Elementary structural analysis for PEPA

Stephen Gilmore, Jane Hillston* and Laura Recalde†

December 18, 1997

Abstract

We consider the adaptation of important structural techniques for analysing behavioural properties of Petri nets to models expressed in process algebras such as PEPA [Hil96].

1 Introduction

Performance Evaluation Process Algebra (PEPA) is a high-level modelling language which is used for performance analysis of models of distributed computer and communication systems. In common with the approach adopted by related modelling languages which also use a process-algebra based approach [GHR93, BBG95] the analysis of system performance is made possible by the inclusion of timing information in a PEPA model. Performance analysis is only an appropriate tool to use in the investigation of a system if certain behavioural criteria are met; one example being freedom from deadlock. Similarly, model solution could give misleading results if some of the activities in the model are never performed. In this paper we concentrate upon such behavioural investigation of PEPA models. One benefit of a separation of the modelling process into behavioural analysis followed by performance analysis is that if the model is found to be unsuitable due to the presence of a deadlock or the absence of liveness then an expensive performance investigation can be avoided at the cost of a more efficient structural analysis. A second, more important benefit is that although any deadlock within a PEPA model would be detected in the performance analysis process the absence of liveness property would not and so the preliminary structural analysis equips the modeller with more confidence in the correctness of the model. This applies in the particular respect that the modeller is assured that there are no activities in the model which are redundant because they never become enabled.

In order to recreate structural analysis techniques from Petri nets [Sil93] for the PEPA language we make use of an intermediate representation which consists of matrices which record behavioural information and a vector which records a state. The identification of the vector is a preliminary step which establishes the dimension of the matrices which are to record the information about transitions from one state to another. The intermediate matrix representation facilitates the structural analysis of PEPA models.

*Both at Laboratory for Foundations of Computer Science, Department of Computer Science, The University of Edinburgh, Scotland. Electronic mail addresses: {stg, jeh}@dcs.ed.ac.uk

†Departamento de Informática e Ingeniería de Sistemas, Universidad de Zaragoza, Spain. Electronic mail address: lrecalde@prometeo.cps.unizar.es

2 Structure of this paper

We focus on a sub-language of PEPA. We describe the conversion of models in the language into a representation by a pair of matrices (termed the *pre- and post-matrices*) which are the basis for the subsequent analysis of the system. At this point we summarise the relevant techniques for structural analysis of Petri nets which we have used for the analysis of models in our stochastic process algebra. We then present a procedure which can be used to extend our sub-language to full PEPA by adding the hiding operator. Finally we point to future directions of this work.

3 The PEPA language

We are not making use here of the added system description information which is provided by the rates which are paired with each occurrence of an activity within a model and so in order to simplify the presentation of the method of structural analysis we remove the specifications of rates from models. This action is justified because in PEPA we have the special case of all activity rates being parameters of negative exponential distributions with infinite support. In this situation we know that rate information can be systematically restored afterwards [CABC93].

In order to facilitate the explanation of our present consideration of structural properties of process algebra models we use the PEPA language without the hiding operator. We will show in Section 7 that the hiding operator could be added without any significant complication. Thus the simplified stochastic process algebra which we are using here is simply a convenient explanatory device and does not compromise the usefulness of our work. The grammar of the language is presented in Figure 1.

$$
\begin{array}{llll}
S & ::= & & \text{sequential components} \\
 & \mid & A & \text{constant} \\
 & \mid & \alpha.S & \text{prefix} \\
 & \mid & S + S & \text{choice} \\
P & ::= & & \text{parallel components} \\
 & \mid & S & \text{degenerate case} \\
 & \mid & P \underset{L}{\bowtie} P & \text{co-operation on } L
\end{array}
$$

Figure 1: Grammar for the PEPA sub-language

Constants are bound by defining equations of the form $A \stackrel{def}{=} P$. The function to determine the activities of a component, written as $\mathcal{A}ct\ A$, calculates the complete alphabet of the component syntactically as the set of all activity names which are used in the description of the component.

A consequence of the grammar is that the collection of equations describing a system can be partitioned into two sets. The equations which specify the behaviour of the sequential components are known as *defining equations*; those which specify how those components interact are termed *model equations*. Note that, via a process of back substitution, we can always reduce the set of model equations to a single *system equation*. We impose the constraint that the constant introduced on the left hand side of the system equation cannot be used recursively. Without loss of generality, throughout the remainder of this paper we will assume that PEPA models are formed by a set of defining equations and a single system equation. Thus those combinators which are used to construct sequential components are *dynamic* combinators, whereas those which are used to construct parallel components are *static* components. The significance of this is that the number of parallel components contained within a model will be invariant over all states.

4 An overview of Petri nets

Petri nets are another formalism which is widely used in the modelling and analysis of systems that exhibit complex behaviour due to the interleaving of parallelism and synchronisation. We concentrate here on Place/Transitions (P/T) net systems and summarise only enough for our present needs. For a thorough introduction to the topic the reader should consult the following references [Mur89, Pet81].

A P/T net is a bipartite directed graph, represented as a quadruple $\mathcal{N}=\langle P, T, \mathbf{Pre}, \mathbf{Post}\rangle$, where P and T are the sets of the two kinds of nodes, *places* and *transitions*, and $\mathbf{Pre}$ and $\mathbf{Post}$ are the $|P| \times |T|$ sized, natural-valued *incidence matrices* which represent the arcs. $\mathbf{Post}[p,t] = w$ means that there is an arc from t to p with *weight* (or *multiplicity*) w, and $\mathbf{Pre}[p,t] = 0$ indicates that there is no arc from p to t. A *marking* $\mathbf{m}$ is a $|P|$ sized, natural-valued vector. In the graphical representation of a P/T net a place is represented by a circle, a transition by a box or bar, and the marking by a number of dots or *tokens* present in each place. A P/T *system* is a pair $\mathcal{S}=\langle \mathcal{N}, \mathbf{m_0}\rangle$, where $\mathbf{m_0}$ is the initial marking. A transition t is *enabled* in a marking $\mathbf{m}$ if and only if $\mathbf{m} \geq \mathbf{Pre}[P,t]$. Its occurrence or firing yields a new marking $\mathbf{m}' = \mathbf{m} + \mathbf{C}[P,t]$, where $\mathbf{C} = \mathbf{Post} - \mathbf{Pre}$ is called the *token flow matrix*.

A P/T system is *live* when every transition is live, i.e. it can ultimately occur from every reachable marking, and it is *deadlock-free* when every reachable marking enables some transition. A P/T system is *bounded* when every place is bounded, i.e. its marking value is less than some bound at every reachable marking. Boundedness precludes overflows and liveness ensures that no single transition in the system can become unattainable.

Structural techniques provide a valuable approach to the analysis of Petri net models. These techniques are based on the idea that useful information about the behaviour of the system can be gained from the structure of the net and the initial marking, without necessarily carrying out a reachability analysis. This has led to the development of some efficient algorithms for the analysis of P/T systems. Two techniques have been predominantly used: (i) graph theory and (ii) linear algebra or convex geometry. Starting from the ideas of liveness and boundedness, in this paper we are seeking to define analogous properties for PEPA models and to develop similar techniques for their investigation. Here "structure" is replaced by "syntax": we aim to analyse the behaviour of the model without generating a derivative graph, but merely by syntactic analysis of the definitional equations (analogous to the net structure) and the system equation (corresponding both to net structure and initial marking).

In [Rib95] it was proved that any PEPA model can be seen as a GSPN. Ribaudo's approach was to define a *labelled* GSPN semantics of PEPA models, in which a corresponding net construction is associated with each of the combinators of the language. One consequence of this work is that, in principle at least, all the structural theory developed for Petri nets is available for the analysis of PEPA models. However, based on Ribaudo's approach it is not immediately apparent how this theory can be exploited at the level of the PEPA model.

In the following sections we will describe how to construct pre- and post-incidence matrices for a PEPA model. This has the consequence that a P/T system can be defined from each PEPA model. However, unlike Ribaudo's work, finding the corresponding P/T system is not the objective of our work. Instead, based on the matrix approach, we will see that some important results from Petri net theory can be used in a straightforward manner to obtain information about the PEPA model. We must point out, however, that the approach of the two formalism is quite different, and this must be taken into account for a correct interpretation of the results, as we shall see.

5 Conversion to a matrix representation

We explain the conversion from the process algebra definitional equations to an equivalent representation as a pair of matrices which record the activities which are performed and the derivatives which result from this. The presentation of the matrices has a column for each activity instance and a row for each component derivative, whether named by an identifier or not. Note that these are not the derivatives of the complete model, which would be generated by application of the semantics to the system equation, but the derivatives of the sequential components generated by application of the semantics to the definitional equations. Thus the resulting matrices have a block structure, each block corresponding to one component in the system equation.

We note that a similar representation of PEPA models in terms of pre- and post-transition vectors for activities was previously presented by Sereno [Ser95] for the purpose of investigating product form equilibrium distributions.

In order to establish the starting state of the PEPA model, in addition to the pre- and post-incidence matrices we must also construct a state vector $\mathbf{s_0}$, analogous to the initial marking $\mathbf{m_0}$. Recall that the specification of a PEPA model consists of a set of defining equations and the system equation. The local states of our model are the sequential derivatives defined by the defining equations. Given this set of "places", each global state $\mathbf{s}$, is the number of instances of each sequential derivative apparent in the syntactic form representing that state. In particular, the initial state $\mathbf{s_0}$ is a vector in which the entry for each derivative is the number of instances of that syntactic form which are found on the right hand side of the system equation. As with the incidence matrices, we can think of this vector as having a block structure corresponding to the components of the model.

Just as the **Pre** and **Post** matrices of a P/T system can be combined to form the token flow matrix, $\mathbf{C} = \mathbf{Post} - \mathbf{Pre}$, the $\boldsymbol{Pre}$ and $\boldsymbol{Post}$ matrices of a PEPA model can be combined to form the *characteristic matrix*, $\boldsymbol{C} = \boldsymbol{Post} - \boldsymbol{Pre}$.

5.1 Prefix

Consider the following very simple component which only performs the activities α and β in alternation.

$$P \stackrel{\mathit{def}}{=} \alpha.\beta.P$$

This component has two states and two activities and the information in its defining equation may be recorded in the following pair of pre- and post-matrices. The pre-matrix is compiled by recording the information that the component P can perform an α activity, thereby placing an entry in the first row and the component $\beta.P$ can perform a β activity, thereby placing an entry in the second row. The post-matrix is compiled in a similar fashion by recording that the component P is reached *after* performing a β activity and that the component $\beta.P$ is reached *after* performing an α activity.

$$\textbf{\textit{Pre}} \quad = \quad \begin{array}{c c} & \begin{array}{cc} \alpha & \beta \end{array} \\ \begin{array}{c} P \\ \beta.P \end{array} & \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \end{array} \qquad \textbf{\textit{Post}} \quad = \quad \begin{array}{c c} & \begin{array}{cc} \alpha & \beta \end{array} \\ \begin{array}{c} P \\ \beta.P \end{array} & \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \end{array}$$

Observe that we can associate with each non-null element of the pre-matrix another non-null element of the post-matrix, i.e. we move from one local state to another. Therefore the number of non-null elements in any column will be the same in the pre-matrix and the post-matrix.

5.2 Choice

Here a component performs a sequence of activities in which each α activity is followed by either a β or a γ activity.

$$\begin{aligned} Q &\stackrel{def}{=} \alpha.R \\ R &\stackrel{def}{=} \beta.Q + \gamma.Q \end{aligned}$$

The component has two states, both of which are named, and three activities. Its representation in terms of pre- and post-matrices follows.

$$\textbf{\textit{Pre}} \quad = \quad \begin{array}{c c} & \begin{array}{ccc} \alpha & \beta & \gamma \end{array} \\ \begin{array}{c} Q \\ R \end{array} & \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{pmatrix} \end{array} \qquad \textbf{\textit{Post}} \quad = \quad \begin{array}{c c} & \begin{array}{ccc} \alpha & \beta & \gamma \end{array} \\ \begin{array}{c} Q \\ R \end{array} & \begin{pmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \end{pmatrix} \end{array}$$

The preservation property which we noted in the previous case does not hold for rows. We see this because the non-null elements of a row in the pre-matrix represent syntactically distinct occurrences of an activity which can be performed in that state in order to evolve to a derivative of that component. However the corresponding row in the post-matrix represents a local state which was reached after performing those activities which are associated with non-null entries. There is no reason for these two numbers to be the same. In our example here the state Q can only perform a single activity but could have been reached by one of two distinct activities. A complementary property can be observed of the state R.

5.3 Cooperation

5.3.1 A first example

We consider a pair of components P and Q which synchronise on their common activities α and β and P performs an individual activity γ before returning to the α activity.

$$\begin{aligned} P &\stackrel{def}{=} \alpha.\beta.\gamma.P \\ Q &\stackrel{def}{=} \alpha.\beta.Q \end{aligned}$$

The system which we are representing in the matrices is $P \underset{\{\alpha,\beta\}}{\bowtie} Q$. This has three states and three activities.

$$\textbf{\textit{Pre}} \quad = \quad \begin{array}{r} \\ P \\ \beta.\gamma.P \\ \gamma.P \\ Q \\ \beta.Q \end{array} \begin{array}{c} \begin{array}{ccc} \alpha & \beta & \gamma \end{array} \\ \left(\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ \hline 1 & 0 & 0 \\ 0 & 1 & 0 \end{array}\right) \end{array} \qquad \textbf{\textit{Post}} \quad = \quad \begin{array}{r} \\ P \\ \beta.\gamma.P \\ \gamma.P \\ Q \\ \beta.Q \end{array} \begin{array}{c} \begin{array}{ccc} \alpha & \beta & \gamma \end{array} \\ \left(\begin{array}{ccc} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ \hline 0 & 1 & 0 \\ 1 & 0 & 0 \end{array}\right) \end{array}$$

Until this point because we were working with a single sequential component the states which were associated with the rows of the matrices were coincident with the states of the system. Now that we have a cooperation between two sequential components we note that the rows present information about the *local states* of subcomponents of the model. The advantage which this decomposition affords is that the sequential subcomponent which is performing the activity is identified. This information would be lost if the state of the whole model—$P \underset{\{\alpha,\beta\}}{\bowtie} Q$, $\beta.\gamma.P \underset{\{\alpha,\beta\}}{\bowtie} Q$ or $\gamma.P \underset{\{\alpha,\beta\}}{\bowtie} Q$—was used as the designator of each row in each matrix.

The row vector representation of the system equation of this model is shown below.

$$\mathbf{s_0} = \begin{array}{ccccc} P & \beta.\gamma.P & \gamma.P & Q & \beta.Q \\ \end{array}$$
$$\left(\begin{array}{ccccc} 1 & 0 & 0 & 1 & 0 \end{array}\right)$$

The column labels of this vector are the row labels from the pre- and post-matrices. We will normally omit these labels when presenting vector representations of system states.

We can observe from this example that synchronisations between the components P and Q here can be clearly seen from the matrix representation where there is any column in the pre-matrix whose sum is greater than one. However, it should be noted that in general a column whose sum is greater than one merely represents a *potential* synchronisation between components. If the model which is being investigated contains activities which are never able to be performed then potential synchronisations might never actually occur.

5.3.2 A second example

In the previous example the matrix representation had more rows than the number of states in the process algebra model of the system. This is because the component Q is tightly synchronised to P in a slave relationship. We now consider an example where the numbers are ordered in the other way, with the number of states in the process algebra model greater than the number of rows in the matrix representation.

Consider the following model which has three components, P, Q and R where Q enforces fairness between P performing activities and R performing activities.

$$\begin{aligned} P &\stackrel{def}{=} \alpha.\beta.P \\ Q &\stackrel{def}{=} \alpha.\gamma.Q \\ R &\stackrel{def}{=} \gamma.\delta.R \end{aligned}$$

The system which we consider is $P \underset{\{\alpha\}}{\bowtie} Q \underset{\{\gamma\}}{\bowtie} R$. The activity which P and Q synchronise upon is α and the activity which Q and R synchronise upon is γ. The matrix representation for this model is shown below. The process algebra model has eight states and twelve transitions. The system equation gives $\mathbf{s_0} = (1\ 0\ 1\ 0\ 1\ 0)$.

$$\textbf{\textit{Pre}} \quad = \quad \begin{array}{r} \\ P \\ \beta.P \\ Q \\ \gamma.Q \\ R \\ \delta.R \end{array} \begin{array}{c} \begin{array}{cccc} \alpha & \beta & \gamma & \delta \end{array} \\ \left(\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right) \end{array} \qquad \textbf{\textit{Post}} \quad = \quad \begin{array}{r} \\ P \\ \beta.P \\ Q \\ \gamma.Q \\ R \\ \delta.R \end{array} \begin{array}{c} \begin{array}{cccc} \alpha & \beta & \gamma & \delta \end{array} \\ \left(\begin{array}{cccc} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{array}\right) \end{array}$$

5.3.3 A third example

Observe that the possible activities in a PEPA model do not always coincide with the columns in the pre- and post-matrices. Even more, since the matrices are derived in a purely syntactic way, there may appear to be synchronisations recorded in the matrices which in fact will never occur. For matrices derived from Petri net representations the presence of these phantom synchronisations indicates a flaw in the given net. However, this is not the case for process algebras.

$$\begin{array}{rcl} P & \stackrel{\text{def}}{=} & \alpha.\beta.\alpha.P \\ Q & \stackrel{\text{def}}{=} & \alpha.\gamma.\alpha.Q \end{array}$$

The system which we consider is $P \underset{\{\alpha\}}{\bowtie} Q$. Components P and Q synchronise upon α, execute β and γ on their own and synchronise again upon α. This model has five states and six transitions.

The matrix representation for this model is shown below. In determining the transcription of α in the matrices, since each component has two instances of α, four apparent synchronisation points are noted. Two of these are real—the leading α activities from P and Q and the trailing α activities from P and Q—and two of these are phantoms—a leading α with a trailing α, in both cases. From the system equation we have $\mathbf{s_0} = (1\ 0\ 0\ 1\ 0\ 0)$.

$$\textbf{\textit{Pre}} \quad = \quad \begin{array}{r} \\ P \\ \beta.\alpha.P \\ \alpha.P \\ Q \\ \beta.\alpha.Q \\ \alpha.Q \end{array} \begin{array}{c} \begin{array}{cccccc} \alpha & \alpha & \alpha & \alpha & \beta & \gamma \end{array} \\ \left(\begin{array}{cccccc} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ \hline 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \end{array}\right) \end{array} \qquad \textbf{\textit{Post}} \quad = \quad \begin{array}{r} \\ P \\ \beta.\alpha.P \\ \alpha.P \\ Q \\ \beta.\alpha.Q \\ \alpha.Q \end{array} \begin{array}{c} \begin{array}{cccccc} \alpha & \alpha & \alpha & \alpha & \beta & \gamma \end{array} \\ \left(\begin{array}{cccccc} 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ \hline 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right) \end{array}$$

Note that Sereno explicitly excludes the case of repeated instances of an activity within a sequential component [Ser95]. He also imposes that when sequential components undertake activities of the same name they must do so in cooperation. We do not include this restriction as the following example shows.

5.3.4 A fourth example

Until now, all of our examples have considered systems where the model components synchronised on all of their common activities. In PEPA it is possible to express pure parallel execution of activities as synchronisation over the empty set of activities. This permits the representation of an *implicit choice* where, for example, cooperations take place pairwise between two of three components. We show this in the present example along with the treatment of self-loops where performing an activity leaves a model component in the same state.

$$\begin{aligned} P &\stackrel{def}{=} \alpha.\beta.P \\ Q &\stackrel{def}{=} \alpha.\gamma.Q \\ R &\stackrel{def}{=} \alpha.R \end{aligned}$$

The system we consider is $(P \underset{\emptyset}{\bowtie} Q) \underset{\{\alpha\}}{\bowtie} R$. Pair-wise synchronisations take place between P and R and between Q and R. This model has four states and eight transitions. The system equation gives $\mathbf{s_0} = (1\ 0\ 1\ 0\ 1)$. The matrix representation for this model is shown below. It is evidently important to represent clearly the pair-wise synchronisations and not to allow the inference of a three-way synchronisation between P, Q and R. For this reason the matrix must devote two columns to the two kinds of synchronisations on the α activity.

$$\boldsymbol{Pre} = \begin{array}{r} \\ P \\ \beta.P \\ Q \\ \gamma.Q \\ R \end{array} \begin{array}{c} \begin{array}{cccc} \alpha & \beta & \alpha & \gamma \end{array} \\ \left(\begin{array}{cccc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \hline 1 & 0 & 1 & 0 \end{array} \right) \end{array} \qquad \boldsymbol{Post} = \begin{array}{r} \\ P \\ \beta.P \\ Q \\ \gamma.Q \\ R \end{array} \begin{array}{c} \begin{array}{cccc} \alpha & \beta & \alpha & \gamma \end{array} \\ \left(\begin{array}{cccc} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ \hline 1 & 0 & 1 & 0 \end{array} \right) \end{array}$$

5.4 A fifth example

For our final example we consider the problem of capturing component duplication in the system equation of a PEPA model. To illustrate one of the reasons why this can be problematic we consider a model where the duplicated components are initiated in different local states by the system equation.

$$\begin{aligned} P &\stackrel{def}{=} \alpha.P' \\ P' &\stackrel{def}{=} \beta.P \\ Q &\stackrel{def}{=} \alpha.\beta.Q \\ R &\stackrel{def}{=} \beta.\alpha.R \end{aligned}$$

The system which we consider is $(P \underset{\{\alpha,\beta\}}{\bowtie} Q) \underset{\emptyset}{\bowtie} (P' \underset{\{\alpha,\beta\}}{\bowtie} R)$. Here because the two copies of the component P are initiated in the orientation which suits their partner component in the cooperation expression the model has four states and eight transitions. In contrast, the system $(P' \underset{\{\alpha,\beta\}}{\bowtie} Q) \underset{\emptyset}{\bowtie} (P \underset{\{\alpha,\beta\}}{\bowtie} R)$ immediately deadlocks. We require these two PEPA models to have different representations and these differ only in their initial state vectors, as shown below.

$$\mathbf{s_0} \;=\; \begin{array}{cccccccccc} & P & P' & Q & \beta.Q & P & P' & R & \alpha.R & \\ (& 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 &) \end{array}$$

$$\mathbf{s_0}' \;=\; \begin{array}{cccccccccc} & P & P' & Q & \beta.Q & P & P' & R & \alpha.R & \\ (& 0 & 1 & 1 & 0 & 1 & 0 & 1 & 0 &) \end{array}$$

As we note from these vectors our method of processing duplicated components is to add rows to the matrices for each of the copies. In net terms this corresponds to duplicating the net structure. We omit the pre- and post-matrices.

6 Application of structural analysis techniques

In this section we show how structural analysis techniques from P/T nets can be applied to stochastic process algebra models expressed in the PEPA language. We begin by introducing relevant terminology which will enable us to classify structurally flawed PEPA models.

6.1 Notions of liveness

The concept of liveness is well understood for Petri nets. We introduce a variation on this (*group liveness*) which can be mirrored in process algebra (*PEPA liveness*) and introduce a stronger notion of liveness for process which implies PEPA liveness.

P/T liveness (PN): From each reachable state of a net there is, for each transition, a sequence of firings which leads to the transition becoming enabled.

Group liveness (PN): From each reachable state of a labelled net there is, for each label, a sequence of firings which will lead to a transition with that label being enabled.

PEPA liveness (PA): For each reachable state and every action type there exists another state which is reachable where an activity of that type can be performed.

Full liveness (PA): For each reachable state and each syntactic occurrence of an activity within a sequential component there exists another state which is reachable where this occurrence of the activity can be performed.

The most interesting relationship between these four notions is the relation of P/T liveness to full liveness. If we consider the P/T system corresponding to a fully live PEPA model, we can see that P/T liveness implies full liveness but that the converse is not true. Problems arise because of the possibility of phantom synchronisations; these are an artifact of the algebraic representation of a model and do not arise in Petri net models.

It should be clear that full liveness implies PEPA liveness and that the notions of group liveness and PEPA liveness are equivalent notions in the two formalisms. The existence of phantom synchronisations does not present a difficulty here because there will be another instance of the synchronisation on the same activity which can be performed from some state.

All of these notions of liveness guarantee freedom from deadlock.

6.2 Boundedness in process algebra models

Boundedness for P/T systems is defined in terms of a bound on the number of tokens which can reside in each place of the net. For a process algebra model syntactic forms of sequential components play a role analogous to places. Thus we can say that a model is bounded if there is a bound on the number of instances of each sequential derivative which will appear in any state of the model.

Given the direct mapping between syntactic states of the PEPA model and the states of the underlying Markov process, boundedness is clearly a necessary condition for finiteness of the stochastic model.

6.3 Significance of the structural analysis

We wish to have the structural analysis determine the strongest form of liveness, namely full liveness. We caution the reader that this does not guarantee the well-known *cyclic* condition for PEPA models but an example which makes clear the difference is not easy to construct and so we do not present such an example here. The curious reader may consult [Col89].

One of the principal tools used in the analysis of P/T systems is the *state equation*. Recall that when one transition t is fired from a state, $\mathbf{m}$, the new state that is reached, $\mathbf{m}'$, can be obtained as $\mathbf{m}' = \mathbf{m} + \mathbf{C}[P, t]$. Analogously, if a sequence σ is fired, the state that is reached can be computed as

$$\mathbf{m}' = \mathbf{m} + \mathbf{C} \cdot \boldsymbol{\sigma}$$

where $\boldsymbol{\sigma}[t]$ represents the number of times transition t occurs in the sequence. Unfortunately, there may be solutions of the equation that do not correspond to firable sequences; these are termed *spurious solutions.*

For any P/T net, *flows* (*semiflows*) are integer (natural) annullers of $\mathbf{C}$. Right and left annullers are called T- and P-(semi)flows, respectively. Flows are important because they induce certain invariant relations which are useful for reasoning about the behaviour of the net. For instance, if $\mathbf{y} \geq \mathbf{0}$ and $\mathbf{y} \cdot \mathbf{C} = \mathbf{0}$ then every reachable state $\mathbf{m}$ satisfies $\mathbf{y} \cdot \mathbf{m} = \mathbf{y} \cdot \mathbf{m_0}$. This implies the existence of a invariance relation which can be expressed in terms of the number of tokens distributed over the places of the net. Similarly, any repetitive transition sequence σ which returns a marking to itself satisfies that $\mathbf{C} \cdot \boldsymbol{\sigma} = 0$.

The existence of certain annullers, or similar vectors, define several behavioural properties over the corresponding net. These properties will hold for any P/T system with the given net structure, i.e. for any initial marking. For example:

- $\mathcal{N}$ is *consistent* (structurally repetitive) if and only if $\mathbf{x} > \mathbf{0}$ exists such that $\mathbf{C} \cdot \mathbf{x} = \mathbf{0}$.
- $\mathcal{N}$ is *conservative* (structurally bounded) if and only if $\mathbf{y} > \mathbf{0}$ exists such that $\mathbf{y} \cdot \mathbf{C} = \mathbf{0}$.

The characteristic matrix of a PEPA model is always conservative, because, as we previously remarked, any non-null element in the pre-matrix has a corresponding element in the post-matrix. In particular this means that the number of copies of processes does not change as the system evolves, which from a Petri net point of view means that the system is bounded.

In bounded P/T systems, with a finite number of states, liveness cannot be achieved unless there is a repetitive sequence covering all the transitions. Therefore, consistency is a necessary condition for liveness in bounded systems. This condition has been improved by the addition of an upper bound to the rank of the token flow matrix. The bound is based on the definition of an equivalence relation between transitions.

Definition 1 (Equal conflict relation) *We say that transitions t and t' are in* equal conflict relation, *if* $\mathbf{Pre}[P,t] = \mathbf{Pre}[P,t'] \neq \mathbf{0}$. *Each equivalence class is an equal conflict set denoted, for a given t,* EQS(t). SEQS *is the set of all the equal conflict sets of a given net.*

Theorem 1 (The Rank Theorem [TS96]) *If $\mathcal{S}$ is a live and bounded system, then it is consistent, conservative and* $\mathrm{rank}(\mathbf{C}) \leq |\mathrm{SEQS}| - 1$.

This theorem cannot be applied directly to PEPA models, because as we previously remarked, full liveness in PEPA models does not imply liveness of the associated P/T system.

Our reasoning in the proof of the rank theorem for PEPA requires a slight restriction on the form of PEPA models. We make use of a condition which we term *uniqueness of guards*, defined as follows:

Definition 2 (Uniqueness of guards) *In each sequential component the action types of activities which are guards on choices (i.e. the activities which are in competition) can never be performed again without first re-visiting the state where the choice was made.*

The Rank Theorem for PEPA models is based on the following Lemma for P/T systems. For the statement of the Lemma we need the following definition.

Definition 3 (Circuit arbiter) *Let $\mathcal{N}$ be a P/T net and let e be an equal conflict set such that $|e| > 1$. A net $\mathcal{C}_e = \langle P_e, e, \mathbf{Pre}_e, \mathbf{Post}_e \rangle$ is an* (ordinary) circuit arbiter *for the equal conflict set e if and only if $\mathcal{C}_e$ is a net in which all arcs are weighted 1, such that $P_e \cap P = \emptyset$ and its underlying graph is an elementary circuit.*

The effect of the circuit arbiter is to impose a fair ordering over the firings of the transitions of the equal conflict set.

Lemma 6.1 *Let $\mathcal{S} = \langle \mathcal{N}, \mathbf{m_0} \rangle$ be a P/T system that is bounded, group live and such that all the transitions that belong to non-trivial equal conflict sets are live. Let $e \in$ SEQS such that $|e| > 1$. Let $\mathcal{C}_e$ be a circuit arbiter for e, and let $\mathcal{N}'$ be the net $\mathcal{N}$ merged with the circuit arbiter $\mathcal{C}_e$ sharing the transitions in e. Then*

1. *There exists a marking $\mathbf{m_0}'$ with $\mathbf{m_0}'[P] = \mathbf{m_0}$ such that $\mathcal{S}' = \langle \mathcal{N}', \mathbf{m_0}' \rangle$ is bounded, group live and all the transitions in non-trivial equal conflict sets are live.*

2. $\mathrm{rank}(\mathbf{C}') = \mathrm{rank}(\mathbf{C}) + |e| - 1$, *where $\mathbf{C}'$ is the token flow matrix of $\mathcal{N}'$.*

Theorem 2 (The PEPA rank theorem) *Let $\mathcal{S}$ be a fully live PEPA model satisifying the uniqueness of guards restriction. Then its associated characteristic matrix, $\boldsymbol{C}$ satisfies that:*

- *There exists* $\mathbf{y} > \mathbf{0}$ *such that* $\mathbf{y} \cdot \boldsymbol{C} = \mathbf{0}$
- *There exists* $\mathbf{x} \geq \mathbf{0}$ *whose support covers all the action types in $\mathcal{S}$ and such that* $\boldsymbol{C} \cdot \mathbf{x} = \mathbf{0}$
- $\mathrm{rank}(\boldsymbol{C}) \leq |\mathrm{SEQS}| - 1$

Proof The existence of the positive left annuller is an immediate consequence of the fact that in PEPA sequential processes are not created, nor destroyed.

Since the number of states of the system is finite, for it to be live there must exist a repetitive sequence of activities. We cannot ensure this sequence will cover all the columns of the matrix, due to the possible existence of "false synchronisations", but it must contain every action type at least once. Observe that this also means that the corresponding P/T system is group live, although not necessarily live. The token flow matrix of this net $\mathbf{C}$ is equal to the characteristic matrix $\boldsymbol{C}$ of the PEPA model.

Furthermore, observe that the uniqueness of guards condition ensures that in the corresponding net all the transitions in non-trivial equal conflict sets are live.

Thus we can apply the Lemma. Let $\mathcal{N}'$ be the corresponding net $\mathcal{N}$ together with integrated circuit arbiters for *every* non-trivial equal conflict set. Applying point 2 of Lemma 6.1 repeatedly after each circuit arbiter is merged, which can be done thanks to point 1 of the Lemma, it follows that $\mathrm{rank}(\mathbf{C}') = \mathrm{rank}(\mathbf{C}) + \sum_{e \in \mathrm{SEQS}}(|e| - 1)$. Moreover, this system remains bounded and group live, hence there must exist a repetitive transition sequence and thus $\mathbf{C} \leq |\mathrm{SEQS}| - 1$. Rearranging the above inequalities we obtain a bound for the rank of the PEPA characteristic matrix $\boldsymbol{C}$:

$$\mathrm{rank}(\boldsymbol{C}) \leq y - \sum_{e \in \mathrm{SEQS}} (|e| - 1) - 1$$

where y is the number of columns in $\boldsymbol{C}$. Since $\sum_{e \in \mathrm{SEQS}} |e| = |T| = y$, this bound is $|\mathrm{SEQS}| - 1$, and the result follows. □

By analysing the defining equations of a PEPA model we can partition the activities of the model into equal conflict sets which are equivalent to the equal conflict sets of transitions in the corresponding P/T system. Thus, this result becomes a procedure of structural analysis which can be applied to any PEPA model to ascertain whether it is possible for it to be fully live.

6.3.1 Determining absence of liveness

In order to clarify the reader's understanding of the notions associated with equal conflict relations and the use of the Rank Theorem, we present a small worked example which builds the equal conflict relation for a small PEPA model with a deadlocking state. The model contains two components, P and R, which synchronise on activities γ and δ.

$$\begin{aligned} P &\stackrel{def}{=} \alpha.\gamma.P + \beta.\delta.P \\ R &\stackrel{def}{=} \gamma.\delta.R \end{aligned}$$

The system which we study is $P \underset{\{\gamma,\delta\}}{\bowtie} R$, which has a deadlock at $\delta.P \underset{\{\gamma,\delta\}}{\bowtie} R$. The matrix representation for this model is shown below.

$$\boldsymbol{Pre} = \begin{array}{r} \\ P \\ \gamma.P \\ \delta.P \\ R \\ \delta.R \end{array} \begin{array}{c} \begin{array}{cccc} \alpha & \beta & \gamma & \delta \end{array} \\ \left(\begin{array}{cccc} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right) \end{array} \qquad \boldsymbol{Post} = \begin{array}{r} \\ P \\ \gamma.P \\ \delta.P \\ R \\ \delta.R \end{array} \begin{array}{c} \begin{array}{cccc} \alpha & \beta & \gamma & \delta \end{array} \\ \left(\begin{array}{cccc} 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{array}\right) \end{array}$$

We observe from the pre-matrix that the activities α and β satisfy the equal-conflict condition for this model, and are thereby considered equal. We can now observe that we have the following set of activities in equal conflict.

$$\text{SEQS} = \{\ \{\alpha, \beta\}, \{\gamma\}, \{\delta\}\ \}$$

For this model, the rank of the matrix $\boldsymbol{C}$ is equal to the cardinality of the set of equal-conflict sets. Thus, solely from this structural analysis of the model, we can use the Rank Theorem to conclude that the model is either not live or it is not bounded. Since all PEPA models are bounded we can conclude that the model is not live. The failure of this property signals a logical error in the model itself which would direct the performance modeller not to proceed onto the performance analysis of the model but rather instead to attempt to diagnose the source of the error.

Recall that for a P/T net structural analysis is based solely on the net structure without the initial marking. To some extent the same is true for structural analysis of PEPA models. However, note that although the system equation is used to form the initial state, analogous to the initial marking, it also plays a crucial role in defining the structure of the model. For example, if the system equation in the model discussed above had been $P \underset{\{\gamma\}}{\bowtie} R$ the pre- and post-matrices would each have had two δ columns corresponding to the two distinct instances of the activity. Moreover, each instance would form a separate element within SEQS. However, the rank of the characteristic matrix remains 3, indicating that the modified model is free from deadlock, which it is.

When a deadlock is indicated one useful diagnostic method is to remove columns from $\boldsymbol{C}$ in order to isolate one of the activities which appears in an equal conflict set of cardinality at least two. The modeller then attempts to solve the consistency condition by seeking a right annuller for the reduced matrix $\boldsymbol{C}'$. This process corresponds to forcing a choice always to be resolved in a particular way, which has the consequence that if an annuller cannot be found for this reduced matrix then it may be concluded that there is a problem which stems from taking this branch of the choice. Repairing problems which are found by this method will eventually result in a model which is worthy of performance-based analysis, possibly with an intervening step to determine freedom from deadlock.

When analysing the possibility of deadlock, the existence of phantom synchronisations will not present a problem and the techniques used for P/T systems can be used without any modification. These techniques are again based on the state equation, although the existence of spurious solutions prevents it from being a characterisation of the absence of deadlock.

Theorem 3 ([TCS93]) *Let $\mathcal{S}$ be a P/T system. If there is no integer solution to*

$$\begin{array}{rcll} \mathbf{m} - \mathbf{C} \cdot \sigma & = & \mathbf{m_0} & \\ \mathbf{m}, \sigma & \geq & \mathbf{0} & \\ \displaystyle\bigvee_{\{p;\mathbf{Pre}[p,t]\neq 0\}} \mathbf{m}[p] & < & \mathbf{Pre}[p,t] & \forall t \in T \end{array}$$

then $\mathcal{S}$ is deadlock-free.

In general this means that the absence of solutions of several systems of equations has to be proved. However, there are cases where the number of systems of equations can be significantly reduced [TCS93]. In particular, when analysing a PEPA model we need to consider only a single system of equations. This follows because even when the PEPA model contains several copies of the same component each of these is represented separately in the pre- and post-matrices and one linear inequality is sufficient to represent the restrictions on a transition [TCS93].

Theorem 4 *Let $\mathcal{S}$ be a PEPA model with n sequential components which we name $P_{10}, \ldots, P_{n0}$ and let the derivatives within P_{i0} be named $P_{i0}, \ldots, P_{in_i}$. Furthermore let A be the set of activity instances represented in the pre- and post-matrices of $\mathcal{S}$. If there is no solution for* $\mathbf{s}$ *for any activity sequence σ (represented in vector form) with*

$$\begin{aligned} \mathbf{s} - \boldsymbol{C} \cdot \sigma &= \mathbf{s_0} \\ \mathbf{s}, \sigma &\geq \mathbf{0} \\ \sum_{\{P_{ij};\mathbf{Pre}[P_{ij},\alpha]\neq 0\}} \mathbf{s}[P_{ij}] &< \sum_{\{P_{ij};\mathbf{Pre}[P_{ij},\alpha]\neq 0\}} \mathbf{Pre}[P_{ij}, \alpha] \quad \forall \alpha \in A \end{aligned}$$

then $\mathcal{S}$ is deadlock-free.

7 Adding a hiding operator

Cooperation on activities in the PEPA language is fully general. Cooperation may involve any number of components and any number of these *cooperands* may play an active role. This generality is in marked contrast to the model of synchronisation in CCS [Mil89] in which exactly two agents are involved in every synchronisation. In CCS a synchronisation between two agents is hidden as an invisible τ action in order to make clear the impossibility of multi-way synchronisation. One purpose of hiding in a process algebra which has multi-way synchronisation, as PEPA does, is to prevent additional components from being able to synchronise on a particular set of activities (those activities which are hidden). Another use of hiding in a stochastic process algebra such as PEPA is to designate activities which are considered insignificant in terms of system performance because rewards cannot be associated with hidden actions. In practice, the effect of hiding is to rename an activity to the reserved name τ, which cannot be used in any synchronisation set. Thus, in order to add a hiding operator to PEPA, we will describe the implementation of the operator in terms of this renaming operation. The impossibility of synchronising on a τ activity is enforced by choosing for each replacement an activity $\tau_0, \tau_1, \tau_2, \ldots$ drawn from an unbounded set. A freshly numbered τ activity is chosen for each replacement so that the PEPA cooperation operator will allow these activities to be performed independently.

The renaming activity can be implemented by a substitution operation combined with the generation of new definitional equations for model components where necessary. Fresh names for components are produced by decorating an identifier with a prime. We think of the prime symbol as an operator which can be applied to a collection of equations in order to generate a related collection of definitions for components with primed identifiers.

Before proceeding to the implementation of the operation we present a simple example in order to reinforce the reader's intuition about this operator. Consider two copies of a component P which are prevented from synchronising on their common α activity by the intervening hiding operator. This model is transformed as shown below.

$$\begin{array}{lcl} P & \stackrel{\mathit{def}}{=} & \alpha.\beta.P \\ (P\backslash\alpha) \underset{\{\alpha\}}{\bowtie} P & & \end{array} \quad \rightsquigarrow \quad \left\{ \begin{array}{lcl} P & \stackrel{\mathit{def}}{=} & \alpha.\beta.P \\ P' & \stackrel{\mathit{def}}{=} & \tau_0.\beta.P' \\ P' \underset{\{\alpha\}}{\bowtie} P & & \end{array} \right.$$

The definition of the implementation of the hiding operator follows. The definition is stated in terms of the hiding of a single name and is expressed in terms of the use of a hiding operator on a PEPA sublanguage expression which, of course, does not use hiding. This has the consequence that it enforces the application of multiple uses of the hiding operation to begin with the removal of inner applications and proceed outwards. The extension from our description in terms of a single hidden name to hiding a set of names is routine. When representing an expression where α is hidden we first choose a fresh τ-identifier, say τ_i, and perform the substitution represented below.

$$\begin{array}{rcll} (\alpha.E)[\tau_i/\alpha] & \rightsquigarrow & \tau_i.E[\tau_i/\alpha] & \\ (\beta.E)[\tau_i/\alpha] & \rightsquigarrow & \beta.E[\tau_i/\alpha] & \\ (E+F)[\tau_i/\alpha] & \rightsquigarrow & E[\tau_i/\alpha] + F[\tau_i/\alpha] & \\ (E \underset{L}{\bowtie} F)[\tau_i/\alpha] & \rightsquigarrow & E[\tau_i/\alpha] \underset{L}{\bowtie} F[\tau_i/\alpha] & \\ A[\tau_i/\alpha] & \rightsquigarrow & A \text{ if } \alpha \notin \mathcal{A}ctA & \\ A[\tau_i/\alpha] & \rightsquigarrow & A' \text{ if } \alpha \in \mathcal{A}ctA & \quad \text{where } A' \stackrel{\mathit{def}}{=} E[\tau_i/\alpha] \text{ if } A \stackrel{\mathit{def}}{=} E \end{array}$$

8 Conclusions and further work

One part of our structural analysis of PEPA models which deserves further study is the treatment of repeated components which suffers from the limitation that it does not exploit any of the symmetry which is implicit in such a model. Improving this aspect of the analysis is a suitable direction for futher study.

In Petri nets the notion of *implicit places* is used to assist with the structural determination of deadlock freedom. We have identified no corresponding notion for process algebra-based modelling and the investigation of the usefulness of a related concept also remains as future work.

We have suggested a separation of the analysis of performance models of distributed computer systems into an initial structural analysis followed by a subsequent performance analysis. As yet, these two phases in the modelling and analysis process are not integrated in the sense that there is no means by which the modeller can take information which is obtained in the structural analysis into the performance analysis phase. The purpose of passing this information forward to the next phase would be for exploitation perhaps in order to make an efficiency saving in the computation of properties such as system throughput or component utilisation. Such a connection between these two phases of the analysis process could potentially be of great benefit in facilitating the modelling of the generation of more complex computer and communication systems which are being developed with the increase in communication capacity and availability and the recent vastly increased use of mobile computing.

Acknowledgements

The authors are grateful to Prof. Manuel Silva of the University of Zaragoza for his help.

References

[BBG95] M. Bernardo, N. Busi, and R. Gorrieri. A distributed semantics for EMPA based on stochastic contextual nets. In Gilmore and Hillston [GH95], pages 492–509.

[CABC93] G. Chiola, M. Ajmone Marsan, G. Balbo, and G. Conte. Generalised stochastic petri nets: A definition at the net level and its implications. *IEEE Transactions on Software Engineering*, 19(2):89–107, February 1993.

[Col89] J. M. Colom. *Análisis Estructural de Redes de Petri. Programación Lineal y Geometría Convexa.* PhD thesis, DIEI. Univ. Zaragoza, June 1989.

[GH95] S. Gilmore and J. Hillston, editors. *Proceedings of the Third International Workshop on Process Algebras and Performance Modelling.* Special Issue of *The Computer Journal*, 38(7), December 1995.

[GHR93] N. Götz, U. Herzog, and M. Rettelbach. Multiprocessor and Distributed System Design: The Integration of Functional Specification and Performance Analysis using Stochastic Process Algebras. In *Performance'93*, 1993.

[Hil96] J. Hillston. *A Compositional Approach to Performance Modelling.* Cambridge University Press, 1996.

[Mil89] R. Milner. *Communication and Concurrency.* Prentice-Hall, 1989.

[Mur89] T. Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580, 1989.

[Pet81] J. L. Peterson. *Petri Net Theory and the Modeling of Systems.* Prentice-Hall, 1981.

[Rib95] M. Ribaudo. *On the Relationship between Stochastic Petri Nets and Stochastic Process Algebras.* PhD thesis, Dipartimento di Informatica, Università di Torino, May 1995.

[Ser95] M. Sereno. Towards a product form solution for stochastic process algebras. In Gilmore and Hillston [GH95], pages 622–632.

[Sil93] M. Silva. In *Practice of Petri Nets in Manufacturing*, chapter Introducing Petri Nets, pages 1–62. Chapman and Hall, 1993.

[TCS93] E. Teruel, J. M. Colom, and M. Silva. Linear analysis of deadlock-freeness of Petri net models. In *Procs. of the Second European Control Conference*, volume 2, pages 513–518. North-Holland, 1993.

[TS96] E. Teruel and M. Silva. Structure theory of equal conflict systems. *Theoretical Computer Science*, 153(1-2):271–300, 1996.