ABSTRACT

We present here an approach using rewriting systems for analysing and proving properties on Petri nets. This approach is implemented in the system PETRIREVE. By establishing a link between the graphic Petri net design and simulation system PETRIPOTE and the term rewriting system generator REVE, PETRIREVE provides an environment for the design and verification of Petri nets. Representing Petri nets by rewriting systems allows easy and direct proofs of the behaviour correctness of the net to be carried out, without having to build the marking graph or to search for net invariants.

I - INTRODUCTION

Petri nets were developed in order to modelize concepts of asynchronous and concurrent processes. Petri nets are widely used: their formalism is simple and allows a number of properties to be modelized. Various proof techniques are available to check these properties [Bra 83]. The design and verification of Petri nets require an integrated and highly interactive environment; such an environment should comprise a graphic editor and tools for checking the behaviour of the net and its structural properties. PETRIREVE provides such an environment by establishing a
link between a graphic system for Petri net design and simulation, PETRIPOTE [Bea 83], and the rewriting "laboratory" REVE [Les 83, FG 84]. PETRIREVE builds a set of equations that corresponds to a net designed under PETRIPOTE. Thanks to an appropriate ordering on places computed by PETRIREVE, orienting the set of equations into rewrite rules is in accordance with the transition firing. The completion algorithm transforms this system into a simpler representation of the net (thus easier to analyse). The operation correctness and structural properties of the net can be checked in different ways: some properties of the net (boundedness, confluence) are related to the properties of the rewriting system; other properties can be checked by adding the corresponding equation to the rewriting system, in particular, invariance, reachability of a marking, proper termination, and quasi-liveness.

We shall briefly sketch the properties one wants to prove for Petri nets, before describing our approach to representing the behaviour of a Petri net with a rewriting system; we then address the meaning of the rules generated by REVE on the representing system w.r.t. the behaviour of the net and of the completed resulting system (that represents a simpler net). Finally, we present the correctness proofs which can thus be carried out directly using the ordering provided by REVE and the completion algorithm.

II - PETRI NETS

II.1 - Specification and design of Petri nets

Among the various tools allowing the representation of concurrent processes, Petri nets can be distinguished by their wide use, and the practical and theoretical results they yield. A Petri net is defined as a tuple (P, T, Pre, Post), P being a set of places, T a set of transitions, Pre and Post being the towards and forwards incidence functions (integer functions defined over the cartesian product of places and transitions). The behaviour of a net is specified by the following rule: a transition t is firable if and only if, for all places p, the number of tokens is greater than Pre(p, t). When a transition t is fired, for all places p, the number of tokens is decreased by Pre(p, t) and increased by Post(p, t). This rule specifies how firing