Romeo: A Tool for Analyzing Time Petri Nets

Guillaume Gardey[1], Didier Lime[2], Morgan Magnin[1], and Olivier (H.) Roux[1]

[1] IRCCyN, CNRS UMR 6597, Nantes, France
{Morgan.Magnin | Guillaume.Gardey | Olivier-h.Roux}@irccyn.ec-nantes.fr
[2] Aalborg University - CISS, Denmark
didier@cs.aau.dk

Abstract. In this paper, we present the features of Romeo, a Time Petri Net (TPN) analyzer. The tool Romeo allows state space computation of *TPN* and on-the-fly model-checking of reachability properties. It performs translations from *TPNs* to Timed Automata (*TAs*) that preserve the behavioural semantics (timed bisimilarity) of the *TPNs*. Besides, our tool also deals with an extension of Time Petri Nets (*Scheduling-TPNs*) modeling preemption.

Key words: Time Petri nets, model-checking, state-space, DBM, polyhedron, scheduling, stopwatch.

1 Introduction

Time Petri Nets (*TPNs*) are a classical formalism, with Timed Automata (*TAs*) to design reactive systems. *TPNs* extend classical Petri nets with temporal intervals associated with transitions. They benefit from an easy representation of real-time systems features (synchronization, parallelism ...). State reachability and boundedness is proven to be undecidable for arbitrary *TPNs*. However, state reachability is decidable for bounded *TPNs*, which is sufficient for virtually all practical purposes.

In real-time applications, it is often necessary to memorize the progress status of an action when this one is suspended then resumed. In this class of models, some extensions of Time Petri Nets have been proposed to express the preemptive scheduling of tasks. Roux and Déplanche [1] propose an extension for Time Petri Nets (*Scheduling-TPNs*) that consists of mapping into the Petri net model the way the different schedulers of the system activate or suspend the tasks. For a fixed priority scheduling policy, *Scheduling-TPNs* introduce two new attributes associated to each place that respectively represent allocation (processor or resource) and priority (of the modeled task). Bucci *et al.* [2] propose a similar model: Preemptive Time Petri Net (*Preemptive-TPN*) by mapping the scheduling policies onto transitions.

2 The tool Romeo

The purpose of the tool Romeo [1] (Available for Linux, MacOSX and Windows platforms) is the analysis and simulation of reactive systems modelled by *TPNs*

[1] Download at: http://www.irccyn.ec-nantes.fr/irccyn/d/fr/equipes/TempsReel/logs

or *Scheduling-TPNs*. It consists of: (1) a graphical user interface (GUI) (written in TCL/Tk) to edit and design *TPNs*, and (2) computation modules (Gpn and Mercutio, written in C++).

2.1 System Design

In a system modelling activity, Romeo, through the GUI, allows to model a reactive system (preemptive reactive system) by *TPNs* (*Scheduling-TPNs*), both of which benefit from an easy graphical representation and from an easy representation of common real-time features (parallelism, synchronization, resource management, watch-dogs. . .).

As a design helper, Romeo implements on-line simulation (*TPNs*, *Scheduling-TPNs*) and reachability model-checking (*TPNs*). It allows the early detection of some modeling issues during the conception stage.

2.2 On-line Model-Checking

In addition to on-line simulation that makes scenarii testing possible, Romeo provides an on-line model-checker for reachability. Properties over markings can be expressed and tested. It is then possible to test the reachability of a marking such that it verifies $M(P_1) = 1 \vee M(P_3) \geq 3$ where $M(P_i)$ is the number of tokens in the place P_i of the net. The tool returns a trace leading to such a marking if reachable.

Such a model-checker allows to verify more complex properties (quantitative properties for instance) expressed by observers (which translate a quantitative property into a reachability test), which is the main method used to study the behavior of a *TPN*.

2.3 Off-line Model-Checking

The modeling of a property using observers requires a good knowledge in *TPNs* and, as far as we know, no automatic observers generation is available to help a system designer.

Romeo implements different theoretical methods to translate the model analyzed into Automata, Timed Automata (*TAs*) or Stopwatch Automata (*SWAs*). The advantages of such translations is that several efficient model-checking tools are available for these models (Mec, Aldebaran, Uppaal, Kronos, Hytech). These translations also extend the class of properties that can be model-checked with observers to temporal logic (LTL, CTL) and quantitative temporal logic (TCTL).

To our knowledge, the translations of *TPNs* to *TAs* implemented in Romeo are currently the only existing methods allowing the verification of quantitative time properties (quantitative liveness and TCTL) on *TPNs*.

Structural Translation Romeo implements a structural transformation of a *TPN* into a timed-bisimilar synchronized product of *TAs* [3] that can be model-checked with the tool Uppaal. The translation is optimized to take at its advantage the management of inactive clocks in Uppaal. It follows that the algorithms implemented in this tool are used efficiently.

State Space Computation Based Translation A first translation consists in the computation of the state class graphs (SCG) that provide finite representations for the behavior of bounded nets preserving their LTL properties [4]. For bounded *TPNs* the algorithm is based on DBM (Difference Bounds Matrix) data structure whereas for *Scheduling-TPNs* the semi-algorithm is based on polyhedra (using the *New Polka* library).

Two different methods are implemented for *TPNs* to generate a *TA* that preserves its semantics (in the sense of *timed bisimilarity*): the first one is derived from *TA* framework [5], the other one from the classical state class graph approach [6]. In the latter method, we reduce the number of clocks needed during the translation, so that the subsequent verification on the resulting *TA* is more efficient. In both methods, the *TAs* are generated in Uppaal or Kronos input format.

Concerning *Scheduling-TPNs*, the method introduced in [7] is implemented. It allows a fast translation into a Stopwatch Automaton (SWA) using an overapproximating semi-algorithm (DBM-based). Despite the overapproximation, it has been proven that the SWA is timed-bisimilar to the original *Scheduling-TPN*. The SWA is produced in the Hytech input format and is computed with a low number of stopwatches. Since the number of stopwatches is critical for the complexity of the verification, the method increases the efficiency of the timed analysis of the system, and in some cases may just make it possible at all compared to a direct modeling of the system with Hytech (see appendix).

2.4 Comparisons

The following tables are an overview of the features of Romeo compared to two others main tools used for the analysis of *TPNs* and *TPNs* extension dealing with preemption. They compare the capabilities of the tools in terms of the properties classes that can be handled.

Tina [8] is a tool for the analysis of *TPNs* mainly using state class graphs techniques. Oris [9] is a tool that analyzes *Preemptive-TPNs* which are equivalent to *Scheduling-TPNs*.

Our major contribution is to bring to *TPNs* frameworks (*Scheduling-TPNs*) methods to efficiently model-check TCTL properties.

3 Case study

3.1 Description

In this section, we use a partial model for the control of an oscillation compensator (hydraulic shock absorber) and a differential blocking on a tractor with

	Reachability	LTL	CTL	Quantitative Liveness [a]	TCTL
TINA	Marking Graph	SCG[b] + MC[c]	Atomic SCG[b] + MC[c]	–	–
ROMEO	On-the-fly checking or Marking Graph	SCG + MC[c] or ZFG[e] + MC[c]	Translation to Timed Automata + UPPAAL [d] or KRONOS		

Table 1. ROMEO capabilities on *TPNs*

[a] Includes response properties like $\forall\Box(\varphi \Longrightarrow \forall\Diamond\Psi)$ where φ or Ψ can contain clock constraints.
[b] SCG = Computation of the State Class Graph.
[c] MC = requires the use of a Model-Checker on the SCG or the ZFG.
[d] UPPAAL implements a subset of TCTL and a special type of liveness defined by formulas of the form $\forall\Box(\varphi \Longrightarrow \forall\Diamond\Psi)$.
[e] ZFG = Computation of the Zone-based Forward Graph.

	State-space computation		Timed analysis	
	Overapproximation	Exact abstraction	RTTL	TCTL
ORIS	DBM	–	DBM-SCG[a] + MC[b]	–
ROMEO	DBM	SCG[c]	Efficient translation to timed bisimilar Stopwatch automata + HYTECH	

Table 2. ROMEO capabilities on *Scheduling-TPNs*

[a] Computation of a DBM over-approximation of the State Class Graph.
[b] Oris supports exact timeliness analysis of traces (with respect to a linear-time variant of Real-Time Temporal Logic (RTTL). Since the analysis may introduce approximate time profiles, the identified traces may be cleaned up.
[c] As for *TPNs*, the SCG preserves LTL properties.

a sowing trailer. The partial system consists of processors running a real-time operating system, linked together with a CAN bus.

We use the translation of a *Scheduling-TPN* into a *SWA* which state space is computed with HYTECH.

We compared the efficiency of our method with a generic direct modelling with HYTECH on this case study. We also tested several simpler and more complex related systems obtained by removing or adding tasks and/or processors. Table 3 gives the obtained results. The case-study in Figure **??** corresponds to example 4 in the table.

Columns 2 and 3 give the number of processors and tasks of the system. Columns 4, 5 and 6 describe the direct modelling in HYTECH results: the number of SWA used to model the system, the number of stopwatches and the time taken by HYTECH to compute the state space. For this generic modelling, we basically used the product of one SWA per task and one SWA for each scheduler. Columns 7, 8, 9 give the results for our method: the number of locations/transitions, stopwatches of the SWA we generated, and the time taken for its generation. Finally, the last column gives the time used by HYTECH to compute the state space of the SWA generated by our method. Times are given in seconds and NA means that the HYTECH computation could not yield a result on the machine used.

	Description		Direct SWA Modelling			Our method (SETPN$\xrightarrow{\text{ROMEO}}SWA\xrightarrow{\text{HYTECH}}$state-space)				
Ex.	Proc.	Tasks	SWA's	Sw.	HYTECH Time	Locations	Transitions	Sw.	ROMEO Time	HYTECH Time
4	3+CAN	7	13	11	NA	297	575	7	0.3	5.3
5	4+CAN	9	15	13	NA	761	1677	8	0.9	29.8
6	5+CAN	11	17	15	NA	1141	2626	9	6	60.1
7	5+CAN	12	18	16	NA	2155	5576	9	8.3	56.5
8	6+CAN	14	.	.	NA	4587	12777	10	59.7	438.8
9	6+CAN	15	.	.	NA	4868	13155	11	96.5	1364.3
10	6+CAN	16	.	.	NA	5672	15102	11	439.1	1372.5
11	7+CAN	18	.	.	NA	8817	25874	12	1146,7	NA

Table 3. Experimental results

These computations have been performed on a POWERPC G4 1.25GHz with 500MB of RAM.

We observe that the computation on a direct modelling as a product of SWA is quickly untractable (Example 3). However, with our method, we are able to deal with systems of much greater size.

References

1. Roux, O.H., Déplanche, A.M.: A t-time Petri net extension for real time-task scheduling modeling. European Journal of Automation (JESA) **36** (2002) 973–987
2. Bucci, G., Fedeli, A., Sassoli, L., Vicario, E.: Time state space analysis of real-time preemptive systems. IEEE transactions on software engineering **30** (2004) 97–111
3. Cassez, F., Roux, O.H.: Structural translation from time Petri nets to timed automata. In: Fourth International Workshop on Automated Verification of Critical Systems (AVoCS'04). ENTCS, London (UK), Elsevier (2004)
4. Berthomieu, B., Diaz, M.: Modeling and verification of time dependent systems using time Petri nets. IEEE trans. on software engineering **17** (1991) 259–273
5. Gardey, G., Roux, O., Roux, O.: State space computation and analysis of time Petri nets. Theory and Practice of Logic Programming (TPLP). Special Issue on Specification Analysis and Verification of Reactive Systems (2005) to appear.
6. Lime, D., Roux, O.H.: State class timed automaton of a time Petri net. In: 10th International Workshop on Petri Nets and Performance Models, (PNPM'03). (2003)
7. Lime, D., Roux, O.H.: A translation based method for the timed analysis of scheduling extended time Petri nets. In: The 25th IEEE RTSS'04, Lisbon, Portugal (2004)
8. Berthomieu, B., Ribet, P.O., Vernadat, F.: The tool tina – construction of abstract state spaces for petri nets and time petri nets. International Journal of Production Research **42** (2004) Tool available at http://www.laas.fr/tina/.
9. Bucci, G., Sassoli, L., Vicario, E.: Oris: A tool for state-space analysis of real-time preemptive systems. Quantitative Evaluation of Systems, First International Conference on (QEST'04) (2004) 70–79