Fundamenta Informaticae 59 (2012) 1–19
IOS Press

Bounded Parametric Verification for Distributed Time Petri Nets with Discrete-Time Semantics *

Michał Knapik, Wojciech Penczek†, Maciej Szreter
Polish Academy of Sciences, ICS,
Ordona 21, 01-237 Warsaw, Poland
{*mknapik,penczek,mszreter*}*@ipipan.waw.pl*

Agata Półrola
University of Łódź, FMCS,
Banacha 22, 90-238 Łódź, Poland
polrola@math.uni.lodz.pl

Abstract. Bounded Model Checking (BMC) is an efficient technique applicable to verification of temporal properties of (timed) distributed systems. In this paper we show for the first time how to apply BMC to parametric verification of time Petri nets with discrete-time semantics. The properties are expressed by formulas of the logic PRTECTL - a parametric extension of the existential fragment of Computation Tree Logic (CTL).

1. Introduction

In order to check whether a system satisfies a given property, both the system and the property in question ought to be abstracted. For complex, timed, multi-process or reactive systems a variety of approaches exist – among them various kinds of Petri nets and timed automata as system abstractions (models), and temporal, epistemic, timed, and dynamic logics as the property specification languages. Checking automatically whether the model conforms to a given specification is known as model checking [9]. One of the approaches used to represent distributed systems with timing dependencies [6, 7, 16], is *time Petri nets* (TPNs) by Merlin and Farber [17]. In this paper we consider (distributed) time Petri nets with discrete-time semantics [19].

*Partly supported by the Polish Ministry of Science and Higher Education under the grant No. N N206 258035.
†Also affiliated with: Institute of Informatics, Podlasie Academy, Sienkiewicza 51, 08-110 Siedlce, Poland

Bounded model checking (BMC) is an efficient verification method whose main idea consists in considering a model truncated up to a specific depth. In this work we use SAT-based BMC verification, consisting in translating a model checking problem solvable on a fraction of a model into a test of propositional satisfiability, which is then made using a SAT-checker. The method has been successfully applied to verification of both timed and untimed systems [2, 3, 4, 8, 12, 22, 27, 29]. This paper shows how to adapt BMC methods, presented in [22, 28, 29, 30] and developed for timed automata, to the case of time Petri nets and parametric properties. The properties are expressed by formulas of the logic PRTECTL [11] - a parametric extension of the existential fragment of Computation Tree Logic (CTL). This is a relatively untouched area of research. We believe that our approach shows how to apply parametric model checking in practice as this problem is known to be of high complexity [5].

The rest of the paper is organized as follows. In Section 2 a related work is discussed. Section 3 recalls from [11] the syntax and semantics of the logics used in this paper. Section 4 introduces k-models together with the bounded semantics for $\mathrm{vRTECTL}$ and $\mathrm{PRTECTL}$. In Section 5 the translation of a model and a property under investigation is presented together with an algorithm for BMC. Section 6 contains an application of the above method to time Petri Nets. Experimental results for the Generic Pipelining Paradigm are shown in Section 7. The concluding remarks are in Section 8.

2. Related Work

The logics investigated in this paper were introduced in [11], whereas an application of BMC to the existential fragment of CTL originates from [21] with a further optimisation shown in [31]. While self-contained, this paper can be considered as an extension of [15] and [20]. The work presented here falls into a broad area of the Parametric Model Checking – an ambiguous term which may mean that we deal with the parameters in models (as in [1] and [13]), in logics (as in [11] and [5]) or in both (as in [26]). There are two reasons limiting the practical applications of the Parametric Model Checking. The first one – computational complexity of the problem – is the result of the presence of satisfiability in the Presburger Arithmetic (PA) as a subproblem. In case of the translation of the existential fragment of TCTL to PA formulae proposed in [5], the joint complexity of the solution is $3\mathrm{EXPTIME}$. The second – undecidability of the problem for Parametric Timed Automata in general [1] – results in the fact that some of the proposed algorithms may not to stop [13].

3. Parameterized Temporal Logics

In this section we recall two extensions of Computation Tree Logic (CTL) [10], introduced in [11]. The main logic considered in this paper, namely PRTCTL logic, is built of expressions containing CTL–like modalities with additional parametric superscripts expressing the time lengths of paths. The universal and existential quantifiers over forementioned parameters are also allowed in PRTCTL. The vRTCTL logic can be perceived as an intermediate logic in building PRTCTL sentences. The formulae of vRTCTL do not contain quantifiers and are evaluated under the accompanied parameter valuations. To give an example, consider vRTCTL formula $\phi(\Theta) = EF(AG^{\leq\Theta} receivingData \implies EG^{\leq 2\Theta}\neg socketOpen)$. Under the parameter valuation υ such that $\upsilon(\Theta) = 2$ we evaluate $[\phi(\Theta)]_\upsilon = EF(AG^{\leq 2} receivingData \implies EG^{\leq 4}\neg socketOpen)$. Introducing a universal quantifier over parameter Θ bounded by 10 we got $\psi = \forall_{\Theta\leq 10} EF(AG^{\leq\Theta} receivingData \implies EG^{\leq 2\Theta}\neg socketOpen)$. The

meaning of ψ is that for each time bound Θ smaller or equal than 10 there exists a future state such that if every execution path starts with consecutive data receiving steps taking Θ time together, then among these paths there is one such that the data input socket is closed for time of 2Θ. Therefore, we have expressed a qualitative property of the system without using any concrete values.

We interpret formulae, following the approach of Emerson and Trefler [11], in timed Kripke structures – i.e. standard Kripke structures having the transitions labelled with natural numbers. The value of a label shows how long it takes to traverse the transition. Throughout this paper by $\mathbb{N}$ we denote the set of all natural numbers (including 0), and by $\alpha(\Theta_1, \ldots, \Theta_n)$ we point out that the formula α contains free parameters $\Theta_1, \ldots, \Theta_n$.

3.1. Syntax

Let $\Theta_1, \ldots, \Theta_n$ be variables, called here *parameters*. An expression of the form $\eta = \sum_{i=1}^{n} c_i \cdot \Theta_i + c_0$, where $c_0, \ldots, c_n \in \mathbb{N}$, is called a *linear expression*. A function $\upsilon : \{\Theta_1, \Theta_2, \ldots, \Theta_n\} \longrightarrow \mathbb{N}$ is called a *parameter valuation*. Let Υ be the set of all the parameter valuations.

Definition 3.1. Let $\mathcal{PV}$ be a set of propositional variables containing the symbol *true*. We define inductively the formulae of vRTCTL :

1. every member of $\mathcal{PV}$ is a formula,
2. if α and β are formulae, then so are $\neg\alpha$, $\alpha \wedge \beta$ and $\alpha \vee \beta$,
3. if α and β are formulae, then so are $EX\alpha$, $EG\alpha$, and $E\alpha U\beta$,
4. if η is a linear expression, α and β are formulae of vRTCTL, then so are $EG^{\leq\eta}\alpha$ and $E\alpha U^{\leq\eta}\beta$.

The conditions $1, 2$, and 3 alone define CTL. Notice that η is allowed to be a constant. The logic defined by a modification of the above definition, where $\eta = a$ for $a \in \mathbb{N}$, is called RTCTL in [11].

Definition 3.2. The formulae of PRTCTL are defined as follows:

1. if $\alpha \in \mathrm{vRTCTL}$, then $\alpha \in \mathrm{PRTCTL}$,
2. if $\alpha(\Theta) \in \mathrm{PRTCTL}$, where Θ is a free parameter, then $\forall_\Theta\alpha(\Theta), \exists_\Theta\alpha(\Theta), \forall_{\Theta\leq a}\alpha(\Theta), \exists_{\Theta\leq a}\alpha(\Theta) \in \mathrm{PRTCTL}$, for $a \in \mathbb{N}$.

3.2. Semantics

We evaluate the truth of the sentences and the formulae accompanied with parameter valuations in timed Kripke structures [11].

Definition 3.3. Let $\mathcal{PV}$ be a set of propositional variables containing the symbol *true*. A timed Kripke structure (a *model*) is a tuple $(S, s^0, \rightarrow, \mathcal{L})$, where S is a finite set of *states*, $s^0 \in S$ is an *initial state*, $\rightarrow \subseteq S \times \mathbb{N} \times S$ is a transition relation such that for every $s \in S$ there exists $s' \in S$ and $n \in \mathbb{N}$ with $(s, n, s') \in \rightarrow$ (i.e., the relation is total), and $\mathcal{L} : S \longrightarrow 2^{\mathcal{PV}}$ is a labelling function satisfying $true \in \mathcal{L}(s)$ for each $s \in S$.

By $s \xrightarrow{n} s'$ we denote that $(s, n, s') \in \rightarrow$. The intuitive meaning of $s \xrightarrow{n} s'$ is that it takes n time units to reach s' from s. The labelling function assigns to each state s a set of propositions which are assumed to be true at s. An infinite sequence $\pi = (s_0, n_0, s_1, n_1, \ldots)$ such that s_i are states of a model and $s_i \xrightarrow{n_i} s_{i+1}$ for $i \in \mathbb{N}$ is called a *path*. By $\pi(i)$ we denote the i–th state present on a path π. We define the *time distance* between positions $\pi(0)$ and $\pi(j)$ as $\delta_\pi^j := \sum_{i=0}^{j-1} n_i$, assume that $\delta_\pi^0 = 0$, and $lim_{j\to\infty}\delta_\pi^j = \infty$ (we consider *progressive* paths only). The number of the states of M is called the size of M and denoted by $|M|$. For a parameter valuation υ and a linear expression η, by $\upsilon(\eta)$ we mean the evaluation of η under υ.

Definition 3.4. (Semantics of vRTCTL**)**
Let M be a model, s – a state, α, β – formulae of vRTCTL. $M, s \models_\upsilon \alpha$ denotes that α is true at the state s in the model M under the parameter valuation υ. We omit M where it is implicitly understood. The relation $\models_\upsilon$ is defined inductively as follows:

1. $s \models_\upsilon p$ iff $p \in \mathcal{L}(s)$
2. $s \models_\upsilon \neg p$ iff $p \notin \mathcal{L}(s)$,
3. $s \models_\upsilon \alpha \wedge \beta$ iff $s \models_\upsilon \alpha$ and $s \models_\upsilon \beta$,
4. $s \models_\upsilon \alpha \vee \beta$ iff $s \models_\upsilon \alpha$ or $s \models_\upsilon \beta$,
5. $s \models_\upsilon EX\alpha$ iff $\exists_\pi\big(\pi(0) = s$ and $\pi(1) \models_\upsilon \alpha\big)$,
6. $s \models_\upsilon EG\alpha$ iff $\exists_\pi\big(\pi(0) = s$ and $\forall_{i\geq 0}\pi(i) \models_\upsilon \alpha\big)$,
7. $s \models_\upsilon E\alpha U\beta$ iff $\exists_\pi\big(\pi(0) = s$ and $\exists_{i\geq 0}\big[\pi(i) \models_\upsilon \beta$ and $\forall_{0\leq j<i}\pi(j) \models_\upsilon \alpha\big]\big)$,
8. $s \models_\upsilon EG^{\leq\eta}\alpha$ iff $\exists_\pi\big(\pi(0) = s$ and $\forall_{i\geq 0}(\delta_\pi^i \leq \upsilon(\eta)$ implies $\pi(i) \models_\upsilon \alpha)\big)$,
9. $s \models_\upsilon E\alpha U^{\leq\eta}\beta$ iff $\exists_\pi\big(\pi(0) = s$ and $\exists_{i\geq 0}\big[\delta_\pi^i \leq \upsilon(\eta)$ and $\pi(i) \models_\upsilon \beta$ and $\forall_{0\leq j<i}\pi(j) \models_\upsilon \alpha\big]\big)$.

Notice that if a vRTCTL formula contains no parameters (i.e., it is an RTCTL formula), then the parameter valuation υ is irrelevant and may be omitted.

Definition 3.5. (Semantics of PRTCTL**)**
Let M be a model, s – a state, and α – a formula of PRTCTL. $M, s \models \alpha$ denotes that α holds at the state s in the model M. The relation $\models$ is defined inductively as follows:

1. $s \models \forall_\Theta\alpha(\Theta)$ iff $\forall_{i_\Theta\geq 0} s \models \alpha(\Theta \leftarrow i_\Theta)$,
2. $s \models \forall_{\Theta\leq a}\alpha(\Theta)$ iff $\forall_{0\leq i_\Theta\leq a} s \models \alpha(\Theta \leftarrow i_\Theta)$,
3. $s \models \exists_\Theta\alpha(\Theta)$ iff $\exists_{i_\Theta\geq 0} s \models \alpha(\Theta \leftarrow i_\Theta)$,
4. $s \models \exists_{\Theta\leq a}\alpha(\Theta)$ iff $\exists_{0\leq i_\Theta\leq a} s \models \alpha(\Theta \leftarrow i_\Theta)$,

where i_Θ is a fresh integer variable and $\alpha(\Theta \leftarrow i_\Theta)$ means that Θ is substituted by i_Θ in α.

The following lemma[1], adapted from Proposition 4 from [11], states that in evaluating formulae of vRTCTL we can restrict the values of parameter valuations by a certain model-induced constant.

Lemma 3.1. Let $M = (S, s^0, \rightarrow, \mathcal{L})$ be a timed Kripke structure, $\alpha \in$ vRTCTL, υ – a parameter valuation, and n_{max} – the greatest time label of a time transition present in M.
Then, $M, s \models_\upsilon \alpha$ iff $M, s \models_{\upsilon'} \alpha$, where $\upsilon'(\Theta) = min\{n_{max} \cdot |M|, \upsilon(\Theta)\}$ for each parameter Θ.

A direct application of Lemma 3.1 yields the next theorem - an extension of Theorem 1 of [11].

Theorem 3.1. Let M be a timed Kripke structure and $Q_1{}_{\Theta_1 \leq a_1} \ldots Q_n{}_{\Theta_1 \leq a_n} f(\Theta_1, \ldots, \Theta_n)$, where $Q_i \in \{\exists, \forall\}$, $a_i \in \mathbb{N} \cup \{\infty\}$, be a PRTCTL sentence. Then, $M, s \models Q_1{}_{\Theta_1 \leq a_1} \ldots Q_n{}_{\Theta_1 \leq a_n} f(\Theta_1, \ldots, \Theta_n)$ iff $M, s \models Q_1{}_{\Theta_1 \leq min\{a_1, n_{max}|M|\}} \ldots Q_n{}_{\Theta_1 \leq min\{a_n, n_{max}|M|\}} f(\Theta_1, \ldots, \Theta_n)$.

Notice that the direct consequence of the above theorem is that each unbounded quantifier can be replaced, without changing the validity of a formula in a fixed model M, with a version bounded with the $n_{max}|M|$ value. Similarly we can argue that non-superscripted modalities can be replaced by similar formulae with $\leq n_{max}|M|$ superscript. Therefore, from now on we omit the unbounded quantifiers and non-superscripted EU and EG in our considerations. The methods presented in this paper, as typical for BMC, are applied to the *existential* parts of the considered logics only.

Definition 3.6. The logics vRTECTL, RTECTL, and PRTECTL are defined as the restrictions of vRTCTL, RTCTL, and the set of sentences of PRTCTL such that the negation can be applied to the propositions only, respectively.

4. Bounded Semantics

We aim at verifying the properties expressed in the existential fragments of the considered logics by means of bounded model checking. Intuitively, to this end we unfold the computation tree of a given model to a limited depth and check the validity of the property in question along such a finite structure.

Definition 4.1. Let M be a model, and $k \in \mathbb{N}$. By $Path_k$ let us denote the set of all the sequences $(s_0, n_0, s_1, n_1, \ldots, s_k)$, where s_i is a state and $s_i \xrightarrow{n_i} s_{i+1}$ for all $0 \leq i < k$. The pair $(Path_k, \mathcal{L})$ is called the *k–model* of M and denoted by M_k. An element $\pi_k \in Path_k$ is called a *k–path*.

Definition 4.2. Let M_k be the k-model of M and $\pi_k \in Path_k$. Define the function $loop : Path_k \longrightarrow 2^{\mathbb{N}}$ as $loop(\pi_k) = \{l \mid l \leq k \wedge \exists_{m \in \mathbb{N}}(\pi_k(k) \xrightarrow{m} \pi_k(l))\}$.

A k-path π_k is called a *loop* if $loop(\pi_k) \neq \emptyset$. Observe that loops are essentially a way of representing some infinite paths in a finite way. The previously defined time distance δ^j_π function extends to time distance $\delta^j_{\pi_k}$ along a k–path π_k in a natural way. We abbreviate $\delta^k_{\pi_k}$ as δ_{π_k}. Similarly, for a k–path π_k and $c \in \mathbb{N}$ we define $\Delta^c_{\pi_k} = max\{i \mid i \leq k \wedge \delta^i_{\pi_k} \leq c\}$ - the maximal index i of π_k s.t. $\delta^i_{\pi_k} \leq c$.

[1] Due to the space limit, the more involved proofs have been moved to Appendix.

Definition 4.3. (Bounded semantics for vRTECTL**)**
Let $M_k = (Path_k, \mathcal{L})$ be the k–model of M, s – a state, $p \in \mathcal{PV}$, $\alpha, \beta \in$ vRTECTL and υ – a parameter valuation. By $M_k, s \models_\upsilon \alpha$ we denote that α holds in M_k under valuation υ. We omit M_k where it is implicitly understood. Define the relation $\models_\upsilon$ as following:

1. $s \models_\upsilon p$ iff $p \in \mathcal{L}(s)$,
2. $s \models_\upsilon \neg p$ iff $p \notin \mathcal{L}(s)$,
3. $s \models_\upsilon \alpha \vee \beta$ iff $s \models_\upsilon \alpha$ or $s \models_\upsilon \beta$,
4. $s \models_\upsilon \alpha \wedge \beta$ iff $s \models_\upsilon \alpha$ and $s \models_\upsilon \beta$,
5. $s \models_\upsilon EX\alpha$ iff $\exists_{\pi_k \in Path_k}\big(\pi_k(0) = s \text{ and } \pi_k(1) \models_\upsilon \alpha\big)$,
6. $s \models_\upsilon EG^{\leq \eta}\alpha$ iff $\exists_{\pi_k \in Path_k}\big[\pi_k(0) = s \text{ and } \big((\delta_{\pi_k} > \upsilon(\eta) \text{ and } \forall_{i \leq \Delta_{\pi_k}^{\upsilon(\eta)}} \pi_k(i) \models_\upsilon \alpha)$ or $(\delta_{\pi_k} \leq \upsilon(\eta) \text{ and } \forall_{i \leq k} \pi_k(i) \models_\upsilon \alpha \text{ and } loop(\pi_k) \neq \emptyset)\big)\big]$,
7. $s \models_\upsilon E\alpha U^{\leq \eta}\beta$ iff $\exists_{\pi_k \in Path_k}\big(\pi_k(0) = s \text{ and } \exists_{i \leq \Delta_{\pi_k}^{\upsilon(\eta)}}(\pi_k(i) \models_\upsilon \beta \text{ and } \forall_{j<i} \pi_k(j) \models_\upsilon \alpha)\big)$.

Definition 4.4. (Bounded semantics for PRTECTL**)**
Let M_k be the k–model of M, n_{max} – the greatest time label of the transitions present in M, s – a state, α – a sentence (a formula without free variables) of PRTECTL and $a \in \mathbb{N}$. Recursively define the relation $\models$ as follows:

1. $M_k, s \models \forall_{\Theta \leq a}\alpha(\Theta)$ iff $\forall_{0 \leq i_\Theta \leq min\{a, k \cdot n_{max}\}} M_k, s \models \alpha(\Theta \leftarrow i_\Theta)$,
2. $M_k, s \models \exists_{\Theta \leq a}\alpha(\Theta)$ iff $\exists_{0 \leq i_\Theta \leq min\{a, k \cdot n_{max}\}} M_k, s \models \alpha(\Theta \leftarrow i_\Theta)$,

where i_Θ is a fresh integer variable.

The following lemma states some fundamental features of the bounded semantics. Namely, the vRTECTL and PRTECTL properties which are valid in some k–model hold also in the whole model M. On the other hand, if a considered property holds in M, it is also valid in some k–model, where $k \leq |M|$.

Lemma 4.1. Let $k, l \in \mathbb{N}$ such that $k \leq l$, M_k be the k–model of M, and s – a state. Consider a formula $\alpha \in$ vRTECTL together with a parameter valuation υ, and a sentence $\beta \in$ PRTECTL. The following conditions hold:

(1). $M_k, s \models_\upsilon \alpha$ implies $M_l, s \models_\upsilon \alpha$, and $M_k, s \models \beta$ implies $M_l, s \models \beta$,

(2). $M_k, s \models_\upsilon \alpha$ implies $M, s \models_\upsilon \alpha$, and $M_k, s \models \beta$ implies $M, s \models \beta$,

(3). $M, s \models_\upsilon \alpha$ implies $M_{|M|}, s \models_\upsilon \alpha$, and $M, s \models \beta$ implies $M_{|M|}, s \models \beta$.

5. Bounded Model Checking

Having presented the bounded semantics, we now apply the bounded model checking method to the verification of properties formulated in vRTECTL and PRTECTL. The BMC method is based on the idea of a translation of a part of the model together with a property in question to a propositional formula. Satisfiability of the result means that the translated formula holds in the model.

5.1. Submodels

In order to obtain an acceptable efficiency, the BMC algorithm works on submodels of the k–model.

Definition 5.1. Let $M_k = (Path_k, \mathcal{L})$ be the k-model. A substructure $M'_k = (Path'_k, \mathcal{L}')$, where $Path'_k \subseteq Path_k$ and $\mathcal{L}'$ is the restriction of $\mathcal{L}$ to the states present in the paths of $Path'_k$, is called a *submodel* of M_k.

The bounded semantics of vRTECTL formulae and PRTECTL sentences over submodels is defined as for k–models. If $M'_k = (Path'_k, \mathcal{L}')$ and $M''_k = (Path''_k, \mathcal{L}'')$ are submodels of some k–model M_k, such that $Path''_k \subseteq Path'_k$, we write $M''_k \subseteq M'_k$.

Lemma 5.1. Let M_k be the k–model, M'_k and M''_k – its submodels, such that $M''_k \subseteq M'_k$ and s – a state present in some path of M''_k. Then, we have:

1. $M''_k, s \models_\upsilon \alpha \Rightarrow M'_k, s \models_\upsilon \alpha$, for $\alpha \in$ vRTECTL and any parameter valuation υ,
2. $M''_k, s \models \alpha \Rightarrow M'_k, s \models \alpha$, for $\alpha \in$ PRTECTL.

Proof:
The first part of the lemma is easily proved by the structural induction. In order to prove the second part, notice that in the bounded semantics the non-modal quantifiers are rewritten as, respectively, conjunctions or disjunctions, and use the result of the first part. □

It was proved in [21] that in order to determine the truth of an ECTL formula in M_k it is sufficient to consider only submodels of a size given by a special function on the checked formula. In this paper we extend these results.

Definition 5.2. Let $\alpha, \beta \in$ PRTECTL, p be an atomic proposition, and η - a linear expression. Define recursively the special function f_k : PRTECTL $\longrightarrow \mathbb{N}$ as follows:

1. $f_k(p) = f_k(\neg p) = 0$,
2. $f_k(\alpha \vee \beta) = max(f_k(\alpha), f_k(\beta))$,
3. $f_k(\alpha \wedge \beta) = f_k(\alpha) + f_k(\beta)$,
4. $f_k(EX\alpha) = f_k(\alpha) + 1$,
5. $f_k(EG^{\leq \eta}\alpha) = (k+1) \cdot f_k(\alpha) + 1$,
6. $f_k(E\alpha U^{\leq \eta}\beta) = k \cdot f_k(\alpha) + f_k(\beta) + 1$,
7. $f_k(\forall_{\Theta \leq c}\beta(\Theta)) = (c+1) \cdot f_k(\beta(\Theta))$,
8. $f_k(\exists_{\Theta \leq c}\beta(\Theta)) = f_k(\beta(\Theta))$.

The following lemmas state that we can determine the truth of vRTECTL and PRTECTL formulae in the k-model using submodels of size bounded by the value of the appropriate function.

Lemma 5.2. Let $\alpha \in$ vRTECTL, M_k be the k-model and υ – a parameter valuation. For any state s present in some path of M_k, $M_k, s \models_\upsilon \alpha$ if and only if there exists a submodel M'_k of M_k such that $M'_k, s \models_\upsilon \alpha$ and $|Path'_k| \leq f_k(\alpha)$.

Proof:
See Appendix. □

Lemma 5.3. Let β be a PRTECTL sentence and M_k be the k-model. For any state s present in some path of M_k, $M_k, s \models \beta$ if and only if there exists a submodel M'_k of M_k such that $M'_k, s \models \beta$ and $|Path'_k| \leq f_k(\beta)$.

Proof:
Straightforward induction with respect to the number of parameters, using Lemma 5.2. □

5.2. Translation to SAT

In order to translate the problem of validity of a sentence $\alpha \in$ PRTECTL in the submodel M'_k to the problem of satisfiability of a propositional formula $\left[\alpha\right]_k$ we encode M'_k and α into propositional logic, and then combine the results together. We present an adapted version of the efficient translation introduced in [31].

Consider a model M. As the number of the states of M is finite, they can be encoded as bit vectors of length $r = \lceil log|M| \rceil$. Therefore, we can represent the states as the valuations of the vector $w = (w_1, \ldots, w_r)$. Due to the fact that we need to count the time passed along a path, we augment the representation of states with an additional bit vector $d = (d_1, \ldots, d_z)$ of length $z = \lceil \log(k \cdot n_{max}) \rceil$ where n_{max} denotes, as previously, the maximal value of transition label present in M. The vector $\mathbf{w} = (w_1, \ldots, w_r, d_1, \ldots, d_z)$ is called a *global state variable*, while each its member is called a *state variable*. Moreover, by $w_\mathbf{w}$ and $d_\mathbf{w}$ we denote the subvectors w and d of the vector $\mathbf{w}$. Denote by $\mathcal{SV}$ a set of state variables. A valuation $V : \mathcal{SV} \longrightarrow \{0,1\}$ naturally extends to the valuations $\hat{V} : \mathcal{SV}^r \longrightarrow \{0,1\}^r$ and $\hat{D} : \mathcal{SV}^z \to \{0,1\}^z$ in such a way that $\hat{V}(w_1, \ldots, w_r) = (V(w_1), \ldots, V(w_r))$ and $\hat{D}(d_1, \ldots, d_z) = (V(d_1), \ldots, V(d_z))$. With a slight notational abuse, we denote by $\hat{V}(\mathbf{w})$ the state encoded by $w_\mathbf{w}$, and by $\hat{D}(\mathbf{w})$ - the value of time encoded by $d_\mathbf{w}$. The *symbolic k-path* is a vector of global state variables. As we need a number of symbolic k-paths to represent the k-paths in a translated submodel, by $(\mathbf{w}_{0,i}, \mathbf{w}_{1,i}, \ldots, \mathbf{w}_{k,i})$ we denote the i-th symbolic k-path, where $\mathbf{w}_{j,i}$ is a global state variable.

Let $\mathbf{w}, \mathbf{w}'$ be global state variables, s a state and p a proposition. In the rules of the translation the following propositional formulae are used:

1. $p(\mathbf{w})$ denotes a formula such that $V \models p(\mathbf{w})$ iff $p \in \mathcal{L}(\hat{V}(\mathbf{w}))$,
2. $T(\mathbf{w}, \mathbf{w}')$ denotes a formula such that $V \models T(\mathbf{w}, \mathbf{w}')$ iff $\hat{V}(\mathbf{w}) \to \hat{V}(\mathbf{w}')$ (i.e., there exists a transition between $\hat{V}(\mathbf{w})$ and $\hat{V}(\mathbf{w}')$ in the model M, but the values of $\hat{D}(\mathbf{w})$, $\hat{D}(\mathbf{w}')$ are not taken into account),
3. $Td(\mathbf{w}, \mathbf{w}')$ denotes a formula such that $V \models Td(\mathbf{w}, \mathbf{w}')$ iff $\hat{V}(\mathbf{w}) \stackrel{n}{\to} \hat{V}(\mathbf{w}')$ for some $n \in \mathbb{N}$, and $\hat{D}(\mathbf{w}') = \hat{D}(\mathbf{w}) + n$ (i.e., there exists a transition between $\hat{V}(\mathbf{w})$ and $\hat{V}(\mathbf{w}')$ in the model

M, and the difference between $\hat{D}(\mathbf{w}')$ and $\hat{D}(\mathbf{w})$ corresponds to the time passed while performing this transition),

4. $H(\mathbf{w},\mathbf{w}')$ is a formula s.t. $V \models H(\mathbf{w},\mathbf{w}')$ iff $\hat{V}(\mathbf{w}) = \hat{V}(\mathbf{w}')$ (encodes equality of states),
5. $L_k(j) = \bigvee_{i=0}^{k} T(\mathbf{w}_{k,j}, \mathbf{w}_{i,j})$ encodes a loop, i.e., $V \models L_k(j)$ iff $loop((\hat{V}(\mathbf{w}_{0,j}), .., \hat{V}(\mathbf{w}_{k,j}))) \neq \emptyset$,
6. $I_{s_0}(\mathbf{w})$ is a formula s.t. $V \models I_{s_0}(\mathbf{w})$ iff $\hat{V}(\mathbf{w}) = s$ and $\hat{D}(\mathbf{w}) = 0$ (encodes the initial state),
7. $Z(\mathbf{w})$ is a formula s.t. $V \models Z(\mathbf{w})$ iff $\hat{D}(\mathbf{w}) = 0$ ($\mathbf{w}$ encodes the first state of a path),
8. $Le(\mathbf{w}, a)$, for $a \in \mathbb{N}$, is a formula such that $V \models Le(\mathbf{w}, a)$ iff $\hat{D}(\mathbf{w}) \leq a$.

Let M be a model and A be a finite subset of $\mathbb{N}$. The *unfolding of the transition relation* is defined as

$$\big[M\big]_k^A := \bigwedge_{j \in A} \bigwedge_{i=0}^{k-1} Td(\mathbf{w}_{i,j}, \mathbf{w}_{i+1,j}).$$

It is easy to see that $V \models \big[M\big]_k^A$ iff for each $j \in A$, $(\hat{V}(\mathbf{w}_{0,j}), \ldots, \hat{V}(\mathbf{w}_{k,j}))$ is a k-path in M. As the translation introduced in [31] was an essential improvement over the original one of [21], we follow Zbrzezny's approach in our work.

Following [31], let A and B be finite subsets of $\mathbb{N}$. By $A \prec B$ we denote that $x < y$ for all $x \in A$ and $y \in B$. Let $k, m, p \in \mathbb{N}$ and $m \leq |A|$, then:

1. $\hat{g}_L(A, m)$ is the subset B of A such that $|B| = m$ and $B \prec A \backslash B$,
2. $\hat{g}_R(A, m)$ denotes the subset B of A such that $|B| = m$ and $A \backslash B \prec B$,
3. $h_X(A)$ is the set $A \backslash \{min(A)\}$,
4. if $k+1$ divides $|A| - 1$ then $h_G(A, k)$ is the sequence of sets $(B_0, \ldots, B_k)$ such that $\bigcup_{i=0}^{k} B_i = A \backslash \{min(A)\}$, $|B_i| = |B_j|$ and $B_i \prec B_j$ for every $0 \leq i < j \leq k$,
5. if k divides $|A| - 1 - p$, then $h_U(A, k, p)$ denotes the sequence of sets $(B_0, \ldots, B_k)$ such that $\bigcup_{i=0}^{k} B_i = A \backslash \{min(A)\}$, $B_i \prec B_j$ for every $0 \leq i < j \leq k$, $|B_0| = \ldots = |B_{k-1}|$ and $|B_k| = p$.

We also need a sequence element selector that is if $h_G(A, k) = (B_0, \ldots, B_k)$ then define $h_G(A, k)(i) = B_i$ for $0 \leq i \leq k$ and if $h_U(A, k, p) = (B_0, \ldots, B_k)$, define $h_U(A, k, p)(i) = B_i$ for $0 \leq i \leq k$. The functions $\hat{g}_L$ and $\hat{g}_R$ are used to divide the set of path indices into the two parts of the sizes sufficient to perform the independent translation of subformulae α and β of formula $\alpha \wedge \beta$. Similarly, h_G and h_U are used to divide the set of path indices into the sequences (hence the use of the selector) of subsets which are of the sizes sufficient to perform the translation of subformulae α and α together with β of, respectively, formulae $EG^{\leq \eta}\alpha$ and $E\alpha U^{\eta}\beta$. A more in-depth description can be found in [31].

Definition 5.3. (Translation of vRTECTL**)**
Let $\alpha, \beta \in$ vRTECTL, p – an atomic proposition, υ – a parameter valuation, η – a linear expression, $(m, n) \in \mathbb{N} \times \mathbb{N}$, and $A \subseteq \mathbb{N}$.

- $\big[p\big]_k^{[m,n,A,\upsilon]} := p(\mathbf{w}_{m,n})$ and $\big[\neg p\big]_k^{[m,n,A,\upsilon]} := \neg p(\mathbf{w}_{m,n})$,

- $\big[\alpha \wedge \beta\big]_k^{[m,n,A,\upsilon]} := \big[\alpha\big]_k^{[m,n,\hat{g}_L(A,f_k(\alpha,\upsilon)),\upsilon]} \wedge \big[\beta\big]_k^{[m,n,\hat{g}_R(A,f_k(\beta,\upsilon)),\upsilon]}$,
- $\big[\alpha \vee \beta\big]_k^{[m,n,A,\upsilon]} := \big[\alpha\big]_k^{[m,n,\hat{g}_L(A,f_k(\alpha,\upsilon)),\upsilon]} \wedge \big[\beta\big]_k^{[m,n,\hat{g}_L(A,f_k(\beta,\upsilon)),\upsilon]}$,
- $\big[EX\alpha\big]_k^{[m,n,A,\upsilon]} := H(\mathbf{w}_{m,n}, \mathbf{w}_{0,min(A)}) \wedge \big[\alpha\big]_k^{[1,min(A),h_X(A),\upsilon]}$,
- $\big[EG^{\leq\eta}\alpha\big]_k^{[m,n,A,\upsilon]} := H(\mathbf{w}_{m,n}, \mathbf{w}_{0,min(A)}) \wedge Z(\mathbf{w}_{0,min(A)}) \wedge$
  $\big(\ (\ Le(\mathbf{w}_{k,min(A)}, \upsilon(\eta)) \wedge (L_k(min(A)) \wedge \bigwedge_{j=0}^{k} \big[\alpha\big]_k^{[j,min(A),h_G(A,k)(j),\upsilon]})\ \big)$
  $\vee\ (\ \neg Le(\mathbf{w}_{k,min(A)}, \upsilon(\eta)) \wedge$
  $\bigwedge_{j=0}^{k}(Le(\mathbf{w}_{j,min(A)}, \upsilon(\eta)) \Rightarrow \big[\alpha\big]_k^{[j,min(A),h_G(A,k)(j),\upsilon]})\ )\ \big)$,
- $\big[E\alpha U^{\leq\eta}\beta\big]_k^{[m,n,A,\upsilon]} := H(\mathbf{w}_{m,n}, \mathbf{w}_{0,min(A)}) \wedge Z(\mathbf{w}_{0,min(A)}) \wedge$
  $\bigvee_{i=0}^{k}\big(\ Le(\mathbf{w}_{i,min(A)}, \upsilon(\eta)) \wedge (\big[\beta\big]_k^{[i,min(A),h_U(A,k,f_k(\beta,\upsilon))(k),\upsilon]} \wedge$
  $\bigwedge_{j=0}^{i-1} \big[\alpha\big]_k^{[j,min(A),h_U(A,k,f_k(\beta,\upsilon))(j),\upsilon]})\ \big)$.

The above encoding is based on the bounded semantics for vRTECTL (see Definition 4.3).

Definition 5.4. (Translation of PRTECTL**)**
Let $\alpha \in$ PRTECTL, $A \subseteq \mathbb{N}$, $(m,n) \in \mathbb{N} \times \mathbb{N}$, and $c \in \mathbb{N}$. If α contains no quantifiers and no free parameters, then:

$$\big[\alpha\big]_k^{[m,n,A]} := \big[\alpha\big]_k^{[m,n,A,\upsilon]}, \text{ where } \upsilon \text{ is any parameter valuation.}$$

As in the above case $\alpha \in$ vRTECTL and it contains no free parameters, the choice of υ is irrelevant.

Let $d = min\{c, k \cdot n_{max}\}$, then:

$$\big[\forall_{\Theta\leq c}\alpha(\Theta)\big]_k^{[m,n,A]} := \big[\alpha(d)\big]_k^{[m,n,\hat{g}_L(A,f_k(\alpha(d)))]} \wedge \big[\forall_{\Theta\leq d-1}\alpha(\Theta)\big]_k^{[m,n,\hat{g}_R(A,f_k(\forall_{\Theta\leq d-1}\alpha(\Theta)))]},$$

$$\big[\exists_{\Theta\leq c}\alpha(\Theta)\big]_k^{[m,n,A]} := \big[\alpha(d)\big]_k^{[m,n,\hat{g}_L(A,f_k(\alpha(d)))]} \vee \big[\exists_{\Theta\leq d-1}\alpha(\Theta)\big]_k^{[m,n,\hat{g}_L(A,f_k(\exists_{\Theta\leq d-1}\alpha(\Theta)))]}.$$

Let M_k be the k-model. If $\alpha \in$ PRTECTL, define $F_k(\alpha) := \{i \in \mathbb{N} \mid 1 \leq i \leq f_k(\alpha)\}$. The set F_k contains the indices of symbolic k-paths used to perform the translation. The formula $\big[M\big]_k^{F_k(\alpha)}$ encodes all the M_k submodels of the size not greater than needed to validate the truth of formula α, as indicated in Lemmas 5.2, 5.3.

We are now ready to complete the translation of the problem of validity in vRTECTL and PRTECTL to the problem of satisfiability of propositional formulae. Let M_k be the k-model, $\alpha \in$ vRTECTL and υ be a parameter valuation. Denote

$$\big[M\big]_k^{\alpha,\upsilon} := \big[M\big]_k^{F_k(\alpha)} \wedge I_{s^0}(\mathbf{w}_{0,0}) \wedge \big[\alpha\big]_k^{[0,0,F_k(\alpha),\upsilon]}.$$

Similarly, let $\beta \in$ PRTECTL, then denote

$$\big[M\big]_k^{\beta} := \big[M\big]_k^{F_k(\beta)} \wedge I_{s^0}(\mathbf{w}_{0,0}) \wedge \big[\beta\big]_k^{[0,0,F_k(\beta)]}.$$

The following theorems ensure the completeness and correctness of the translation.

Theorem 5.1. Let M_k be the k-model of M, v – a parameter valuation, α – a formula of vRTECTL containing at least one modality, and s a state. Then, the following equivalence holds: $M_k, s \models_v \alpha$ iff $\left[M\right]_k^{\alpha,v}$ is satisfiable.

Proof:
The proof is divided into two parts: the proof of correctness and the proof of completeness of the translation, both obtained by the induction on the length of the formula. Both the parts are similar to the counterparts of the theorem for ECTL that is Theorems 3.1 and 3.2 from [31]. □

Theorem 5.2. Let M_k be the k-model of M, β – a sentence of PRTECTL containing at least one modality, and s – a state. Then, the following equivalence holds: $M_k, s \models \beta$ iff $\left[M\right]_k^{\beta}$ is satisfiable.

Proof:
This can be shown by replacing the non-modal quantifiers in a formula of PRTECTL with, appropriately, conjunctions or disjunctions, and by using Theorem 5.1. □

6. Implementation for Time Petri Nets

In this section we show how to implement the above verification method for the case of time Petri nets. To this aim, we start with recalling some basic definitions.

6.1. Time Petri Nets

In the paper we consider time Petri nets and a notable subclass, called *distributed time Petri nets*. The standard definition of time Petri nets is as follows:

Definition 6.1. A time Petri net (TPN, for short) is a six-element tuple $\mathcal{N} = (P, T, FR, Eft, Lft, m^0)$, where $P = \{p_1, \ldots, p_{n_P}\}$ is a finite set of *places*, $T = \{t_1, \ldots, t_{n_T}\}$ is a finite set of *transitions*, $FR \subseteq (P \times T) \cup (T \times P)$ is the *flow relation*, $Eft, Lft : T \to \mathbb{N}$ are functions describing the *earliest* and the *latest firing time* of the transition, where for each $t \in T$ we have $Eft(t) \leq Lft(t)$, and $m^0 \subseteq P$ is the *initial marking* of $\mathcal{N}$.

For a transition $t \in T$ we define its *preset* $\bullet t = \{p \in P \mid (p,t) \in FR\}$ and *postset* $t\bullet = \{p \in P \mid (t,p) \in FR\}$, and consider only the nets such that for each transition the preset and the postset are nonempty. We need also the following notations and definitions:

- a *marking* of $\mathcal{N}$ is any subset $m \subseteq P$,
- a transition $t \in T$ is *enabled* at m ($m[t\rangle$ for short) if $\bullet t \subseteq m$ and $t\bullet \cap (m \setminus \bullet t) = \emptyset$; and *leads from m to m'*, if it is enabled at m, and $m' = (m \setminus \bullet t) \cup t\bullet$. The marking m' is denoted by $m[t\rangle$ as well, if this does not lead to misunderstanding.
- $en(m) = \{t \in T \mid m[t\rangle\}$;
- for $t \in en(m)$, $newly\_en(m,t) = \{u \in T \mid u \in en(m[t\rangle) \ \wedge \ (t\bullet \cap \bullet u \neq \emptyset \vee u\bullet \cap \bullet t \neq \emptyset)\}$.

In this work we assume a discrete-time semantics of TPNs, i.e., consider integer time passings only (cf. [19]). A *concrete state* σ of a net $\mathcal{N}$ is then a pair $(m, clock)$, where m is a marking, and $clock : T \to \mathbb{N}$ is a function which for each transition $t \in en(m)$ gives the time elapsed since t became enabled most recently, and assigns zero to other transitions. Given a state $(m, clock)$ and $\delta \in \mathbb{N}$, denote by $clock + \delta$ the function defined by $(clock + \delta)(t) = clock(t) + \delta$ for each $t \in en(m)$, and $(clock + \delta)(t) = 0$ otherwise. By $(m, clock) + \delta$ we denote $(m, clock + \delta)$. The *concrete state space* of $\mathcal{N}$ is a structure $(\Sigma, \sigma^0, \to_c)$, where Σ is the set of all the concrete states of $\mathcal{N}$, $\sigma^0 = (m^0, clock^0)$ with $clock^0(t) = 0$ for each $t \in T$ is the initial state of $\mathcal{N}$, and $\to_c \subseteq \Sigma \times (T \cup \mathbb{N}) \times \Sigma$ is a timed consecution relation defined by:

- for $\delta \in \mathbb{N}$, $(m, clock) \xrightarrow{\delta}_c (m, clock + \delta)$ iff $(clock + \delta)(t) \leq Lft(t)$ for all $t \in en(m)$ (*time successor*),
- for $t \in T$, $(m, clock) \xrightarrow{t}_c (m', clock')$ iff $t \in en(m)$, $Eft(t) \leq clock(t) \leq Lft(t)$, $m' = m[t\rangle$, and for all $u \in T$ we have $clock'(u) = 0$ for $u \in newly\_en(m, t)$ and $clock'(u) = clock(u)$ otherwise (*action successor*).

Notice that firing of a transition takes no time.

Given a set of propositional variables PV, we introduce a valuation function $V_c : \Sigma \to 2^{PV}$ which assigns the same propositions to the states with the same markings. We assume the set PV to be such that each $q \in PV$ corresponds to exactly one $p \in P$, and use the same names for the propositions and the places. The function V is then defined by $p \in V_c(\sigma) \iff p \in m$ for each $\sigma = (m, \cdot)$. The structure $M_c(\mathcal{N}) = (\Sigma, \sigma^0, \to_c, V_c)$ is called a *concrete model* of $\mathcal{N}$. It is easy to see that $M_c(\mathcal{N})$ induces a timed Kripke structure, obtained from $M_c(\mathcal{N})$ by replacing labels corresponding to transitions by the label 0.

In order to benefit from a structure of the net and obtain a greater efficiency, we consider separately a subclass of TPNs - *distributed time Petri nets* defined as follows:

Definition 6.2. Let $\mathcal{I} = \{\mathrm{i}_1, \ldots, \mathrm{i}_n\}$ be a finite ordered set of indices, and let $\mathfrak{N} = \{N_\mathrm{i} = (P_\mathrm{i}, T_\mathrm{i}, FR_\mathrm{i}, m^0_\mathrm{i}, Eft_\mathrm{i}, Lft_\mathrm{i}) \mid \mathrm{i} \in \mathcal{I}\}$ be a family of 1-safe, sequential time Petri nets (called *processes*), indexed with $\mathcal{I}$, with the pairwise disjoint sets P_i of places, and satisfying the condition $(\forall \mathrm{i}_1, \mathrm{i}_2 \in \mathcal{I})(\forall t \in T_{\mathrm{i}_1} \cap T_{\mathrm{i}_2})\,(Eft_{\mathrm{i}_1}(t) = Eft_{\mathrm{i}_2}(t) \wedge Lft_{\mathrm{i}_1}(t) = Lft_{\mathrm{i}_2}(t))$. A *distributed time Petri net* $\mathcal{N} = (P, T, FR, m^0, Eft, Lft)$ is the union of the processes N_i, i.e., $P = \bigcup_{\mathrm{i}\in\mathcal{I}} P_\mathrm{i}$, $T = \bigcup_{\mathrm{i}\in\mathcal{I}} T_\mathrm{i}$, $FR = \bigcup_{\mathrm{i}\in\mathcal{I}} FR_\mathrm{i}$, $m^0 = \bigcup_{\mathrm{i}\in\mathcal{I}} m^0_\mathrm{i}$, $Eft = \bigcup_{\mathrm{i}\in\mathcal{I}} Eft_\mathrm{i}$, and $Lft = \bigcup_{\mathrm{i}\in\mathcal{I}} Lft_\mathrm{i}$.

Notice that the function Eft_{i_1} (Lft_{i_1}) coincides with Eft_{i_2} (Lft_{i_2}, resp.) for the joint transitions of each two processes i_1 and i_2. The interpretation of such a system is a collection of sequential, non-deterministic processes with communication capabilities (via joint transitions). Moreover, we assume that the initial marking of a DTPN contains exactly one place of each process of the net, and that all its processes are *state machines* (i.e., for each $\mathrm{i} \in \mathcal{I}$ and each $t \in T_\mathrm{i}$, $|\bullet t| = |t \bullet| = 1$), which implies that in any marking of $\mathcal{N}$ there is exactly one place of each process. It is important to observe that a large class of distributed nets can be decomposed to satisfy the above requirement [14].

The structure of the above nets allows to define concrete states of these nets in a different way. Instead of assigning a clock to a transition, we assign it to a process; the clock shows the time passed since the most recently marked place of its process was marked. Such a semantics is equivalent to the standard one [23], but results in reducing the number of clocks, which influences efficiency of implementations.

In order to be able to perform parametric verification, we introduce an additional restriction on the nets under consideration, i.e., require they contain no cycle C of transitions such that for each $t \in C$ we have $Eft(t) = 0$ (which guarantees that the time flows when the net progresses, and is a typical assumption when analysing timed systems).

6.2. Implementation

Given a time Petri net $\mathcal{N}$, let c_{max} denote the greatest finite value of the function Lft of $\mathcal{N}$. It is easy to see that for each $(m, clock) \in \Sigma$ and each $t \in T$ we have $clock(t) \leq c_{max}$. Thus, the states of Σ can be encoded by valuations of a vector of state variables w being a pair (w_m, w_c), where $w_m = (w_m[1], \ldots, w_m[n_P])$ encodes the marking part of a state, whereas $w_c = (w_c[1], \ldots, w_c[b])$, with $b = \lceil \log(n_T \cdot c_{max}) \rceil$, encodes the clock part. Let $\mathcal{I} = \{1, \ldots, n_P\}$ and $\mathcal{I}(m) = \{i \in \mathcal{I} \mid p_i \in m\}$. The functions needed to encode the transition relation are now of the following form:

- $H(\mathbf{w}, \mathbf{v}) = \bigwedge_{1 \leq i \leq n_P} w_m[i] \iff v_m[i] \wedge \bigwedge_{1 \leq i \leq b} w_c[i] \iff v_c[i]$,
- $p_i(w) = w_m[i]$,
- $I_{s^0}(\mathbf{w}) = \bigwedge_{i \in \mathcal{I}(m^0)} w_m[i] \wedge \bigwedge_{i \in \mathcal{I} \setminus \mathcal{I}(m^0)} \neg w_m[i] \wedge \bigwedge_{i=1,\ldots,b} \neg w_c[i]$,
- $T(\mathbf{w}, \mathbf{v}) = T_{trans}(\mathbf{w}, \mathbf{v}) \vee T_{time}(\mathbf{w}, \mathbf{v})$, where $T_{trans}(\mathbf{w}, \mathbf{v})$ encodes that the state represented by $w_{\mathbf{v}}$ is a successor of the state represented by $w_{\mathbf{w}}$ obtained by firing a transition, and $T_{time}(w, v)$ encodes that the state represented by $w_{\mathbf{v}}$ is a time successor of the state represented by $w_{\mathbf{w}}$

(the details of the last two encodings, as well as of the remaining functions, are omitted for simplicity). Moreover, it is reasonable to consider sequences of transitions in which time- and action steps alternate.

Contrary to the general case, for DTPNs and their semantics with clocks assigned to processes we cannot assume any upper bound on clock values. However, the length of time steps can be restricted (without loss of generality) to a value depending on c_{max}, which in turn allows to bound values of clocks on a k-path by a value depending on k and c_{max} (the details can be found in [20]). This allows to encode the states on a k-paths by valuations of vectors of state variables, similarly as in the general case. The functions needed to encode the transition relation, as well as the structure of sequences of transitions considered, are of a similar form to the ones listed above (again, the details are omitted for brevity).

7. Example – Generalized Timed Pipelining Algorithm

We consider the Generic Pipeline Paradigm Petri net model [18]. It consists of three parts: Producer producing data ($ProdReady$), Consumer receiving data ($ConsReady$), and a chain of n intermediate Nodes – $(Node_iReady)$ - ready for receiving data, $(Node_iProc)$ - processing data, and $(Node_iSend)$ - sending data.

Notice that the example can be scaled in order to see how the size of the system influences performance, and whether the truth of the verified formulae is affected. The parameters a, b, c, d, e, f are used to adjust the time properties of Producer, Consumer, and the intermediate Nodes. So, it is possible to change the constraints on the processing and idle time of each the processes. We test the following formula: $\exists_\Theta EF^{\leq\Theta}(\neg ProdReady \wedge \neg ConsReady \wedge EG^{\leq\Theta}\neg ConsReady)$. Intuitively, it expresses that the system can reach, in the time smaller or equal than some value Θ, such a state that neither Producer is ready to produce nor Consumer is ready to receive, and for the next Θ time steps the Consumer

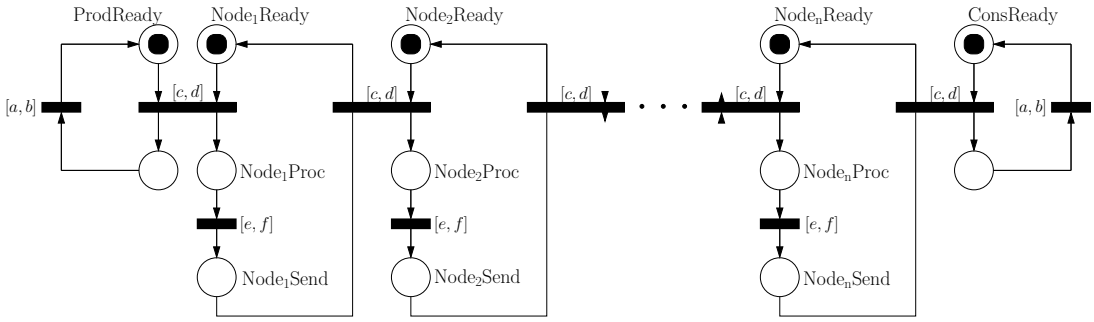

Figure 1. Generic Timed Pipelining Paradigm

n	k	time params						PBMC				RSat	SAT?
		a	b	c	d	e	f	| clauses |	| vars |	time	mem	time	Y/N
0	4	0	5	0	5	0	5	18752	6173	0.3	3.6	0.04	Y
1	20	0	5	0	5	2	1	949002	313710	78.93	44.47	6.46	N
1	8	0	5	0	2	1	3	131783	43424	14.96	9.02	5.02	Y
1	20	0	1	0	2	1	3	670398	222073	87.7	32.22	88.05	N
1	8	0	5	0	2	1	3	131783	43424	15.36	9.02	4.96	Y
2	20	1	3	2	5	1	3	2096085	694835	192.8	94.23	176.11	N
2	20	1	5	2	5	1	5	2080957	690067	208.2	93.58	202.35	N
2	12	1	12	1	12	1	12	1086932	358616	79.95	50.27	156.03	Y
3	20	0	5	0	5	0	5	2462343	817103	298.2	110.1	249.25	N
3	16	1	10	1	10	1	10	3189145	1057781	255.6	141.4	363.43	Y
4	20	0	8	2	8	1	5	5344604	1774816	707	234.6	747.06	N
4	20	0	10	1	10	1	5	6612127	2196285	875.7	289.5	868.55	Y

Table 1. Testing the formula $\exists_\Theta EF^{\leq\Theta}(\neg ProdReady \wedge \neg ConsReady \wedge EG^{\leq\Theta}\neg ConsReady)$ (time measured in seconds, memory in MB).

receiving abilities are disabled. As the experiments (see Table 1) confirm, the validity of this formula heavily depends on the combination of the constants. For example, comparing the fourth and fifth row reveals that in a simple one-process pipeline, the time in which Producer and Consumer are allowed to be idle is crucial to the system processing capabilities. This is quite natural, because if b is shortened from 5 time units to 1, keeping the remaining constants fixed, then Consumer can remain in the idle state ($\neg ConsReady$) for just one time step. The times of the translation to SAT (PBMC) and of solving satisfiability by the solver (RSat) are given for the value of k of the second column.

We implemented the procedure above on top of Verics BMC tool by combining the encoding of the transition relation for distributed time Petri nets [20] with the encoding of PRTECTL operators [15]. The experiments were performed on 1.6 GHz Intel Atom with 1GB RAM.

8. Conclusions and Further Work

The aim of this paper was to present both a general theoretical approach to SAT-based verification of PRTECTL properties of discrete-timed systems, and its implementation (supported by some experimental results) for time Petri nets with discrete-time semantics. In the second case we focused on distributed time Petri nets as their structure allowed us to obtain a higher efficiency of the implementation due to a reduced number of clocks.

Our future work will involve an implementation of the method also for standard TPNs with discrete semantics. Moreover, we expect that using such a semantics is sufficient to reason about PRTECTL properties also in the case of TPNs with dense time (which is based on results of Popova [24, 25]). Proving this will be a topic of a next paper.

References

[1] R. Alur, T. Henzinger, and M. Vardi. Parametric real-time reasoning. In *Proc. of the 25th Ann. Symp. on Theory of Computing (STOC'93)*, pages 592–601. ACM, 1993.

[2] G. Audemard, A. Cimatti, A. Kornilowicz, and R. Sebastiani. Bounded model checking for timed systems. In *Proc. of the 22nd Int. Conf. on Formal Techniques for Networked and Distributed Systems (FORTE'02)*, volume 2529 of *LNCS*, pages 243–259. Springer-Verlag, 2002.

[3] M. Benedetti and A. Cimatti. Bounded model checking for Past LTL. In *Proc. of the 9th Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'03)*, volume 2619 of *LNCS*, pages 18–33. Springer-Verlag, 2003.

[4] A. Biere, A. Cimatti, E. Clarke, M.Fujita, and Y. Zhu. Symbolic model checking using SAT procedures instead of BDDs. In *Proc. of the ACM/IEEE Design Automation Conference (DAC'99)*, pages 317–320, 1999.

[5] V. Bruyére, E. Dall'Olio, and J-F. Raskin. Durations and parametric model-checking in timed automata. *ACM Transactions on Computational Logic*, 9(2), 2008.

[6] G. Bucci, A. Fedeli, L. Sassoli, and E. Vicaro. Modeling flexible real time systems with preemptive time Petri nets. In *Proc. of the 15th Euromicro Conference on Real-Time Systems (ECRTS'03)*, pages 279–286. IEEE Computer Society, 2003.

[7] G. Bucci and E. Vicaro. Compositional validation of time-critical systems using communicating time Petri nets. *IEEE Trans. on Software Eng.*, 21(12):969–992, 1995.

[8] E. Clarke, A. Biere, R. Raimi, and Y. Zhu. Bounded model checking using satisfiability solving. *Formal Methods in System Design*, 19(1):7–34, 2001.

[9] E. Clarke, O. Grumberg, and D. Peled. *Model Checking*. MIT Press, 1999.

[10] E. A. Emerson and E. Clarke. Using branching-time temporal logic to synthesize synchronization skeletons. *Science of Computer Programming*, 2(3):241–266, 1982.

[11] E. A. Emerson and R. Trefler. Parametric quantitative temporal reasoning. In *Proc. of the 14th Symp. on Logic in Computer Science (LICS'99)*, pages 336–343. IEEE Computer Society, July 1999.

[12] K. Heljanko. Bounded reachability checking with process semantics. In *Proc. of the 12th Int. Conf. on Concurrency Theory (CONCUR'01)*, volume 2154 of *LNCS*, pages 218–232. Springer-Verlag, 2001.

[13] T. Hune, J. Romijn, M. Stoelinga, and F. Vaandrager. Linear parametric model checking of timed automata. In *Proc. of the 7th Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'01)*, volume 2031 of *LNCS*, pages 189–203. Springer-Verlag, 2001.

[14] R. Janicki. Nets, sequential components and concurrency relations. *Theoretical Computer Science*, 29:87–121, 1984.

[15] M. Knapik, W. Penczek, and M. Szreter. Bounded parametric model checking for elementary net systems. In *Proc. of Int. Workshop on Petri Nets and Software Engineering (PNSE'09)*, pages 97–117. University of Hamburg, 2009.

[16] R. Mascarenhas, D. Karumuri, U. Buy, and R. Kenyon. Modeling and analysis of a virtual reality system with time Petri nets. In *Proc. of the 20th Int. Conf. on Software Engineering (ICSE'98)*, pages 33–42. IEEE Computer Society, 1998.

[17] P. Merlin and D. J. Farber. Recoverability of communication protocols – implication of a theoretical study. *IEEE Trans. on Communications*, 24(9):1036–1043, 1976.

[18] D. Peled. All from one, one for all: On model checking using representatives. In *Proc. of the 5th Int. Conf. on Computer Aided Verification (CAV'93)*, volume 697 of *LNCS*, pages 409–423. Springer-Verlag, 1993.

[19] W. Penczek and A. Półrola. *Advances in Verification of Time Petri Nets and Timed Automata: A Temporal Logic Approach*, volume 20 of *Studies in Computational Intelligence*. Springer-Verlag, 2006.

[20] W. Penczek, A. Półrola, and A. Zbrzezny. SAT-based (parametric) reachability for distributed time Petri nets. In *Proc. of the Int. Workshop on Petri Nets and Software Engineering (PNSE'09)*, pages 133–154. University of Hamburg, 2009.

[21] W. Penczek, B. Woźna, and A. Zbrzezny. Bounded model checking for the universal fragment of CTL. *Fundamenta Informaticae*, 51(1-2):135–156, 2002.

[22] W. Penczek, B. Woźna, and A. Zbrzezny. Towards bounded model checking for the universal fragment of TCTL. In *Proc. of the 7th Int. Symp. on Formal Techniques in Real-Time and Fault Tolerant Systems (FTRTFT'02)*, volume 2469 of *LNCS*, pages 265–288. Springer-Verlag, 2002.

[23] A. Półrola and W. Penczek. Minimization algorithms for time Petri nets. *Fundamenta Informaticae*, 60(1-4):307–331, 2004.

[24] L. Popova. On time Petri nets. *Elektronische Informationsverarbeitung und Kybernetik*, 27(4):227–244, 1991.

[25] L. Popova-Zeugmann. Time Petri nets state space reduction using dynamic programming. *Journal of Control and Cybernetics*, 35(3):721–748, 2006.

[26] J-F. Raskin and V. Bruyère. Real-time model checking: Parameters everywhere. In *Proc. of the 23rd Conf. on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'03)*, volume 2914 of *LNCS*, pages 100–111. Springer-Verlag, 2003.

[27] O. Strichman. Tuning SAT checkers for bounded model checking. In *Proc. of the 12th Int. Conf. on Computer Aided Verification (CAV'00)*, volume 1855 of *LNCS*, pages 480–494. Springer-Verlag, 2000.

[28] B. Woźna. ACTL* properties and bounded model checking. *Fundamenta Informaticae*, 63(1):65–87, 2004.

[29] B. Woźna, A. Zbrzezny, and W. Penczek. Checking reachability properties for timed automata via SAT. *Fundamenta Informaticae*, 55(2):223–241, 2003.

[30] A. Zbrzezny. Improvements in SAT-based reachability analysis for timed automata. *Fundamenta Informaticae*, 60(1-4):417–434, 2004.

[31] A. Zbrzezny. Improving the translation from ECTL to SAT. *Fundamenta Informaticae*, 85(1-4):513–531, 2008.

A. Appendix

In this section we provide the proofs omitted from the text. Throughout this part we put $s \to s'$, where s, s' are states of a model M, if there exists some $i \in \mathbb{N}$ such that $s \xrightarrow{i} s'$. Let π be a path in a model M and $c \in \mathbb{N}$, and denote $\Delta_\pi^c = max\{i \mid \delta_\pi^i \leq c\}$.

Lemma 3.1. *Let $M = (S, s^0, \to, \mathcal{L})$ be a timed Kripke structure, $\alpha \in$ vRTCTL and υ – a parameter valuation. Denote by n_{max} the greatest time label of a time transition present in M. Then $M, s \models_\upsilon \alpha \iff M, s \models_{\upsilon'} \alpha$, where $\upsilon'(\Theta) = min(n_{max} \cdot |M|, \upsilon(\Theta))$ for each parameter Θ.*

Proof:
The proof goes by the structural induction. We omit the easy basic cases and focus on EU and EG modalities. Notice that $\upsilon'(\Theta) \leq \upsilon(\Theta)$ for all the parameters Θ, therefore also $\upsilon'(\eta) \leq \upsilon(\eta)$ for any linear expression η.
Let us start with showing that $s \models_\upsilon EG^{\leq\eta}\alpha$ iff $s \models_{\upsilon'} EG^{\leq\eta}\alpha$. If $s \models_\upsilon EG^{\leq\eta}\alpha$, then from the existence of a path π such that $\pi(0) = s$ and $\pi(i) \models_\upsilon \alpha$ for all $i \leq \Delta_\pi^{\upsilon(\eta)}$, the fact that $\Delta_\pi^{\upsilon'(\eta)} \leq \Delta_\pi^{\upsilon(\eta)}$ and the inductive assumption we obtain $s \models_{\upsilon'} EG^{\leq\eta}\alpha$.
On the other hand, if $s \models_{\upsilon'} EG^{\leq\eta}\alpha$, we consider two cases. The first case, when $\upsilon'(\eta) = \upsilon(\eta)$ is trivial, so let us assume that $\upsilon'(\eta) < \upsilon(\eta)$. Notice that $\upsilon'(\eta) \geq n_{max} \cdot |M|$, and consider a path π' satisfying $\pi'(0) = s$ and $\pi'(i) \models_{\upsilon'} \alpha$ for all $i \leq \Delta_{\pi'}^{\upsilon'(\eta)}$. Now, due to the fact that the longest finite path built from the distinct states in M is of length at most $|M|$, and its time length is smaller or equal to $(|M|-1)\cdot n_{max}$, we obtain that $\Delta_{\pi'}^{\upsilon'(\eta)} \geq |M|$. Therefore, there exist $l < k \leq |M|$ such that $\pi'(k) \to \pi'(l)$. Define the path π as follows:

$$\pi(i) = \begin{cases} \pi'(i) & \text{for } i < k \\ \pi'(l + (i-k) mod (k-l)) & \text{for } i \geq k. \end{cases}$$

This path satisfies $\pi(0) = s$ and $\pi(i) \models_{\upsilon'} \alpha$ for all $i \leq \Delta_\pi^{\upsilon(\eta)}$, from which $s \models_\upsilon EG^{\leq\eta}\alpha$ follows by the inductive argument.
Consider the case of $s \models_\upsilon E\alpha U^{\leq\eta}\beta$ iff $s \models_{\upsilon'} E\alpha U^{\leq\eta}\beta$. If $s \models_\upsilon E\alpha U^{\leq\eta}\beta$ then there exists a path π such that $\pi(0) = s$, and for some $i \leq \Delta_\pi^{\upsilon(\eta)}$ we have $\pi(i) \models_\upsilon \beta$, and $\pi(j) \models_\upsilon \alpha$ for all $j < i$. Again, we consider two cases and omit the easy case of $\upsilon'(\eta) = \upsilon(\eta)$. Assume that $\upsilon'(\eta) < \upsilon(\eta)$, from which we obtain the fact that $\Delta_\pi^{\upsilon(\eta)} > |M|$. Therefore, as previously, there exist $l < k < |M|$ such that $\pi(k) \to \pi(l)$. By consecutive removal of blocks such as $\pi(l), \ldots, \pi(k-1)$ from π, we eventually arrive at the path π' such that for some $i < |M| \leq \Delta_{\pi'}^{\upsilon'(\eta)}$ we have $\pi'(0) = s$, $\pi'(i) \models_\upsilon \beta$, and $\pi'(j) \models_\upsilon \alpha$ for all $j < i$. Thus, by the inductive assumption $s \models_{\upsilon'} E\alpha U^{\leq\eta}\beta$ follows.
Finally, if $s \models_{\upsilon'} E\alpha U^{\leq\eta}\beta$, then $s \models_\upsilon E\alpha U^{\leq\eta}\beta$ follows from the fact that $\Delta_\pi^{\upsilon'(\eta)} \leq \Delta_\pi^{\upsilon(\eta)}$ and the inductive assumption. □

Lemma 4.1. *Let $k, l \in \mathbb{N}$ such that $k \leq l$, M_k be the k–model of M, and s – a state. Consider a formula $\alpha \in$ vRTECTL together with a parameter valuation υ, and a sentence $\beta \in$ PRTECTL. The following conditions hold:*

(1). $M_k, s \models_\upsilon \alpha$ implies $M_l, s \models_\upsilon \alpha$, and $M_k, s \models \beta$ implies $M_l, s \models \beta$,

(2). $M_k, s \models_\upsilon \alpha$ implies $M, s \models_\upsilon \alpha$, and $M_k, s \models \beta$ implies $M, s \models \beta$,

(3). $M, s \models_\upsilon \alpha$ *implies* $M_{|M|}, s \models_\upsilon \alpha$, *and* $M, s \models \beta$ *implies* $M_{|M|}, s \models \beta$.

Proof:
Let us focus on the implications concerning α formulae of vRTECTL. We start with the first implication and omit the basic case of atomic subformulae and their negations as trivial. Let α, β be formulae satisfying the considered property, then $M_k, s \models_\upsilon \alpha \wedge \beta$ iff $M_k, s \models_\upsilon \alpha$ and $M_k, s \models_\upsilon \beta$, from which by the inductive assumption follows that $M_l, s \models_\upsilon \alpha$ and $M_l, s \models_\upsilon \beta$ which is equivalent to $M_l, s \models_\upsilon \alpha \wedge \beta$. The case of disjunction follows similarly. For the case of $M_k, s \models_\upsilon EX\alpha$ let us notice that a finite path π_k in M_k such that $\pi_k(0) = s$ and $M_k, \pi_k(1) \models_\upsilon \alpha$ is a prefix of some finite path in M_l, therefore $M_l, s \models_\upsilon EX\alpha$. Now, the case of $M_k, s \models_\upsilon EG^{\leq\eta}\alpha$ can be divided into two subcases. If $\upsilon(\eta) < \delta_{\pi_k}$, then define π_l as some finite path in M_l containing π_k as a prefix. If $\upsilon(\eta) \geq \delta_{\pi_k}$, then there exists an infinite path π along which α holds (see the proof of Lemma 3.1). Define π_l as a prefix of length l of π. In both the cases α holds along π_l, therefore $M_l, s \models_\upsilon EG^{\leq\eta}\alpha$. In the final case of $M_k, s \models_\upsilon E\alpha U^{\leq\eta}\beta$ let us notice that if for some path π_k in M_k we have $\pi_k(0) = s$, $M_k, \pi_k(i) \models_\upsilon \beta$, $M_k, \pi_k(j) \models_\upsilon \alpha$ for all $0 \leq j < i$ and $i \leq \Delta_{\pi_k}^{\upsilon(\eta)}$ then the same path is a prefix of some path in M_l, thus $M_l, s \models_\upsilon E\alpha U^{\leq\eta}\beta$.
The second implication is proved by the structural induction, similarly to the above reasoning. The only nontrivial case is when we consider $M_k, s \models_\upsilon EG^{\leq\eta}\alpha$. Notice that if $\upsilon(\eta) < \delta_{\pi_k}$, then a finite path along which α is satisfied up to $\Delta_{\pi_k}^{\upsilon(\eta)}$ steps can be extended to an infinite path (due to the fact that the transition relation is total). On the other hand, if $\upsilon(\eta) \geq \delta_{\pi_k}$, then the finite path along which α is satisfied contains a loop, and can be transformed into infinite path by traversing the loop, as in the proof of Lemma 3.1.
In the proof of the last implication we focus on two modalities, starting from the case of $M, s \models_\upsilon EG^{\leq\eta}\alpha$. If this formula is valid, then there exists an infinite path π in M such that $\pi(0) = s$ and $M, \pi(i) \models_\upsilon \alpha$ for all $0 \leq i \leq \Delta_{\pi_k}^{\upsilon(\eta)}$. There are two possible subcases. We omit the easier subcase when $\Delta_{\pi_k}^{\upsilon(\eta)} \leq |M|$. If $\Delta_{\pi_k}^{\upsilon(\eta)} > |M|$, then the path π contains a loop with a return from position $l \leq |M|$. Therefore, by unwinding a loop in π we can create an infinite path such that α is satisfied in each of its positions – also along a prefix of length $|M|$. Considering $M, s \models_\upsilon E\alpha U^{\leq\eta}\beta$, notice that there exists an infinite path π in M such that $\pi(0) = s$ and for some $l \leq \Delta_{\pi_k}^{\upsilon(\eta)}$ we have $M, \pi(l) \models_\upsilon \beta$ and $M, \pi(i) \models_\upsilon \alpha$ for all $0 \leq i < l$. Again, there are two subcases. We omit the easier case of $l \leq |M|$. If $l > |M|$, then there exist $n, m \in \mathbb{N}$ such that $n < m \leq l$, and $\pi(n) = \pi(m)$. By consecutive elimination of blocks of type $\pi(n+1), \ldots, \pi(m)$, as in the proof of Lemma 3.1, we eventually obtain a path π' such that $\pi'(0) = s$, $\pi'(j) \models_\upsilon \beta$ for some $j \leq |M|$, and $\pi'(i) \models_\upsilon \alpha$ for all $i < j$.
The part concerning β sentences of PRTECTL is obtained by straightforward induction with respect to the number of quantifiers, and using the results proven above. □

Lemma 5.2. *Let* $\alpha \in$ vRTECTL, M_k *be the* k*-model and* υ *– a parameter valuation. For any state* s *present in some path of* M_k, $M_k, s \models_\upsilon \alpha$ *if and only if there exists a submodel* M'_k *of* M_k *such that* $M'_k, s \models_\upsilon \alpha$ *and* $|Path'_k| \leq f_k(\alpha)$.

Proof:
The "if" part follows directly from Lemma 5.1. For the "only if" part, we use the structural induction. The base cases of $M_k, s \models_\upsilon p$ and $M_k, s \models_\upsilon \neg p$ are trivial. Notice that $M_k, s \models_\upsilon \alpha \vee \beta$ iff $M_k, s \models_\upsilon \alpha$ or $M_k, s \models_\upsilon \beta$. From the inductive assumption there is M'_k such that $M'_k, s \models_\upsilon \alpha$ and $|Path'_k| \leq f_k(\alpha)$,

or $M'_k, s \models_\upsilon \beta$ and $|Path'_k| \leq f_k(\beta)$. Thus $M'_k, s \models_\upsilon \alpha \vee \beta$ and $|Path'_k| \leq max(f_k(\alpha), f_k(\beta)) = f_k(\alpha \vee \beta)$.

Recall that $M_k, s \models_\upsilon \alpha \wedge \beta$ iff $M_k, s \models_\upsilon \alpha$ and $M_k, s \models_\upsilon \beta$. By the inductive assumption there exist submodels M''_k and M'''_k of M_k such that $M''_k, s \models_\upsilon \alpha, |Path''_k| \leq f_k(\alpha)$ and $M'''_k, s \models_\upsilon \beta, |Path'''_k| \leq f_k(\beta)$. Consider the submodel M'_k such that $Path'_k = Path''_k \cup Path'''_k$, then from Lemma 5.1 and the inclusions $M''_k \subseteq M'_k, M'''_k \subseteq M'_k$ we obtain $M'_k, s \models_\upsilon \alpha \wedge \beta$. Moreover, $|Path'_k| \leq |Path''_k| + |Path'''_k| \leq f_k(\alpha) + f_k(\beta) = f_k(\alpha \wedge \beta)$.

Now consider the case of $M_k, s \models_\upsilon EX\alpha$. From the definition of bounded semantics we obtain that there is some k–path $\pi_k \in Path_k$, such that $\pi_k(0) = s$ and $M_k, \pi_k(1) \models_\upsilon \alpha$. From the inductive assumption there exists a submodel M''_k such that $M''_k, \pi_k(1) \models_\upsilon \alpha$ and $|Path''_k| \leq f_k(\alpha)$. Define the submodel M'_k having $Path'_k = Path''_k \cup \{\pi_k\}$. Then from $\pi_k \in Path'_k$ and Lemma 5.1 we obtain $M'_k, \pi_k(1) \models_\upsilon \alpha$, therefore $M'_k, s \models_\upsilon EX\alpha$. Moreover, $|Path'_k| \leq |Path''_k| + 1 \leq f_k(\alpha) + 1 = f_k(EX\alpha)$.

Let us move to the case of $M_k, s \models_\upsilon EG^{\leq\eta}\alpha$. We have to consider two subcases. In the first case there exists a path $\pi_k \in Path_k$ such that $\Delta^{\upsilon(\eta)}_{\pi_k} < k$, $\pi_k(0) = s$ and $M_k, \pi_k(i) \models_\upsilon \alpha$ for all $0 \leq i \leq \Delta^{\upsilon(\eta)}_{\pi_k}$. Let M^i_k denote a submodel of M_k such that $M^i_k, \pi_k(i) \models_\upsilon \alpha$ and $|Path^i_k| \leq f_k(\alpha)$ for $0 \leq i \leq \Delta^{\upsilon(\eta)}_{\pi_k}$. Define M'_k such that $Path'_k = \bigcup_{0\leq i\leq\Delta^{\upsilon(\eta)}_{\pi_k}} Path^i_k \cup \{\pi_k\}$. Then by Lemma 5.1 we have $M'_k, \pi_k(i) \models_\upsilon \alpha$ for all $0 \leq i \leq \Delta^{\upsilon(\eta)}_{\pi_k}$, thus $M'_k, s \models_\upsilon EG^{\leq\eta}\alpha$. From the inductive assumption we obtain

$$|Path'_k| \leq \sum_{0\leq i\leq\Delta^{\upsilon(\eta)}_{\pi_k}} |Path^i_k| + 1 \leq (k+1) \cdot f_k(\alpha) + 1 = f_k(EG^{\leq\eta}\alpha).$$

We deal with the subcase of $\Delta^{\upsilon(\eta)}_{\pi_k} \geq k$ in a similar way.

In the final case of $M_k, s \models_\upsilon E\alpha U^{\leq\eta}\beta$ there exists a path $\pi_k \in Path_k$ and $0 \leq j \leq \Delta^{\upsilon(\eta)}_{\pi_k}$ such that $M_k, \pi_k(j) \models_\upsilon \beta$ and $M_k, \pi_k(i) \models_\upsilon \alpha$ for all $0 \leq i \leq j$. From the inductive assumption there exists a submodel M^j_k satisfying $M^j_k, \pi_k(j) \models_\upsilon \beta$ and $|Path^j_k| \leq f_k(\beta)$, and submodels M^i_k such that $M^i_k, \pi_k(i) \models_\upsilon \alpha$ for all $0 \leq i < j$ and $|Path^i_k| \leq f_k(\alpha)$. Define M'_k such that $Path'_k = \bigcup_{0\leq l\leq j} Path^l_k \cup \{\pi_k\}$. Then, by Lemma 5.1 we have $M'_k, \pi_k(i) \models_\upsilon \alpha$ for all $0 \leq i < j$, and $M'_k, \pi_k(j) \models_\upsilon \beta$. As from the latter follows $M'_k, s \models_\upsilon E\alpha U^{\leq\eta}\beta$ and

$$|Path'_k| \leq \sum_{0\leq i<j} |Path'_k| + |Path^j_k| + 1 \leq j \cdot f_k(\alpha) + f_k(\beta) + 1$$

$$\leq k \cdot f_k(\alpha) + f_k(\beta) + 1 = f_k(E\alpha U^{\leq\eta}\beta),$$

we conclude the proof of this case and of the lemma. □