Decolourisation of Stochastic Symmetric Nets with Bags

Michal Žarnay
Transportation Networks Department
University of Žilina
Žilina, Slovak Republic
michal.zarnay@fri.uniza.sk

Manuel Silva
Computer Science and Systems Engineering Department
University of Zaragoza
Zaragoza, Spain
silva@unizar.es

***Abstract*—This paper deals with decolourisation of Stochastic Symmetric Nets with Bags (SSNB, a descendant of Well-Formed Nets), a class of coloured Petri nets (CPN), to generalised stochastic Petri nets (GSPN). In this process, the procedure for autonomous nets has been enriched to deal with bags, and supplemented by rules for adjustment of transition parameters (firing rates of timed and weights of immediate transitions) in order to decolourise a SSNB on the timed level as well, and be able to compute performance indices more efficiently.**

***Index Terms*—Stochastic Symmetric Nets with Bags, Generalized Stochastic Petri Nets, decolourisation**

I. Introduction

Coloured Petri nets (CPN) is a kind of high-level Petri nets for modelling of a discrete event system (DES), where through colours, identities are represented [1], [2]. They offer a natural description of systems with elements of various attributes. Compared to formalisms with undistinguished tokens, under symmetries on behaviours of components, CPN models lead to more compact and meaningful net structures.

In the effort to reduce the state space, we try to *decolourise* the original CPN to obtain a place/transition (P/T) net system. More precisely, remove colour identities as much as possible while creating a net with "analogous" structure and non-distinguished tokens, thus creating "populations" of entities. Doing this for timed models, performance results should be preserved. This process may be continued eventually by fluidification later to obtain a continuous P/T net system [3].

For timed CPN we use *Symmetric Nets with Bags* (SNB) [4] provided with a T-timed stochastic interpretation that we call Stochastic Symmetric Nets with Bags (SSNB). Their non-coloured counterpart is Generalized Stochastic Petri Nets (GSPN) [5].

II. Formalisms

A. Symmetric Nets with Bags (SNB)

Compared to usual components of Petri net classes, the Symmetric Nets class [4] (formerly called Well-Formed Nets, [2]) adds colour classes, domains and functions. It is a syntactical subclass of CPN [1]: it has a more strict definition of colour classes, but provides the same modelling power. Its enhancement with the use of *bags* – sets of coloured tokens manipulated together in the net system – leads to SNB [4] (here closely followed, including only necessary notation).

The set of *classes* of a Symmetric Net is denoted by $\{C_1, ..., C_k\}$. The partition of a class C_i is denoted $C_i = \biguplus_{q\in 1..s_i} C_{i,q}$ where s_i is the number of static *subclasses* of C_i.

Let C be a set. A *bag* over C, denoted $Bag(C)$, is a mapping a: $C \to \mathbb{N}$ such that the set, called *support* of a, $\|a\| = \{c \mid a(c) \neq 0\}$ is finite. Let $a, b \in Bag(C)$. Then $a \cup b$ is defined by $(a \cup b)(c) = a(c) + b(c)$ and $a \geq b$ holds iff $\forall c \in C$, $a(c) \geq b(c)$. When $a \geq b$, $a \setminus b$ is defined by $(a \setminus b)(c) = a(c) - b(c)$. The size of a is defined by $size(a) = \sum_{c\in C} a(c)$.

A *colour domain* D is a Cartesian product of classes and *sets of bags over classes*: $D = \bigotimes_{i=1,...,k}(C_i)^{e_i} \times \bigotimes_{i=1,...,k} Bag(C_i)^{e'_i}$, where e_i (e'_i) is the number of occurrences of class C_i ($Bag(C_i)$) in D. An item $d \in D$ is denoted by $d = \bigotimes_{i=1,...,k,j=1,...,e_i} c_i^j \times \bigotimes_{i=1,...,k,j=1,...,e'_i} b_i^j$ with $c_i^j \in C_i$ and $b_i^j \in Bag(C_i)$.

Basic colour functions (*for elements*) deal with colour domains and are defined $D \to Bag(C_i)$ ($\forall j$: $1 \leq j \leq e_i$, where applicable): $x^j_{C_i}(d) = c_i^j$, $x^j_{C_i}(d)$++ = the successor of c_i^j in C_i (C_i is supposed to be ordered) and $C_i.all(d) = \sum_{x\in C_i} x$ and $C_{i,q}.all(d) = \sum_{x\in C_{i,q}} x$.

Let $b_i^j = \sum_{x\in C_i} \alpha_x.x$. Then the *basic colour functions for bags* produce a single element which is a bag and are defined $D \to Bag(C_i)$ ($\forall j, j'$: $1 \leq j, j' \leq e'_i$, where applicable): $X^j_{Bag(C_i)}(d) = b_i^j$ denotes the function *dispatching* items of a bag, $\sim X^j_{Bag(C_i)}(d) = \sum_{x\in C_i|\alpha_x=0} 1.x$ the function producing the *complement* of a bag, $(X^j_{Bag(C_i)} \cup X^{j'}_{Bag(C_i)})(d) = \sum_{x\in C_i}(\alpha_x + \alpha'_x).x$ the *union* of two bags and $(X^j_{Bag(C_i)} \setminus X^{j'}_{Bag(C_i)})(d) = \sum_{x\in C_i} max(0, (\alpha_x - \alpha'_x)).x$ the *difference* between two bags.

To be able to manipulate with bags, we use an operator *whole* that, applied to a bag, produces a single bag containing it. The $whole_C(c)$ is the mapping $C \to Bag(C)$ defined by $c \in C$: $whole_C(c) = 1.\{1.c\} \in Bag(C)$, and $Bag(C) \to Bag(Bag(C))$ by $b \in Bag(C)$: $whole_C(b) = 1.b \in Bag(Bag(C))$.

Note: To distinguish, we use names starting with an *uppercase* letter for bag functions and with a *lowercase* letter for token functions in SNB models, while omitting the co-domain of colour functions. Furthermore, the *whole* colour operator may be used in colour functions composition like $whole \circ X$, and in this case it is denoted $whole(X)$. ■

A *class colour function* $f : D \to Bag(C_i)$ is a linear combination of basic colour functions for elements and for

bags such that $\forall d \in D, \forall c \in C_i, f(d)(c) \geq 0$. A colour function labelling an arc between a transition t and a place p is either a natural number or a tuple of colour functions with or without compositions of a class colour function $C(t) \rightarrow Bag(C_{\alpha'_i})$ and the $whole_{C_{\alpha'_i}}$ mapping.

A (basic) *guard* for bags is a boolean mapping defined on a colour domain D with $b_i^j = \sum_{c \in C_i} \alpha_c.c$. Let $\bowtie$ be either the $=$ or the $<$ relation. SNB guards are syntactically built with 3 basic colour functions for elements and 3 for bags. The former are $[x_{C_i}^{i_1} \bowtie x_{C_i}^{i_2}](c)$ (that equals true iff $c_i^{i_1} \bowtie c_i^{i_2}$), $[x_{C_i}^{i_1} = x_{C_i}^{i_2}{+}{+}](c)$ (true iff $c_i^{i_1}$ is the successor of $c_i^{i_2}$ in C_i) and $[x_{C_i}^{i_1} \in C_{i,q}](c)$ (true iff $c_i^{i_1}$ belongs to the static subclass $C_{i,q}$). The latter include $[Unique(X^j_{Bag(C_i)})](c)$ (true iff $\forall c \in C_i, \alpha_c \leq 1$), $[card(X^j_{Bag(C_i)}) \bowtie n](c)$ (true iff $size(X^j_{Bag(C_i)}(c)) \bowtie n$) and $[X^j_{Bag(C_i)} \bowtie X^{j'}_{Bag(C_i)}](c)$ (true iff $X^j_{Bag(C_i)}(c) \bowtie X^{j'}_{Bag(C_i)}(c)$). Guards in general are inductively defined from basic guards using operators $\vee$, $\wedge$ and $\neg$.

A *Symmetric Net with Bags* is a nine-tuple $\mathcal{N} = \langle P, T, \mathbf{Pre}, \mathbf{Post}, \mathbf{Inh}, \mathbf{pri}, Cl, C, \Phi \rangle$ where P is a finite non-empty set of places; T is a finite non-empty set of transitions, $T \cap P = \emptyset$; $Cl = \{C_1, ..., C_k\}$ is the set of classes, each being partitioned into s_i static sub-classes ($C_i = \uplus_{q=1..s_i} C_{i,q}$); C defines colour domain $C(s)$ (finite Cartesian product of classes and of bags of classes) for each node s; **Post** (**Pre**, **Inh**) is the forward (backward, inhibitor) incidence mapping associating, with each pair $(p,t) \in P \times T$, a general colour function for bags defined $C(t) \rightarrow Bag(C(p))$; **pri** is the priority mapping associating a number in $\mathbb{N}$ with each transition instance $C(t)$; Φ is a mapping associating a guard with each $t \in T$ (by default, $\Phi(t) = true$).

We use ${}^\bullet s$ ($s^\bullet$) to denote a set of *preceding* (*following*) nodes to node s. Symbol p° represents a set of transitions that follow place p via an *inhibitor* arc.

Extended conflict set (ECS) is a set of transitions created by transitive closure of symmetric structural conflict relation, which is relation of two transitions with the same priorities in structural or indirect structural conflict relation. For formal definitions, see p. 95 in [5].

We also use the following notation:

- The set of transitions T is partitioned to two disjoint subsets: T_T of *timed* and T_I of *immediate* transitions.
- The set of transitions T is partitioned to n subsets according to existing ECS-s in the net T_{ECS_i}, $i = 1..n$, $|T_{ECS_i}| \geq 1$, $\bigcup_{i=1}^{n} T_{ECS_i} = T$.
- $var(t)$ represents a set of all variables present on a transition t. We denote $var(p \rightarrow t)$ a set of variables on input and $var(p \leftarrow t)$ a set of variables on output arcs of a transition t ($var(p \rightarrow t) \cup var(p \leftarrow t) = var(t)$).
- $bexp_i(t)$ represents a set of all expressions composed from basic colour functions for bags present on a transition t.

Colour-safe place is a place that never contains two copies of the same tuple in any marking. Analogously, *colour-safe bag* is a bag that does not contain two copies of the same colour.

As an abbreviation of a colour expression, we also use an order function, denoted *ord*, for colour domain $D_C = C_1^{e_1} \times ... \times C_k^{e_k}$ and a tuple function $f : D_C \rightarrow Bag(D_C)$ and defined: $ord(x_i^j).f \equiv \sum_{1 \leq q \leq s_i} [x_i^j \in C_{i,q}] q.f$.

Bag-unfolding is unfolding of information about cardinality of bags in case this is not an exact number: every original transition with an uncertain cardinality of bag(s) in its colour domain is replaced by as many new transitions as there are possible values of cardinality of the bags. Similarly, every place that can contain bags of uncertain cardinality is replaced by a set of new places. The new nodes are connected with arcs according to the bag cardinalities. Similarly as unfolding of colours, the bag-unfolding operation enlarges the net structure. However, unlike by colours, the enlarging refers to cardinality of bags, not their content.

Ord-unfolding is unfolding of information from the *ord* function that can be used in transition guards or arcs: $ord(x)$ delivers order of value x in its colour set. Similarly to the bag-unfolding, it produces a set of transitions and arcs according to all potential values of the $ord(x)$ function in the given context.

B. Stochastic Symmetric Nets with Bags (SSNB)

SSNB, whose version without bags has been formerly called Stochastic Well-Formed Nets [2] is a t-timed extension of the SNB formalism. The type of timing allowed corresponds to timing of GSPN [5]: essentially exponential probability distribution functions on timed transitions and deterministic zero delay on immediate transitions. A *Stochastic Symmetric Net* is a ten-tuple $\mathcal{N} = \langle P, T, \mathbf{Pre}, \mathbf{Post}, \mathbf{Inh}, \mathbf{pri}, Cl, C, \Phi, \mathbf{w} \rangle$, where P, T, **Pre**, **Post**, **Inh**, **pri**, Cl, C and Φ are defined as in SNB, while $\mathbf{w}$ is a T-indexed vector of functions $\mathbf{w}[t]$: $C(t) \times \left(\bigotimes_{p \in P} Bag(C(p))\right) \rightarrow \mathbb{R}^+$, defining a *rate* for each (exponential) *timed* transition instance and a *weight* for each *immediate* transition instance. The transition rate/weight function may depend on the transition instance and it may be marking-dependent, satisfying certain restrictions [2].

If not stated otherwise, we assume *infinite server semantics* (ISS) for transitions, i.e. no limit on number of transition instances enabled at once, but *single server semantics* (SSS) per colour, i.e. the same transition instance is enabled only once, although available tokens may enable more of them at once ([6], p. 463). A parameter assigned to a timed transition in SSNB, e.g. λ, represents firing rate of one instance of the timed transition and all instances are assumed to have the same firing rate. This means that if there are n transition instances enabled, the resulting firing rate of the whole timed transition is the sum of rates of all enabled transition instances, i.e. $n.\lambda$ (for more details see [5], p. 110).

III. Decolourisation of SSNB

Our goal is to preserve not only *qualitative*, but also *quantitative* model properties between original coloured and obtained non-(less-)coloured models. Our focus is on decolourisation at structural level, not on decolourisation based on reachability graph (that naturally is more powerful).

Decolourisation procedure of SSNB comprises these steps:

1) Decolourisation as autonomous SNB to preserve qualitative properties of the model.

2) Unfolding of net parts that cannot be decolourised (reachability graph corresponds to Markov chain).
3) Adjustment of parameters of transitions in non-coloured net to preserve quantitative results of the model.

A. Decolourisation of Autonomous SNB

Based both on reachability graph (behaviour) and on net structure, the decolourisation of autonomous Symmetric nets has been introduced in [7]. The net structure approach is further developed as *reduction rules* for coloured nets in [8] and the detailed specification, to which we refer here, is given in [6].

Here, we introduce decolourisation of Symmetric Nets with Bags (SNB), i.e. we include the original procedure for autonomous Symmetric nets briefly and point out, what must be adjusted, so that it covers use of bags as well. In addition, there are a few details that we suggest to add in the original definition.

The structural decolourisation procedure consists of:

- full decolourisation:
  1) Identifying a colour flow in the coloured net
  2) Defining a colour shrinking function for it
  3) Projecting of the net over the colour flow
  4) Checking the flow in projected net for redundancy
  5) Decolourising the net with respect to the colour flow and the colour shrinking function
- partial decolourisation:
  6) elimination of a colour not immediately used
  7) elimination of a colour consumed by a transition

First, the full decolourisation steps 1-5 are performed iteratively until there are no more colour flows identified that would meet the conditions of the procedure. After that, the last obtained net is exposed to partial decolourisation steps 6-7 that look for net patterns meeting specific conditions.

1) Colour Flow Identification: The colour components of a place p are all the components of its colour domain $C(p)$. We denote $p \Downarrow_{C_i^j}$ the colour component of p corresponding to the j-th occurrence of C_i in $C(p)$. The colour components of transitions and arc expressions are defined analogically: $t \Downarrow_x$ represents the colour component of t corresponding to variable $x \in var(t)$, and $(p \to t) \Downarrow_{C_i^j}$ / $(p \leftarrow t) \Downarrow_{C_i^j}$ / $(p \multimap t) \Downarrow_{C_i^j}$ corresponds to the element which is, for each tuple in the formal sum defining the arc expression, in the same position in the expression associated with the input/output/inhibitor arc as C_i^j in the place p.

A colour flow $\mathcal{F}$ consists of a set $cf\_nodes(\mathcal{F})$ of nodes and a set $cf\_comp(\mathcal{F})$ of colour components for each node in $cf\_nodes(\mathcal{F})$. It is well-defined, when it satisfies a set of five properties ([6], def. 7.8, p. 256). We propose some modifications of properties 1 and 5. The first property can include also transitions succeeding places via inhibitor arcs (p°) to be as following: if $p \in cf\_nodes(\mathcal{F})$ then ${}^\bullet p \cup p^\bullet \cup p^\circ \subset cf\_nodes(\mathcal{F})$.

The last property of the same definition omits type of predicates $[d(x) = d(y)]$, since this is not defined in the SNB formalism. According to [9], the frequency of use of the predicate is minimal and it can be replaced by $x \in C_{i,q} \wedge y \in C_{i,q}$ for all involved static subclasses $C_{i,q}$. On the other hand, the predicate type for bags $X \bowtie Y$ is appended to the list of predicate types of the same property. The whole property is then: if $t \Downarrow_x \in cf\_comp(\mathcal{F})$ and a basic predicate of type $[x = y]$, $[x = y{+}{+}]$, or $[X \bowtie Y]$ appears in $\Phi(t)$, then $t \Downarrow_y \in cf\_comp(\mathcal{F})$.

2) Colour Shrinking Function Definition: A colour shrinking function sh does the conversion from colour elements in a basic colour class C_i to colour elements in a shrunk colour class C_i'. Because of the absence of predicate type $[d(x) = d(y)]$, the original definition for SN is formally amended for SNB: $sh : C_i \to C_i'$ such that $|C_i'| \leq |C_i|$ and C_i' has a number of static subclasses less than or equal to the one of C_i and $\forall c_1, c_2 \in C_i, \quad c_1 \in C_{i,q} \wedge c_2 \in C_{i,q} \Rightarrow sh(c_1) \in C'_{i,q} \wedge sh(c_2) \in C'_{i,q}$.

Colour shrinking function for bags produced by the operator *whole* from $Bag(C_i)$ could be analogous to the colour shrinking function for colour elements, however it makes very little sense decolourising such bags, since they exist in coloured models in order to carry information about grouping of coloured tokens. Hence, colour flows containing the bags with the operator should also contain synchronization expressions in transition guards preventing the flows to be decolourised. If the link between bags and ordinary colour elements of the same colour class C_i is missing in the model, then the flow characterised by the $Bag(C_i)$ colour components can be substituted by a new basic colour class $C_j = Bag(C_i)$ and every bag in the $Bag(C_i)$ by a colour element $c \in C_j$ (see the second example in the section III-B). The colour shrinking function sh can be then applied to C_j.

3) Net Projection over Colour Flow: Projection of the given SNB $\mathcal{N}$ over the identified colour flow $\mathcal{F}$ differs from projection of SN given in def. 7.12 in [6] in the specification of arc matrices $\mathbf{Pre}'$, $\mathbf{Post}'$ and $\mathbf{Inh}'$ - it includes bag expressions.

Let $\mathbf{Pre}[p,t] = \sum_k \delta_k \bigotimes_{C_i \in C} \bigotimes_{j=1,\ldots,e_i} \left\{\sum_{q=1}^{n_i} \alpha_{i,q}^j.C_{i,q}.all + \sum_{x \in var_i(t)}(\beta_x^j.x + \gamma_x^j.x{+}{+}) + \sum_{X \in bexp_i(t)} \theta_X^j.X\right\}$ be the arc expression associated with an arc $p \in cf\_nodes(\mathcal{F}) \to t \in cf\_nodes(\mathcal{F})$ in the original net, then $\mathbf{Pre}'[p,t] = \sum_k \delta_k \eta_k \bigotimes_{C_i \in C} \bigotimes_{j:p\Downarrow_{C_i^j} \in cf\_comp(\mathcal{F})} \left\{\sum_{q=1}^{n_i} \alpha_{i,q}^j.C_{i,q}.all + \sum_{x \in var_i(t)}(\beta_x^j.x + \gamma_x^j.x{+}{+}) + \sum_{X \in bexp_i(t)} \theta_X^j.X\right\}$ where $\eta_k = \prod_{C_i \in C} \prod_{j:p\Downarrow_{C_i^j} \notin cf\_comp(\mathcal{F})} \left|\sum_{q=1}^{n_i} \alpha_{i,q}^j.C_{i,q}.all + \sum_{x \in var_i(t)}(\beta_x^j.x + \gamma_x^j.x{+}{+}) + \sum_{X \in bexp_i(t)} \theta_X^j.X\right| = \prod_{C_i \in C} \prod_{j:p\Downarrow_{C_i^j} \notin cf\_comp(\mathcal{F})} \left(\sum_{q=1}^{n_i} \alpha_{i,q}^j.|C_{i,q}| + \sum_{x \in var_i(t)}(\beta_x^j + \gamma_x^j) + \sum_{X \in bexp_i(t)} \theta_X^j.|X|\right)$

Matrices $\mathbf{Post}'$ and $\mathbf{Inh}'$ are derived from $\mathbf{Post}$ and $\mathbf{Inh}$ in the same way as $\mathbf{Pre}'$ is derived from $\mathbf{Pre}$.

4) Redundancy Check of the Colour Flow: The key condition for decolourisation is that the colour flow $\mathcal{F}$ contains *no colour synchronization*, i.e. there are no structural constructs in the net projection over colour flow, that would allow distinguishing tokens of certain colours from other colours of the colour flow's domain in any reachable marking. In other words, if by permutation of colours of tokens, we can

get another marking $\mathbf{m}'$ from the current marking $\mathbf{m}$ in the projected net, $\mathbf{m}'$ must not be prevented by expressions in the projected net structure.

Redundancy of the flow $\mathcal{F}$ in the projected net $\mathcal{N}'$ is checked with respect to static subclass $C_{i,q}$. For each transition t of $\mathcal{N}'$, certain properties must be satisfied. We modify three out of four properties stated in the original definition, and add two more.

According to [6] (def. 7.13), property 1 allows presence of inhibitor arcs in the projected net with synchronization arc function $\langle C_{i,q}.all \rangle$ only. We propose to include inhibitor arcs with projection arc function $\langle x \rangle$ and bag functions as well, providing the inhibiting place is colour-safe. In addition, in case of bag functions, the adjacent transition's guards must specify their cardinality exactly (a single number) and the set of all possible bags from C_i must contain only colour-safe bags by default.

Properties 3 and 4 are to be modified to include bag variables. The property 3 is then formed as follows. Let x (X) be a variable that can be assigned an element (a bag) in $C_{i,q}$ ($C_{i,q}$ or $Bag(C_{i,q})$), i.e. the transition guard does not imply that $x \notin C_{i,q}$. No more than one occurrence of either x, $x{+}{+}$ or X is allowed in the labeling functions of all input arcs of t (otherwise there would be a colour synchronization between input places, that is not desirable). The property 4 handling constraints on guards includes apart from element variables x, y, z, etc. also bag variables X, Y, Z, etc.

We add the following constraints:

- In bag functions on arcs, no occurrence of difference $X \setminus Y$ and of complement of union $\sim(X \cup Y)$ of bags X and Y, providing they can contain tokens of the same colours ($X \cap Y \neq \emptyset$) in any marking, is allowed.
- No occurrence of the $ord(x)$ function is allowed in the place of constants determining number of tokens or bags, neither in guards, nor in arc expressions. (When the function is present, the net must be *ord-unfolded*.)

5) Decolourisation of the Model: Finally, we can perform the structural decolourisation of the SNB model $\mathcal{N}$ with respect to the minimal colour flow $\mathcal{F}$ and the shrinking function sh.

We enhance the third operation in the def. 7.15 ([6]) to include decolourisation of bags and inhibitor arcs. The proposed content is then: The arc expression components selected by $cf\_comp(\mathcal{F})$ are transformed as follows:

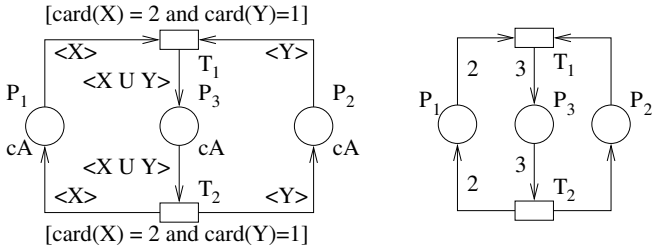
- On input or output arcs: x remains unchanged, $C_{i,q}.all$ is substituted by $\frac{|C_{i,q}|}{|C'_{i,q}|}C_{i,q}.all$, $C_i.all$ is substituted by $\sum_q \frac{|C_{i,q}|}{|C'_{i,q}|}C_i.all$ and X is replaced by $|X|.x$, where x is a result of the colour shrinking function.
- On inhibitor arcs: x is substituted by $|C_i|C_i.all$, $C_i.all$ is substituted by x, and X is substituted by its cardinality that is defined in the guard of the adjacent transition.

6) Elimination of a colour not immediately used [8]: This decolourisation rule applies when a new colour is "created" by a transition but it is not immediately used in a synchronization. The creation of the new colour can therefore be postponed.

7) Elimination of a colour consumed by a transition [8]: This reduction rule applies when a colour is "destroyed" (or swallowed) by a transition T, but it could have been already eliminated before by the firing that put tokens into the input place of T. The reduction anticipates the colour elimination of one step.

B. Illustration of Decolourisation of SNB

In the first example (fig. 1), use of union of bags is shown. Input bags X and Y, of prescribed cardinalities 2 and 1 respectively, are united by firing transition T_1. The three tokens introduced in place P_3 as union of the bags are undistinguished, and thus their distribution in bags X and Y by the following firing of transition T_2 follows one of several options. For instance for $\mathbf{m}(P_3) = \{a, b, c\}$, there are 3 options of tokens distribution in bags X and Y corresponding to the 3 possible partitions.

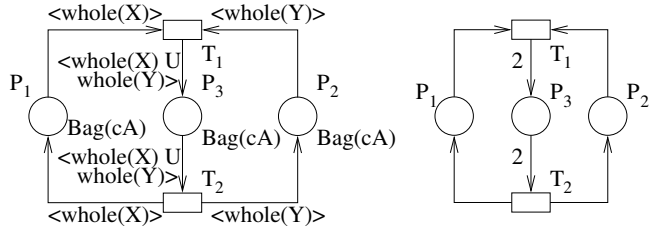


(a) Coloured net (b) Decolourised net

Fig. 1: Symmetric net with use of union of bags.

This model can be decolourised thanks to the prescribed sizes of all bags present in it. The bags in arc expressions are replaced by the number indicating their size and coloured tokens are replaced by non-coloured ones (fig. 1b).

In the second example (fig. 2), use of bags as *wholes* is shown ($whole(X)$ means that elements in the bag X are treated together as a unit). Every bag stays without changes in all transition firings, what represents the same behaviour as if the $Bag(cA)$ was replaced by a simple colour set, e.g. cB, and every bag from $Bag(cA)$ was substituted by a colour from the set cB and there was a bijection between $Bag(cA)$ and cB. Such a model can be then decolourised even according to rules for SNB without bags (fig. 2b).

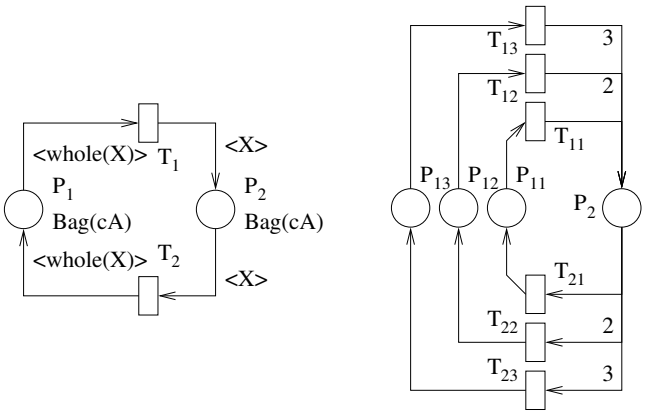


(a) Coloured net (b) Decolourised net

Fig. 2: Symmetric net with using bags only.

The third example (fig. 3) combines use of bags as wholes and as groups of elements. Unlike in the example in the fig. 1,

cardinality of the bag X is not determined here, so it can have from one element to as many elements as they are present in the initial marking of both places (if $\mathbf{m}(P_1) = \{\{a, b\}, \{c\}\}$ and $\mathbf{m}(P_2) = \emptyset$, then there are three elements circulating in the net). This corresponds also to a case, when the cardinality of the bag is given by inequality allowing more than one integer value (e.g. $card(X) \leq 2$). In this example, the net cannot be decolourised directly. It must be first *bag-unfolded*, i.e. every transition with such an inequality must be replaced by a set of transitions with all possible cardinalities of the bag expressed in equalities (three transitions for the given case: $card(X) = 1$, $card(X) = 2$ and $card(X) = 3$) and every place adjacent to such transition must be replaced by a set of places connected to the new transitions accordingly, and the obtained model can be then decolourised (fig. 3b).



(a) Coloured net (b) Decolourised net

Fig. 3: Symmetric net combining bags as wholes and as groups of elements.

C. Transition Parameters Adjustment Rules

Timing parameters in decolourisation have been first considered in [8] for two partial decolourisation rules named there CRR_2 and CRR_3 (our decolourisation steps 6 and 7 described above) and briefly also for rule CRR_1 (steps 1-5). It is the latter rule that we develop here to further details.

In order to preserve flow through transitions between original and decolourised net, the rules for transition parameters adjustment (TPA) seek to adjust *firing rates* of timed transitions and *weights* of immediate transitions according to existing ECS for transition instances in the SSNB. In some cases, firing semantics of timed transitions is also changed.

Since we do not follow the decolourisation process via reachability graph, the TPA rules are defined at net structural level: from net to net. In the form of simple net patterns, they define necessary changes in timing parameters of transitions so that rates of underlying Continuous-time Markov chain between original and decolourised models remain preserved. Every rule decides what is to be changed in firing rate or weight and firing semantics of the transition after transformation.

Explanation of each rule is based on the original coloured (or partially coloured) and non-coloured net patterns. Each pattern contains one transition for individual transformation, or more transitions for conflicts transformation. Every transition is assigned a parameter. For coloured transitions, the firing rate λ or weight w is *per one transition instance*, for non-coloured ones, firing rate μ or weight v relates to the *whole transition*. The patterns were chosen to be simple (minimal) and clearly describing the given aspect – they are easily extensible from the minimal configuration of 1 or 2 elements (places, transitions, variables, tuple elements) to a general one of n elements. At the same time, the patterns are meant to correspond with common net structures, i.e. they contain additional structural elements occurring in practical models, but not necessary for the description of the rule.

Each TPA rule depicts adjustment of transition parameters arising from one aspect. In practical models, the aspects may be combined in a single net pattern considered, what then expects application of more TPA rules on the same pattern.

In the TPA rules, we can distinguish two groups. Net patterns of rules 1-6, contain only one timed transition, since they are typically applied to timed transitions. They are valid for timed transitions that are persistent, and for immediate transitions. Net patterns of rules 7-8 containing 2 transitions in a conflict (outside of transitive structural conflicts), are designed for immediate transitions only.

When applying TPA rules, we consider transition parameters of coloured net and non-coloured net. The former will be called *coloured reference* (CR) net, the latter *non-coloured reference* (NR) net. If unfolding of any net part is necessary in the process from original coloured to non-coloured net, it is the *coloured net with the unfolded subnet* that is used as CR net.

While basic TPA rules are summarized in tables I and II, we provide comments to a few of them.

The **TPA rule** 1 deals with a variable $y \in var(t) - var(p \to t)$ that represents a token with any of possible values from colour set cB. Then in the CR net pattern, there are as many transition instances in conflict in the ISS as there are combinations of current marking of place $P1$ that will be assigned to variable x and all possible values of variable y from the colour set cB. Since each of the transition instances has the firing rate λ, the total transition rate in ISS is $\lambda.|cB|$. In the NR net, the transition T' is $\mathbf{m}(P'_1)$-enabled in ISS. As a result, its firing rate μ must be $|cB|$-times bigger than the firing rate λ of the coloured transition T. Server semantics of T' stays infinite, its flow is: $f(T') = \mu.\mathbf{m}(P_1)$.

Place P_2 with its input arc are, in fact, not necessary for description of this aspect and their omitting would not change the result.

The **TPA rule** 1*bis* is a special case of the previous one, where the new variable on output is decolourised in already partially decolourised net.

In the **TPA rule** 2, there are tokens from multiple places combined on the transition. Flow of T' depends on a product of current marking of all input places and the transition T'

TABLE I: TPA rules 1-4. In all the cases, places in ${}^\bullet T$ are assumed to be colour-safe.

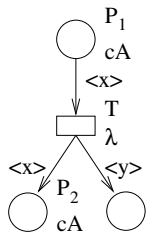
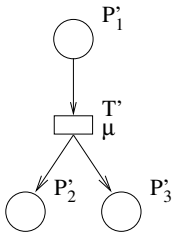
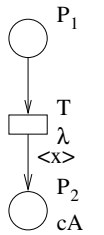
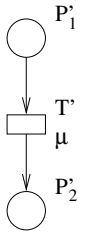
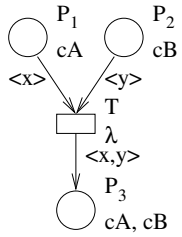
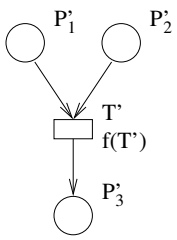
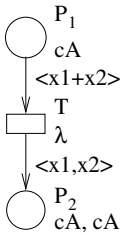
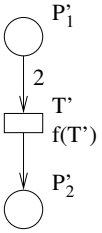
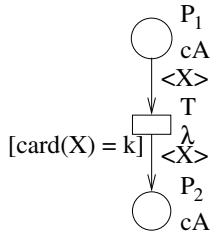
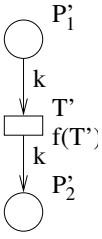
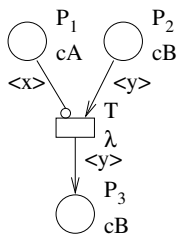
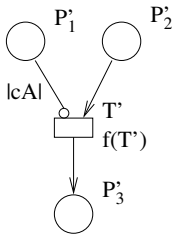
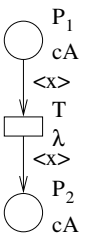
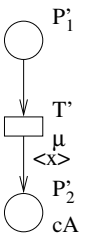
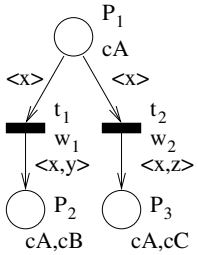
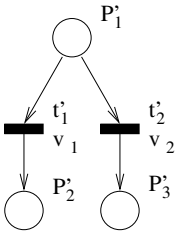
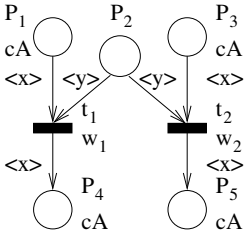
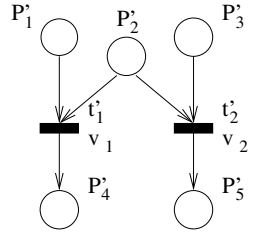
CR net	NR net
TPA rule 1 **(New Variable on Output)**	
	
$\mu = \lambda.\lvert cB\rvert \wedge f(T') = \mu.\mathbf{m}(P_1)$	
TPA rule 1*bis* **(Decolourisation of Output Only)**	
	
$\mu = \lambda.\lvert cA\rvert \wedge f(T') = \mu.\mathbf{m}(P_1)$	
TPA rule 2 **(Multiple Input Places)**	
	
$f(T') = \lambda.\mathbf{m}(P_1).\mathbf{m}(P_2)$	
TPA rule 3 **(Addition on Input Arc)**	
	
$f(T') = \lambda.\mathbf{m}(P_1).(\mathbf{m}(P_1) - 1)$	
TPA rule 4 **(Bag on Input Arc)**	
	
$f(T') = \lambda.\binom{\mathbf{m}(P_1)}{k}$	

TABLE II: TPA rules 5-8. In all the cases, places in ${}^\bullet T$ are assumed to be colour-safe.

CR net	NR net
TPA rule 5 **(Inhibitor arc)**	
	
$f(T') = \lambda.(\lvert cA\rvert - \mathbf{m}(P_1)).\mathbf{m}(P_2)$	
TPA rule 6 **(Decolourisation of Input Only)**	
	
$\mu = \frac{\lambda}{\lvert cA\rvert} \wedge f(T') = \mu.\mathbf{m}(P_1)$	
TPA rule 7 **(Free-choice Conflict with New Variables)**	
	
$v_1 = w_1.\lvert cB\rvert \wedge v_2 = w_2.\lvert cC\rvert$	
TPA rule 8 **(Non-free-choice Conflict)**	
	
$v_1 = w_1.\mathbf{m}(P_1).\mathbf{m}(P_2) \wedge v_2 = w_2.\mathbf{m}(P_3).\mathbf{m}(P_2)$	

requires the *product-based* semantics [3].

The **TPA rule** 3 deals with an addition of tokens from the same colour set on an input arc. The resulting marking-dependent flow will be built from number of possible *sequences of length k* of token colours on the arc from current marking of the place P_1. The sequences of length k result from the principle used in SNB where the order of assignment of colours to variables matters, e.g. assignments $x1 = A_1 \wedge x2 = A_2$ and $x1 = A_2 \wedge x2 = A_1$ are different (2 sequences of length 2). If the arc expression involves more than two variables, cor-

responding polynomial is derived from laws of combinatorics (for more details, see [10]).

If the differentiation is not needed, then the bags shall be used as described in the **TPA rule** 4, where variable X represents a bag formally defined as $X \in bag(cA)$ of cardinality specified in the guard of the transition T (in the discussed example: $k = 2$).

Note: The place P_2 in patterns related to last 2 rules has no influence and could be omitted. This also means that replacing the expression $\langle x1, x2\rangle$ by $\langle x1 + x2\rangle$ on the output arc and changing the colour domain of P_2 to single cA has no effect on the TPA rule 3.■

The **TPA rule** 8 is one of the 2 here considered, that are applicable to immediate transitions only, changing their corresponding weights respectively. Here, the number of transition instances in the coloured net depends on current marking in input places, resulting weights/flows are thus marking-dependent.

Note: Colour sets of input places P_1 and P_3 may have different colour domains. Output places P_4 and P_5 with adjacent arcs have no influence on the rule's effect and could be omitted.■

Table III shows some other examples of modifying arc weight for corresponding arc expressions in the TPA rule 8. In fact, the adjustment of transition parameters in this rule and the TPA rule 7 can be constructed according to parameters adjustment in the other TPA rules.

TABLE III: Weight modification for various arc expressions in the pattern of the TPA rule 8, when $C(P_2) = cB$.

Arc expression of $\mathbf{Pre}(P_2, t_1)$	modified weight v_1
$\langle y1 + y2\rangle$	$w_1.\mathbf{m}(P_1).\mathbf{m}(P_2).(\mathbf{m}(P_2) - 1)$
$\langle cB.all\rangle$	$w_1.\mathbf{m}(P_1).1$
$\langle X\rangle$ (bag with $card(X) = 2$)	$w_1.\mathbf{m}(P_1).\binom{\mathbf{m}(P_2)}{2}$

In all discussed rules, we expect that the *input places are colour-safe*. If they were not, the decolourisation on timed level with the goal of preserving performance results would not be successful. The cause of this is that we assume single server semantics for colours on a transition instance, what differs from infinite server semantics for instances of one transition (i.e. three tokens of the same colour in a single input place will enable a transition one time, while three tokens of different colours will enable it three times). To preserve performance results, we need to keep the information about which tokens have the same colours and which not. However by decolourisation, we lose it.

D. Transition Parameters Adjustment Algorithm

Using the outlined rules, the TPA algorithm is the following:

1) In the CR net, select a conflict set of immediate transitions $T_{proc} = T_{ECS_i} \cap T_I$ such that $|T_{proc}| \geq 2$ and it has not been processed yet. If there are no such sets, go to step 3.
2) If the conflict in the T_{ECS_i} is
   - a free-choice conflict, apply the TPA rule 7,
   - a non-free-choice conflict of two transitions, apply the TPA rule 8, if possible,
   - another conflict, save all of its transitions for later individual processing.

   In any case, go to step 1.
3) From the *timed* transitions and *immediate* transitions that are a part of an ECS not processed yet, choose one – denoted t. If there is no such a transition, *algorithm terminates*. Otherwise based on the presence of following aspects, include appropriate TPA rules in modification of the transition's flow:
   - If only input part of t is coloured in the CR net, consider the TPA rule 6.
   - If only output part of t is coloured in the CR net, consider the TPA rule 1*bis*.
   - If $var(t) - var(p \rightarrow t) \neq \emptyset$, i.e. there are variables that appear on output arcs only in the CR net, consider the TPA rule 1.
   - If there are at least 2 coloured input places of t in CR net, consider the TPA rule 2.
   - For every input arc of t with more than one variable in an addition in the CR net, consider the TPA rule 3.
   - For every input arc of t with a bag, consider the TPA rule 4.
   - For every input arc of t that is an inhibitor, consider the TPA rule 5.
4) Go to step 3.

The algorithm does not consider processing immediate transitions that do not form a part of any ECS, because although their behaviour (number of transition instances) after decolourisation may change, it does not influence performance results of the net. Also the non-persistent timed transitions are not included.

IV. Application Example

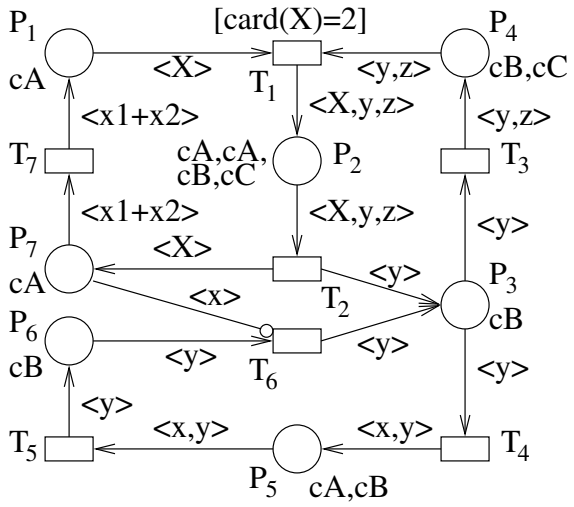
For illustration of introduced rules, we'll use a simple model in the fig. 4. Its coloured version uses three basic colour sets cA, cB and cC. Transitions T_i (T'_i) have firing rates λ_i (μ_i) in CR (NR) net.

On the level of autonomous net, the model is fully decolourised after finding four colour flows consecutively:

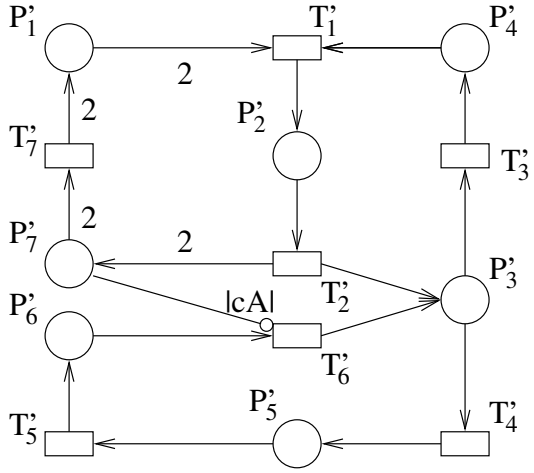
1) Nodes P_1, T_1, P_2, T_2 and T_6 with colour components related to cA and variables x, $x1$ and $x2$,
2) Nodes T_4, P_5 and T_5 with a colour component related to cA and variable x,
3) Nodes T_1, P_2, T_2, P_3, T_3, P_4, T_4, P_5, T_5, P_6 and T_6 with colour components related to cB and variable y,
4) Nodes T_3, P_4, T_1, P_2 and T_2 with colour components related to cC and variable z.

Parameters of transitions are adjusted as follows:

- T_1: TPA rules 2 and 4 (two input places P_1 and P_4 with a bag on one input arc) leading to a marking-dependent flow $f(T'_1) = \lambda_1.\binom{\mathbf{m}(P_1)}{2}.\mathbf{m}(P_4)$.
- T_2: no application of TPA rules: $f(T'_2) = \lambda_2.\mathbf{m}(P_2)$.



(a) CR net



(b) NR net

Fig. 4: Illustration model - flow definitions of 4 transitions change after decolourisation: $f(T'_1) = \lambda_1.\binom{\mathbf{m}(P_1)}{2}.\mathbf{m}(P_4)$, $f(T'_3) = \lambda_3.|cC|.\mathbf{m}(P_3)$, $f(T'_4) = \lambda_4.|cC|.\mathbf{m}(P_3)$, $f(T'_6) = \lambda_6.\mathbf{m}(P_6).(|cA| - \mathbf{m}(P_7))$ and $f(T'_7) = \lambda_7.\mathbf{m}(P_7).(\mathbf{m}(P_7) - 1)$.

- T_3: TPA rule 1 (one input place P_3 and one new variable z introduced on output arc): $f(T'_3) = \lambda_3.|cC|.\mathbf{m}(P_3)$.
- T_4: also TPA rule 1 (one input place P_3, new variable x on its output arc): $f(T'_4) = \lambda_4.|cC|.\mathbf{m}(P_3)$.
- T_5: no application of TPA rules: $f(T'_5) = \lambda_5.\mathbf{m}(P_5)$.
- T_6: TPA rule 5 (one input place P_6 and one inhibiting place P_1) leading to a marking-dependent flow $f(T'_6) = \lambda_6.\mathbf{m}(P_6).(|cA| - \mathbf{m}(P_7))$.
- T_7: TPA rule 3 (addition on the input arc) leading to a marking-dependent flow $f(T'_7) = \lambda_7.\mathbf{m}(P_7).(\mathbf{m}(P_7) - 1)$.

V. Conclusion

Contribution of this paper is twofold: enhancement of the decolourisation procedure of autonomous Symmetric Nets to include use of bags and specification of transition parameter adjustment rules for stochastic Symmetric Nets with Bags (SSNB).

In the former, several restrictions exist for direct application of decolourisation procedure on SNB, mainly involving some bag expressions on arcs and colour-safety condition of inhibitor places.

In the latter, we have found that existence of non-colour-safe places obstructs decolourisation of SSNB on timed level (although it may not be an obstacle on autonomous level), because losing the colour information, the precise adjustment formulas in decolourised nets cannot be constructed.

As for completeness, a set of rules is like a set of tools: *more tools make the operation more efficient*. However, if a toolkit is too big, a problem of managing it may occur. In this sense, the list of transition parameter adjustment rules may also be enlarged, if it is convenient and still manageable.

Future work will include formulation of TPA rules for non-persistent timed transitions.

Acknowledgment

Our work has been enriched by contributions of Giuliana Franceschinis, Fabrice Kordon and Laure Petrucci.

This work has been done during a post-doc stay of M. Žarnay in the University of Zaragoza. It has been partially supported by the Spanish project CICYT - FEDER DPI2006-15390 and the European Community project DISC (7FP, Grant Agr., INFSO-ICT-224498).

References

[1] K. Jensen and L. Kristensen, *Coloured Petri nets. Modelling and validation of concurrent systems*. Springer-Verlag, July 2009.

[2] G. Chiola, C. Dutheillet, G. Franceschinis, and S. Haddad, "Stochastic well-formed colored nets and multiprocessor modelling applications," in *High-level Petri nets, theory and application, LNCS*, K. Jensen and G. Rozenberg, Eds. Springer Verlag, 1991, pp. 504–530.

[3] M. Silva and L. Recalde, "Petri nets and integrality relaxations: A view of continuous Petri nets," *IEEE Trans on Systems Man and Cybernetics*, vol. 32, no. 4, pp. 314–327, 2002.

[4] S. Haddad, F. Kordon, L. Petrucci, J.-F. Pradat-Peyre, and N. Trèves, "Efficient state-based analysis by introducing bags in Petri nets color domains," in *28th American Control Conference (ACC'09)*. Omnipress IEEE Catalog, June 2009, pp. 5018–5025.

[5] M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis, *Modelling with generalised stochastic Petri nets*. John Wiley & Sons, 1995.

[6] G. Balbo and M. Silva, Eds., *Performance models for discrete event systems with synchronisations: formalisms and analysis techniques*. KRONOS, Zaragoza, 1998, MATCH - Human Capital and Mobility CHRX-CT94-0452 Draft for the Summer School, Jaca (Spain), 3-11 September, 1998.

[7] G. Chiola and G. Franceschinis, "Structural colour simplification in well-formed coloured nets," in *Proceedings 4th International Workshop on Petri Nets and Performance Models, Melbourne, Australia*, December 1991, pp. 144–153.

[8] M. Ajmone Marsan, S. Donatelli, G. Franceschinis, and F. Neri, "Reductions in GSPN and SWN: An overview and an example of application," in *Network Performance Modeling and Simulation*, J. Walrand, K. Bagchi, and G. Zobrist, Eds. Gordon and Breach, 1998, ch. 3, pp. 59–104.

[9] F. Kordon and L. Petrucci, "Phone conversation," 10 March 2010.

[10] P. J. Cameron, *Combinatorics, topics, techniques, algorithms*. Cambridge University Press, 1994.