$, there are exactly two states s_a and s'_a such that $\Pi(s_a) = \Pi(s'_a) = a$, $s'_a \not\to_R$ and s_a is related by R with all the other states in S.

7 Conclusion

We propose in this paper a logical framework for the specification and the verification of processes with infinite state spaces. We provide mainly a recursive verification procedure for nonregular properties w.r.t. context-free processes. This procedure concerns properties that are definable in an expressively powerful logic PCTL combining a classical temporal logic (CTL) with Presburger arithmetic. The arithmetical part of the logic allows to express constraints on numbers of occurrences of state formulas. Naturally, our decidability results still hold if we consider any decidable extension of Presburger arithmetic.

The work we present can be extended straightforwardly to the specification and verification of *timed processes* with a discrete time domain: For instance, time "ticks" can be seen as occurrences of some particular event (corresponding to the truth of some special atomic proposition) and then, time constraints can be expressed as any other occurrence constraints. Moreover, we can also consider the notion of *duration* of a state formula, i.e., time during which the fixed state formula holds [CHR91]. Indeed, in the case of a discrete time domain, the notion of duration coincides with the notion of number of occurrences at states where time ticks appear. In this framework, the results of this paper extend some results given in [BES93].

References

[BBK87] J.C.M. Baeten, J.A. Bergstra, and J.W. Klop. Decidability of Bisimulation Equivalence for Processes Generating Context-Free Languages. Tech. Rep. CS-R8632, 1987. CWI.

[BES93] A. Bouajjani, R. Echahed, and J. Sifakis. On Model Checking for Real-Time Properties with Durations. In *8th Symp. on Logic in Computer Science*. IEEE, 1993.

[BJ74] G.S. Boolos and R.C. Jeffrey. *Computability and Logic*. Cambridge Univ. Press, 1974.

[BS92] O. Burkart and B. Steffen. Model Checking for Context-Free Processes. In *CONCUR'92*. Springer-Verlag, 1992. LNCS 630.

[Buc62] J.R. Buchi. On a Decision Method in Restricted Second Order Arithmetic. In *Intern. Cong. Logic, Method and Philos. Sci.* Stanford Univ. Press, 1962.

[CES83] E.M. Clarke, E.A. Emerson, and E. Sistla. Automatic Verification of Finite State Concurrent Systems using Temporal Logic Specifications: A Practical Approach. In *10th ACM Symp. on Principles of Programming Languages*. ACM, 1983.

[CHR91] Z. Chaochen, C.A.R. Hoare, and A.P. Ravn. A Calculus of Durations. *Information Processing Letters*, 40:269–276, 1991.

[CHS92] S. Christensen, H. Hüttel, and C. Stirling. Bisimulation Equivalence is Decidable for all Context-Free Processes. In *CONCUR'92*. Springer-Verlag, 1992. LNCS 630.

[EH83] E.A. Emerson and J.Y. Halpern. 'Sometimes' and 'Not Never' Revisited : On Branching versus Linear Time Logic . In *POPL*. ACM, 1983.

[EL86] E.A. Emerson and C.L. Lei. Efficient Model-Checking in Fragments of the Propositional μ-Calculus. In *First Symp. on Logic in Computer Science*, 1986.

[Eme87] E.A. Emerson. Uniform Inevitability is Tree Automaton Ineffable. *Information Processing Letters*, 24, 1987.

[FL79] M.J. Fischer and R.E. Ladner. Propositional Dynamic Logic of Regular Programs. *J. Comp. Syst. Sci.*, 18, 1979.

[GH91] J.F. Groote and H. Hüttel. Undecidable Equivalences for Basic Process Algebra. Tech. Rep. ECS-LFCS-91-169, 1991. Dep. of Computer Science, Univ. of Edinburgh.

[GPSS80] D. Gabbay, A. Pnueli, S. Shelah, and J. Stavi. On the Temporal Analysis of Fairness. In *7th Symp. on Principles of Programming Languages*. ACM, 1980.

[Har78] M.A. Harrison. *Introduction to Formal Language Theory*. Addison-Wesley Pub. Comp., 1978.

[HP84] D. Harel and M.S. Paterson. Undecidability of PDL with $L = \{a^{2^i} \mid i \geq 0\}$. *J. Comp. Syst. Sci.*, 29, 1984.

[HPS83] D. Harel, A. Pnueli, and J. Stavi. Propositional Dynamic Logic of Nonregular Programs. *J. Comp. Syst. Sci.*, 26, 1983.

[HR90] D. Harel and D. Raz. Deciding Properties of Nonregular Programs. In *31th Symp. on Foundations of Computer Science*, pages 652–661. IEEE, 1990.

[Koz83] D. Kozen. Results on the Propositional μ-Calculus. *Theo. Comp. Sci.*, 27, 1983.

[KP83] T. Koren and A. Pnueli. There Exist Decidable Context-Free Propositional Dynamic Logics. In *Proc. Symp. on Logics of Programs*. Springer-Verlag, 1983. LNCS 164.

[Niw88] D. Niwinski. Fixed Points vs. Infinite Generation. In *LICS*. IEEE, 1988.

[Pnu77] A. Pnueli. The Temporal Logic of Programs. In *FOCS*. IEEE, 1977.

[Pra81] V.R. Pratt. A Decidable Mu-Calculus: Preliminary Report. In *FOCS*. IEEE, 1981.

[QS82] J-P. Queille and J. Sifakis. Specification and Verification of Concurrent Systems in CESAR. In *Intern. Symp. on Programming, LNCS 137*, 1982.

[Rab69] M.O. Rabin. Decidability of Second Order Theories and Automata on Infinite Trees. *Trans. Amer. Math. Soc.*, 141, 1969.

[Rog67] H. Rogers. *Theory of Recursive Functions and Effective Computability*. McGraw-Hill Book Comp., 1967.

[SE84] R.S. Streett and E.A. Emerson. The Propositional μ-Calculus is Elementary. In *ICALP*. Springer-Verlag, 1984. LNCS 172.

[Str82] R.S. Streett. Propositional Dynamic Logic of Looping and Converse is Elementary Decidable. *Information and Control*, 54, 1982.

[Tho87] W. Thomas. On Chain Logic, Path Logic, and First-Order Logic over Infinite Trees. In *2nd Symp. on Logic in Computer Science*, 1987.

[Var88] M.Y. Vardi. A Temporal Fixpoint Calculus. In *POPL*. ACM, 1988.

[VW86] M.Y. Vardi and P. Wolper. Automata-Theoretic Techniques for Modal Logics of Programs. *J. Comp. Syst. Sci.*, 32, 1986.

[Wol83] P. Wolper. Temporal Logic Can Be More Expressive. *Inform. and Control*, 56, 1983.