On state space decomposition for the numerical analysis of stochastic Petri nets*

Carlos J. Pérez-Jiménez and Javier Campos
Dpto. de Informática e Ingeniería de Sistemas, Universidad de Zaragoza
María de Luna 3, 50015, Zaragoza, Spain
E-mail: cjperez,jcampos@posta.unizar.es

Abstract

Net-driven decomposition techniques are considered in this paper in order to reduce the state explosion problem for the computation of performance indices of stochastic Petri nets. Basically, the idea is to represent (or partially represent) in a decomposed manner the reachability graph of the model so it can be used for exact and/or approximated performance analysis. In that way, the complete storing of the graph is avoided and, for the case of approximate analysis, the solution of the isomorphous continuous time Markov chain is substituted by the solution of smaller components. The techniques are applied to a couple of non-trivial models.

1. Introduction

One of the major drawbacks for the use of *stochastic Petri nets* (SPN's) [9] in performance evaluation of real systems is probably the *state explosion problem*. This paper tries to contribute to the solution of that problem proposing some ideas and techniques for a *net-driven decomposition* of the *reachability graph* (RG) of the original system, leading to the generation of the continuous time Markov chains (CTMC's) associated to several smaller submodels whose solution can be combined for an approximated performance analysis. An iterative *response time approximation algorithm* is the technique selected here for the combination of the solutions of the (CTMC's of the) submodels in order to approximate the *steady-state* throughput of transitions (number of firings per time unit) in the original model (provided that such steady-state behaviour exists). However, one of the techniques that will be presented gives an exact decomposed representation of the RG of the original system, therefore it could be useful also for exact performance analysis.

Previous works in the same direction by the authors and other (co-)authors are [1, 11, 12, 10] in the framework of approximate throughput computation for particular net subclasses and [2] in the framework of *tensor algebra-based exact solution* of general SPN's. Other related work is [4].

We assume that the reader is familiar with concepts and notation of P/T nets [6] and SPN's [9]. The paper is organised as follows. In section 2, we review a decomposition technique for general stochastic Petri nets that was proposed and applied to the exact solution of the underlying CTMC in [2]. For our purpose, that decomposition technique has a main problem: *the derived subsystems may be non-ergodic* (thus, the solution of their underlying CTMC's could make no sense). In section 3, we solve the mentioned problem by presenting a very simple technique to build ergodic CTMC's from the subsystems. Even though that technique allows to compute a (meaningful) solution in all cases, in some situations the result may be not very accurate due to the inclusion of *spurious states* in the submodels that do not correspond to actual ones in the original system and to the 'inadequate aggregation' of different states of the original system into the same state in the subsystems. For solving more accurately those cases, we present in section 4 a structured view of the RG of the original model that allows to compute and store it in a decomposed manner. In section 4.1, the decomposed representation of the RG is used to eliminate all the spurious states, while in section 4.2 the aggregation of states not connected by internal transitions is avoided. The structured view of the RG of the original model presented in section 4 is applied here to derive approximated values of throughput but it could be also an alternative technique to the tensor algebra-based solution presented in [2] for exact analysis, since it is based on the construction of an exact decomposed representation of the RG of the model. In section 5, an iterative response time approximation algorithm (similar to that presented in [1]) is explained as well as an example of application to a couple of non-trivial models. Some concluding remarks are stressed in section 6.

*This work has been developed within the project TAP98-0679 of the Spanish CICYT.

2. Structural decomposition of PN systems and two-level abstract views

In this section, we review a decomposition technique for general PN systems that was proposed in [2] in the framework of *tensor algebra-based exact solution* of SPN's. See [2] for the details.

2.1. Structured view of PN's (net level)

An arbitrary PN system can always be observed as a set of *modules* (disjoint simpler PN systems) that asynchronously communicate by means of a set of *buffers* (places).

Definition 1 (SAM) [2] *A strongly connected PN system, $\mathcal{S} = \langle P_1 \cup \ldots \cup P_K \cup B, T_1 \cup \ldots \cup T_K, \mathbf{Pre}, \mathbf{Post}, \mathbf{m_0} \rangle$, is a* System of Asynchronously Communicating Modules*, or simply a SAM, if:*
1. $P_i \cap P_j = \emptyset$ for all $i, j \in \{1, \ldots, K\}$ and $i \neq j$;
2. $T_i \cap T_j = \emptyset$ for all $i, j \in \{1, \ldots, K\}$ and $i \neq j$;
3. $P_i \cap B = \emptyset$ for all $i \in \{1, \ldots, K\}$;
4. $T_i = P_i^{\bullet} \cup {}^{\bullet}P_i$ for all $i \in \{1, \ldots, K\}$.
The net systems $\langle \mathcal{N}_i, \mathbf{m_0}_i \rangle = \langle P_i, T_i, \mathbf{Pre}_i, \mathbf{Post}_i, \mathbf{m_0}_i \rangle$ with $i \in \{1, \ldots, K\}$ are called modules *of $\mathcal{S}$ (where $\mathbf{Pre}_i$,$\mathbf{Post}_i$, and $\mathbf{m_0}_i$ are the restrictions of $\mathbf{Pre}$, $\mathbf{Post}$, and $\mathbf{m_0}$ to P_i and T_i). Places in B are called* buffers*. Transitions belonging to the set $\mathrm{TI} = {}^{\bullet}B \cup B^{\bullet}$ are called* interface *transitions. Remaining ones $((T_1 \cup \ldots \cup T_K) \setminus \mathrm{TI})$ are called* internal *transitions.*

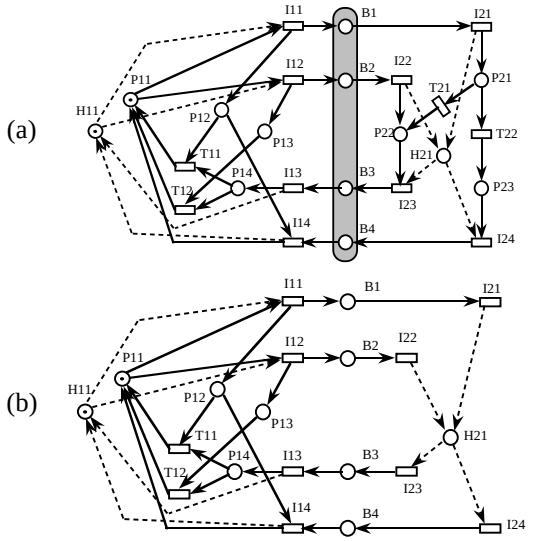

Figure 1. (a) A SAM and (b) its LS1.

A SAM with two modules (the subnets generated by nodes whose tag starts with P_i or T_i for $i = 1, 2$) and four buffers (B_1, B_2, B_3, B_4) is depicted in Fig. 1.a.

All the strongly connected PN systems belong to the SAM class with the only addition of a *structured view* of the model (either given by construction or decided after observation of the model). Many structured views are possible, ranging from the extreme consideration of each transition as a different module (all the places being buffers) to consider that the system is a single module (and there are no buffers).

With respect to timing interpretation, we assume that independent, exponentially distributed random variables are associated to the firing of transitions with single-server semantics as in classical SPN's [9]. The techniques presented in this paper can be easily extended to infinite-server semantics. We suppose that a unique steady-state behaviour exists to compute steady-state performance indices of the model. Even more, we restrict to *structurally live, structurally bounded* (therefore *consistent* and *conservative*) and *reversible* (therefore *ergodic*) PN systems.

2.2. Reduction rule and abstract views

In [2], the reduction rule that follows has been introduced for the *internal behaviour* of modules of a SAM. Each module is decomposed into several pieces and each piece is substituted by a set of new special places called *marking structurally implicit places* (MSIP's). Later, using that reduction, the original model can be decomposed into a collection of *low level systems* ($\mathcal{LS}_i$ with $i = 1, \ldots, K$) and a *basic skeleton* ($\mathcal{BS}$). In each $\mathcal{LS}_i$, only one module is kept while the internal behaviour of the others is reduced. In [2], the $\mathcal{LS}_i$ and the $\mathcal{BS}$ are used for a tensor algebra-based exact computation of the underlying CTMC. In this paper, we adapt the decomposition for a non-exact but more efficient approximate analysis.

Definition 2 [2] *Let $\mathcal{S} = \langle P, T, \mathbf{Pre}, \mathbf{Post}, \mathbf{m_0} \rangle$ be a SAM with $P = P_1 \cup \ldots \cup P_K \cup B$ and $T = T_1 \cup \ldots \cup T_K$. The equivalence relation R is defined on $P \setminus B$ by: $\langle p, p' \rangle \in R$ for $p, p' \in P_i$ iff there exists a non-directed path Π in $\mathcal{N}_i$ from p to p' such that $\Pi \cap \mathrm{TI} = \emptyset$ (i.e., containing only internal transitions). The $r(i)$ different equivalence classes defined in P_i with $i = 1, \ldots, K$ by the relation R are denoted as P_i^j with $j = 1, \ldots, r(i)$.*

The next step is to define and compute the sets of MSIP's needed for the reduction process.

Definition 3 [5] *Let $\mathcal{N}$ be a net and p be a place with incidence vector $l_p = \mathbf{C}[p, \cdot]$. The place p is a MSIP in $\mathcal{N}$ if there exists $\mathbf{y} \geq \mathbf{0}$ such that $\mathbf{y}[p] = 0$ and $l_p = \mathbf{y} \cdot \mathbf{C}$. The set of places in $\|\mathbf{y}\|$ are called* implying places *of p (where $\|\mathbf{y}\|$, called support of $\mathbf{y}$, is the set of non-zero components of $\mathbf{y}$).*

An algorithm for the computation of a set H_i^j of MSIP's for each equivalence class P_i^j defined in a module by means of relation R was proposed in [2]. The basic idea is to consider all the MSIP's $p_{\mathbf{y}}$ derived from the *minimal*

P-semiflows $\mathbf{y}$ of the subnet induced by P_i^j (i.e., $p_{\mathbf{y}}$ is the place with incidence vector $l_{p_{\mathbf{y}}} = \mathbf{y} \cdot \mathbf{C}$, where $\mathbf{y}$ is such that $\mathbf{y} \cdot \mathbf{C}[P_i^j, T_i^j] = 0, \mathbf{y} \geq 0$, $\mathbf{y}$ has minimal support).

A place is *implicit*, under interleaving semantics, if it can be deleted without changing the firing sequences. Each MSIP of H_i^j added to $\mathcal{N}$ needs an initial marking for making it implicit. In [5], an efficient method for computing such marking is presented.

The next step for the definition of the $\mathcal{LS}_i$ and the $\mathcal{BS}$ is to define an *extended system* ($\mathcal{ES}$).

Definition 4 [2] *Let $\mathcal{S} = \langle P, T, \mathbf{Pre}, \mathbf{Post}, \mathbf{m_0} \rangle$ be a SAM with $P = P_1 \cup \ldots \cup P_K \cup B$ and $T = T_1 \cup \ldots \cup T_K$. The extended system $\mathcal{ES}$ is obtained from $\mathcal{S}$ by adding all the places in H_i^j, $j = 1, \ldots, r(i), i = 1, \ldots, K$ with their incidence vectors and the initial marking necessary for making them implicit.*

Consider, for instance, the SAM given in Fig. 1.a. The original net system $\mathcal{S}$ is the net without the places H_{11} and H_{21}. These places are the MSIP's computed to summarise the internal behaviour of the two modules. Place H_{11} summarises the module 1 (the subnet generated by nodes whose tags begin with P_1 or T_1), and place H_{21} summarises the module 2. The $\mathcal{ES}$ is the net system of Fig. 1.a (adding to $\mathcal{S}$ the places H_{11} and H_{21}).

From the $\mathcal{ES}$, we can build the $\mathcal{LS}_i$ and $\mathcal{BS}$.

Definition 5 [2] *Let $\mathcal{S} = \langle P_1 \cup \ldots \cup P_K \cup B, T_1 \cup \ldots \cup T_K, \mathbf{Pre}, \mathbf{Post}, \mathbf{m_0} \rangle$ be a SAM and $\mathcal{ES}$ its extended system.*
i) The low level system $\mathcal{LS}_i$ for $i = 1, \ldots, K$ of $\mathcal{S}$ is the net system obtained from $\mathcal{ES}$ by deleting all the nodes in $\bigcup_{j \neq i}(P_j \cup (T_j \setminus \mathrm{TI}))$ and their adjacent arcs.
ii) The basic skeleton $\mathcal{BS}$ of $\mathcal{S}$ is the net system obtained from $\mathcal{ES}$ by deleting all the nodes in $\bigcup_{j=1}^{K}(P_j \cup (T_j \setminus \mathrm{TI}))$ and their adjacent arcs.

In each $\mathcal{LS}_i$ all the modules $\mathcal{N}_j$ with $j \neq i$, are reduced to their interface transitions and to the implicit places that were added in the $\mathcal{ES}$, while $\mathcal{N}_i$ is fully preserved. Systems $\mathcal{LS}_i$ represent different low level views of the original model. In the $\mathcal{BS}$ all the modules are reduced, and it constitutes a high level view of the system.

In Fig. 1.b. the $\mathcal{LS}_1$ of Fig. 1.a is depicted. The $\mathcal{BS}$ is obtained by deleting from Fig. 1.b the nodes whose tags begin with P_1 and T_1.

By construction, since the original net is conservative then the $\mathcal{LS}_i$ and the $\mathcal{BS}$ are also conservative, so the reachability sets of all these subsystems are finite. The main result about the behaviour of the constructed subsystems is the following.

Property 6 [2] *Let $\mathcal{S}$ be a SAM, $\mathcal{LS}_i$ its low level systems for i=1,..., K, $\mathcal{BS}$ its basic skeleton, and $\mathrm{L}(\mathcal{S})$ the language of firing sequences of $\mathcal{S}$. Then:*
i) $\mathrm{L}(\mathcal{S})|_{T_i \cup \mathrm{TI}} \subseteq \mathrm{L}(\mathcal{LS}_i)$ *for* $i = 1, \ldots, K$.
ii) $\mathrm{L}(\mathcal{S})|_{\mathrm{TI}} \subseteq \mathrm{L}(\mathcal{BS})$.

The above property states that the reduction technique recalled here does not remove but possibly adds new paths between interface transitions.

3. Guaranteeing subsystems ergodicity

From the result stated in Property 6, it follows that the RG's of $\mathcal{LS}_i$ and $\mathcal{BS}$ include at least the projections (on the corresponding preserved nodes) of the reachable markings of the original system. They also reproduce the projections on the preserved transitions of the firing sequences of the original system. But since the inclusion in the statement of Property 6 is not strict, the RG's of the subsystems may possibly include new (let us say) *spurious* markings and firing sequences that do not correspond to actual markings and firing sequences of the original system. In some cases, this non desired behaviour can lead to non ergodic systems. In this section we present a technique to avoid such undesired behaviour, guaranteeing ergodicity of the subsystems. Ergodicity of subsystems is needed to apply this decomposition technique to approximate analysis since the underlying CTMC's of subsystems must be solved.

Consider, for example, the system given in Fig. 1.a. Cutting the system through the places B_1 to B_4 and applying the reduction technique of the previous section to the right hand side subnet, the $\mathcal{LS}_1$ of Fig. 1.b is obtained. In the original system after the firing of interface transition I_{22} only transition I_{23} can be fired, but in $\mathcal{LS}_1$ it is also possible to fire transition I_{24} after the firing of transition I_{22}. This new possible firing makes $\mathcal{LS}_1$ non live (the sequence $\sigma = I_{12}I_{22}I_{24}$ is firable in $\mathcal{LS}_1$ and the marking produced by the firing of σ is B_4P_{13} that is a total deadlock).

In other words, the underlying CTMC's of the subsystems may be non ergodic.

There is a direct way to adjust these CTMC's to assure ergodicity. In general, the RG's of the subsystems may have several strongly connected components. To obtain an ergodic CTMC, only the strongly connected component of the initial marking in each subsystem must be selected. It will be proved that these strongly connected components include, at least, all the projected states and firing sequences of the original net system.

Theorem 7 *Let $\mathcal{S}$ be a SAM, $\mathcal{ES}$, $\mathcal{LS}_i$ with $i = 1, \ldots, K$ and $\mathcal{BS}$ its extended, low level and the basic skeleton systems, respectively. Let $\mathrm{RG}^*(\mathcal{LS}_i)$ and $\mathrm{RG}^*(\mathcal{BS})$ be the strongly connected components of $\mathrm{RG}(\mathcal{LS}_i)$ and $\mathrm{RG}(\mathcal{BS})$, resp., that contain the initial marking. Let $\mathrm{RS}^*(\mathcal{LS}_i)$ and $\mathrm{RS}^*(\mathcal{BS})$ be the states of $\mathrm{RG}(\mathcal{LS}_i)$ and $\mathrm{RG}(\mathcal{BS})$, resp. Let $\mathrm{L}^*(\mathcal{LS}_i)$ and $\mathrm{L}^*(\mathcal{BS})$ be the language of firing sequences of $\mathrm{RG}^*(\mathcal{LS}_i)$ and $\mathrm{RG}^*(\mathcal{BS})$, respectively. Then:*
i) $\mathrm{L}(\mathcal{S})|_{T_i \cup \mathrm{TI}} \subseteq \mathrm{L}^*(\mathcal{LS}_i)$ *for* $i = 1, \ldots, K$ *and* $\mathrm{L}(\mathcal{S})|_{\mathrm{TI}} \subseteq \mathrm{L}^*(\mathcal{BS})$.
ii) $\mathrm{RS}(\mathcal{ES})|_{P_i \cup H \cup B} \subseteq \mathrm{RS}^*(\mathcal{LS}_i)$ *for* $i = 1, \ldots, K$ *and* $\mathrm{RS}(\mathcal{ES})|_{B \cup H} \subseteq \mathrm{RS}^*(\mathcal{BS})$.

Proof: Consider the extended system $\mathcal{ES}$ of $\mathcal{S}$. By definition, all the places in $H = \bigcup_{i=1}^{K} H_i$ are implicit in $\mathcal{ES}$. Therefore $\mathrm{L}(\mathcal{S}) = \mathrm{L}(\mathcal{ES})$. Since $\mathcal{S}$ is assumed reversible, $\mathbf{m_0}$ is a *home state* (if this is not the case, but $\mathcal{S}$ has home state, any home state can be considered as the new initial marking and the result is still true). Given a marking $\mathbf{m} \in \mathrm{RS}(\mathcal{ES})$, by reversibility of $\mathcal{ES}$, there exists $\sigma, \tau \in \mathrm{L}(\mathcal{ES})$ such that $\mathbf{m_0} \xrightarrow{\sigma} \mathbf{m} \xrightarrow{\tau} \mathbf{m_0}$. Let $\mathbf{m_0}_i = \mathbf{m_0}|_{P_i \cup H \cup B}$ be the initial marking of $\mathcal{LS}_i$ and $\mathbf{m}' = \mathbf{m}|_{P_i \cup H \cup B}$ be the projection of $\mathbf{m}$ over the places of $\mathcal{LS}_i$. Let $\sigma' = \sigma|_{T_i \cup \mathrm{TI}}$ and $\tau' = \tau|_{T_i \cup \mathrm{TI}}$. It must be proved that $\mathbf{m}' \in \mathrm{RS}^*(\mathcal{LS}_i)$ and $\sigma' \in \mathrm{L}^*(\mathcal{LS}_i)$. By definition of $\mathcal{ES}$, it is clear that the marking of the places in $H \cup B$ is only changed by the firing of transitions in TI, and the marking of the places in P_i is only changed by the firing of transitions in T_i. Then, in $\mathcal{LS}_i$, $\mathbf{m_0}_i \xrightarrow{\sigma'} \mathbf{m}' \xrightarrow{\tau'} \mathbf{m_0}_i$, so $\mathbf{m}' \in \mathrm{RS}^*(\mathcal{LS}_i)$ and $\sigma' \in \mathrm{L}^*(\mathcal{LS}_i)$. ♢

The strongly connected components of a directed graph can be efficiently computed with a time complexity of $O(max(n, |E|))$, where n is the number of nodes of the graph and $|E|$ the number of edges [7].

The above theorem gives a general technique applicable to any structurally live, structurally bounded and reversible SAM to obtain, from the subsystems, ergodic CTMC's available for subsequent computations.

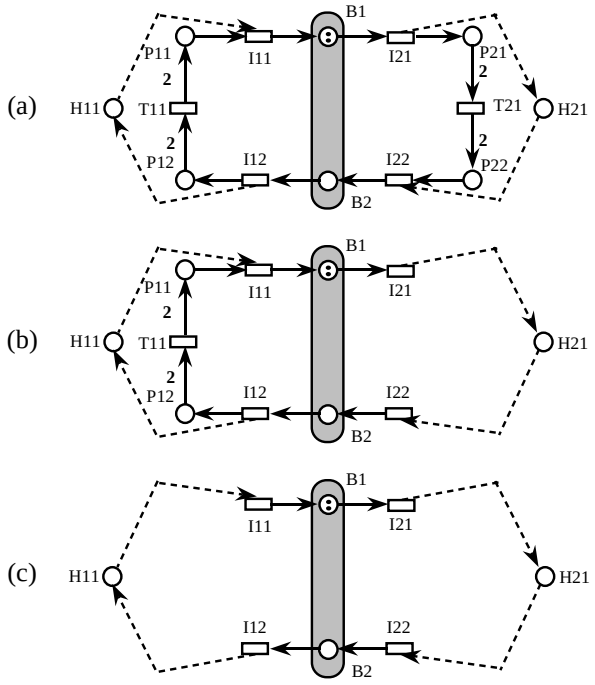

Figure 2. (a) A SAM, (b) its LS1 and (c) its BS.

In the case of Fig. 1, $\mathrm{RG}(\mathcal{LS}_1)$ has two strongly connected components. One of them with all the states but B_4P_{13} and the other one with only the state B_4P_{13} (the deadlock marking). The strongly connected component of the initial marking is the first one and then, the deadlock marking B_4P_{13} is removed in the process.

It must be pointed out that $\mathrm{RG}^*(\mathcal{LS}_i)$ and $\mathrm{RG}^*(\mathcal{BS})$ may still include spurious markings (and/or spurious firing sequences) that do not correspond to the projection of any marking (firing sequence) of the original system over the preserved nodes.

For example, in Fig. 2.a a weighted marked graph is depicted. Cutting the system through the places B_1 and B_2 and applying the reduction technique of the previous section, the $\mathcal{LS}_1$ of Fig. 2.b is obtained. Now, the underlying CTMC of $\mathcal{LS}_1$ is ergodic, so by computing in it the strongly connected component of the initial marking, the entire CTMC is obtained. But in this CTMC, there are spurious markings and firing sequences that were not possible in the original system. For example, in the original system, $\mathbf{m}[B_1] \cdot \mathbf{m}[B_2] = 0$, for any reachable marking, but in $\mathcal{LS}_1$, the marking B_1B_2 is reachable. This marking is not a projection of any reachable marking of the original system over the places of $\mathcal{LS}_1$.

4. Structured view of PN's (RG level)

In many cases the technique developed in previous section is enough for getting good approximations with iterative algorithms like that presented in section 5, but in some cases these approximations may be very poor. In our opinion, the explanation comes from the following two problems: (**P1**) The introduction of spurious markings (reachable markings in the subsystems that do not correspond to projections of reachable markings of the original system). In the example of Fig. 2 the state B_1B_2 is not reachable in the original system, but it becomes reachable in the subsystems. And the second problem (**P2**) is the 'inadequate' aggregation of different states of the original system (aggregation of states not connected by internal transitions) into the same state in the subsystems. In the example of Fig. 2 the states $P_{11}P_{21}$ and $P_{12}P_{22}$ are aggregated in the same state $H_{11}H_{21}$ in the $\mathcal{BS}$. This aggregation in the $\mathcal{BS}$ introduces spurious firing sequences in the submodel. Now, from state $H_{11}H_{21}$, it is possible to fire transitions I_{11} and I_{22} (this situation was not possible in the original system). Making computations with this aggregated CTMC leads to very bad results. The same problems can appear in more complex nets.

In this section we develop the basis for the solution of these problems. First, a structured view of the RG of a SPN will be explained. With this view it is possible to compute and store in a decomposed manner the RG of the original model. Thus, this view can be applied not only for approximate analysis but also for other analysis techniques. After that, this representation of the RG of the original system is used for the generation of the subsystems. Several possible subsystems can be generated. In subsection 4.1 the problem (**P1**) is completely solved (any reachable state of any subsystem is the projection of a state of the original system). In subsection 4.2 also the second problem (**P2**) is solved.

Definition 8 *Let $\mathcal{S}$ be a SAM and RG($\mathcal{S}$) its RG. Let R_i with $i = 1, \ldots, K$ be the following equivalence relations defined on the set of vertices of RG($\mathcal{S}$): $\forall s_1, s_2$ vertices of RG($\mathcal{S}$), $\langle s_1, s_2 \rangle \in R_i$ iff there exists a non-directed path Π in RG($\mathcal{S}$) from s_1 to s_2 containing only transitions in $T_j \setminus \mathrm{TI}$ with $j \neq i$. Let R_0 be the following equivalence relation defined on the set of vertices of RG($\mathcal{S}$): $\forall s_1, s_2$ vertices of RG($\mathcal{S}$), $\langle s_1, s_2 \rangle \in R_0$ iff there exists a non-directed path Π in RG($\mathcal{S}$) from s_1 to s_2 containing only transitions in $T \setminus \mathrm{TI}$.*

The following property trivially follows from the definition.

Property 9 *Let $\mathcal{S}$ be a SAM and R_i with $i = 0, \ldots, K$ the equivalence relations defined above. Then for each $i = 1, \ldots, K$ we have $R_i \subseteq R_0$. Let D_i with $i = 1, \ldots, K$ be the set of states of RG($\mathcal{S}$) belonging to an arbitrary equivalence class induced by R_i. Then, there exists an equivalence class D_0 induced by R_0 such that $D_i \subseteq D_0$.*

Definition 10 *Let $G_1 = (V_1, E_1, L_1)$, $G_2 = (V_2, E_2, L_2)$ be two directed labelled graphs. The product G of G_1 and G_2 (denoted by $G_1 \times G_2$) is another directed labelled graph $G = (V, E, L)$ such that: $V = V_1 \times V_2$ (Cartesian product), $L = L_1 \cup L_2$ and $E((s_{11}, s_{21}), (s_{12}, s_{22})) = t$ iff $(s_{11} = s_{12}$ and $E_2(s_{21}, s_{22}) = t)$ or $(s_{21} = s_{22}$ and $E_1(s_{11}, s_{12}) = t)$.*

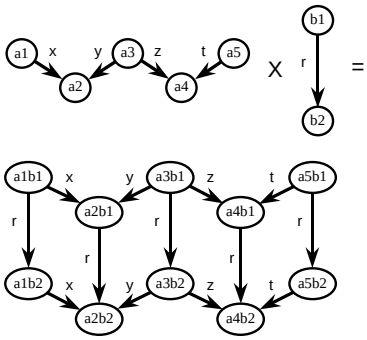

Figure 3. Product of directed labelled graphs

Fig. 3 shows the product of two directed labelled graphs.

Definition 11 *Let $\mathcal{S}$ be a SAM and RG($\mathcal{S}$) its RG. Let D be the directed labelled subgraph of RG($\mathcal{S}$) generated by the states with a common given fixed marking in the buffers (the subgraph composed by the states and the labelled edges joining them). A breadth-first search in RG($\mathcal{S}$) induces a breadth-first search in D. Consider the breadth-first spanning forest computed in D. The root H of a tree of the spanning forest is a* head *of D iff there is not any cross edge*[1] *arriving at H.*

[1]See [7] for the definition of cross edge in a breadth-first search.

Property 12 *Let $\mathcal{S}$ be a SAM and RG($\mathcal{S}$) its RG. Let D be the directed labelled subgraph of RG($\mathcal{S}$) generated by the states with a common given fixed marking in the buffers. Let H be a head of D, D_H be the directed labelled subgraph generated by the set of successors of H in D and D_H^i with $i = 1, \ldots, K$ the aggregation of D_H induced by R_i (the quotient set D_H / R_i). Then $D_H = D_H^1 \times \cdots \times D_H^K$.*

Proof: Let H^i with $i = 1, \ldots, K$ be the equivalence class of H in D_H / R_i. Let $J \in D_H$. Then there exists σ such that $H \xrightarrow{\sigma} J$ and σ composed only by internal transitions of $\mathcal{S}$. Let σ_i with $i = 1, \ldots, K$ be the projection of σ on the internal transitions of $\mathcal{N}_i$ and J^i be such that $H^i \xrightarrow{\sigma_i} J^i$ (σ_i is firable from H^i in D_H^i by concurrency among the internal transitions of $\mathcal{N}_i$ and $\mathcal{N}_j$ with $i \neq j$). Then $J = (J^1, \ldots, J^K) \Rightarrow J \in D_H^1 \times \cdots \times D_H^K$.
Let $(J^1, \ldots, J^K) \in D_H^1 \times \cdots \times D_H^K$. Then there exists σ_i with $i = 1, \ldots, K$ such that $H^i \xrightarrow{\sigma_i} J^i$. Let $\sigma = \sigma_1 \cdots \sigma_K$. By concurrency among internal transitions in $\mathcal{N}_i$ and $\mathcal{N}_j$ with $i \neq j$, σ is firable from H in D_H. Let J be such that $H \xrightarrow{\sigma} J$. Then $J = (J^1, \ldots, J^K) \Rightarrow (J^1, \ldots, J^K) \in D_H$. ♢

The above property exploits the concurrency among internal transitions in a SAM to compute the successors of a head locally in each module and storing only the aggregated classes in each subnet so reducing the time and memory requirements for the computation of these states.

Property 13 *Let $\mathcal{S}$ be a SAM and RG($\mathcal{S}$) its RG. Let D be the directed labelled subgraph of RG($\mathcal{S}$) generated by the set of states with a common given fixed marking in the buffers. Let $H_1, \ldots, H_n$ be the set of heads of D, and D_{H_i} with $i = 1, \ldots, n$ the directed labelled subgraph generated by the successors of H_i in D. Then $D = \bigcup_{i=1}^{n} D_{H_i}^1 \times \cdots \times D_{H_i}^K$, where $D_{H_i}^j$ with $i = 1, \ldots, n$ and $j = 1, \ldots, K$ is the aggregated subgraph of D_{H_i} by means of R_j.*

This property immediately follows from the previous one. Then the minimum information needed to generate the states of an equivalence class of R_0 is the set of heads of the class with their corresponding heads in each aggregated class. In different classes of the original RG the same head can appear in an aggregated class, so locally each aggregated class can be computed and stored only once, exploiting the concurrency of internal transitions at the maximum degree.

Now we need to know what happens with the interface transitions.

Property 14 *Let $\mathcal{S}$ be a SAM and RG($\mathcal{S}$) its RG. Let D be the directed labelled subgraph of RG($\mathcal{S}$) generated by the states with a common given fixed marking in the buffers. Let H be a head of D, D_H the directed labelled subgraph generated by the set of successors of H in D and D_H^i with $i = 1, \ldots, K$ the aggregation of D_H by means of relation R_i. Let $S \in D_H^i$. An interface transition $t \in T_j \cap \mathrm{TI}$ with $i \neq j$ is enabled in S iff there exists $S' \in D_H^j$ such that t is enabled in S'.*

Proof: $\Rightarrow$) Let S' be the state of D_H^j in which t is enabled and H^i the class of H in D_H^i (the head of D_H^i). By property 12 the state $M = (\ldots, H^i, \ldots, S', \ldots) \in D$. t is enabled in M so, by concurrency among internal transitions of $\mathcal{N}_i$ and interface transitions of $\mathcal{N}_j$ with $i \neq j$, t is enabled in all the states of D_H^i, thus t is enabled in S.
$\Leftarrow$) If t is enabled in S, since S is an equivalence class in D, there exists a state $M = (\ldots, S, \ldots, S', \ldots) \in D$ such that t is enabled in M (S and S' are the i and j-component of M). Aggregating by means of R_j, $S' \in D_H^j$ and t is enabled in S'. $\diamondsuit$

Last property allows to compute locally in each module i the interface transitions of $\mathcal{N}_i$ that can be fired in a given local marking D_H^i and exporting them to all the local markings of the other modules D_H^j with $j \neq i$. Again, in different classes of RG($\mathcal{S}$) by means of R_0 the same head can appear in an aggregated class, so taking into account the marking of buffers and the internal state in $\mathcal{N}_i$ (in other words the class D_H^i) is enough to compute the interface transitions enabled in D_H. Moreover, a spurious interface transition is never computed because the interface transitions are computed in the corresponding module with all the information about its input places.

Now we can present the algorithm to compute the decomposed description of RG($\mathcal{S}$).

Algorithm 4.1

input: A SAM $\mathcal{S}$
$B_0 := \mathbf{m_0}|_B$; initial marking on buffers
for $i := 1$ **to** K **do**
  $H_i := \mathbf{m_0}|_{P_i}$; breadth-first search from H_i in $\mathcal{N}_i$
$H := (H_1, \ldots, H_K)$
insert head (B_0, H) in the queue
while queue of heads non empty **do**
  $(B, H) :=$ first head of the queue; $L := \emptyset$
  for $j := 1$ **to** K
    $L := L \cup \{I_j \in T_j \cap \mathrm{TI} \mid I_j$ enabled in $H_j\}$
  for each $I_j \in L$ **do**
    for each successor h_j of H_j in $\mathcal{N}_j$ s. t. I_j is enabled **do**
      $(B, h_j) \xrightarrow{I_j} (B', H_j')$; $H' := (H_1, \ldots, H_j', \ldots, H_K)$
      if H_j' is new **then** breadth-first search from H_j' in $\mathcal{N}_j$
      if B' is new **then** insert head (B', H')
      else isnew := TRUE
        for each H'' s. t. (B', H'') is head **do**
          if $H'' \xrightarrow{\sigma} H'$, $\sigma \subseteq T \setminus \mathrm{TI}$ **then** isnew := FALSE
          else if $H' \xrightarrow{\sigma} H''$ with $\sigma \subseteq T \setminus \mathrm{TI}$ **then**
              replace H'' by H'; isnew = FALSE
          else if H' and H'' have a common successor **then**
              make H' and H'' related in B'
        if isnew = TRUE **then** insert head (B', H')
output: Set of heads
  $\mathrm{RG}(\mathcal{S})|_{\mathcal{N}_i}$ for $i = 1, \ldots, K$

In the above algorithm given the initial marking of a SAM, all the firing sequences composed only by internal transitions are computed locally in each module by means of a breadth-first search in each module. The local states computed in each module are stored locally (not all the possible combinations among them). Then a queue of unexplored heads is maintained in the while-loop. A head in the output of the algorithm is a state of $\mathcal{S}$ that is reachable only after the firing of an interface transition. During the algorithm the heads are replaced by other ones at the moment in which a predecessor by means of internal transitions is computed. In each iteration of the while-loop, the first head (B, H) of the queue is studied. B is the marking of the buffers and $H = (H_1, \ldots, H_K)$ the local markings in each module. Given a head H, locally in each module j, the set of interface transitions of T_j that can be fired after any firing sequence of internal transitions are computed. The union of these sets is the set L of interface transitions enabled in the equivalence class of (B, H) by means of R_0. After that, each transtition I_j of the set L is fired (j is the number of the module). The firing of I_j only changes buffers (B to B') and the local marking of module j. In general several successors h_j of H_j in $\mathcal{N}_j$ can enable I_j. Then for each h_j we must fire I_j to produce a new local marking H_j'. The next step is to check if the local marking H_j' is new. If this is the case a new local breadth-first search is done in module j computing and storing new local states. Then we must test if the reached marking (B', H') is a new head to insert in the queue. To do that we test if B' is new (then it is clearly a new head). If B' is not new, we must compare H' with all the computed heads of the form (B', H''). Four cases are possible. If (B', H') is reachable from (B', H'') by means of internal transitions (B', H') is erased. If (B', H'') is reachable from (B', H') then (B', H') replaces (B', H''). If (B', H'') and (B', H') have a common successor then we mark the two heads as related. It is important to store this information for the subsequent phase. In other case (B', H') is a new head. The previous tests about reachability between states can also be done locally in each module.

The output of this algorithm is a decomposed description of RG($\mathcal{S}$) (computed with less memory and time requirements than in the classical way). Applying properties 12 and 14 we could precisely compute RG($\mathcal{S}$). Now, we apply this reduced description of RG($\mathcal{S}$) to generate the RG's of the reduced submodels $\mathcal{LS}_i$ and $\mathcal{BS}$. Two different reduced submodels will be computed. In subsection 4.1 the projections of the markings of RG($\mathcal{S}$) on the preserved nodes in the submodels will be computed and in subsection 4.2 the aggregations of RG($\mathcal{S}$) by means of relations R_i with $i = 0, \ldots, K$ will be computed.

4.1. Reducing spurious markings

In this subsection we use the decomposed description of the RG($\mathcal{S}$) computed with algorithm 4.1 for the generation of the projections of RG($\mathcal{S}$) on the preserved places of $\mathcal{LS}_i$ and $\mathcal{BS}$. So, the difference of this second decomposition technique with respect to the first one presented in section 3 is that the subsystems have no spurious states (thus, solving the first problem **P1** mentioned in the beginning of this section). In this way we obtain the smallest possible RS for each subsystem.

Definition 15 *Let $\mathcal{S}$ be a SAM and RG($\mathcal{S}$) its RG. Let R'_i with $i = 1, \ldots, K$ be the following equivalence relation defined on the set of states of RG($\mathcal{S}$): $\forall s_1, s_2$ states of RG($\mathcal{S}$), $\langle s_1, s_2 \rangle \in R'_i$ iff $s_1|_{P_i \cup B} = s_2|_{P_i \cup B}$. Let R_0 be the following equivalence relation defined on the set of states of RG($\mathcal{S}$): $\forall s_1, s_2$ states of RG($\mathcal{S}$), $\langle s_1, s_2 \rangle \in R_0$ iff $s_1|_B = s_2|_B$. We denote by RG$'_i$ with $i = 0, \ldots, K$ the RG resulting after the aggregation of all the states in RG($\mathcal{S}$) belonging to the same equivalence class of R'_i.*

RG$'_0$ is the reduced RG somehow related to the $\mathcal{BS}$ (it is the result of aggregating all the states with the same marking on the buffers) and RG$'_i$ with $i = 1, \ldots, K$ is the reduced RG somehow related to $\mathcal{LS}_i$ (i.e., the result of aggregating all the states with the same marking in the places of $\mathcal{LS}_i$).

Here is the algorithm for the generation of these reduced RG's.

Algorithm 4.2

input: Set of heads of a SAM
RG($\mathcal{S}$)$|_{\mathcal{N}_i}$ for $i = 1, \ldots, K$
$l \in \{0, \ldots, K\}$ (RG$'_l$ to generate)
$H :=$ first head (initial marking)
$M_l := H_l$ the local initial marking
insert (H, M_l) in the queue
while queue non empty **do**
  $P := (H, M_l)$ first pair of the queue
  $L := \emptyset$ (list of enabled transitions)
  in local subnet $\mathcal{N}_l$:
    add to L interface transitions enabled in (H_0, M_l)
    add to L internal transitions enabled in M_l
  add to L interface transitions of T_j with $j \neq l$ enabled in H
  for each $t \in L$ **do**
    $(H, M_l) \xrightarrow{t} (H', M'_l)$
    $P' := (H', M'_l)$
    if (H'_0, M'_l) is new in RG$'_l$ **then**
      insert P' in the queue
      insert $(H_0, M_l) \xrightarrow{t} (H'_0, M'_l)$ in RG$'_l$
    else
      for $t \in \mathrm{TI} \setminus T_l$ enabled in H' **do**
        if t not enabled in (H'_0, M'_l) **then** insert P' in the queue
output: RG$'_l$

In the above algorithm we maintain a queue of pairs (H, M_l) (H is a head and M_l a local marking) like in a breadth-first search of a directed graph. Given a pair (H, M_l), the list L of enabled transitions must be computed. This list L is composed by internal transitions (in the case of RG$_l$ with $l > 0$) and interface transitions. The internal transitions and interface transitions of module l are computed in $\mathcal{N}_l$ according to local state (H_0, M_l) (H_0 is the marking of the buffers). The interface transitions t of module j with $j \neq l$ are the transition associated to the head H. The firing of a transition in the list generates a new pair (H', M'_l). If the state (H'_0, M'_l) is new in RG$_l$ the pair (H', M'_l) is inserted in the queue to explore the new marking. If (H'_0, M'_l) has been already computed, the interface transitions t of module j with $j \neq l$ enabled in H' must be tested because new transitions for the marking can appear due to this head (the state (H'_0, M'_l) may be successor of several heads with different interface transitions enabled).

In the case $L = 0$ (to compute the RG$_0$) only the heads must be processed (not the local markings). RG($\mathcal{S}$) can also be computed maintaining pairs of heads with the local marking in all the modules.

By construction, all the states of the reduced RG's are projections of reachable states in $\mathcal{S}$. Therefore, all the spurious states have been deleted and the inclusion of theorem 7.ii becomes an equality now.

Note that the time and memory complexity of the genration of these reduced RG's is less than or equal to the complexity of the generation of the RG($\mathcal{LS}_i$) of section 3 due to the fact that a spurious marking is never computed.

4.2. Reducing spurious firing sequences

In this subsection we use the decomposed description of the RG($\mathcal{S}$) computed with algorithm 4.1 to generate different RG's related to the $\mathcal{LS}_i$ and $\mathcal{BS}$ with respect to the ones in the previous subsection. In this case we will compute the aggregations of RG($\mathcal{S}$) by means of relations R_i with $i = 0, \ldots, K$ defined before. With this third decomposition technique the subsystems have no spurious states (as in the second one) but also the aggregation of states not connected by internal transitions in RG($\mathcal{S}$) is avoided (solving the second problem **P2** mentioned in the beginning of this section). With respect to the first technique (section 3) no spurious states are computed in the subsystems and with respect to the second one (subsection 4.1), the RG's of the subsystems may have more states but without anomalous aggregations, thus eventually improving the approximations.

Definition 16 *Let $\mathcal{S}$ be a SAM and RG($\mathcal{S}$) its RG. Let R_i with $i = 0, \ldots, K$ be the equivalence relations defined in Definition 8. We denote by RG$_i$ with $i = 0, \ldots, K$ the RG resulting after the aggregation of all the states in RG($\mathcal{S}$) belonging to the same equivalence class of R_i.*

RG$_0$ is the reduced RG somehow related to the $\mathcal{BS}$ (it is the result of aggregating all the states connected by internal transitions of $\mathcal{S}$) and RG$_i$ with $i = 1, \ldots, K$ is the reduced RG related to $\mathcal{LS}_i$ (it is the result of aggregating all the states connected by internal transitions of subnets $\mathcal{N}_j$ with $i \neq j$). In the example of Fig. 2, the states $P_{11}P_{21}$ and $P_{12}P_{22}$ are not aggregated in RG$_0$ (while they do were aggregated in the first and second technique) because they are not connected through an internal transition. Note that the time and memory complexity of this operation is less than or equal to the complexity of the generation of the RG($\mathcal{LS}_i$) of section 3.

Here is the algorithm for the generation of the reduced RG's.

Algorithm 4.3
input: Set of heads of a SAM
RG($\mathcal{S}$)$|_{\mathcal{N}_i}$ for $i = 1, \ldots, K$
$l \in \{0, \ldots, K\}$ (RG$_l$ to generate)
$H :=$ initial marking; $M_l := H_l$ the local initial marking
insert (H, M_l) in the queue
while queue non empty **do**
$P := (H, M_l)$ first pair of the queue
$L := \emptyset$ (list of enabled transitions)
in local subnet $\mathcal{N}_l$:
add to L interface transitions enabled in (H_0, M_l)
add to L internal transitions enabled in M_l
add to L interface transitions of T_j with $j \neq l$ enabled in H
for each $t \in L$ **do**
$(H, M_l) \xrightarrow{t} (H', M_l')$
$P' = (H', M_l')$
if P' is new in RG$_l$ **then**
insert P' in the queue
insert $P \xrightarrow{t} P'$ in RG$_l$
else if $\exists (H'', M_l')$ in RG$_l$ with H'' related to H'
for $t \in \mathrm{TI} \setminus T_l$ enabled in H' **do**
if t not enabled in (H_0', M_l') **then** insert P' in the queue
output: RG$_l$

This algorithm is very similar to the previous one. They differ only in the merging proccess of the states of the subsystems. In algorithm 4.2 all the states of a given subsystem with the same marking on the preserved places are merged. Now only the states with the same marking belonging to related heads can be merged because two given states are connected through a path of internal transitions iff they belong to the same head or related heads.

5. Application to iterative throughput approximation methods

In this section we apply the previous decomposition techniques for approximate throughput computation. The method that we consider is, basically, the same *response time preservation method* used in [1, 12, 10, 11]. In each $\mathcal{LS}_i$ a unique module of $\mathcal{S}$ with all its places and transitions is kept. So, in $\mathcal{LS}_i$ interface transitions of module j (for $j \neq i$) approximate the response time of module j. The algorithm is the following:

Algorithm 5.1
select the modules
derive RG($\mathcal{LS}_i$) for $i = 1, \ldots, K$ and RG($\mathcal{BS}$)
give an initial service rate $\boldsymbol{\mu}_i^{(0)}$ for $i = 2, \ldots, K$
$j := 0$ {counter for iteration steps}
repeat
$j := j + 1$
for $i := 1$ **to** K **do**
solve $\mathcal{LS}_i$: In: $\boldsymbol{\mu}_l^{(j)}$ for $l < i$ and $\boldsymbol{\mu}_l^{(j-1)}$ for $i < l \leq K$
Out: initial rates $\boldsymbol{\mu}_i$ and thr. $\boldsymbol{\chi}_i^{(j)}$ of $\mathrm{TI} \cap T_i$
solve $\mathcal{BS}$: In: $\boldsymbol{\mu}_l^{(j)}$ for $l < i$ and $\boldsymbol{\mu}_l^{(j-1)}$ for $i < l \leq K$
$\boldsymbol{\mu}_i$ and $\boldsymbol{\chi}_i^{(j)}$ for transitions in $\mathrm{TI} \cap T_i$
Out: actual rates $\boldsymbol{\mu}_i^{(j)}$ $\mathrm{TI} \cap T_i$
until convergence of $\boldsymbol{\chi}_1^{(j)}, \ldots, \boldsymbol{\chi}_K^{(j)}$

In the above procedure, once the K modules have been selected and given some initial values $\boldsymbol{\mu}_i^{(0)}$ of the rates of the interface transitions of $\mathcal{N}_i$ with $i = 2, \ldots, K$, a certain CTMC associated to $\mathcal{LS}_1$ is solved. The computation of that CTMC varies depending on which of the three techniques presented in previous sections is applied.

The selection of $\boldsymbol{\mu}_i^{(0)}$ for $i = 2, \ldots, K$ does not affect (under our experience) to the approximation obtained with the response time preservation method. A simple option is putting the initial rate of the transitions in the original model. From the solution of that CTMC, the first estimation $\boldsymbol{\chi}_1^{(1)}$ of the throughput of the interface transitions of $\mathcal{N}_1$ can be computed. Then, the initial estimated values of service rates of interface transitions of $\mathcal{N}_1$ must be derived. To do that, we take the initial values $\boldsymbol{\mu}_1^{(0)}$ for service rates of interface transitions of $\mathcal{N}_1$ and we scale them iteratively in the $\mathcal{BS}$ until the throughput of interface transitions of $\mathcal{N}_1$ in the $\mathcal{BS}$ and $\boldsymbol{\chi}_1^{(1)}$ are equal. The same procedure is executed for each $\mathcal{LS}_i$ in a cyclic way. Each time we solve $\mathcal{LS}_i$ we obtain in the $\mathcal{BS}$ a new estimation of the interface transitions rates of $\mathcal{N}_i$. The previous steps are repeated until *convergence* of the throughput approximations of the subsystems is achieved (if there exists).

The initial estimated values of the interface transitions rates of $\mathcal{N}_i$ in $\mathcal{LS}_i$ are computed by means of the implicit places H_i in $\mathcal{LS}_i$. Let p_t be the probability that the transition $t \in T_i \cap \mathrm{TI}$ is enabled in the reduced subnet (similar to [1], it is an estimation of the transition utilization with single server semantics). Then we put $\mu_t^{(j)} = \chi_t / p_t$ (it is an estimation of the transition rate).

The computation of the actual rates of the corresponding interface transitions in $\mathcal{BS}$ can be implemented with an iterative search (the initial estimated transition rates are weighted until the throughput of the involved transitions are equal in $\mathcal{BS}$ and in $\mathcal{LS}_i$). Now the net system ($\mathcal{BS}$) has considerably fewer states than the original one. In each iteration of this search, the underlying CTMC of the $\mathcal{BS}$ is solved. Note that only in the first iteration the CTMC is completely derived. For later iterations only some values must be changed. In some cases (for instance, for FRT-nets [3]), the relative throughputs (or visit ratios) of transitions of the $\mathcal{BS}$ are independent on the transition service time (they only depend on the net structure and on the conflict resolution rates). In these cases, only one parameter must be tuned up in the $\mathcal{BS}$, thus a unidimensional search can be implemented. The case of multidimensional search shows convergence problems so it is still an open topic.

Now, convergence of the entire method and the uniqueness of the solution should be addressed. Using the results in [8] we were not able yet to obtain a formal proof of the convergence of the method due to the use of the $\mathcal{BS}$ (the iterative scheme is more difficult for formal studies). But extensive testing with nets in which an unidimensional search can be implemented in the $\mathcal{BS}$ shows that there exists one

and only one solution, computable in a finite number of steps, typically between 2 and 6 if the convergence criterion is that the difference between the two last estimations of the throughput is less than 0.1%.

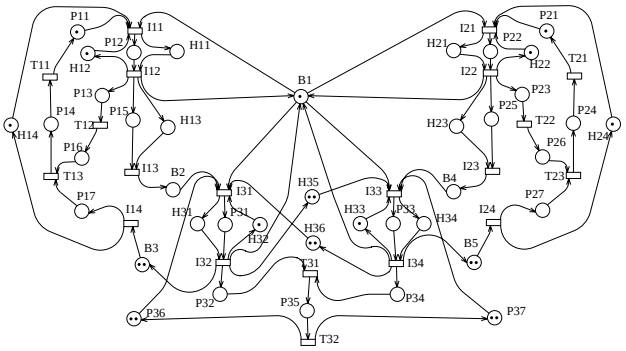

Figure 4. The ES of a SAM.

Now we are going to apply the three decomposition techniques to the throughput approximation of two different non trivial examples by means of the previous response time preservation algorithm. The first one is the SAM of Fig. 4. It is composed of three modules interconnected through 5 buffers (places B_1 to B_5). Places that start with H are the implicit places (the first number is the subsystem in which they have been computed) and places that start with P are the internal ones. Transitions that start with I are the interface and the other are internal. The underlying CTMC has 94080 states. In this case, the three decomposition techniques produce the same aggregated submodels $\mathcal{LS}_i$ with $i = 1, 2, 3$ and $\mathcal{BS}$ so the approximation will be the same. These submodels have 14640, 14640, 14400 and 5600 states, respectively.

The rates of transitions have been arbitrarily fixed as: 1.0 for transitions T_{11}, T_{13},T_{21}, T_{23}, I_{32}, I_{34}; 2.0 for T_{12}, T_{22},I_{11}, I_{21}, I_{31}, I_{33}; 3.0 for I_{12}, I_{14}, I_{22}, I_{24}; 4.0 for T_{31}, T_{32} and 5.0 for I_{13}, I_{23}.

$\mathcal{LS}_1$		$\mathcal{LS}_2$		$\mathcal{LS}_3$	
$\chi(I_{11})$	scale f.	$\chi(I_{21})$	scale f.	$\chi(I_{31})$	scale f.
0.230496	1.025007	0.223043	1.089387	0.222493	1.005589
0.223777	1.090070	0.223461	1.091612	0.223462	1.005589
0.223472	1.091621	0.223471	1.091612	0.223473	1.005589
0.223471	1.091621	0.223471	1.091612	0.223473	1.005589
Error: -0.51%					

Table 1. Iteration results for the SAM in Fig. 4.

In this case, the exact throughput of transition I_{11} is 0.224607. In Table 1 we present the iteration steps of the method. Columns $\chi(I_{ij})$ are the estimated values for throughput of transition I_{ij} at each iteration step. Columns 'scale f.' are the scale factors modifying the previous estimated service rates, computed with the $\mathcal{BS}$. Convergence of the method is obtained in this case in the four iteration step and the error was -0.51%.

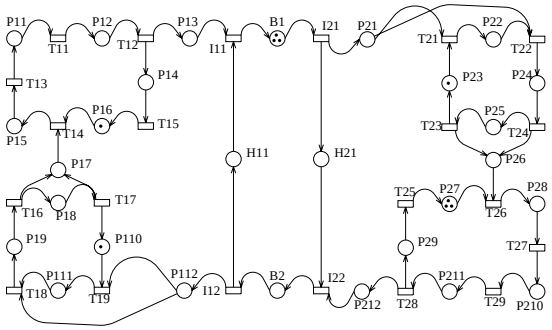

Figure 5. The ES of another SAM.

As a second example we have selected the SAM of Fig. 5 to stress the differences among the three techniques. The names of places and transitions follow the same rules of the previous example. The model is composed of 2 modules interconnected through 2 buffers. All transition rates have been set to 1.0. It has 2698 reachable states and the exact throughput of transition I_{11} is 0.104304. In Table 2 we present the iteration steps of the three techniques for this case.

Applying the first decomposition technique, the CTMC's associated to $\mathcal{BS}$, $\mathcal{LS}_1$ and $\mathcal{LS}_2$ have 20, 310 and 828 states, respectively. The throughput approximation in this case is 0.113586 (5 iterations), in other words an error of 8.9%.

With the second decomposition technique, the CTMC's associated to $\mathcal{BS}$, $\mathcal{LS}_1$ and $\mathcal{LS}_2$ have 18, 211 and 487 states, respectively. The throughput approximation in this case is 0.076916 (6 iterations), in other words an error of -26.26%.

With the third decomposition technique, the CTMC's associated to $\mathcal{BS}$, $\mathcal{LS}_1$ and $\mathcal{LS}_2$ have 22, 211 and 487 states, respectively. The throughput approximation in this case is 0.105939 (4 iterations), in other words an error of 1.57%.

In this case, the first technique fails due to the great number of spurious states in the CTMC's of the subsystems (compare the number of states of $\mathcal{LS}_2$ in the first technique with respect to the other ones). More interesting is the great difference between the approximation of the second and third techniques. This difference is due to the inappropriate aggregation of 4 states in the CTMC associated to the $\mathcal{BS}$ for the second technique (the CTMC of the $\mathcal{LS}_i$ are the same in the second and third techniques).

We do not compare here the overall computation time of our algorithms with respect to the time needed for solving the entire CTMC due to two facts. The first one is

$\mathcal{LS}_1$		$\mathcal{LS}_2$		$\mathcal{LS}_1$		$\mathcal{LS}_2$		$\mathcal{LS}_1$		$\mathcal{LS}_2$	
$\chi(I_{11})$	scale f.	$\chi(I_{21})$	scale f.	$\chi(I_{11})$	scale f.	$\chi(I_{21})$	scale f.	$\chi(I_{11})$	scale f.	$\chi(I_{21})$	scale f.
0.174202	1.030663	0.124153	1.209259	0.158793	0.998916	0.105566	1.139154	0.158793	1.048865	0.110149	1.031971
0.114948	1.229055	0.113760	1.255224	0.082152	0.949469	0.079294	1.253320	0.105799	1.003225	0.105927	1.034708
0.113605	1.236202	0.113590	1.256088	0.077370	0.950758	0.077126	1.265756	0.105940	1.003435	0.105941	1.034708
0.113587	1.236291	0.113587	1.256088	0.076956	0.950928	0.076934	1.266884	0.105939	1.003435	0.105941	1.034708
0.113586	1.236291	0.113587	1.256088	0.076919	0.950950	0.076917	1.266979				
				0.076916	0.950950	0.076916	1.267026				
First technique			Error: 8.90%	Second technique			Error: -26.26%	Third technique			Error: 1.57%

Table 2. Iteration results for the SAM in Fig. 5 with the three decomposition techniques.

that the implementation of our algorithm is just a prototype to make experiments about convergence and relative error. The second one is that we made experiments with 'simple' examples (500000 states at most) in order to obtain the exact solution of the original model (to compare with the approximation). Theses examples take just a few minutes of computation and the reduction in time is not significant. If the original model had millions of states then we could not compare the time complexity because it would be impossible to solve the original model with the classical technique.

6. Conclusions

State space decomposition techniques of stochastic Petri nets have been considered for numerical computation of performance indices and, in particular, for throughput approximation based on response time preservation of a submodel. The decomposition is induced by the net structure of the original model, for which an structured view is assumed. Within this framework, three different variants for obtaining the CTMC of the auxiliary submodels have been presented. In the first one, only ergodicity of the CTMC of the submodels is assured, in order to be able to compute meaningful results. In the second technique, a reduction of spurious states is achieved with the goal of reducing the gap between the behaviour of the submodels and that of the original system. The third technique seeks for the same objective but reducing the firing sequences of the submodels that do not correspond to real (projections of) sequences of the original model.

The accuracy of the techniques has been compared using a couple of models that are specially sensible to spurious behaviour. In most practical cases, our experience is that the obtained results do not differ so much.

The gain of state space decomposition techniques with respect to the classical exact solution algorithm (solution of the underlying CTMC of the original model) is in both memory and time requirements. With respect to space, the infinitesimal generator of the CTMC of the whole system is never stored. Instead of that, the generator matrices of smaller subsystems are stored. With respect to time complexity, we do not solve the CTMC isomorphous to the original system but those isomorphous to the derived subsystems.

References

[1] J. Campos, J. M. Colom, H. Jungnitz, and M. Silva. Approximate throughput computation of stochastic marked graphs. *IEEE Trans. on Soft. Eng.*, 20(7):526–535, Jul. 1994.

[2] J. Campos, S. Donatelli, and M. Silva. Structured solution of asynchronously communicating stochastic modules. *IEEE Trans. on Soft. Eng.*, 25(2):147–165, Mar. 1999.

[3] J. Campos and M. Silva. Structural techniques and performance bounds of stochastic Petri net models. In G. Rozenberg, ed., *Adv. in Petri Nets 1992*, vol. 609 of *LNCS*, pp. 352–391. Springer-Verlag, Berlin, 1992.

[4] G. Ciardo and K. Trivedi. A decomposition approach for stochastic reward net models. *Perf. Eval.*, 18(1):37–59, Sep. 1993.

[5] J. M. Colom and M. Silva. Improving the linearly based characterization of P/T nets. In G. Rozenberg, ed., *Adv. in Petri Nets 1990*, vol. 483 of *LNCS*, pp. 113–145. Springer-Verlag, Berlin, 1991.

[6] F. DiCesare, G. Harhalakis, J. M. Proth, M. Silva, and F.B. Vernadat. *Practice of Petri Nets in Manufacturing.* Chapman & Hall, London, 1993.

[7] A. Gibbons. *Algorithmic Graph Theory*. Cambridge U. Press, London, 1985.

[8] V. Mainkar and K. Trivedi. Fixed point iteration using stochatic reward nets. In *Proc. of the* 6^{th} *Int. Work. on PNPM*, pp. 21–30, Durham, NC, USA, Oct. 1995. IEEE-Comp. Soc. Press.

[9] M. K. Molloy. Performance analysis using stochastic Petri nets. *IEEE Trans. on Comp.*, 31(9):913–917, Sep. 1982.

[10] C. J. Pérez-Jiménez, J. Campos, and M. Silva. Approximate throughput computation of a class of cooperating sequential processes. In *Proc. of the Rensselaer's V Int. Conf. on Comp. Integrated Manuf. and Autom. Tech. (CIMAT'96)*, pp. 382–389, Grenoble, France, May 1996.

[11] C. J. Pérez-Jiménez, J. Campos, and M. Silva. On approximate performance evaluation of manufacturing systems modelled with weighted T-systems. In *Proc. of the IMACS/IEEE-SMC Multiconf. on Comp. Eng. in Sys. Appl. (CESA'96)*, pp. 201–207, Lille, France, Jul. 1996.

[12] C. J. Pérez-Jiménez, J. Campos, and M. Silva. State machine reduction for the approximate performance evaluation of manufacturing systems modelled with cooperating sequential processes. In *Proc. of the 1996 IEEE Int. Conf. on Rob. and Aut.*, pp. 1159–1165, Minneapolis, MN, USA, Apr. 1996.