Implementation of Weighted Place / Transition Nets based on Linear Enabling Functions *

J.L. Briz and J.M. Colom

Depto. Ingeniería Eléctrica e Informática
Universidad de Zaragoza, c/ María de Luna, 3 (Pol. Actur)
50.015 Zaragoza, Spain
e-mail: briz@etsii.unizar.es

Abstract. Petri Nets should be implemented in an efficient and reliable way, specially when they are going to be used for critical problems, like that of giving support to Discrete Event Systems Simulation, whichever sequential or parallel strategies are adopted. One of the critical points while implementing a Petri Net, is that of determining whether a transition is enabled. In this contribution we classify transitions in several classes. The enabling of a transition is characterized by means of a Linear Enabling Function (LEF), that depends on the class. For some classes a transformation must be applied, preserving the behavior of the net. We show how LEFs can be applied to build a Simulation Engine that uses as data structure a DES described in terms of a Timed Petri Net, taking benefit of the properties of LEFs.

Keywords. Simulation of Weighted Place-Transition Systems, Timed Petri Nets, Linear Enabling Functions, Structure and Behavior of nets.

1 Introduction

The interest in parallel and distributed systems grows constantly as they are introduced in new domains. Their complex nature makes essential simulation and verification techniques during the design process, to prevent the occurrence of bad behaviors, to ensure that certain good properties hold, and to evaluate the performance. Petri Nets (PNs) have been pointed out as a good modelling tool, since many properties may be easily analyzed in a great number of cases. Moreover, when formal analysis becomes impracticable, the model may be simulated. As simulation tool, PNs allow the formulation of models with realistic features (as the competition for passive resources) absent in other paradigms (as nude queueing networks).

Given a PN model of a discrete physical system, we simulate the system by playing the token game on that PN, i.e. by firing transitions as a result of the available tokens. This is also referred as implementing the PN. If a deterministic or stochastic time interpretation is associated to transitions – Timed PNs (TPNs)

* Partially supported by grants CICYT TIC-91-0354 of the Plan Nacional de Investigación and P IT-6/91 of the CONAI (Diputación General de Aragón)
or Stochastic PNs (SPNs) —, the implementation of the TPN or SPN yields, actually, a Discrete Event Simulation system. In fact, SPNs have been proposed as the minimal discrete event notation [13], and both TPNs and SPNs are in the scope of recent works on Discrete Event Systems (DES) and Parallel DES (PDES) simulation [1, 11, 5, 27, 12].

PDES has arisen as a promising way of reducing simulation costs. However, sequential DES is still more efficient in a great number of cases, because of the overhead produced by conservative and optimistic strategies, which avoid causality errors or detect and recover from them. Whichever approach is used — sequential DES or PDES —, the final goals are always efficiency and reliability. Sequential event-driven simulation methods try to improve the list processing capabilities, and both conservative and optimistic approaches to PDES try to reduce communication costs [2, 8]. When PNs are used as the underlying frame, some of the heavy simulation components may be lightweighted, but the problem of how the token game can be efficiently carried out in a reliable way becomes critical. Particularly, the enabling test of the transitions plays a paramount role, since it is a rather time-consuming operation. This PN implementation problem, that has been present in the PN community since more than ten years ago (see [20, 16, 7, 24, 22, 25, 10] as some significant different approaches), recovers now a new meaning.

According to different criteria, the implementation of a PN may be compiled or interpreted, sequential or concurrent, centralized or distributed, synchronous or asynchronous [25]. For sequential, centralized methods, the solutions addressed to reduce the costs of the enabling test broadly fall into one of the following classes:

1. **Place-driven approaches.** Only the output transitions of some representative marked places are tested for firability. This gives a characterization of the partial enabling of transitions.

2. **Transition-driven approaches.** A characterization of the enabling of transitions is supplied, and only enabled transitions are considered. The firing of a given transition modifies the enabling conditions of the transitions connected to its input and output places. In general, it is not necessary an explicit representation of the marking.

Place-driven approaches have been extensively considered under different names [6, 21, 18, 24, 25]. Its main problem is the selection of a good set of representative places. A representative place is good when, if it is marked, its output transition is enabled. Today there is no practical way to compute this set nor to measure its quality. Nevertheless, very few works concern the transition-driven approach. These works either deal only with binary PNs [21, 18], or make a very inefficient characterization of the enabling, e.g. based on counters, which represents a clear disadvantage regarding to the place-driven approach [6].

For distributed implementations of PNs, the enabling test highly depends on the way in which places and transitions are represented. Thus, it may be carried out by means of complex reservation protocols, as in [22], or by following a client-server approach [3, 10].