PAPETRI : Environment for the Analysis of PETRI nets

G. BERTHELOT (*), C. JOHNEN (**), L. PETRUCCI (***)

(*) CEDRIC-IIE, 18 Allée Jean Rostand, BP 77, 91002 EVRY CEDEX

(**) L.R.I., U.A. au CNRS 410, Bât. 490, Université Paris-Sud, 91405 ORSAY CEDEX

(***) MASI, tour 65, Université Pierre et Marie Curie, 4 place Jussieu, 75252 PARIS CEDEX 05

Work supported by ESPRIT project DEMON (BRA 3148).

Abstract

In this paper, we present PAPETRI, a general and integrated environment for editing and analysing Petri nets. PAPETRI allows us to work with different classes of nets. Several analysis tools are available for each of these classes.

The kernel of PAPETRI is the graphical and interactive editor, PETRIX. Five analysis tools are available : a graphical simulator of Petri nets, two constructors of covering graphs, a generator of semi-flows (COMBAG), a method using rewriting techniques (PETRIREVE), an analyser and simulator of algebraic nets (VAERA). Each of these tools, as well as PETRIX, is detailed hereafter.

1/ PAPETRI : overview

Petri nets[1] and their extensions are special classes of transitions systems where states are specified by tuples and transitions by a couple of tuples representing respectively precondition and modification of the state.

The goal of PAPETRI (in French, *Poste pour l'Analyse des réseaux de PETRI*) is to provide a friendly editing environment and to afford a great deal of analysis tools

[1] Basic notions and definitions on Petri nets can be found in [Rei85].

for several classes of Petri nets. These tools are not only based on nets classical methods but also on original approaches using rewriting techniques and abstract data types.

A particular care was accorded, in the conception of PAPETRI, on one hand to the flexibility of edition and on the other hand to the easy integration of new tools. The actual functionnalities are given hereafter, for each class of nets :

- *Petri nets* : simulation, construction of covering graphs, computation of a generative family of semi-flows, analysis with rewriting techniques;
- *Coloured nets* [Jen81] : construction of covering graphs, semi-flows calculus;
- *Algebraic nets* [Vau85] : syntactic verification, simulation, skeleton analysis (Petri net), normed net analysis (algebraic net).

This list will be enlarged : the edition of nets may be extended to other classes, and other tools can be integrated in PAPETRI.

The main function of PAPETRI is PETRIX, a graphical and interactive environment for the edition of nets. The characteristics of PETRIX are detailed in section 2. The commands which may be used are called throughout a set of unfolding menus and dialog windows. They also allow us to call analysis tools for the class of the edited net.

Five analysis tools are presently integrated in PAPETRI :

- *Simulator* : graphical simulation of token game, described in section 2;
- *Covering graphs* : a first procedure builds the classical Karp and Miller covering graph and allows to check some net properties. A second one computes the minimal covering graph. These two tools are presented in section 3;
- *COMBAG* : semi-flows computation, introduced in section 4;
- *PETRIREVE* : analysis of Petri nets using rewriting techniques, described in section 5;
- *VAERA* : tool for syntactic verification, simulation and analysis of algebraic nets, detailed in section 6.

2/ PETRIX : the editor-simulator

The graphical and interactive editor of nets, PETRIX, uses the XWINDOWS environment on UNIX workstations.

PETRIX allows to draw a net, using the mouse and unfolding menus. It is possible to create, modify, drag and suppress the basic objects of a Petri net that are places, transitions, arcs and tokens. An arc joining a place and a transition together may be created as a broken arrow between the two objects. A token must be created inside a place but may be moved into another one. So, the graph obtained is a correct Petri net. Each of the objects may be named. It is possible to create texts which are either free (comments, title of the net, ...), or linked to another object. The shape and the size of graphical objects may be changed.

Several nets can be edited at the same time, each of them being displayed in separate windows. Operations such as cut (or copy)/paste allow the user to insert a part of a net into another net.

The size of edited nets is only limited by the available memory (several thousands of objects for 1Mb).

The simulation of an ordinary net allows to play the token game using the usual enabling rules. Evolutions can be viewed, as the marking of each place is modified in the window. The token game can be performed according to three modes : manual, semi-automatic or automatic. In manual mode, the simulator selects the enabled transitions and the user chooses the one he wants to occur. In semi-automatic mode, the evolution is done as long as only one transition can occur. Otherwise, all enabled transitions are highlighted and the user chooses the one to occur. In automatic mode, choices are randomly made. The simulation stops when no transition can occur. The user can stop the simulation whenever he wants, in any mode.

3/ Covering graphs

The more classical technique for the analysis of Petri nets consists in the construction of the *covering graph* : all the reachable markings are listed; when the set of reachable markings is infinite, a shortened list leads to build a bounded graph [KM69]. Important properties (boundedness of places, quasi-live transitions, regularity of the language) are decidable by procedures which analyze the graph.

A procedure to build this graph is available for places/transitions nets and coloured nets in PAPETRI. Once the graph is computed, an analyser of covering graphs provides diagnosis functions for the net as well as a visualisation of the graph. These functions are obtained through a menu.

The analysis functions concern the following properties : regularity of the language, existence of repetitive sequences of transitions, list of unbounded places and quasi-live transitions.

The analyser can compute the covering graph terminal connected components, which gives more information concerning liveness of transitions, home states, deadlocks of the net.

The mutual exclusion between places can be tested. The users specifies the places he wants to test the mutual exclusion of, then the analyser tests it on all nodes labels. Moreover, the analyser indicates whether these places can be empty all together.

The user can specify a sequence of transitions to occur, starting from the initial marking, and the analyser indicates the node obtained if the sequence labels a path in the graph. In the case of a bounded net, this function allows a behavioural study.

The graph may be scanned : the analyser lists all nodes labels.

The construction of a covering graph is expensive as well in time as in space. These difficulties may be reduced by the use of the minimal covering graph [Fin89] : are only kept the nodes with uncomparable labels. The number of nodes may thus be drastically decreased (several orders of magnitude). The construction of such a graph, both for places/transitions nets and coloured nets, was integrated in PETRIX [Mor89]. For the moment, the diagnosis functions are not available with this sort of graph.

For coloured nets, it could be possible to reduce furthermore the size of covering graphs, by the use of this technique combined with Jensen's, which are based on symmetries between colours ([HJJJ86]).

The construction of the covering graph allows to analyse the net for only one initial marking. However, a lot of nets model systems with various initial states. In our example, the protocol must be valid whichever the number of trains departing from West_Terminus or East_Terminus is. This goal can be reached by the study of semi-flows.

4/ COMBAG : semi-flows computation

In this section, we present COMBAG [Tre88] : a structural analysis method (i.e. which relies only on edges between places and transitions) which allows the validation of net properties independently of the initial marking. COMBAG is a tool for semi-flows computation in a Petri net. Two kinds of semi-flows are derived from the incidence matrix C :

- a *p-semi-flow* f weights every place such that $f^T.C = 0$. It leads to a net invariant : the weighted sum of tokens in places is constant for any reachable marking;

- a *t-semi-flow* f associates a number of occurrences with every transition such that C.f = 0. It denotes the repetitive and stationnary sequences of transitions.

In the following, we will use the term *semi-flow* to denote as well p-semi-flows as t-semi-flows.

The semi-flows over $\mathbb{N}$ (weights are positive integers) allow a finite generative family : every semi-flow over $\mathbb{N}$ can be expressed as a linear combination of the semi-flows in the generative family.

COMBAG calculates a generative family of semi-flows over $\mathbb{N}$, using Farkas' algorithm, the complexity of which is exponential w.r.t. time. To decrease the answering delays, COMBAG proposes, at each step, two optimization techniques. The first one consists in looking for the incidence matrix column that generates less operations, and the other one suppresses unusefull lines.

The use of coloured nets, which shorten Petri nets, leads to more concise nets. The arcs can be valued by variables (in this case, any token may be used, with no constraint as concerns its colour), or by colours (the token must have the same colour as the valuation).

COMBAG provides semi-flows for coloured nets having a finite number of colours. It calculates a generative family for three classes of semi-flows over $\mathbb{Z}$ (weights are signed integers) of a coloured net [VM84] :

- *type 1* : semi-flows f such that $f^T.|C| = 0$. They indicate invariant assertions on the number of tokens in places, without considering their colour.
- *type 2* : semi-flows f such that $f^T.C = 0$. They denote invariant assertions on the number of tokens of a colour, valid for every colour (except the distinguished neutral one).
- *type 3* : semi-flows f such that $f^T.\Pi_a(C) = 0$ (where $\Pi_a(C)$ is the projection of the incidence matrix on colour a). They represent invariant assertions on the number of tokens with colour a. This sort of semi-flows can be calculated for each colour (except the black one).

This technique enables us to find out structural properties of a net, but it does not give any piece of information concerning the net behaviour. In the next section, we introduce an original approach for the analysis of nets w.r.t. a class of initial markings.

5/ PETRIREVE : analysis with rewriting techniques

PETRIREVE [CJ85] analyses Petri nets with validation techniques based on rewriting systems. This technique studies the behaviour of a net independently of the initial martking, and thus validates structural properties of the net (termination).

PETRIREVE builts a set of oriented equations representing the behaviour of the net. Knuth-Bendix's completion transforms these equations into a rewriting system. To do so, PETRIREVE uses system REVE [FG83] - rewriting laboratory - developped by the CRIN (Nancy, France) and the MIT (Cambridge, USA). Studying the obtained rewriting system leads to straightly deduce some net properties (confluence, boundedness, ...). Other properties (invariants, quasi-liveness, reachability, termination, ...) may be validated by testing the completion of the rewriting system to which a specific equation is added.

We conceived a general technique to represent the behaviour of a net by a canonical rewriting system. A marking is represented by a term with operator *state* as header, this operator having as many arguments as there are places in the net. Each argument corresponds to a place : the value of the argument is the number of tokens in the place.

Each transition is represented by an equation. The left term denotes the precondition necessary for the transition to occur. Every term that unifies with the left term denotes a marking for which the transition can occur. The right term represents the state obtained after the occurrence of the transition.

PETRIREVE creates other equations in order to rewrite any term representing a deadlock as an identical term where the header is *deadlock.*

The rewriting system obtained after performing Knuth-Bendix's completion must converge to be used for proofs. Moreover, the equations have to be oriented as the occurrences of transitions. These two requirements leaded us to design an order on place-arguments of *state* to ensure termination, if possible, taking into account the orientation induced by transitions.

PETRIREVE automatically constructs the equations related to the net behaviour. These equations are transmitted to REVE which transforms them into a rewriting system with Knuth-Bendix's completion. The completion reduces the two members of an equation into their normal form and orientates the equation. Critical pairs eventually generate new equations. So, on one hand, rules can be redundant and thus suppressed of the system. On the other hand, these new equations may lead to reduction of already existing rules.

As the rewriting system is canonical, proofs which usually need a recurrent procedure can be performed. The equation to satisfy is added to the rewriting system. If the system (with this new equation) completion does not build any equation between

generators (s, 0), then the equation is satisfied in the initial algebra [HH80]. To verify a p-semi-flow, the user adds a rule corresponding to the associated invariant to the system. It is as follows :

inv(state(before occuring) == inv(state(after occuring)).

If these equations create rules between generators, the invariant is not coherent with the occurrence of the transition. As the completion does not create any equation between generators, the invariant is valid.

We deduce, with this technique, the termination of the net for a class of initial markings (and not for only one initial marking).

PETRIREVE is the first tool to verify the termination of a net for a class of initial states (a net terminates from an initial state if it always reaches a unique terminal state without successors). For the moment, PETRIREVE only analyses Petri nets. It may be extended to other classes of nets. However, the delays of completion will certainly be quite long.

6/ VAERA : Verification, Analysis and Evolution of algebRAic nets

The algebraic nets defined by J. Vautherin [Vau85] add another extent to Petri nets. The main interest of such a sort of high level nets is structuring data by the use of a specification of abstract data types. This is consistent with recent programming methods.

Let us shortly recall the way an algebraic net works. Such a net consists in the association of two components : a specification of abstract data types, which describes the data types used and the operations on them, and a Petri net. The places of the net are sorted, the tokens being constants of the specification, of the same sort as their containing place. The arcs are valued by terms of the specification.

The transitions are enabled according to the following criteria : first of all, the conditions on the amount of tokens in places must be valid, as in places/transitions nets, secondly, conditions between arcs valuations and tokens values must be satisfied. Each arc entering a transition is valued either by a term without variables, or by a single variable. In the first case, a token with this value must be used to fire the transition. In the second case, the variable takes the value of one of the tokens for the occurrence of the transition. Afterwards, if the transition occurs, the term associated with each arc exiting the transition is evaluated (variables occurring in the term have the value given when examining the entering arcs), and a token having this value is

created in the corresponding place. Transitions may also have an auxiliary condition (sound equation) associated with them. This equation, the hand-sides of which are functions of the entering variables, must be valid for the transition to occur.

The analysis of algebraic nets is more complex than the analysis of places/transitions nets, due to the powerfulness of abstract data types. Tokens meaning is denoted as well by their presence in places as by their value. Two tokens within a same place may have different interpretations. We will first explain how PETRIX deals with the definition of an algebraic net, and we will detail afterwards the analysis tools we have designed.

When the user wants to design an algebraic net, he describes the Petri net part with PETRIX and uses VAERA for everything related to algebraic specifications. These must have been created by ASSPEGIQUE (in French, ASsistance à la SPEcification algébrIQUE, see [Cho88]), environment for algebraic specification developped at the LRI (Orsay, France), which can be called by VAERA.

Once the net is created, VAERA provides a tool for syntactic verification of the terms in the net and checks the coherence of the sorts used (tokens must be of sort of the place they are in, ...).

It is also possible to use the simulator included in VAERA to observe the behaviour of the net : the user selects, with the mouse, the transition to occur. The tool verifies that the conditions necessary for the transition to occur are satisfied, and if true, fires the transition as described before. As tokens may have different values, two modes of simulation are available : either the user chooses which tokens will be used for the transition to occur, or the tokens are randomly chosen by VAERA.

Let us now introduce our various analysis tools. It may be interesting, in order to analyse algebraic nets, to yield simpler models by ignoring some pieces of information. These models are the skeleton and the normed net. Indeed, the difficulty of analysis of algebraic nets countervails their expression power.

The skeleton of an algebraic net :

The first model that we will study is the underlying Petri net, named *skeleton* (see [Vau85]). VAERA can automatically generate the skeleton of an algebraic net : auxiliary conditions for transitions to occur, arcs valuations and tokens values are forgotten. Thus, we obtain a net which can be analysed by the places/transitions tools previously described.

Some properties such as the boundedness of a place, the non quasi-liveness of a transition and the termination of a net are true for the algebraic net if they are true for its skeleton.

The normed net :

Another model extracted from the algebraic net is the *normed net* (see [Pet88]). The normed net is a model between the algebraic net (the most structured model) and the skeleton (the less structured model). This net contains less information than the algebraic net, but more than the skeleton. The analysis of the normed net is indeed more complex than the analysis of the skeleton, but its importance leans upon the degree of information it contains. This is the reason why its analysis allows us to detect properties of the algebraic net which are not valid for the skeleton. The main analysis which can be performed concerns the termination of the net.

The normed net is derived from the algebraic net by forgetting the auxiliary conditions for transitions to occur and by changing the valuation of arcs as well as the values of tokens into their norm. The norm maps a term to its "size". For example, as concerns files, function *norm* counts the number of elements in the file.

To generate this net, the user will have to include in his specification the definition of function *norm*. The net obtained is then such that the values of the tokens are unsigned integers. A transition will not occur if this operation leads to create strictly negative tokens.

We have proved [Pet88] that the termination of the normed net implies the termination of the algebraic net. To validate this property, we observe the behaviour of the net transitions. If a transition cannot infinitely occur, we will say that it can be blocked. If all the transitions can be blocked, the net terminates. VAERA decides, in a lot of cases, of the blocking of transitions. It tells the user if the net terminates, and otherwise gives the list of blockable transitions.

For the moment, we only have semi-decision procedures. They cannot infirm termination. Hence, to improve the analysis of algebraic nets, we intend to add a tool for the construction of finite reachable markings graphs.

7/ Conclusion and future work

We presented, in this paper, PAPETRI. It offers, in a graphical environment, various analysis tools. Moreover, it deals with different classes of nets : ordinary nets, coloured nets, algebraic nets. The mixture of techniques from various theories allows to study all the aspects of a system and to fully validate it.

As concerns the examples we studied, the answering delays are reasonable w.r.t. the calculus complexity (a few minutes for the longer operations).

PAPETRI is an open system which may easily receive other tools and may be extended to other classes of nets. The next foreseen extensions deal with the construction of reachable markings graphs for algebraic nets and verification of home states.

Bibliography

[Cho88] Choppy C. : *ASSPEGIQUE user's manual*, rapport de recherche L.R.I. n°452, Orsay, 1988.

[CJ85] Choppy C., Johnen C. : *PETRIREVE : proving Petri net properties with rewriting systems*, LNCS vol.202, rewriting techniques and applications, Jouannaud J.P. Ed., Springer Verlag, 1985.

[FG83] Forgaard R., Guttag J.V. : REVE : *a term rewriting system generator with failure resistant Knuth Bendix*, Proc. of an NSF Workshop on the rewrite rule laboratory, 1983.

[Fin89] Finkel. A. : *A minimal coverability graph for Petri nets*, rapport de recherche L.R.I., Orsay, 1989.

[Jen81] Jensen K. : *Coloured Petri nets and the invariants method*, T.C.S., 14-3, pp. 317-336, 1981.

[KM69] Karp R.M., Miller R.E. : *Parrallel program schemata*, JCSS,4, pp. 147-195, 1969.

[HH80] Huet G., Hullot J.J. : *Proofs by induction in equational theories with constructors*, JCSS, 25, pp. 239-266, 1982.

[HJJJ86] Huber P., Jensen M., Jensen K., Jepsen O. : *Reachability trees for High-Level Petri nets*, T.C.S., 45, 1986.

[Mor89] Moreau J. M. : *Graphe de couverture minimal dans les réseaux de Petri*, rapport de stage de D.E.A., Université d'Orléans, 1989.

[Pet88] Petrucci L. : *Etude d'un environnement d'exécution de réseaux de Petri algébriques*, rapport de stage de D.E.A., Université de Paris-Sud, Orsay, 1988.

[Rei85] Reisig W. : *Petri Nets* , Springer Verlag, 1985.

[Tre88] Treves N. : *A comparative study of different techniques of semi-flows computation in higher-level nets*, Proc. IX European Workshop on Applications and Theory of Petri nets, 1988.

[Vau85] Vautherin J. : *Un modèle algébrique, basé sur les réseaux de Petri, pour l'étude des systèmes parallèles*, Thèse de doctorat d'ingénieur, Université Paris-Sud, Orsay, 1985.

[VM84] Vautherin J., Memmi G. : *Computation of flows for unary-Predicate/Transition nets*, LNCS 188, G. Rozenberg ed., Springer Verlag, pp. 307-327, 1984.