Why discrete-time Petri Nets with stopwatches?

Morgan MAGNIN
Jean-Pierre Elloy, Pierre Molinaro, Olivier (H.) Roux
JDOC 2006

Introduction

Before selling a new model to its clients, a car manufacturer must be sure of the quality of its vehicle. Among all, the correctness of the whole computational architecture constitutes a critical factor (a car can now contain more than thirty processors). If some bugs remain, the fall-outs may be disastrous not only for the image of the company, but also for the safety of the driver.

As systems demanding correctness proofs increase in complexity, their validation cannot be achieved by an exhaustive simulation of all their potential behaviors. Thus efficient formalisms, like Time Petri Nets (TPNs), and model-checking techniques are needed.

TPNs are however not expressive enough to model the preemptive scheduling of tasks, i.e. actions that can be suspended and resumed. That is why an extension of TPNs that addresses the modeling of stopwatches (instead of simple clocks) have been introduced: Stopwatches Petri Nets (SwPNs). The complexity of the SwPN model is such that many problems have been proven undecidable as long as a dense-time analysis is considered. That means there will never be any automatic procedure that computes the state space of the net. In order to obtain a finite abstraction of the state space, we apply a discrete-time approach instead of a dense-time one.

Petri Nets with Stopwatches

Petri Nets with stopwatches [1] (SwPNs for short) are a time extension of classical Petri Nets. They allow to represent easily common structures in timed systems: synchronisation, parallelism, mutual exclusion, preemption … Informally, to each transitions of the net is associated a stopwatch and a time interval. The clock measures the time since the transition was enabled and the time interval is interpreted as a firing condition: the transition may fire if its clock value belongs to the time interval.

In the dense-time approach, time is considered as a continuous variable whose evolution goes at rate 1. By contrast, in the discrete time approach, time is seen as “jumping” from one integer to the other, with no care of what may happen in between. The latter is an under-approximation of the former.

The semantics of a discrete-time SwPN consists in two different actions:

- a discrete transition: the firing of an enabled transition,
- a discrete-time transition: the net stays in its current marking while time is increasing of one time unit.

Analyzing a discrete-time bounded SwPN

Why working with discrete-time instead of dense-time?

The marking reachability problem has been proven undecidable for dense-time SwPNs. On the contrary, we have proven the following theorem:

**Theorem 1** The marking reachability problem is decidable for discrete-time bounded SwPNs.

This implies the discrete-time approach leads to a finite state space abstraction even for nets whose dense-time state space computation does not terminate.

An efficient method for computing the state space of SwPNs

In the time extensions of Petri nets, temporal information is modelled implicitly. In [2], we have proven the following result: as far as discrete-time is considered, it is possible to simulate the elapsing of time with a special tick transition added to the classical Petri net formalism. The clock associated to each transition is then viewed as a place whose marking ranges over the domain of natural numbers and is incremented by one with every tick transition. Thus the state space of discrete-time bounded SwPNs can be computed directly by using existing tools for classical (untimed) Petri nets like Markg [3].

Figure 2: A simple example: translating a TPN into an untimed Petri net

Experimental results comparing the discrete-time state space computation and the dense-time one show that the discrete-time approach is interesting for many systems of practical interest.

References

