ASYNCHRONOUS COMPOSITION OF HIGH LEVEL PETRI NETS: A QUANTITATIVE APPROACH

Serge Haddad Patrice Moreaux

LAMSADE – URA CNRS 825, Université Paris Dauphine, Place du Maréchal de Lattre de Tassigny, 75775 PARIS Cedex 16, FRANCE

Abstract. Stochastic Well Formed Nets (SWNs) are a powerful Petri Net model which allows the computation of performance indices with an aggregation method. Decomposition methods initiated by B. Plateau are another way to reduce the complexity of such a computation. We have shown in a previous work, how to combine these two approaches for systems with synchronous composition. Despite similarities between the asynchronous and synchronous cases, it turns out that the former presents specificities that need theoretical foundations. We undertake this task in the present paper. We derive necessary conditions on the modeled systems that allow for the two methods to be combined. For parallel systems satisfying these necessary conditions we develop a model with the corresponding algorithm. This model, based upon synchronization of "global" tokens moving across submodels, covers a large range of real life systems. An example shows the intuitive ideas behind these developments.

Introduction

It is well known that complex systems with synchronization forbid the use of analytical results to find the steady state probabilities of the corresponding stochastic models. We have then to work at the Markov chain level or to use approximate methods.

Our work deals with Markov chains. In this case, because of the huge size of the state space, one is lead to find efficient methods that avoid the building of the whole chain. The most effective methods are based upon the structure of the models generating the Markov chain. Two main methods follow this approach, aggregation and decomposition using tensor products.

The aggregation method builds a partition of the state space compatible with stochastic behaviour to get a new Markov chain on the classes of the partition with much fewer states. Then this chain is solved providing steady state probabilities of the aggregates and the results are possibly used to find the steady state probabilities of the original chain. This method is very effective since, for instance, with the Stochastic Well formed Net (SWN) model ([3]), we can build

the partition a priori, and furthermore it has an interpretation at the system level (a class is a set of states for which a condition is satisfied, ...).

The decomposition method describes the state space as a product of smaller state spaces and from this, gives an expression of the infinitesimal generator of the Markov chain using only generators on these smaller spaces and operators from tensor algebra. The chain is then solved using directly this expression which contains only "small" matrices. This method, introduced by Plateau [11], with the Stochastic Automata Network (SAN) model was extended to Generalized Stochastic Petri Nets (GSPN) by Donatelli [6] and Buchholz [2].

The purpose of our work is to merge these two methods keeping the benefits of both. We have shown in [8] how to combine the SWN model and the Plateau approach for synchronous composition of subsystems. In this paper, we show how to develop the same combination in the asynchronous case already tackled by Buchholz for GSPNs. As shown by Buchholz in several papers, the asynchronous composition needs the definition of the environment of each subnet: we propose the formally defined notion of *abstract view* of a subnet, based on qualitative criteria, which allows application of the decomposition method.

However, as in the synchronous case, we first point out the two main difficulties of the combination approach, that is to say the specification problem of external/internal synchronization and the resolution problem of synchronization memory. If we have a system with several objects of the same kind (processes for instance), which are synchronized together (through common resources sharing for instance), we may design a model of this system as a synchronized product of models of the activities of these objects, but this product does not allow any aggregation; the synchronization occurs between objects and there are no object classes. We say in this case that the system exhibits an internal synchronization. Therefore we focus on systems with external synchronization, that is to say with synchronization between objects of different kinds. In this context, we observe the synchronization memory phenomenon: the successive firings of synchronization transitions change the state space in such a way that, as a general rule, the original chain cannot be lumped as composition of smaller aggregated chains, preventing the combination of the two methods.

Fortunately, for models involving synchronization memory, which is the general case, we show that a control of the synchronization memory may be managed for restricted asynchronous composition of SWNs. We derive the steady state probabilities computation algorithm which is composed of four stages: definition of an abstract view of each component (subnet), definition of enlarged SWNs, derivation of a tensor expression of the generator of the model, computation of steady state probabilities via iterative methods with that expression.

The outline of the paper is as follows: in Sect. 1 we remind the reader of the methods we want to merge and review related works. We then summarize the basic problems and our proposed framework in Sect. 2 before giving theoretical results together with an illustrative example in Sect. 3. For concision of the paper, proofs of results are carried over to appendix A and definitions about

flows and implicit places may be found in App. B.

1 General framework and related works

In this presentation, we restrict ourselves for the sake of simplicity to Continous Time Markov Chain (CTMC).

1.1 Aggregation and decomposition

Let us recall that aggregation methods may be summarized in the following steps: given a CTMC with state space E and infinitesimal generator $Q = [q_{ij}]$:

- find *a partition* of the state space E, say $(E^{(k)})_{k=1,\ldots,K}$, so that the behaviour of states belonging to the same $E^{(k)}$ are stochastically equivalent, that is to say:

$$\forall k, h \in \{1, \ldots, K\}, \quad \forall e, e' \in E^{(k)} \quad \sum_{e_h \in E^{(h)}} q_{e,e_h} = \sum_{e_h \in E^{(h)}} q_{e',e_h} = \widetilde{q}_{k,h} \quad (1)$$

- construct a new CTMC with state space $\widetilde{E} = \{E^{(k)} \mid k = 1, \ldots, K\}$, and infinitesimal generator $\widetilde{Q} = [\widetilde{q}_{k,h}]$,
- solve this CTMC,
- possibly compute the probabilities of the states $e_k \in E^{(k)}$ using the previous solution (note that this step requires additional information about the e_k).

Kemeny and Snell ([10]) showed (first for Discret Time Markov Chain (DTMC), the result was later on extended to CTMC) that *the strong lumpability condition* (1) is necessary and sufficient for the aggregated process to be markovian hence a CTMC. Usually we have $K \ll \sum_k |E^{(k)}|$ so the computation of the steady state probabilities is easier for $\widetilde{Q}$ than for Q. SWN ([3]) is a Petri Net model which supports such a method.

Now, the basic steps in decomposition methods are:

- describe the CTMC state space E as a subset of *a cartesian product* of smaller spaces, say $E \subseteq \prod_{k=1}^{K} E_k$,
- use this decomposition to get an expression of Q as function $f(Q_1, \ldots, Q_K)$, where Q_k is the infinitesimal generator of the CTMC restricted to E_k,
- compute the solution π with $\pi.f(Q_1, \ldots, Q_K) = 0$.

In our context, the functions f are sums of tensor products of the Q_k (see [5] and [11] for details about tensor algebra and its use in the area of stochastic transition systems).

The main interest of this method is to enable the steady state probabilities computation *without computing the Q matrix* but instead, *directly using the tensor expression of Q*.

Trying to merge the two methods above means: from a CTMC C with state space E and infinitesimal generator Q, get an aggregated CTMC of C as a "tensor composition" of smaller aggregated CTMCs. Hence, the successive steps for such a method are:

- build a state space decomposition of E, getting $E \subseteq E' = \prod_{k=1}^{K} E_k$,
- use an aggregation method verifying the strong lumpability condition (1) on each of the CTMCs (E_k, Q_k) leading to $\widetilde{E_k} = \{E_k^{(j)} \mid j = 1, \ldots, n_k\}$ and infinitesimal generators $\widetilde{Q_k}$,
- build the product $(\widetilde{E'} = \prod_{k=1}^{K} \widetilde{E_k}$, $\widetilde{Q} = f(\widetilde{Q_1}, \ldots, \widetilde{Q_K}))$ of the aggregated CTMCs and define the aggregated image $\widetilde{E} \subseteq \widetilde{E'}$ of E.

Unfortunately, as a general rule, $(\widetilde{E}, \widetilde{Q})$ *is not an aggregation* of (E, Q) verifying (1). *So the problem is to find additional conditions on the initial CTMC* C which ensure that the combination satisfies the condition (1). We give in Sect. 3 a solution for SWNs models, using tensor expression for the function f.

1.2 GSPNs asynchronous composition

Asynchronous composition of Petri nets models communicating subsystems (subnets), with entities (tokens) moving from one subsystem to another one. The communication links between subnets are common places which are not input places of transitions of the source net (the PO_k places in the source, the input places in the destination) and transitions from the source subnet with at least one output place in the destination subnet. The formal expression of asynchronous composition for a given class of Petri nets (GSPN, SWN, ...) is summarized in the following

Definition 1. The Petri net $N = (P, T, \ldots)$ is the asynchronous composition of the nets $(N_k = (P_k, T_k, \ldots))_{k \in K}$ iff[1]

- For all k there is a subset PO_k of P_k s.t. $\forall t \in T_k \ {}^\bullet t \bigcap PO_k = \emptyset$ (the set of output places); we denote $TO_k = \{t \in T_k \text{ s.t. } t^\bullet \bigcap PO_k \neq \emptyset\}$ the set of output transitions of N_k, and $TO = \bigcup_{k \in K} TO_k$ the set of all output transitions,
- $P = \bigcup_{k \in K} P_k$ and $P_k \bigcap P_{k'} \subseteq [PO_k \bigcap (P_{k'} \backslash PO_{k'})] \bigcup [(P_k \backslash PO_k) \bigcap PO_{k'}]$ for $k \neq k'$ (the set of places and the input/output places),
- $T = \bigcup_{k \in K} T_k$ and $T_k \bigcap T_{k'} = \emptyset$ for $k \neq k'$ (the set of transitions); $T \backslash TO$ is the set of local transitions,
- Each additional parameter of N for the class of nets involved (designated by ...) is such that its restriction to N_k is the corresponding parameter of N_k.

In the rest of the paper, for any marking M of N, we denote $M_k = M(P_k \backslash PO_k)$, so that $M = (M_k)_{k \in K}$.

P. Buchholz studied the asynchronous composition of GSPNs and other kinds of Petri nets in several papers. His approach may be summarized as follows.

The global net N is decomposed in K subnets N_k as defined above. For each subnet, one define an *aggregated view*, discarding local behaviour of the subnet (in his papers he proposed several definitions of such aggregated views and we refer here to the one proposed in [1, 2]: a *virtual place* p_k and a *virtual transition* t_k summarize the global behaviour of the subnet).

[1] from now on, we shall use for ease of writing, K as set of k indexes or as maximal k index when no confusion can arise.

The global behaviour of the net is summarized in the net N_0 composed with aggregated views of all the subnets and is studied for itself giving a Reachability Set (RS) RS_0.

The RS RS_k of the subnet k is computed using N_k and the behaviour of all other subnets, also summarized with a single virtual place transition pair. RS_k is decomposed in partition ($RS_k(m_k)$), all markings of $RS_k(m_k)$ providing the same marking of the virtual place p_k.

Buchholz has proved that the generator of the CTMC of the tangible states of N may be expressed as linear combinations of tensor products of three kinds of matrices: $Q_k(m_k)$ giving local transition firings of N_k, $U_k(n, m_k)$ for marking changes due to incoming bags n in N_k and $S_k(m_k, c)$ for those due to firings of output transitions of N_k for a colour c. So, to compute the steady state probabilities, it is sufficient to use these "small" matrices.

The main advantages of this method are:

- reduced data structures allow the study of large nets
- elimination of vanishing states may be done at the subnet level which reduces both state space sizes and time computation w.r.t to global elimination
- aggregated views may be defined at different levels leading to hierarchical decomposition from coarsest views to more detailed ones.

These works also point out the very important fact that the study of a subnet in isolation requires to define its *environment*. We propose in Sect. 3.1 such a definition for SWNs in a formal way.

However the following points must be highlighted:

- although [2] deals with SWNs, only Tangible Reachability Graphs (TRG) – not Symbolic Reachability Graph (SRG) – are used to compute the solution, the Well Formed aspects of the net being used only to compute ordinary markings and firings: the Q matrix relates to the unfolded net, and *no aggregation –in the stochastic meaning– is exploited.*
- *no automatic method to build aggregated views* based upon the net description is provided, which may lead, as pointed out by the author ([1]), to consistency problems.

The present work provides solutions for these two important problems.

2 Theoretical Context

In this section we first set the framework of our research in order to extend the results that we have reviewed above, then point out the key problems about such extensions. Let us recall that we want to develop an aggregation method based upon the SWN formalism while keeping the advantages of the decomposition methods for asynchronous composition of subsystems. Because of space constraint, we refer the reader to [3] for a detailled presentation of SWN and SRG, and to [7] for a complete study.

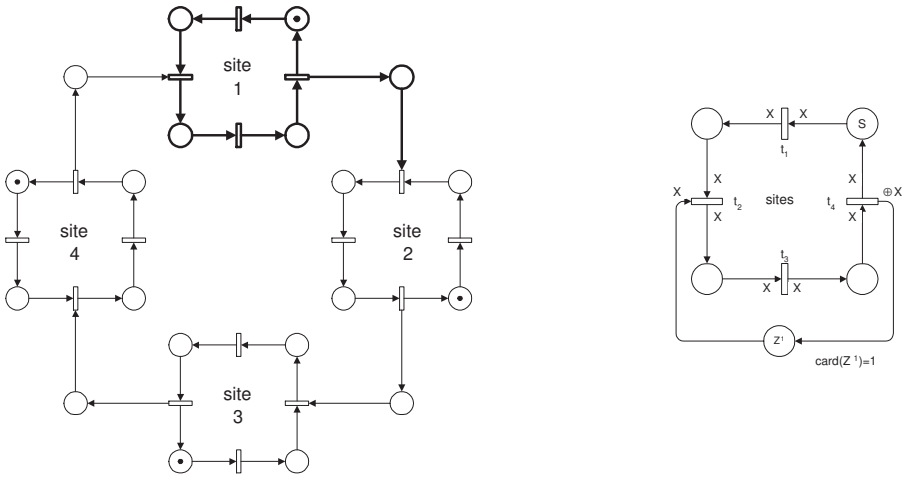

Fig. 1. GSPN and SWN of a logical token ring

2.1 The specification problem

The first problem with this approach relates to the kind of synchronization found at the system specification level: if several objects of the same "type" (processes for instance) are synchronized together, then we may build a model with a product structure , each of the terms of this synchronized product being a model of one object behaviour, leading to a product of CTMCs; but then, as we modelize *one* object behaviour with *one* subnet, there is no object class at all, and therefore we cannot use aggregation. In these situations, we say that the system exhibits *internal synchronization*, "internal" meaning "between objects of the same kind". Alternatively, we may also build SWN models of such systems, and we get an aggregated CTMC of the whole model, but no decomposition may be used.

We say that the system exhibits *external synchronization* if there is synchronization between objects of *different classes*.

A simple example of internal synchronization is a system of sites executing sequential code with a section in mutual exclusion, the enabling of critical section execution being allocated in a cyclic manner to each site (logical token ring). The GSPN and SWN of this system (with 4 sites) is shown in Fig. 1: starting from the idle state, each site does a first job (first transition) and then waits for the mutual exclusion token to continue its work (second transition). When the critical section work is done, it releases the mutual exclusion token (fourth transition) and returns to idle state. In the SWN model, we have only one basic class C_s for the Sites. The S marking indicates that all sites are idle in the initial state and the Z^1 dynamic marking means that this place holds *any single token* of the colour class C_s.

As we see, the SWN is the "folding" of the GSPN and the synchronization between sites is embedded in it; on the other side, we could try to decompose the GSPN as two by two synchronization between four identical GSPNs (one example of which is drawn in bold in Fig. 1).

So, in such situations, we have to choose between two models for the specification of the system: keeping a whole SWN or unfolding the net to get a GSPN or SAN decomposition.

In systems with external synchronization, objects of different classes are synchronized and, following the general method presented in Sect. 1.1, we want to build a "synchronized product" of subnets such that each subnet modelizes the behaviour of an object class. In this way, we could use aggregation at the subnets level using the SWN model and at the same time, a composition of the CTMCs underlying these SWNs, at the global level.

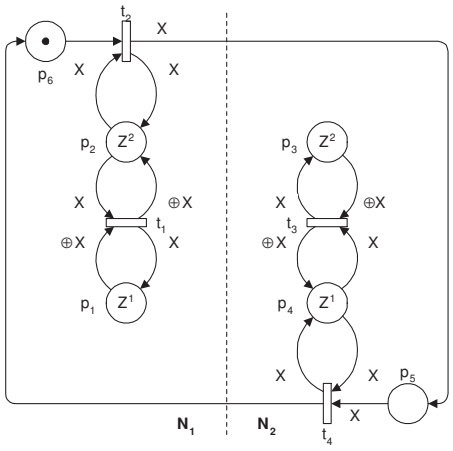

Fig. 2. Asynchronous composition of SWNs without combined aggregation and decomposition

2.2 The resolution problem

Given a system with external synchronization only, we can modelize it as a SWN N, *asynchronous composition* of SWNs $(N_k)_{k\in K}$ each N_k being a SWN model of an object class. Unfortunately, as a general rule, a direct extension of the GSPNs composition cannot be used *to solve* the initial CTMC because composition (i.e. cartesian product) of aggregates provided by the SRGs of the N_k *does not provide an aggregation of the whole CTMC of the model* verifying the strong lumpability condition (1). The main reason for this is that synchronization of *coloured* tokens is not preserved in such a direct composition.

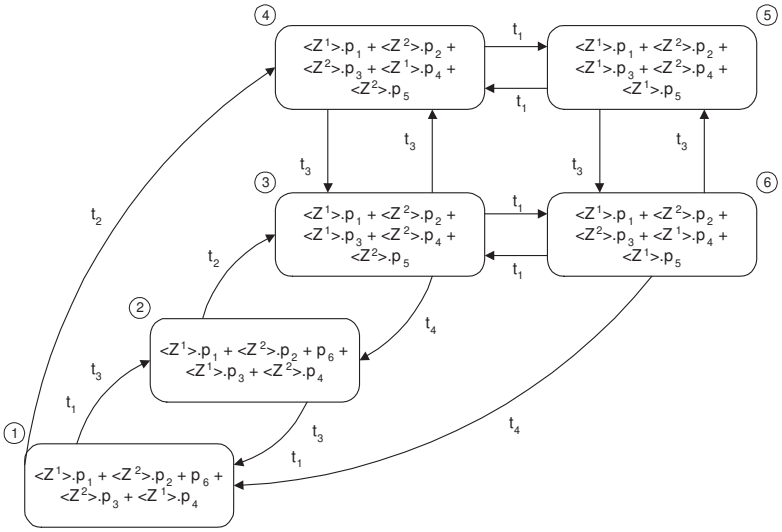

Fig. 3. SRG of the net of Fig. 2

Let us give an example of this problem with the SWN of Fig. 2. N is the composition of 2 SWNs N_1 and N_2. The colour domain of all places except p_6 and all transitions is a single ordered colour class C with $|C| = 2$. Firing of t_1 (t_3) exchanges the colour of places p_2 and p_1 (p_4 and p_3). Firing of t_2 provides the colour of p_2 to p_5 and firing of t_4 returns an non coloured token in p_6 when p_4 and p_5 have the same token colour. The SRG of N is given in Fig. 3: symbolic markings 1 and 2 are the only ones with one token in p_6 and they differ by the colour of tokens in p_2 and p_4 (same – Z^2 – or different – Z^1 and Z^2 – colours). Let us emphasis that *these two markings cannot be aggregated because the firing of t_2 produces the markings (3 and 4) from which t_4 is enabled (in 3) or nor (in 4)*. Figure 4 shows two subnets $\overline{N}_1$ and $\overline{N}_2$ which are extensions of N_1 and N_2 with an abstract view of the complementary net reduced to one place and one transition and Fig. 5 gives the SRGs of $\overline{N}_1$, $\overline{N}_2$ and their "product". The marking 12 corresponds to markings 1 *and* 2 in the original SRG. The key point is that *the abstract view does not catch the colour synchronization which will append in t_4*, and it is easy to see that no other abstract view could do it.

Furthermore, even if we can define a "synchronized product" of SRGs, we have to find how to express the label associated with an output transition, that is to say its rate, with information provided by these SRGs.

All of the following work consists of building nets in which the colour synchronization are never conditioned by earlier firings in more than one subnet. This is done through:

– the definition of an abstract view of each component (subnet),

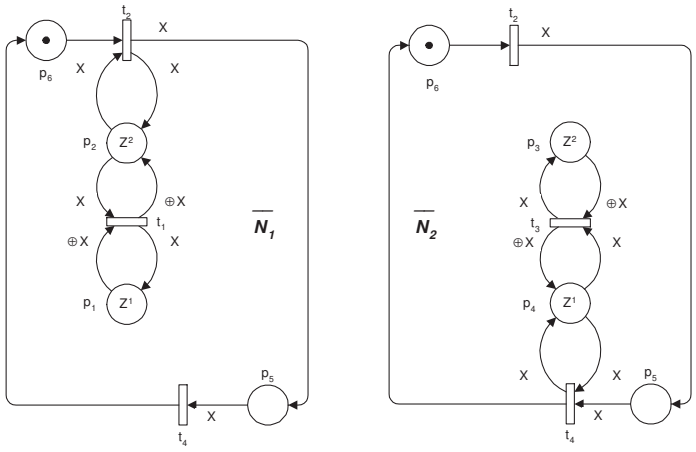

Fig. 4. Subnets $\overline{N_1}$ and $\overline{N_2}$ for the net of Fig. 2

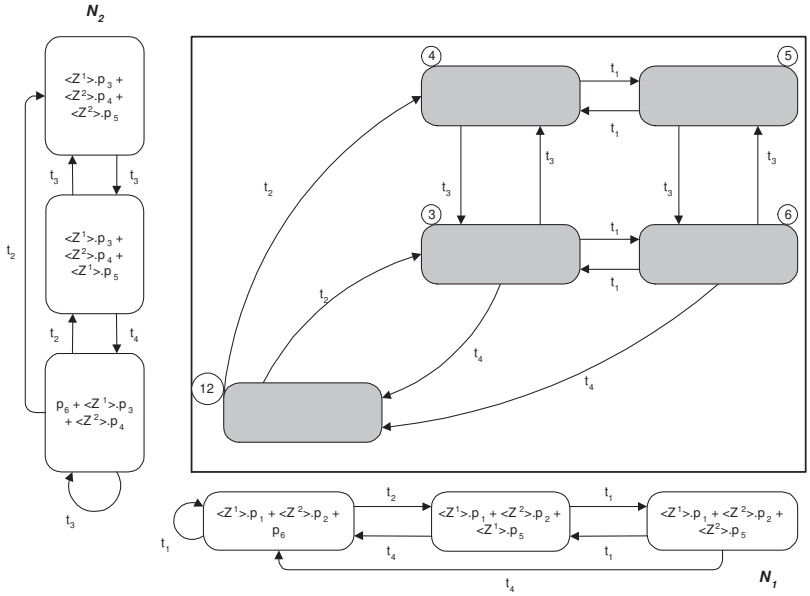

Fig. 5. SRGs of $\overline{N_1}$, $\overline{N_2}$ (Fig. 4) and their "product"

– the definition of *modified* – that is to say enlarged – SWNs (denoted $\overline{N_k}$) of the subsystems including *a representation of their environment* via abstract views of all other subnets.

From the SRGs of these SWNs we derive of a tensor expression of the generator of the model, and we compute performance measures with iterative methods.

3 Asynchronous composition of SWNs

In the present work we assume that each transition has exponential firing time, so that the stochastic process defined by N is a CTMC, and that the transition rate is marking independent (future work will partially relax these conditions).

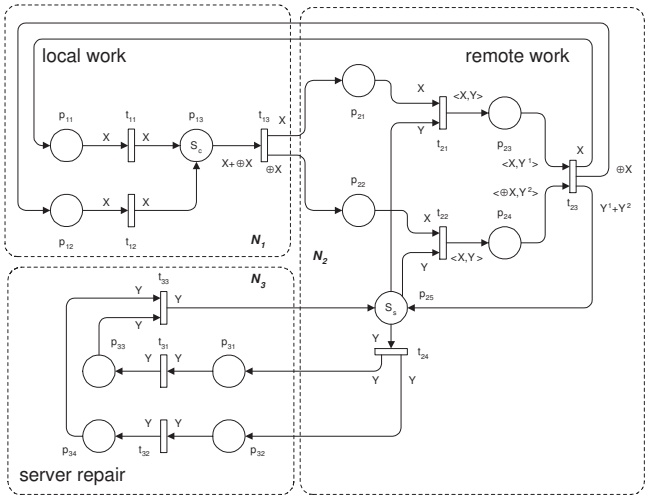

Fig. 6. Example of asynchronous composition of SWNs

We give in Fig. 6 an example of asynchronous composition of SWNs to which we shall refer throughout this section to explain our definitions and results. A brief explanation of the system is as follows: we have a client-server system in which clients are initially located in N_1 (place p_{13}) and servers in N_2 (place p_{25}). Clients emit a server request by pairs of neighbour (variables X and $\bigoplus X$, the client class C_c is ordered). The requests are treated in N_2 where servers (class C_s) execute the requests (transitions t_{21}, t_{22} and t_{23}). A server may fails: in this case it must be repaired with two jobs located in N_3 (transitions t_{31}, t_{32} and t_{33}).

The basic colour classes are hence C_c and C_s and the colour domains of each node (not shown in the figure for ease of reading) are: C_c for p_{11}, p_{12}, p_{13}, p_{21},

p_{22}, t_{11} ,t_{12} and t_{13}, C_s for p_{25}, p_{31}, p_{32}, p_{33}, p_{34}, t_{24}, t_{31} , t_{32} and t_{33}, $C_c \times C_s$ for p_{23}, p_{24}, t_{21} and t_{22}, $C_c \times C_s^2$ for t_{23}.

3.1 The abstraction process

As we have seen in previous sections, the first problem is to be able to define subnets $\overline{N_k}$ embedding N_k and *an aggregated view of its environment*. This means that the modelizer has to define an abstract view of each N_k which will be used in the description of the environment of other subnets.

An abstract view has to enforce a set of constraints:

- it allows to hide details of behaviour of the subnet.
- it is consistent, that is to say: if $M[\delta\rangle M'$ with a sequence of firings $\delta = \tau_1^* \sigma \tau_2^*$ composed of local firings τ_1^* and τ_2^* to be hidden, then we must have: $a(M)[\sigma\rangle a(M')$ with $a(M)$ the abstraction of the marking M.
- it must let "visible" interactions between global entities.
- it has to be formally defined from the net description.
- it is compatible with a combined aggregation/decomposition method.

To take into account the previous constraints, this abstraction should be *guided by qualitative considerations*, especially by observing interactions between global entities (colour classes) inside each subnet keeping these interactions "visible" in the abstraction.

Furthermore, more generally and unlike Buchholz, it does not seem possible to abstract a subnet with *only one place*: we need at least one place to modelize entities of each basic colour class moving from one subnet to another one.

We propose to define each place through a partial semiflow[2] to ensure consistency of the abstraction and deal with formal definitions deduced from the net description.

Definition 2. An *abstraction semiflow* f of N_k is a partial semiflow of N with respect to $T_k \backslash TO_k$ s.t.:

- there is a colour class C_i s.t. $C(f) = C_i$.
- $\forall p \in P_k \backslash PO_k$, f_p is 0 or b times a projection (with b a positive constant).

Let F_k be a set of abstraction semiflows of N_k. The *abstract view* of N_k w.r.t F_k is the set of places $PA_k = \{p_f \ ; \ f \in F_k\}$ with:

- $C(p_f) = C(f)$ (the colour domain of p_f).
- $M_0(p_f) = \sum_{p \in P_k \backslash PO_k} f(M_0(p))$ (the initial marking of p_f).

For any marking $M = (M_k)_{k \in K}$ of N, the *abstract marking* of M_k is $\mathrm{am}(M_k) = (\sum_{p \in P_k \backslash PO_k} f(M(p)))_{f \in F_k}$, also denoted $\mathrm{am}_k(M)$ or $M(PA_k)$ (the values of the semiflows of F_k in the marking M_k)

The *global colours of* N_k are $\{C(f) \ ; \ f \in F_k\}$.

[2] f is a partial semiflow on $N = (P, T, \ldots)$ w.r.t a set $T' \subseteq T$ of transitions iff f is a semiflow on the net $N' = (P, T', \ldots)$

The objects of the global classes of the (N_k) are the only ones which move between subnets. Let us note that a global class for N_k may be a non global one for another $N_{k'}$. Such classes may be renamed with different names in each N_k where it is non global. From now on, *we assume that such a renaming has been done.*

In our example SWN, we have two partial semiflows in N_2 ($f_{2c} = X.p_{21} + X.p_{23} + X.p_{22} + X.p_{24}$ and $f_{2s} = X.p_{25} + Y.p_{23} + Y.p_{24}$) and one partial semiflow in N_3 ($f_{3s} = X.p_{31} + X.p_{33}$).

We can now define the modified subnets allowing decomposition under appropriate conditions, set additional notations and define the aggregation function we shall use. In the sequel, the abstractions of the (N_k) are given.

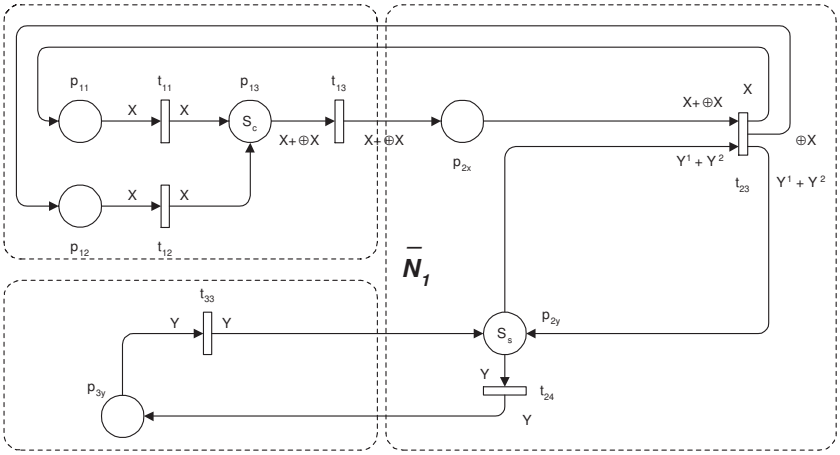

Fig. 7. Subnet $\overline{N}_1$ for the example SWN

Definition 3. Let N be an asynchronous composition of the $(N_k)_{k\in K}$. The *extension* $\overline{N}_k$ of N_k is the SWN $(\overline{P}_k, \overline{T}_k, \mathcal{C}, J, \overline{W}_k^-, \overline{W}_k^+, \Phi, \omega, \overline{M}_{0,k}, \theta_k)$ with:

- $\overline{P}_k = (P_k \backslash PO_k) \bigcup_{k' \neq k} PA_{k'}$ (For each marking M of N, the corresponding marking of $\overline{N}_k$ is $\overline{M}_k = (M_k, (M(PA_{k'}))_{k' \neq k})$).
- $\overline{T}_k = T_k \bigcup_{k' \neq k} TO_{k'}$
- $\forall p \in P_k \backslash PO_k$, $\forall t \in \overline{T}_k$, $\overline{W}_k^-(p,t) = W_k^-(p,t)$ and $\overline{W}_k^+(p,t) = W_k^+(p,t)$
- $\forall p_f \in PA_{k'}$, $\forall t \in \overline{T}_k$:
  $\overline{W}^-(p_f,t) = \sum_{p' \in {}^\bullet t} f_{p'} \circ W^-(p',t)$ and $\overline{W}^+(p_f,t) = \sum_{p' \in t^\bullet} f_{p'} \circ W^+(p',t)$

$\overline{M}_k(PA_{k'})$ is still named the *abstract marking* of $N_{k'}$ (in $\overline{M}_k$).

Fig. 7 shows the extension $\overline{N}_1$ of N_1. We see that in PA_2 we have only two places p_{2x} and p_{2y} for the colours C_c and C_s, and in PA_3 one place for C_s. The transitions t_{13}, t_{23}, t_{24} and t_{33} are also modified accordingly with our definition.

We could also define a full abstract view $\overline{N}$ of N with abstract views only of all subnets, useful in a hierarchical design process, however our method does not use $\overline{N}$.

Notations

- $\overline{SRS}_k$ (resp. $\overline{SRG}_k$) is the SRS (resp. the SRG) of $\overline{N}_k$,
- $\overline{XSRS} = \prod_{k\in K} \overline{SRS}_k$.
- $\overline{\mathcal{M}}_k$ is the symbolic marking of $\overline{M}_k$ in $\overline{SRS}_k$ and $\overline{\mathcal{M}} = (\overline{\mathcal{M}}_k)_{k\in K} \in \overline{XSRS}$.
- $\mathcal{D}(\overline{XSRS}) = \{M \mid \exists\, \overline{\mathcal{M}} \in \overline{XSRS} \text{ s.t. } \forall k \in K,\ \overline{M}_k \in \overline{\mathcal{M}}_k\}$

Definition 4. Let N be an asynchronous composition of the $(N_k)_{k\in K}$. The aggregation function $\mathcal{A}$ is: $\mathcal{A}(M) = (\overline{\mathcal{M}}_k)_{k\in K} = \overline{\mathcal{M}} \in \overline{XSRS}$

3.2 Syntactic conditions of aggregation

In order to be able to apply a combined aggregation/decomposition method of resolution, we have to add syntactic conditions to the model: roughly speaking, they mean that for each global colour class and every marking, the subsets of colours in the abstract views are a partition of this class.

We give a first set of conditions for which the algorithm proposed below may be used to compute performance measures of N. As usual with (stochastic) Petri nets, such conditions must be expressed at the *syntactic* level that is to say relative to either structural properties like colour domains, incidence function expressions, ... or to properties which may be checked only using these structural properties like semiflows, ..., to avoid checking of the RG of N.

Definition 5. We say that N fulfills the syntactic conditions of aggregation iff $\forall k \in K$ we have the following properties:

1. $\forall p \in {}^\bullet TO_k$ with $C(p) = \prod_{i\in I} C_i^{e_i}$, $\forall i \in I$ s.t. C_i is a global colour of N_k and $e_i > 0$ we have: $\forall 1 \le j \le e_i$, X_i^j (the jth projection on C_i) is in one abstraction semiflow $f \in F_k$.
2. $\forall t \in TO_k$ $\forall X \in Var(t)$ corresponding to a global colour class of N_k, X is in a positive term of some $W^-(p,t)$.
3. $\forall p \in PA_k$ with $C(p) = C_i$, $\forall k' \neq k$, $\forall p' \in P_{k'}$ with $C(p') = \prod_{i\in I} C_i^{e'_i}$:
   $\forall\, 1 \le j' \le e'_i$ there is a semiflow $g = (g_q)_{q\in \overline{P}_{k'}}$ on C_i in $\overline{N}_{k'}$, s.t. $\forall M\ g(M) = S_i$ with:
   $$g_q = \begin{cases} 0 \text{ or a projection} & \text{if } q \neq p, q \neq p' \\ Identity & \text{if } q = p \\ j'\text{th projection} & \text{if } q = p' \end{cases}$$

Condition 1 ensures that the abstract view of N_k is not too coarse and memorizes colours moving between components: the basic colour classes involved in

the firings of $t \in TO_k$ have to be in the colour domain of PA_k. Condition 2 means that the firing colours of $t \in TO_k$ must be in the input places of t. At last, condition 3 implies that in each marking, we have a partition of each occurrence of each global colour C_i, between colours usable to fire t and colours in other subnets $N_{k'}$.

It is easy to show that our example SWN fulfills these conditions.

Let us notice these conditions are fulfilled for many nets which are models of systems with several components each having some kind of autonomy.

3.3 Performance measures computation algorithm

As in the synchronous case, the basis of the algorithm is a combination of the tensor expression of the generator Q of the CTMC of the synchronized product of the $\overline{SRG}_k$ and of the regular computation for SWNs.

The CTMC transitions come from the firing of a transition $t \in T \backslash TO$ hence changing *only the kth component of the global state*, or from the firing of an output transition $t \in TO$ changing several components $k_1, \ldots, k_l$ of the state.

As transition firing rates depend only on the static subclasses of the chosen colors, the rate $r(t)$ from $\mathcal{M}$ to another $\mathcal{M}'$ sums the rates $r(t, d, \langle \lambda, \mu \rangle)$ of all symbolic firings $\langle \lambda, \mu \rangle$ of t which fit the choice d, for all given color static subclasses choices d. Moreover $r(t, d, \langle \lambda, \mu \rangle)$ is $\mathrm{card}(\mathcal{F})$ (also denoted $|\mathcal{F}|$) times the mean firing time $\theta(t, d)$ of t for d where $\mathcal{F}$ is the set of ordinary firings of t from any ordinary marking of $\mathcal{M}$ to ordinary markings of the symbolic marking $\mathcal{M}'$.

It can be shown then, that the matrix Q of the CTMC can be written as sub-matrix of

$$Q' = \bigoplus_{k=1}^{K} Q'_k + \sum_{t \in TO} \sum_{d} \theta(t, d) \left[\bigotimes_{k=1}^{K} C_k(t, d) - \bigotimes_{k=1}^{K} A_k(t, d) \right] \tag{2}$$

where $d = (d_i^j)_n^{e_i(t)}$ (with n_i the number of static subclasses of C_i and $1 \leq d_{ij} \leq n_i$) is a choice of static subclasses for the symbolic firings of t ($(a_i^j)_n^{b_i}$ denotes the tuple $(a_1^1, \ldots, a_1^{b_1}, \ldots, a_i^1, \ldots, a_i^{b_i}, \ldots, a_n^1, \ldots, a_n^{b_n})$).

The Q'_k matrices come from the generator of the CTMC of the $\overline{N}_k$ nets in isolation using only local transitions: they can be built from classical SWN technics, discarding any output transition effect.

The $\bigoplus$ operator means that these CTMCs are independent stochastic processes. The A_k and C_k matrices are obtained as consequence of output transition firings: any such firing produces a state change in each component $\overline{N}_k$ involved in the marking change.

By sub-matrix we mean that non zero terms (for same pair of states) of Q equal those of Q' and that if $\overline{\mathcal{M}}$ is reachable and $\overline{\mathcal{M}}'$ is unreachable then $q'_{\overline{\mathcal{M}}, \overline{\mathcal{M}}'} = 0$.

The $C_k(t, d)$ and $A_k(t, d)$ matrices are computed by the algorithm below.

Let us denote:

– $\forall\ \langle\lambda,\mu\rangle$ (instantiation functions in $\overline{N}_k$ for t), $\overline{\mathcal{M}}_k$ and $\overline{\mathcal{M}}'_k$ in $\overline{SRS}_k$:

$$1_{(t,d,\langle\lambda,\mu\rangle,\overline{\mathcal{M}}_k,\overline{\mathcal{M}}'_k)} = \begin{cases} 1 & \text{if} \quad d = (d(Z_i^{\lambda_i(j)}))_n^{e_i(t)} \\ & \text{and} \quad \overline{\mathcal{M}}_k[t(\langle\lambda,\mu\rangle)\rangle\overline{\mathcal{M}}'_k \\ 0 & \text{else} \end{cases}$$

$$1_{(t,d,\overline{\mathcal{M}}_k,\overline{\mathcal{M}}'_k)} = \bigvee_{\langle\lambda,\mu\rangle} 1_{(t,d,\langle\lambda,\mu\rangle,\overline{\mathcal{M}}_k,\overline{\mathcal{M}}'_k)}$$

with $\bigvee$ denoting the Boolean addition (logical or).

– for $t \in TO_k$

$$F_{(\langle\lambda,\mu\rangle,\overline{\mathcal{M}}_k,\overline{\mathcal{M}}'_k)} = \prod_{i=1}^{h}\prod_{j=1}^{m_i} \frac{\mathrm{card}(Z_i^j)!}{(\mathrm{card}(Z_i^j)-\mu_i^j)!}$$

with h the highest index of non ordered basic colour classes of $C(t)$.

Then we have the following algorithm.

Algorithm:

1. for each $k \in K$ compute $\overline{SRG}_k$ (hence $\overline{SRS}_k$) ($r_k = |\overline{SRS}_k|$)
2. for each $k \in K$ compute the Q'_k matrices from $\overline{SRG}_k$, using only local transitions
3. for each $t \in TO$ (say $t \in TO_k$)
   for each $h \in K$ compute $C_h(t,d)$ and $A_h(t,d)$ from $\overline{SRG}_h$:
   if $h \neq k$ and $t^\bullet \bigcap P_h = \emptyset$ then $C_h(t,d) = A_h(t,d) = I_{r_h}$
   if $h = k$ then $c_h(t,d)_{\overline{\mathcal{M}}_h,\overline{\mathcal{M}}'_h} = \sum_{\langle\lambda,\mu\rangle} 1_{(t,d,\langle\lambda,\mu\rangle,\overline{\mathcal{M}}_h,\overline{\mathcal{M}}'_h)} F_{(\langle\lambda,\mu\rangle,\overline{\mathcal{M}}_h,\overline{\mathcal{M}}'_h)}$
   if $h \neq k$ and $t^\bullet \bigcap P_h \neq \emptyset$ then $c_h(t,d)_{\overline{\mathcal{M}}_h,\overline{\mathcal{M}}'_h} = 1_{(t,d,\overline{\mathcal{M}}_h,\overline{\mathcal{M}}'_h)}$
   $a_h(t,d)_{\overline{\mathcal{M}}_h,\overline{\mathcal{M}}'_h} = \begin{cases} \sum_{\overline{\mathcal{M}}''_h} c_h(t,d)_{\overline{\mathcal{M}}_h,\overline{\mathcal{M}}''_h} & \text{if} \quad \overline{\mathcal{M}}_h = \overline{\mathcal{M}}'_h \\ 0 & \text{else} \end{cases}$
4. compute the performance measure using the tensor expression of Q'.

To use the tensor expression of Q', the numerical method computing a given measure has to verify the conditions:

– only linear functions of products $V.Q^m$ are used, with V a vector
– no unreachable state is involved in the computation

Let us emphasize that such computations never use Q' directly but instead the Q'_k, C_k and A_k matrices.

An important example of performance measure is the steady state probability distribution vector of the aggregated CTMC. We can then use iterative methods like the power method or the faster GMRES method ([12]) to compute these probabilities with the proposed algorithm. However, to ensure the above conditions, the initial vector must have non zero components for reachable states only: we can for instance, choose the vector $v^{(0)}$ with $v^{(0)}_{\mathcal{M}_0} = 1$ and $v^{(0)}_{\mathcal{M}} = 0$ if $\mathcal{M} \neq \mathcal{M}_0$.

We state in the following theorem the results giving the correctness of the algorithm.

Theorem 6. *Let N be an asynchronous composition of the $(N_k)_{k\in K}$, then:*

1. *$RS \subseteq \mathcal{D}(\overline{XSRS})$*
2. *the function $\mathcal{A}$ defined by $\mathcal{A}(M) = (\overline{\mathcal{M}}_k)_{k\in K} = \overline{\mathcal{M}} \in \overline{XSRS}$ is an aggregation function verifying the strong lumpability condition (1)*
3. *the transition rate from any marking $\overline{\mathcal{M}}$ of $\overline{XSRS}$ is given in the previous algorithm.*

The proof is given in App.A.

Conclusion

In this paper we have shown how to combine two methods, aggregation and decomposition, to deal with the increasing complexity of parallel systems. The studied systems are composed of subsystems communicating via entities moving between them. Decomposition expresses the state space of the system as a cartesian product of smaller spaces. Aggregation reduces the state space by grouping states and solving the Markov chain on the set of state classes; the SWN model moreover enables the reduction directly from the net description.

We have shown that in the case of internal synchronization, we have to choose between either decomposition or aggregation to specify the studied system. Now, in the case of external synchronization, we have shown that it is the synchronization memory phenomenon which allows or disallows the merging of the two methods to solve its Markov chain.

We have given a new method, allowing a combined aggregation–composition approach: we construct new subnets, built from original ones and abstract views of the others to deal with synchronization memory – capturing the synchronization memory – and we apply the decomposition approach on their SRGs. We have established a set of syntactic conditions under which such a method can be used.

Future work will extend the results to SWNs with immediate transitions and will experiment the method for large nets with a Petri net tool ([4]).

A Proof of the algorithm

The proof[3] of theorem 6 is established in several steps:

- definition of a set of semantic conditions, that is to say at the marking level (definition 8) for which
- we prove that they verify the strong lumpability condition (1) (theorem 10 based upon lemma 9).
- proof that the syntactic conditions of aggregation imply those semantic conditions (theorem 11).
- proof that the generator of the aggregated CTMC is a submatrix of Q' given in the algorithm.

[3] This section may be skipped by readers how have no interest in theoretical developments

We use a semantic intermediate level in this proof. The reason of which is threefold:

- the semantic conditions allow a structured definition of syntactic conditions: for each of the first one, we try to find a syntactic translation.
- given the same set of semantic conditions, we shall be able to find other sets of syntactic conditions, for special classes of nets, reducing the new proof to the fact that syntactic conditions imply semantic conditions.
- the correctness of the expression of Q' depends on syntactic, not semantic, conditions.

A.1 Semantic conditions of aggregation

We now exhibit the marking level conditions under which we can define a combined aggregation and decomposition. From the above aggregation function definition, two markings M and M' will be in the same aggregate if there is a K-tuple of admissible permutations s_k such that $\forall k \in K$, $s_k(\overline{M}_k) = M'_k$. However, for any marking, the (s_k) must be equal on the markings of the common abstract places of the $(\overline{N}_k)$ so we introduce the definition:

Definition 7. Two admissible permutations s and r are *compatible w.r.t.* M_k (or $\overline{M}_k$) iff $s(\mathrm{am}(M_k)) = r(\mathrm{am}(M_k))$.
A *compatible family of permutations* $s = (s_k)_{k\in K}$ w.r.t M is a family of admissible permutations s_k s.t. for any $k, k', k"$, s_k and $s_{k'}$ are compatible w.r.t. $M_{k"}$; for such a family we denote $s(M) = (s_k(M_k))_{k\in K}$.

Definition 8. We say that N fulfills the semantic conditions of aggregation iff for every k and every reachable marking $\overline{M}_k$ we can find a subset $C_{i,k}$ of every $C_i \in \mathcal{C}$ with the following properties:

1. for every admissible permutation s:
   $s(\mathrm{am}(\overline{M}_k)) = \mathrm{am}(\overline{M}_k) \Rightarrow (\forall i \in I\ s(C_{i,k}) = C_{i,k})$.
2. $\forall\ t \in TO_k$: if $\overline{M}_k[t(c)\rangle$ then c is composed of colours of $(C_{i,k})_{i\in I}$.
3. for every compatible permutations r and s w.r.t $\overline{M}_k$ s.t. $\forall i \in I\ r_{|C_i \setminus C_{i,k}} = s_{|C_i \setminus C_{i,k}}$, we have:
   $\forall k', \forall \overline{M}'_{k'}$: $\overline{M}'_{k'}(PA_k) = \mathrm{am}(\overline{M}_k) \Rightarrow r(\overline{M}'_{k'}) = s(\overline{M}'_{k'})$.

Intuitively, condition 2 means that the $C_{i,k}$ are the set of basic colours enabling the firings of $t \in TO_k$. Condition 1 ensures that the abstract view of $\overline{M}_k$ is consistent with respect to firings of $t \in TO_k$. Condition 3 expresses the fact that only colours not in the $C_{i,k}$ are relevant outside M_k.

Lemma 9. *Let N fulfill the semantic conditions above and M s.t. $\forall k \in K$ $\overline{M}_k$ is reachable in $\overline{N}_k$. Let $s = (s_k)_{k\in K}$ be a compatible family of permutations w.r.t M.*

If $M[t(c)\rangle M'$ then there is a permutation r and a compatible family $q = (q_k)_{k\in K}$ w.r.t M' s.t. $s(M)[t(r(c))\rangle q(M')$, with r depending only upon t and s.

Proof. We have to find r and q such that $\forall k \in K$, $s_k(M_k)[t(r(c))\rangle q_k(M'_k)$. From the definition of the $(\overline{M}_k)_{k\in K}$ and the compatibility of the $(s_k)_{k\in K}$, it is equivalent to show: $\forall k \in K$, $s_k(\overline{M}_k)[t(r(c))\rangle q_k(\overline{M}'_k)$.

• *First case:* $t \in T_{k_0} \backslash TO_{k_0}$ is a local transition.

We have then $\forall k \neq k_0$ $\overline{M}'_k = \overline{M}_k$; and $t \notin T_k \Rightarrow s_k(\overline{M}_k)[t(s_{k_0}(c))\rangle s_k(\overline{M}_k) = s_k(\overline{M}'_k)$.

As s_{k_0} is admissible $s_{k_0}(\overline{M}_{k_0})[t(s_{k_0}(c))\rangle s_{k_0}(\overline{M'}_{k_0})$.
The result follows with $r = s_{k_0}$ and $\forall k \in K$ $q_k = s_k$ i.e. $q = s$.

• *Second case:* $t \in TO_{k_0}$ is an output transition.
We apply the semantic conditions with $s_{k_0}(\overline{M}_{k_0})$ and show that we can choose $r = s_{k_0}$.

Since s_{k_0} is admissible, $s_{k_0}(\overline{M}_{k_0})[t(s_{k_0}(c))\rangle s_{k_0}(\overline{M'}_{k_0})$

We now prove that for each $k \neq k_0$ we have $s_k(\overline{M}_k)[t(s_{k_0}(c))\rangle q_k(\overline{M}'_k)$ with adapted q_k.

Let us denote $M'' = s_{k_0}(M)$ and $(C_{i,k_0})_{i\in I}$ the subsets from the semantic conditions w.r.t. M''_{k_0}.

For each $k \neq k_0$, let us denote $u_k = s_k \circ s_{k_0}^{-1}$. As s is a compatible family and by definition of $\overline{M}''_{k_0}$ and u_k, we have[4]: $u_k(\mathrm{am}(\overline{M}''_{k_0})) = \mathrm{am}(u_k(\overline{M}''_{k_0})) = \mathrm{am}(s_k(\overline{M}_{k_0})) = \mathrm{am}(s_{k_0}(\overline{M}_{k_0})) = \mathrm{am}(\overline{M}''_{k_0})$. So we have, from semantic condition 1, $u_k(C_{i,k_0}) = C_{i,k_0}$ for every i.

Let us now define the permutation v_k:
$\forall i \in I$, $v_{k|C_{i,k_0}} = Id_{C_{i,k_0}}$ and $v_{k|C_i \backslash C_{i,k_0}} = u_{k|C_i \backslash C_{i,k_0}}$. It is clear that v_k is admissible, hence so is $v_k \circ s_{k_0}$ and $v_k \circ s_{k_0}(\overline{M}_k)[\, t(v_k \circ s_{k_0}(c))\rangle\ v_k \circ s_{k_0}(\overline{M'}_k)$

From semantic condition 2, s_{k_0} is composed of colours from the C_{i,k_0}, hence, by definition of v_k, $v_k \circ s_{k_0}(c) = s_{k_0}(c)$.

We have for every $i \in I$, $v_{k|C_i \backslash C_{i,k_0}} = u_{k|C_i \backslash C_{i,k_0}}$. Applying the semantic condition 3, we get $v_k(\overline{M}''_k) = u_k(\overline{M}''_k)$, that is to say $v_k(\overline{M}''_k) = s_k(\overline{M}_k)$, and finally $s_k(\overline{M}_k)[t(s_{k_0}(c))\rangle q_k(\overline{M}'_k)$ with $q_k = v_k \circ s_{k_0}$ $(k \neq k_0)$ and $q_{k_0} = s_{k_0}$ $(q = (q_k)_{k\in K}$ is clearly a compatible family w.r.t $\overline{M}')$.

From this lemma we deduce the following theorem:

Theorem 10. *Let N fulfill the semantic conditions above. Then the aggregation function $\mathcal{A}$ fulfills the strong lumpability condition (1) on $\mathcal{D}(\overline{XSRS})$.*

Proof is omitted (see [9]).

A.2 From syntactic conditions to semantic conditions

Theorem 11. *Let N fulfill the syntactic conditions of aggregation. Then N fulfills the semantic conditions of aggregation.*

[4] from the definition of the flows $f \in F_k$, $s(am(M_k)) = am(s(M_k))$ for any admissible permutation s.

Sketch of proof (see [9] for a detailed proof): Let k and $\overline{M}_k$ be given. We define the partition of the semantic conditions as: $C_{i,k} = \{\text{colours of } C_i \text{ in am}(\overline{M}_k)\}$ (let us remark that $C_{i,k} = \emptyset$ for any non global colour of N_k).

We then prove successively the semantic condition 1 from the definition above, the second semantic condition from syntactic conditions 1 and 2 and the third semantic condition from syntactic condition 3.

A.3 Generator of the aggregated CTMC

The sketch of the proof is the following (see [9] for a detailed proof):

Let $\theta(\overline{\mathcal{M}}, \overline{\mathcal{M}'})$ be the rate from the reachable state $\overline{\mathcal{M}}$ to the reachable state $\overline{\mathcal{M}'}$ in the aggregated CTMC. We have $\theta(\overline{\mathcal{M}}, \overline{\mathcal{M}'}) = \sum_t \sum_d \theta(t,d)|U_{t,d}|$ with $U_{t,d} = \{\overline{\mathcal{M}} \xrightarrow{t(c)} \overline{\mathcal{M}}' \mid \overline{M'} \in \overline{\mathcal{M}'},\ d(c) = d\}$ and $\theta(t,d)$ the rate of the transition t for any colour with static partition d and $\overline{M}$ fixed in $\overline{\mathcal{M}}$.

For $t \notin TO$ it is clear that $\bigoplus_{k=1}^{K} Q'_k$ gives the correct rates.

For $t \in TO_h$ we rewrite $U_{t,d}$ with firing sets of the $(\overline{N}_k)_{k\in K}$ which leads to the expression of the algorithm.

B Flows, semiflows and implicit places in WNs

Flows are structural invariant of Petri nets: to each flow is associated a constant sum of weighed markings of places which gives information about the behaviour of the net. For coloured PN, the definition of flows uses *place colour functions* instead of constants hence the name *symbolic* flows.

Definition 12. Let N be a WN with places P and incidence matrix W (of linear functions from $\text{Bag}_{\mathbb{Q}}(\text{C(t)})$ to $\text{Bag}_{\mathbb{Q}}(\text{C(p)})$). Let A be a set. A (symbolic) flow of N on A is a vector $f = (f_p)_{p\in P} \neq 0$ of linear functions from $\text{Bag}_{\mathbb{Q}}(\text{C(p)})$ to $\text{Bag}_{\mathbb{Q}}(\text{A})$ s.t.:

$$\forall t \in T,\ \sum_{p\in P} f_p \circ W(p,t) = 0$$

For any reachable marking M, we have then:

$$\sum_{p\in P} f_p(M(p)) = \sum_{p\in P} f_p(M_0(p)) = \sum_{a\in A} \alpha(a).a \quad \text{(a constant)}$$

A semiflow is a flow with positive functions f_p: $\forall a \in A,\ \forall c \in \text{Bag(C(p))}$, $f_p(c)(a) \geq 0$

For WNs, following linear functions play an important role (for a place p with colour domain $\prod_{i\in I} C_i^{e_i}$ and a colour c of p):

- Identity: $Id(c) = c$ (and also $n.Id$ with n a constant number)
- Projections: for a colour class C_i the jth projection is $\text{Proj}_i^j(c) = c_i^j$

An implicit place with respect to a set P' of places, does not disable the firing of any transition for which preconditions are satisfied in P'.

Definition 13. A place p of a coloured Petri net $N = (P, T, \ldots)$ is implicit w.r.t. $P' \subseteq P$ ($p \notin P'$) iff:

1. there is a symbolic flow f with domain $C(p)$ s.t.: $f_p = Id$ and $\forall q \in P'$, $f_q < 0$
2. $\forall t \in T$, we have $\forall c \in C(t)$:

$$f_p(M_0(p)) + \sum_{q \in P'} f_q(M_0(q)) \;\geq\; f_p(W^-(p,t)(c)) + \sum_{q \in P'} f_q(W^-(q,t)(c))$$

An *implicit place* is an implicit place w.r.t some P'.

References

1. P. Buchholz. Hierarchies in colored GSPNs. In *Proc. of the 14th International Conference on Application and Theory of Petri Nets*, number 691 in LNCS, pages 106–125, Chicago, Illinois, USA, June 1993. Springer–Verlag.
2. P. Buchholz. Aggregation and reduction techniques for hierarchical GCSPN. In *Proc. of the 5th International Workshop on Petri Nets and Performance Models*, pages 216–225, Toulouse, France, October 19–22 1993. IEEE Computer Society Press.
3. G. Chiola, C. Dutheillet, G. Franceschinis, and S. Haddad. Stochastic well-formed colored nets and symmetric modeling applications. *IEEE Transactions on Computers*, 42(11):1343–1360, November 1993.
4. G. Chiola, G. Franceschinis, R. Gaeta, and M. Ribaudo. GreatSPN 1.7: GRaphical Editor and Analyser for Timed and Stochastic Petri nets. *Performance Evaluation*, 24(1,2):47–68, 1995.
5. M. Davio. Kronecker products and shuffle algebra. *IEEE Transactions on Computers*, 30(2):116–125, 1981.
6. S. Donatelli. Superposed generalized stochastic Petri nets: definition and efficient solution. In Robert Valette, editor, *Proc. of the 15th International Conference on Application and Theory of Petri Nets*, number 815 in LNCS, pages 258–277, Zaragoza, Spain, June 20–24 1994. Springer–Verlag.
7. C. Dutheillet. *Symétries dans les réseaux colorés. Définition, analyse et application à l'évaluation de performance.* Thèse, Université Paris VI, Paris, France, 28 janvier 1991.
8. S. Haddad and P. Moreaux. Evaluation of high level Petri nets by means of aggregation and decomposition. In *Proc. of the 6th International Workshop on Petri Nets and Performance Models*, pages 11–20, Durham, NC, USA, October 3–6 1995. IEEE Computer Society Press.
9. S. Haddad and P. Moreaux. Aggregation and decomposition for performance evaluation of asynchronous product of high level Petri nets. Document du Lamsade 102, LAMSADE, Université Paris Dauphine, Paris, France, May 1997.
10. J. G. Kemeny and J. L. Snell. *Finite Markov Chains.* V. Nostrand, Princeton, NJ, 1960.
11. B. Plateau and J.M. Fourneau. A methodology for solving Markov models of parallel systems. *Journal of parallel and distributed computing*, 12:370–387, 1991.
12. Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7:856–869, 1986.

Table of Contents

List of Figures