Parametric Quantitative Temporal Reasoning *

E. Allen Emerson
Richard J. Trefler
Computer Sciences Department and Computer Engineering Research Center
University of Texas, Austin, TX 78712, USA
{emerson, trefler}@cs.utexas.edu

Abstract

We define Parameterized Real-Time Computation Tree Logic (PRTCTL), which allows quantitative temporal specifications to be parameterized over the natural numbers. Parameterized quantitative specifications are quantitative specifications in which concrete timing information has been abstracted away. Such abstraction allows designers to specify quantitative restrictions on the temporal ordering of events without having to use specific timing information from the model. A model checking algorithm for the logic is given which is polynomial for any fixed number of parameters. A subclass of formulae are identified for which the model checking problem is linear in the length of the formula and size of the structure. PRTCTL is generalized to allow quantitative reasoning about the number of occurrences of atomic events.

1. Introduction

Pnueli [18] pioneered the use of temporal logic as a language for describing the behavior of reactive systems [16]. Qualitative temporal logics, PLTL [18] and CTL [10] for example, express properties of the temporal ordering of events without, in general, a regard for any quantitative measure on the elapsed time between the occurrences of the events. Real-time temporal logics ([17] [5] [2] [13] [14]) and more generally quantitative logics ([7] [8] [6] [15] c.f. [9]) cater for the expression of quantitative bounds on the occurrence of events. Such logics allow operators of the form $\mathsf{AF}^{\leq 5}P$ meaning that inevitably event P will occur within five time steps. The use of constants, 5 in this case, can be problematic, requiring detailed knowledge of the specific timing information of a system – design information which may change frequently as the design moves towards completion.

Parameterization of quantitative specifications is a type of abstraction which allows the system designer to express quantitative information which is valid over an entire design process. For example, it may not be known exactly what the delay is between when a process sends a request for a resource and when the server receives the request, but it is expected that it takes no more than twice that amount of time for the requester to receive the grant for the resource from the server. This type of specification can be expressed in Parameterized Real-Time Computation Tree Logic (PRTCTL) as

$$\forall x[\mathsf{AG}(\mathit{request} \Rightarrow \mathsf{AF}^{\leq x}\mathit{receive}) \\ \Rightarrow \mathsf{AG}(\mathit{request} \Rightarrow \mathsf{AF}^{\leq 2\cdot x}\mathit{grant})]$$

This says that for any natural number x if it takes at most x steps for a request to be received by the server then it takes at most $2 \cdot x$ steps for the requester to receive the grant from the server. This is much different from the purely qualitative CTL specification

$$\mathsf{AG}(\mathit{request} \Rightarrow \mathsf{AF}\mathit{receive}) \Rightarrow \mathsf{AG}(\mathit{request} \Rightarrow \mathsf{AF}\mathit{grant})$$

which simply says that if it is inevitable that requests are received, then it is inevitable that requests are granted. Furthermore, the use of parameterization has obviated the need for specifying exactly how long it takes for requests to be received.

While PRTCTL allows parameterization over all natural numbers we show that the model checking problem [10] (c.f. [19]) for this logic can be reduced to checking a finite number of instances of the parameterized formula. The number of instances depends on the size of the model to be checked. This leads to a model checking algorithm of time complexity $\mathcal{O}(|M|^{k+1}|f|)$ where k is the number of parameters in f, $|M|$ is the

*This work was supported in part by NSF grant CCR-980-4736 and SRC contract 98-DP-388.

size of M and $|f|$ is the length of f. For a large class of formulae this complexity can be lowered to linear in the size of the formula and the size of the structure.

A PRTCTL formula is monotone in parameter x, if when put in positive normal form, x appears either exclusively in modalities of the form $\mathsf{U}^{\leq x}$ or $\mathsf{F}^{\leq x}$ or exclusively in $\mathsf{G}^{\leq x}$ modalities. A formula is monotone if it is monotone in all its parameters. We show that the model checking problem for monotone formulae can be solved in time $\mathcal{O}(|M||f|)$.

We also describe an algorithm for solving the PRTCTL model checking problem over timed structures. A discrete timed structure allows transitions to be labeled with natural numbers. We show that if m is the largest transition label then PRTCTL formulae f with k parameters can be model checked on timed structure M in time $\mathcal{O}((m|M|)^k|M|^2|f|)$.

PRTCTL, and indeed most quantitative logics, are designed for the expression of quantitative properties concerning the number of transitions or the number of time units represented by transitions until the occurrence of an event. However, these logics cannot naturally express quantitative measures on the number of occurrences of independent events. We define and give model checking algorithms for General Parameterized CTL, GPCTL, (c.f. [17] [7] [8] [6] [21] [15]) which allows quantitative parameterization over the occurrence of events. For example, $\mathsf{AF}^{\leq 3 \cdot request} response$ says that it is inevitable that a response will be issued before more than 4 occurrences of request. $\exists x \mathsf{AG}(request \Rightarrow \mathsf{AF}^{\leq x \cdot request} response)$ says that there is a bound on the number of requests which will be issued before a response is seen, something that is not guaranteed by the formula $\mathsf{AG}(request \Rightarrow \mathsf{AF} response)$.

There have been numerous models of computation proposed for the modeling of real-time systems (see [3] for a survey). Such models tend to be extensions of Kripke structures which add resettable clocks (or timers) and transitions labeled with time values from discrete or dense time domains. These more elaborated models are in many circumstances more accurate models of real-time systems than the less complex structures. However, for many systems the appropriate level of abstraction for verification of temporal properties is often the conventional Kripke structure. For example, an interleaving model of true concurrency is a realistic assumption for many on-board embedded systems. Since hardware may be quite limited, there is often a single multiprogrammed processor on which sequential processes are running concurrently when viewed at a sufficient level of granularity. We argue that it may be necessary and useful to verify quantitative temporal properties of the less sophisticated models of computation.

The rest of the paper is organized as follows. In the next section PRTCTL is defined and given a semantics. Section 3 gives model checking algorithms for the logic and for the monotone subclass of formulae. Section 4 extends the logic and model checking algorithms to timed models of computation. Section 5 contains a discussion of GPCTL and model checking algorithms for that logic. Finally, section 6 contains a short conclusion and comparison with other work.

2. PRTCTL

This section contains a description of the logic PRTCTL and its related logics CTL, RTCTL and vRTCTL. vRTCTL is used only as a convenience for generating the formulae of PRTCTL. We use $\mathbb{N}$ to denote the set of natural numbers and AP to denote a finite set of atomic propositions. CTL is the set of formulae formed from the standard boolean operators $\wedge, \vee$ and $\neg$, atomic propositions $P \in AP$ and temporal modalities $\mathsf{AX}, \mathsf{AU}, \mathsf{AF}, \mathsf{AG}, \mathsf{EX}, \mathsf{EU}, \mathsf{EF}$ and EG. For more details we refer to [12].

To formulate the logic RTCTL superscripts of the form $\leq n$, for $n \in \mathbb{N}$, are allowed over the path operators U, F and G. For example $\mathsf{AG}^{\leq 5}P$ is a formula of RTCTL. vRTCTL extends RTCTL by allowing variables and linear combinations of variables and constants in the superscripts. $\mathsf{AG}^{\leq 5x+y}P$, for example, is a formula of vRTCTL.

PRTCTL formulae are formed as follows: Let $f(x)$ be a formula of vRTCTL or PRTCTL in which the variable x appears free and let $a \in \mathbb{N}$. Then $\forall x f(x)$, $\forall x \leq af(x)$, $\exists x f(x)$ and $\exists x \leq af(x)$ are formulae of PRTCTL.

A formula is in positive normal form (PNF) iff $\neg$ operators appear only in front of atomic propositions and the only boolean operators are $\wedge$ and $\vee$.

Sentences, formulae with no free variables, of the above logics are given semantics relative to Kripke structures. A Kripke structure M is a tuple (S, R, L) where S is a set of states, $R \subseteq S \times S$ (such that R is left total) is the transition relation and $L : S \rightarrow 2^{AP}$ is the labeling function. $\sigma = s_0 s_1 \ldots$ is a (full) path in M iff for all $i \in \mathbb{N}$, $s_i \in S$ and $(s_i, s_{i+1}) \in R$. $M, s \models f$ denotes that state s in structure M satisfies formula f. $M \models f$ signifies that there is some state s in S such that $M, s \models f$. We define $\models$ for CTL formulae in the usual way; see [12] for more details.

RTCTL modalities are given semantics by the following recursive rules and noting that $\mathsf{AF}^{\leq n}$ and $\mathsf{EF}^{\leq n}$ modalities are special cases of $\mathsf{AU}^{\leq n}$ and $\mathsf{EU}^{\leq n}$, respectively.

- $M, s \models \mathsf{A} f \mathsf{U}^{\leq n} g$ iff for all paths $\sigma = s_0 \ldots$ such that $s_0 = s$, there exists an $i \leq n$ such that $M, s_i \models g$ and for all $j < i$, $M, s_j \models f$.

- $M, s \models \mathsf{AG}^{\leq n} f$ iff for all paths $\sigma = s_0 \ldots$ such that $s_0 = s$, and for all $i \leq n$, $M, s_i \models f$.

- $M, s \models \mathsf{E} f \mathsf{U}^{\leq n} g$ iff there exists a path $\sigma = s_0 \ldots$ such that $s_0 = s$, there exists an $i \leq n$ such that $M, s_i \models g$ and for all $j < i$, $M, s_j \models f$.

- $M, s \models \mathsf{EG}^{\leq n} f$ iff there exists a path $\sigma = s_0 \ldots$ such that $s_0 = s$, and for all $i \leq n$, $M, s_i \models f$.

Finally, for formulae of PRTCTL the following are added to the definition of $\models$.

- $M, s \models \forall x f(x)$ iff for all $n \in \mathbb{N}$, $M, s \models f(n)$.

- $M, s \models \forall x \leq a f(x)$ iff for all $n \leq a$, $M, s \models f(n)$.

- $M, s \models \exists x f(x)$ iff there exists $n \in \mathbb{N}$ such that $M, s \models f(n)$.

- $M, s \models \exists x \leq a f(x)$ iff there exists $n \leq a$ such that $M, s \models f(n)$.

Proposition 1 *Suppose $Qx \leq af(x)$ is a sentence of PRTCTL where $a \in \mathbb{N}$ and Q is either $\forall$ or $\exists$. $M, s \models \forall x \leq af(x)$ iff $M, s \models f(0) \wedge f(1) \wedge \ldots \wedge f(a)$ and $M, s \models \exists x \leq af(x)$ iff $M, s \models f(0) \vee f(1) \vee \ldots \vee f(a)$.*

Proposition 1 shows that PRTCTL formulae whose quantifiers range over finite sets can be seen as shorthands for RTCTL formulae. In the sequel $f(0) \wedge f(1) \wedge \ldots \wedge f(a)$ $(f(0) \vee f(1) \vee \ldots \vee f(a))$ will sometimes be abbreviated by $\wedge_{n \in [0..a]} f(n)$ $(\vee_{n \in [0..a]} f(n))$. The following proposition shows that the set of PRTCTL sentences is semantically closed under negation.

Proposition 2 *Let f be a PRTCTL sentence. There is a sentence f' such that for all $M = (S, R, L)$ and $s \in S$, $M, s \models f$ iff $M, s \not\models f'$.*

3. Model Checking PRTCTL

CTL and RTCTL can both be model checked in time $\mathcal{O}(|M||f|)$. Below we show that the model checking problem for PRTCTL is decidable and give an algorithm for solving the problem. The algorithm runs in time $\mathcal{O}(|M|^{k+1}|f|)$ where k is the number of distinct parameters in f. Thus for any fixed number of parameters the problem is polynomial in the size of the formula and the structure. In practice, we expect that most useful formulae will have only one or two parameters and thus the algorithms can be said to be efficient.

The model checking problem for parameterized formulae is reduced to a finite state problem by showing that $M, s \models \forall x f(x)$ $(M, s \models \exists x f(x))$ can be determined by checking only a finite set of instances of the form $M, s \models f(a)$ where a is a natural number no bigger than the size of M.

Theorem 1 *Let $M = (S, R, L)$ be a finite Kripke structure and $Q_n x_n \ldots Q_0 x_0 f(x_0, \ldots x_n)$ be a PRTCTL sentence. $M, s \models Q_n x_n \ldots Q_0 x_0 f(x_0, \ldots x_n)$ iff $M, s \models Q_n x_n \leq |S| \ldots Q_0 x_0 \leq |S| f(x_0, \ldots x_n)$.*

Proof: By induction on the number of quantifiers. If f has no quantifiers then the theorem holds vacuously.
$M, s \models \forall x_{n+1} Q_n x_n \ldots Q_0 x_0 f(x_0, \ldots x_n, x_{n+1})$ iff for all $a \in \mathbb{N}$, $M, s \models Q_n x_n \ldots Q_0 x_0 f(x_0, \ldots x_n, a)$ iff for all $a \in \mathbb{N}$ $M, s \models Q_n x_n \leq |S| \ldots Q_0 x_0 \leq |S| f(x_0, \ldots x_n, a)$. This formula can be interpreted as a short hand for a set of formulae where the quantifiers are replaced by $\wedge$ and $\vee$. For all $(a_0, \ldots, a_n) \in |S|^{n+1}$ it was shown [14] that for all $a > |S|$, $M, s \models f(a_0, \ldots, a_n, a)$ iff $M, s \models f(a_0, \ldots, a_n, |S|)$. So, for all $a \in \mathbb{N}$ $M, s \models Q_n x_n \leq |S| \ldots Q_0 x_0 \leq |S| f(x_0, \ldots x_n, a)$ iff $M, s \models \forall x_{n+1} \leq |S| Q_n x_n \leq |S| \ldots Q_0 x_0 \leq |S| f(x_0, \ldots x_n, x_{n+1})$.
$M, s \models \exists x_{n+1} Q_n x_n \ldots Q_0 x_0\ f(x_0, \ldots x_n, x_{n+1})$ iff there exists $a \in \mathbb{N}$ such that $M, s \models Q_n x_n \ldots Q_0 x_0\ f(x_0, \ldots, x_n, a)$ iff there exists $a \in \mathbb{N}$ $M, s \models Q_n x_n \leq |S| \ldots Q_0\ x_0 \leq |S| f(x_0, \ldots, x_n, a)$. This formula can be interpreted as a short hand for a set of formulae where the quantifiers are replaced by $\wedge$ and $\vee$. For all $(a_0, \ldots, a_n) \in |S|^{n+1}$ it was shown that if $a > |S|$, $M, s \models f(a_0, \ldots, a_n, a)$ iff $M, s \models f(a_0, \ldots, a_n, |S|)$. So, there exists $a \in \mathbb{N}$ $M, s \models Q_n x_n \leq |S| \ldots Q_0 x_0 \leq |S| f(x_0, \ldots x_n, a)$ iff $M, s \models \exists x_{n+1} \leq |S| Q_n x_n \leq |S| \ldots Q_0 x_0 \leq |S| f(x_0, \ldots x_n, x_{n+1})$. $\Box$
The above theorem shows how to replace $\forall x$ by $\forall x \leq |S|$ and $\exists x$ by $\exists x \leq |S|$. In fact, the theorem can be strengthened to replace $\forall x \leq a$ $(\exists x \leq a)$ by $\forall x \leq |S|$ $(\exists x \leq |S|)$ whenever $|S| \leq a$. [14] showed that any formula f of RTCTL can be model checked in time $\mathcal{O}(|M||f|)$. This leads immediately to the following corollary based on the extension to the model checking algorithms of [14] given below. In particular see [14] for details regarding the procedures *ModelCheckEU*, *ModelCheckEUn*, *ModelCheckEG*, *ModelCheckEGn*, *ModelCheckAU*, *ModelCheckAUn*, *ModelCheckAG* and *ModelCheckAGn*. *ModelCheckEU*, for example, takes as input a model $M = (S, R, L)$ and two formulae f and g such that for all $s \in S$, $f \in L(s)$ (respectively, $g \in L(s)$) iff $M, s \models f$ $(M, s \models g)$. *ModelCheckEU* then labels $s \in S$ with $\mathsf{E} f \mathsf{U} g$ iff $M, s \models \mathsf{E} f \mathsf{U} g$. *ModelCheckUn* works similarly for formulae of the form $\mathsf{E} f \mathsf{U}^{\leq a} g$ for

some constant $a \in \mathbb{N}$. Analogously, *ModelCheckEG* and *ModleCheckEGn* check formulae of the form $\mathsf{EG}f$ and $\mathsf{EG}^n f$. The A formulae are handled similarly. In the sequel we show how the complexity can be substantially reduced for monotone formulae.

Procedure *Models*(M, f)
    Checks whether $M, s \models f$
    by labeling all states s of M with f iff $M, s \models f$

    Case f is of the form:
        $\forall x g(x)$:begin
                let $h = g(0) \wedge \ldots \wedge g(|M|)$;
                Models(M, h);
                for all $s \in S$, label s with f iff s is labeled with h;
            end
        $\exists x g(x)$:begin
                let $h = g(0) \vee \ldots \vee g(|M|)$;
                Models(M, h);
                for all $s \in S$ label s with f iff s is labeled with h;
            end
        $\mathsf{EX}g$: begin
                Models(M, g);
                for all $s \in S$ label s with f iff there exists a t such that $(s, t) \in R$ and t is labeled with g;
                end
        $\mathsf{E}g\mathsf{U}^{\leq c}h$: begin
                Models(M, g);
                Models(M, h);
                ModelCheckEUn(M, g, h, c);
            end
        $\mathsf{EG}^{\leq c}g$: begin
                Models(M, g);
                ModelCheckEGn(M, g, c);
                end
        $\mathsf{E}g\mathsf{U}h$: begin
                Models(M, g);
                Models(M, h);
                ModelCheckEU(M, g, h);
            end
        $\mathsf{EG}g$: begin
                Models(M, g);
                ModelCheckEG(M, g);
            end
        $\mathsf{AX}g$: begin
                Models(M, g);
                for all $s \in S$ label s with f iff for all t such that $(s, t) \in R$, t is labeled with g;
                end
        $\mathsf{A}g\mathsf{U}^{\leq c}h$: begin
                Models(M, g);
                Models(M, h);
                ModelCheckAUn(M, g, h, c);
            end
        $\mathsf{AG}^{\leq c}g$: begin
                Models(M, g);
                ModelCheckAGn(M, g, c);
                end
        $\mathsf{A}g\mathsf{U}h$: begin
                Models(M, g);
                Models(M, h);
                ModelCheckAU(M, g, h);
            end
        $\mathsf{AG}g$: begin
                Models(M, g);
                ModelCheckAG(M, g);
            end
        $g \wedge h$: begin
                Models(M, g);
                Models(M, h);
                for all $s \in S$ label s with f iff s is labeled with g and h;
        $g \vee h$: begin
                Models(M, g);
                Models(M, h);
                for all $s \in S$ label s with f iff s is labeled with g or h;
            end
        $\neg g$: begin
                Models(M, g);
                for all $s \in S$ label s with f iff s is not labeled with g;
            end
        P: skip;
    end

Corollary 1 *Given $M = (S, R, L)$ and PRTCTL formula f with k parameters, the model checking problem, does $M \models f$, can be solved in time $\mathcal{O}(|M|^k |M||f|)$.*

Proposition 3 *Suppose $f(x)$ is a vRTCTL formula in positive normal form whose only parameter is x, $M = (S, R, L)$ is a structure and s is a state in M.*

- *If the only parameterized operators in $f(x)$ are $\mathsf{EU}^{\leq x}$ and $\mathsf{AU}^{\leq x}$ then $M, s \models f(0)$ iff for all $x \leq |S|$, $M, s \models f(x)$.*
- *If the only parameterized operators in $f(x)$ are $\mathsf{EU}^{\leq x}$ and $\mathsf{AU}^{\leq x}$ then $M, s \models f(|S|)$ iff there exists $x \leq |S|$, $M, s \models f(x)$.*
- *If the only parameterized operators in $f(x)$ are $\mathsf{EG}^{\leq x}$ and $\mathsf{AG}^{\leq x}$ then $M, s \models f(|S|)$ iff for all $x \leq |S|$, $M, s \models f(x)$.*

- *If the only parameterized operators in $f(x)$ are $\mathsf{EG}^{\leq x}$ and $\mathsf{AG}^{\leq x}$ then $M, s \models f(|0|)$ iff there exists $x \leq |S|$, $M, s \models f(x)$.*

Theorem 2 *Let $Q_0 x_0 \ldots Q_n x_n f(x_0, \ldots, x_n)$ be a PRTCTL formula in positive normal form, and M a structure. If for all $i \in [0..n]$, the only parameterized operators mentioning x_i are either all $\mathsf{U}^{\leq x_i}$ or all $\mathsf{G}^{\leq x_i}$, then $M \models Q_0 x_0 \ldots Q_n x_n f(x_0, \ldots, x_n)$ iff $M \models f(a_0, \ldots, a_n)$ where for $i \in [0..n]$,*

- *$a_i = 0$ if $Q_i = \forall$ and x_i only appears in U operators;*
- *$a_i = 0$ if $Q_i = \exists$ and x_i only appears in G operators;*
- *$a_i = |M|$ if $Q_i = \forall$ and x_i only appears in G operators;*
- *$a_i = |M|$ if $Q_i = \exists$ and x_i only appears in U operators.*

Theorem 2 immediately gives rise to the following corollary.

Corollary 2 *Given $M = (S, R, L)$ and monotone PRTCTL formula f, the model checking problem 'does $M \models f$' can be solved in time $\mathcal{O}(|M||f|)$.*

4. Timed Structures

Timed (Kripke) structures enhance the structures of the previous sections by labeling the transitions between states by integer values. For example, $(s, a, t) \in R$ if there is a transition from state s to state t taking a time units. $M = (S, R, L)$ is as before except that $R \subseteq S \times \mathbb{N} \times S$. In the sequel we will make a clear distinction between timed and untimed structures. CTL modalities have exactly the same meaning in timed and untimed structures; that is, these modalities are oblivious to the arc labels. Defined below are the relevant timed modalities for RTCTL where s is a state in the timed structure M and $n \in \mathbb{N}$. In particular, note that $\mathsf{EF}^{\leq n}$ modalities cannot be defined in terms of EX as is the case with CTL and RTCTL (c.f. [14] [3] [10]). Because A modalities can be seen to be shorthands for E modalities we leave out any explicit mention of them.

- $M, s \models \mathsf{E} f \mathsf{U}^{\leq n} g$ iff there is a path $\sigma = s_0 n_1 s_1 \ldots$ in M such that $s_0 = s$ and for some $i \in \mathbb{N}$, $M, s_i \models g$, $\sum_{j=1}^{i} n_j \leq n$ and for all $j < i$, $M, s_j \models f$.
- $M, s \models \mathsf{EG}^{\leq n} g$ iff there is a path $\sigma = s_0 n_1 s_1 \ldots$ in M such that $x_0 = s$ and for all $i \in \mathbb{N}$, if $\sum_{j=1}^{i} n_j \leq n$ then $M, x_i \models g$.

The following proposition shows that the model checking problem for timed modalities can be reduced to modalities whose quantities are no bigger than the length of the longest simple path in the structure M.

Proposition 4 *(c.f. [14]) Let $M = (S, R, L)$ be a timed structure with finite state set S. Let $c = |S| \max(\{m \in \mathbb{N} | (s, m, t) \in R\})$.*

- *$M, s \models \mathsf{E} f \mathsf{U}^{\leq c} g$ iff for all $n \in \mathbb{N}$, $n \geq c$, $M, s \models \mathsf{E} f \mathsf{U}^{\leq n} g$.*
- *$M, s \models \mathsf{EG}^{\leq c} g$ iff for all $n \in \mathbb{N}$, $n \geq c$, $M, s \models \mathsf{EG}^{\leq n} g$.*

Next, we show that the model checking problem for timed modalities can be solved efficiently in discrete timed structures.

Theorem 3 *Given timed structure $M = (S, R, L)$ and $a \in \mathbb{N}$, the model checking problem 'does $M \models \mathsf{E} p \mathsf{U}^{\leq a} q$' can be solved in time $\mathcal{O}(|M|^2)$ [1].*

Theorem 4 *Given timed structure $M = (S, R, L)$ and $a \in \mathbb{N}$, the model checking problem 'does $M \models \mathsf{EG}^{\leq a} p$' can be solved in time $\mathcal{O}(|M|)$.*

The following corollaries are a direct consequence of the model checking complexity of the algorithms for checking individual modalities and the semantics of parameterized formulae.

Corollary 3 *Given timed structure $M = (S, R, L)$ such that m is the largest natural number labeling a transition in M, and PRTCTL sentence f with k parameters, the model checking problem, does $M \models f$, can be solved in $\mathcal{O}((m|M|)^k |M|^2 |f|)$.*

Corollary 4 *Given timed structure $M = (S, R, L)$, such that m is the largest natural number labeling a transition in M, and monotone PRTCTL sentence f, the model checking problem, does $M \models f$, can be solved in time $\mathcal{O}(|M|^2 |f|)$.*

5. Event Sensitive Parametric Reasoning

Previously, the parameters of PRTCTL formulae referred to the passage of time only through a measure of the number of transitions in a path. Each transition represented some unit of time. It is often useful to be able to count the number of events, such as the number of times proposition p occurs along a path. To this end,

[1] A more detailed analysis shows that this problem can be solved in time $\mathcal{O}(|R| + |S| \log m)$ for m the largest transition label (c.f. [1][11]).

we enhance PRTCTL formulae with the ability to refer to independent events or propositions. For example, $\mathsf{AF}^{\leq 4send} response$ says that along all futures *response* occurs before more than 5 occurrences of *send*.

Let $p \in AP$, $n \in \mathbb{N}$ and $x \in \mathrm{Var}$. An atomic expression ae is of the form $\leq n \cdot p$ or $\leq x \cdot p$. Our methods outlined in the sequel are sufficient to handle arbitrary boolean combinations over AP in place of p; however, for simplicity we only discuss the expressions as defined below. An expression e is either an atomic expression or any positive boolean combination of expressions. Formulae of General Parametric CTL (GPCTL) are any formulae of PRTCTL combined with the modalities EU^e, AU^e, EG^e and AG^e for any expression e. Semantics over untimed structures $M = (S, R, L)$ are given for these new operators as follows.

- $M, s \models \mathsf{E}f\mathsf{U}^e g$ iff there is a path $\sigma = s_0 \ldots$ such that $s = s_0$ and there is an $i \in \mathbb{N}$ such that $M, s_i \models g$, for all $j < i$, $M, s_j \models f$ and $s_0 \ldots s_{i-1} \models e$.
- $M, s \models \mathsf{A}f\mathsf{U}^e g$ iff for all paths $\sigma = s_0 \ldots$ such that $s = s_0$, there is an $i \in \mathbb{N}$ such that $M, s_i \models g$, for all $j < i$, $M, s_j \models f$ and $s_0 \ldots s_{i-1} \models e$.
- $M, s \models \mathsf{EG}^e f$ iff there is a path $\sigma = s_0 \ldots$ such that $s = s_0$ and for all an $i \in \mathbb{N}$ such that $s_0 \ldots s_{i-1} \models e$, $M, s_i \models f$.
- $M, s \models \mathsf{AG}^e f$ iff for all paths $\sigma = s_0 \ldots$ such that $s = s_0$, for all $i \in \mathbb{N}$ such that $s_0 \ldots s_{i-1} \models e$, $M, s_i \models f$.
- $s_0 \ldots s_m \models \leq n \cdot p$ iff there are no more than n, positions, i, $i \in [0..m]$ such that $p \in L(s_i)$.
- $s_0 \ldots s_m \models e_0 \vee e_1$ iff $s_0 \ldots s_m \models e_0$ or $s_0 \ldots s_m \models e_1$.
- $s_0 \ldots s_m \models e_0 \wedge e_1$ iff $s_0 \ldots s_m \models e_0$ and $s_0 \ldots s_m \models e_1$.

We use $e(x)$ to refer to an expression in which the variable x appears. Given $e(x)$, and $n \in \mathbb{N}$ then $e(n)$ refers to the expression every where the same except that x has every where been replaced by n. Below we show that satisfaction of event quantified modalities can also be restricted to consideration of simple paths.

Proposition 5 *Let $M = (S, R, L)$, $s \in S$ and let $n > |S|$ then the following hold.*

- $M, s \models \mathsf{E}f\mathsf{U}^{e(n)} g$ *iff* $M, s \models \mathsf{E}f\mathsf{U}^{e(|S|)} g$.
- $M, s \models \mathsf{EG}^{e(n)} f$ *iff* $M, s \models \mathsf{EG}^{e(|S|)} f$.
- $M, s \models \mathsf{A}f\mathsf{U}^{e(n)} g$ *iff* $M, s \models \mathsf{A}f\mathsf{U}^{e(|S|)} g$.
- $M, s \models \mathsf{AG}^{e(n)} f$ *iff* $M, s \models \mathsf{AG}^{e(|S|)} f$.

Model checking for atomic event quantitative modalities can be done in a completely analogous manner to model checking timing properties, hence we have the following theorems. Let $p, q, r \in AP$ and $n \in \mathbb{N}$.

Theorem 5 *The model checking problem 'does $M, s \models \mathsf{E}q\mathsf{U}^{\leq n \cdot p} r$' can be solved in time $\mathcal{O}(|M|)$. The model checking problem 'does $M, s \models \mathsf{EG}^{\leq n \cdot p} q$' can be solved in time $\mathcal{O}(|M|)$.*

However, for more complex modalities the problem becomes decidedly more complex.

Theorem 6 *Let e be an expression with no variables. The model checking problem 'does $M, s \models \mathsf{E}p\mathsf{U}^e q$' is NP-complete.*

The following theorem and corollaries are a direct consequence of the semantics of parameterization and the previous proposition.

Theorem 7 *Let $M = (S, R, L)$ be a finite structure and $Q_n x_n \ldots Q_0 x_0 f(x_0, \ldots x_n)$ be a GPCTL sentence. $M, s \models Q_n x_n \ldots Q_0 x_0 f(x_0, \ldots x_n)$ iff $M, s \models Q_n x_n \leq |S| \ldots Q_0 x_0 \leq |S| f(x_0, \ldots x_n)$.*

Corollary 5 *Given $M = (S, R, L)$ and GPCTL formula f with k parameters and atomic exponents, the model checking problem 'does $M \models f$' can be solved in $\mathcal{O}(|M|^k |M||f|)$.*

Corollary 6 *Given $M = (S, R, L)$ and GPCTL formula f, the model checking problem 'does $M \models f$' can be solved in $\mathcal{O}(|M|^{|f|+1} |f|)$.*

Next we apply our results about monotone formulae to the GPCTL model checking problem.

Proposition 6 *$f(x)$ is a GPCTL sub-formula in positive normal form, with no quantifiers, M is a structure and s is a state in M.*

- *If the only parameterized operators in $f(x)$ are $\mathsf{EU}^{e(x)}$ and $\mathsf{AU}^{e(x)}$ then $M, s \models f(0)$ iff for all $x \leq |S|$, $M, s \models f(x)$.*
- *If the only parameterized operators in $f(x)$ are $\mathsf{EU}^{e(x)}$ and $\mathsf{AU}^{e(x)}$ then $M, s \models f(|S|)$ iff there exists $x \leq |S|$, $M, s \models f(x)$.*
- *If the only parameterized operators in $f(x)$ are $\mathsf{EG}^{e(x)}$ and $\mathsf{AG}^{e(x)}$ then $M, s \models f(|S|)$ iff for all $x \leq |S|$, $M, s \models f(x)$.*

- *If the only parameterized operators in $f(x)$ are $\mathsf{EG}^{e(x)}$ and $\mathsf{AG}^{e(x)}$ then $M, s \models f(|0|)$ iff there exists $x \leq |S|$, $M, s \models f(x)$.*

Theorem 8 *Let $Q_0 x_0 \ldots Q_n x_n f(x_0, \ldots, x_n)$ be a GPCTL formula in positive normal form, and M a structure. If for all $i \in [0..n]$, the only parameterized operators mentioning x_i are either all $\mathsf{U}^{e(x_i)}$ or all $\mathsf{G}^{e(x_i)}$, then $M \models Q_0 x_0 \ldots Q_n x_n f(x_0, \ldots, x_n)$ iff $M \models f(a_0, \ldots, a_n)$ where for $i \in [0..n]$,*

- *$a_i = 0$ if $Q_i = \forall$ and x_i only appears in U operators;*
- *$a_i = 0$ if $Q_i = \exists$ and x_i only appears in G operators;*
- *$a_i = |M|$ if $Q_i = \forall$ and x_i only appears in G operators;*
- *$a_i = |M|$ if $Q_i = \exists$ and x_i only appears in U operators.*

Corollary 7 *Given $M = (S, R, L)$ and monotone GPCTL formula f with atomic superscripts, the model checking problem, 'does $M \models f$' can be solved in time $\mathcal{O}(|M||f|)$.*

6. Conclusion

We have presented PRTCTL, a parameterized quantitative temporal logic, and given an efficient model checking algorithm for a large subclass of the logic. PRTCTL is an extension to RTCTL that allows the expression of abstract quantitative formulae which cannot be expressed in the standard quantitative logics without parameterization.

[4] shows that parametric reasoning is possible for a restricted class of parameterized timed automata, although the general problem is undecidable. There, the parameterization is given directly in terms of the automaton definition. [20] defines PTCTL, a parametric version of TCTL [2] and uses PTCTL to characterize the parameter valuations for which a timed automaton will satisfy the instantiated TCTL formula. This characterization is given in terms of a set of linear equations. [20] shows that this parametric timing analysis problem is solvable in time double exponential in the size of the timed automaton or structure. Our approach has been to work with a more restrictive type of structure and a different kind of parameterized formula. Specifically, we do not allow the use of resettable clocks or arcs labeled with elements of a dense domain in the definition of structures but we do allow quantification over parameter values, something which is not allowed in PTCTL. With these differences we are able to show a substantial improvement in complexity and give a more standard model checking approach to the problem of evaluating parameterized formulae over structures. We note that our approach is amenable to recovering instantiations of the parameters for which the formula holds in the structure, if such exist (c.f. [4] [20]).

Finally we have given algorithms for PRTCTL over timed structures and GPCTL over untimed structures. Our analysis of PRTCTL draws directly from and enhances [14], particularly theorem 3. We have shown that the GPCTL model checking problem is NP-complete even for the case when there are no parameters. However, for sentences with only atomic superscripts and a fixed k number of parameters, the GPCTL model checking problem is solvable in time polynomial in the size of the structure. Our work differs from that of [21] in that our logic makes no use of so called *freeze quantifiers* ([2] [5]). Freeze quantification is a type of local quantification whereas the parameterization used here is a global quantification. There does not seem to be an obvious translation from PRTCTL into TCTL but this remains an interesting open question. The fact that GPCTL can express quantitative properties of independent actions differentiates this logic from many others in the literature. Along any computation the number of occurrences of each event must be monotonically non-decreasing but they may each may increase at their own rate. This type of independence is not allowed in either [21] or in [2] although the exact restrictions are somewhat different in the two papers. The logics defined here are not as general as those that combine the full power of Presberger arithmetic with temporal logics but they also avoid the increased complexity for model checking associated with the added power [7] [8].

References

[1] R. Ahuja, K. Mehlhorn, B. Orlin, and R. Tarjan. Faster algorithms for the shortest path problem. Technical Report 193, MIT Operations Research Center, 1988.

[2] R. Alur, C. Courcoubetis, and D. Dill. Model checking for real-time systems. In *Proceedings of the 5th Annual Symposium on Logic in Computer Science*, pages 414–425. IEEE Computer Society Press, New York, 1990.

[3] R. Alur and T. Henzinger. Logics and models of real time: A survey. In J. de Bakker, K. Huizing, W.-P. de Roever, and G. Rozenberg, editors, *Real Time: Theory in Practice*, pages 74–106. Springer-Verlag, New York, 1992. Lecture Notes in Computer Science vol. 600.

[4] R. Alur, T. Henzinger, and M. Y. Vardi. Parametric real-time reasoning. In *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing.* ACM, 1993.

[5] R. Alur and T. A. Henzinger. A really temporal logic. In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science*, pages 164–169. IEEE Computer Society Press, New York, 1989.

[6] I. Beer, S. Ben-David, C. Eisner, and A. Landver. Rulebase: an industry-oriented formal verification tool. In *33rd Design Automation Conference.* ACM, 1996.

[7] A. Bouajjani, R. Echahed, and P. Habermehl. On the verification problem of nonregular properties for nonregular processes. In *Symposium on Logics in Computer Science*, pages 123–133. IEEE, 1995.

[8] A. Bouajjani, R. Echahed, and P. Habermehl. Verifying infinite state processes with sequential and parallel composition. In *POPL*, pages 95–106. ACM, 1995.

[9] S. Campos, E. Clarke, W. Marrero, M. Minea, and H. Hirashi. Computing quantitative characteristics of finite-state real-time systems. In *Real-Time Systems Symposium.* IEEE, 1994.

[10] E. M. Clarke and E. A. Emerson. Design and verification of synchronization skeletons using branching time temporal logic. In *Logics of Programs Workshop, IBM Yorktown Heights, New York*, pages 52–71. Springer, May 1981. Lecture Notes in Computer Science vol. 131.

[11] H. Cormen, C. Leiserson, and R. Rivest. *Introduction to Algorithms.* MIT Press, 1990.

[12] E. A. Emerson. *Temporal and Modal Logic*, chapter 16, pages 996–1072. Elsevier Science Publishing, 1990. J. van Leeuwen, Editor.

[13] E. A. Emerson. Real–time and the μ–calculus. In *Proceedings of Real-Time: Theory in Practice*, pages 176–194. Springer, June 1992. Lecture Notes in Computer Science vol. 600.

[14] E. A. Emerson, A. K. Mok, A. P. Sistla, and J. Srinivasan. Quantitative temporal reasoning. In E. M. Clarke and R. Kurshan, editors, *CAV90: Computer-aided Verification*, pages 136–145. Springer-Verlag, 1990. Lecture Notes in Computer Science vol. 531. Journal version appears in Journal of Real Time Systems, vol. 4, pp. 331-352, 1992.

[15] E. A. Emerson and R. J. Trefler. Generalized quantitative temporal reasoning: An automata- theoretic approach. In M. Bidoit and M. Dauchet, editors, *TAPSOFT 97: Theory and Practice of Software Development*, pages 189–200. Springer-Verlag, 1997.

[16] D. Harel and A. Pnueli. On the development of reactive systems. In K. Apt, editor, *Logics and Models of Concurrent Systems*, pages 477–498. Springer-Verlag, 1985. NATO Advanced Summer Institutes, vol. F-13.

[17] F. Jahanian and A. Mok. Safety analysis of timing properties in real-time systems. *IEEE Transactions on Software Engineering*, 1986.

[18] A. Pnueli. The temporal logic of programs. In *18th annual IEEE-CS Symp. on Foundations of Computer Science*, pages 46–57. IEEE, 1977.

[19] J. P. Queille and J. Sifakis. Specification and verification of concurrent programs in cesar. In *Proc. 5th Int. Symp. on Programming*, pages 195–220. Springer, 1982.

[20] F. Wang. Parametric timing analysis for real-time systems. *Information and Computation*, pages 131–150, 1996.

[21] J. Yang, A. Mok, and F. Wang. Symbolic model checking for event-driven real-time systems. In *ACM Transactions on Programming Languages and Systems.* ACM, ACM, 1997.