ARTICLE IN PRESS YINCO:4055

Information and Computation ••• (••••) •••–•••

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

Parametric multisingular hybrid Petri nets: Formal definitions and analysis techniques

Hassan Motallebi, Mohammad Abdollahi Azgomi *

Trustworthy Computing Laboratory, School of Computer Engineering, Iran University of Science and Technology, Tehran, Iran

ARTICLE INFO

Article history:
Received 29 December 2012
Received in revised form 17 August 2014
Available online xxxx

Keywords:
Parametric multisingular hybrid Petri nets ($\mathcal{P}$-MSHPNs)
Parametric speed-based partitioning graph
Parametric state class graph
Parametric invariant checking
Optimum configuration
Linear programming

ABSTRACT

Multisingular hybrid Petri net (MSHPN) is an extension of hybrid Petri nets enriched with the capabilities of hybrid automata to achieve the practical expressive power of multisingular hybrid automata. In this paper, we define parametric multisingular hybrid Petri nets ($\mathcal{P}$-MSHPNs), as a parametric extension of MSHPNs. We present the parametric reachability analysis techniques and algorithms and prove that the parametric reachability analysis of $\mathcal{P}$-MSHPNs amounts to the analysis of standard MSHPNs. Once the parametric state space of a $\mathcal{P}$-MSHPNs model is computed, it can either be used for parametric model checking analysis using the existing techniques or be instantiated to obtain non-parametric state spaces. $\mathcal{P}$-MSHPNs models can be analysed to obtain the set of feasible configurations for the system parameters. We also give a method for deriving the set of constraints on the parameters that ensure the correctness of an invariant property and a method for finding the optimum parameter configuration.

1. Introduction

Hybrid systems are a wide family of systems that involve both continuous evolution and discrete behaviour. In the specification of these systems, continuous variables may be used not only for continuous quantities but also as an approximation for natural variables of a discrete event system where the changes in the quantity of that variable happen at a constant rate. The transformation of such natural variables into non-negative reals is quite reasonable, especially in very populated or high traffic systems [1]. Although in real systems, the variables may be actually discrete, usually their continuous approximation leads to suitable results. Most importantly, since the huge number of events in discrete-event systems causes the so called state explosion problem, this relaxation (fluidification) technique can solve or reduce this problem significantly. In addition, this technique in which discrete changes with constant rate in the value of a discrete variable are approximated with continuous evolution of its corresponding real variable, leads to improvements in both complexity and decidability [2].

Hybrid automata (HA) [3] are a mathematical model for the specification of hybrid systems. Due to their ease of analysis [4], linear hybrid automata (LHA), a subclass of hybrid automata, is known as a powerful analysis tool for hybrid systems. However, hybrid automata are not a suitable formalism for the specification of some behavioural aspects of hybrid systems. In the following, we mention some of the problems of modelling hybrid systems with hybrid automata:

* Corresponding author at: School of Computer Engineering, Iran University of Science and Technology, Hengam St., Resalat Sq., Tehran, Iran, Postal Code: 16846-13114. Fax: +98 21 73225322.

E-mail addresses: hmotallebi@iust.ac.ir (H. Motallebi), azgomi@iust.ac.ir (M. Abdollahi Azgomi).

http://dx.doi.org/10.1016/j.ic.2014.11.001
0890-5401/

Please cite this article in press as: H. Motallebi, M. Abdollahi Azgomi, Parametric multisingular hybrid Petri nets: Formal definitions and analysis techniques, Inf. Comput. (2014), http://dx.doi.org/10.1016/j.ic.2014.11.001

- The communications among different automata in a network of hybrid automata [3] is a very restricted form of communication happening only through hand-shaking. This simple form of communication is not appropriate for the specification of systems with more sophisticated types of communications. For example, consider a communication system consisting of several machines whose communications are continuously happening through the uploading channels (continuously changing real variables). Since the components of a network of automata cannot communicate through real variables, such systems cannot be described naturally as the networks of hybrid automata. Therefore, although, many hybrid systems with concurrent components may be specified as networks of hybrid automata, in the above mentioned situations, different system components cannot be specified as the networks of hybrid automata. Consequently, as the complexity and the level of parallelism of these systems increases, the dimension of the hybrid automata model increases dramatically. For this reason, the specification of complex hybrid systems using hybrid automata may need exhaustive encoding of too many graph nodes.
- In the specification of a hybrid system using hybrid automata, the modeller may have to deal with conflict resolution and speed computation issues beside the specification issues. For example consider the following simple example:

Example 1 (*File compressor program*). Consider a simple file compressor program with two buffers in a memory of capacity N. Assume the source file stream is read by a reader thread with the rate 6 and stored in a buffer x. The data in x is consumed by a compressor thread with the rate 3 and is compressed with the compression factor $2/3$. Also, assume that the data in the buffer x can be processed only if the amount of data in x is more than the threshold equal to 6. The result is stored in another buffer y until it is written to a destination file by a writer thread with the rate 1. In order to continue their processing, both the reader and compressor threads need some empty space in the memory to put their output results. However, shortly after these threads start their work, the memory becomes full and the reader and compressor threads have to compete with each other for the empty space provided by the execution of the writer thread. In such a situation, a conflict arises between the reader and the compressor threads and the relative execution speed of these threads need to be computed. In fact, if we model this system as a hybrid automaton with the real variables x and y, for different valuations of these variables their evolution rates should be computed. Therefore, while modelling many systems using linear hybrid automata, first the modeller has to deal with conflict resolution and speed computation issues. The modeller need to compute the flow conditions of the real variables of the system, which in many situations can be quite complicated. Some examples of such situations are given in this paper.

Also, we have the same problems in modelling hybrid systems with other automaton-based models like timed automata (TA) [5,6].

In addition to automaton-based models, there are also some Petri net-based formalisms for specification of hybrid systems. Different flavours of timed, continuous and hybrid Petri nets [7–11] are examples of these formalisms. Although, Petri net-based models may provide powerful tools for describing some behavioural aspects of hybrid systems, there are systems that can be described using the automaton-based models, but cannot be specified using the Petri net-based models [2,12–14].

Recently, the relative expressive power and capabilities of these models are investigated and it is shown that these Petri net-based models have less expressive power than their automaton counterparts [2,12–15]. As discussed in [2,12], models like timed Petri net (TPN) [8] and hybrid Petri net (HPN) [7] are special cases of timed automata and linear hybrid automata (LHA), respectively. Therefore, there are hybrid systems that can be described using timed automata (respectively, LHA), but cannot be specified using TPN (respectively, HPN).

Therefore, Petri net-based formalisms do not have enough expressive power and automaton-based models have problems in specification of some concurrent systems. Bridging the gap between Petri net-based formalisms and automaton-based models and associating the modelling capacities of Petri net-based formalisms with the analysis power of automaton-based models have been considered as a challenging issue [7,14,16,17]. Some works have tried to address this issue by proposing algorithms for translations among these formalisms [8,17–19] while some others have given modelling formalisms with the capabilities of both families of models [20].

Multisingular hybrid Petri net (MSHPN) [20], is an extension of hybrid Petri nets enriched with the capabilities of hybrid automata to achieve the practical expressive power of multisingular hybrid automata (MSHA). In MSHPNs, hybrid Petri net model is equipped with the capabilities of hybrid automata to control the execution and firing of transitions. In order to control its execution and firing, to each timed transition in an MSHPN model are assigned: (1) an execution predicate, (2) a firing predicate and (3) a deadline predicate.

In order to perform model checking analysis on a system, a prior knowledge of the system and its environment is needed. We use this knowledge to specify the model and perform analysis on it. However, if the system environment changes, then, the model checking process should be repeated again for the new environment. Besides, obtaining a complete knowledge of a system can be impossible. In many applications, a system is defined by parameters that are in relation with several other systems. While traditional approaches can only be used to verify concrete timing properties (without parameters), there are parametric models and tools which can be used for parametric modelling and verification of timed and hybrid systems. In some of these tools, parameters must be instantiated to perform analyses; however, some other methods allow direct analysis of the parametric models without the need for instantiating parameters [21].