Universität Dortmund

Fachbereich Informatik

Hierarchical Reachability Graph Generation for Petri Nets

Peter Buchholz, Peter Kemper

Forschungsbericht

Nr. 660/1997

Hierarchical Reachability Graph Generation for Petri Nets

Peter Buchholz and Peter Kemper

Informatik IV, Universität Dortmund
D-44221 Dortmund, Germany
e-mail: {buchholz,kemper}@ls4.informatik.uni-dortmund.de

Abstract

Reachability analysis is the most general approach to analyze Petri nets. Due to state space explosion, generation of the reachability set and reachability graph becomes intractable even for moderately sized nets with the known approaches. This paper presents a new method to generate and represent the reachability set and reachability graph of large Petri nets in a compositional and hierarchical way. The representation is related to previously known Kronecker based representations and it contains the complete information about reachable markings and possible transitions. The central idea of the new technique is a divide and conquer approach. Based on net-level results, nets are decomposed and reachability graphs for parts are generated and combined. The whole approach can be realized in a completely automated way and has been integrated in a Petri Net based analysis tool.

Keywords: Petri Nets, Reachability Set, Reachability Graph, Hierarchical Structure, Invariant Analysis

1 Introduction

Petri Nets (PNs) are an established formalism to describe and analyze dynamic systems. Among the large number of available analysis techniques, the generation of the set of all reachable markings and all possible transitions is the most general approach, which is theoretically applicable for every bounded net. The resulting graph is denoted as the reachability graph (RG) or occurrence graph. The set of reachable markings is denoted as the reachability set (RS). Reachable markings of the PN build the vertices of the graph and transitions describe the edges. Edges may be labeled with the corresponding transition identifier from the PN description. The RG contains the full information about the dynamic behavior of the PN and can be easily analyzed to gain results about the functional behavior as required for the verification of system properties. RGs are generated by an algorithm computing all successor markings for discovered markings, starting with the initial marking of the net. This approach is conceptually simple and is integrated in most software tools developed for the analysis of PNs. In practice, unfortunately, the size of RGs often grows exponentially with the size of the PN in terms of places and tokens. Hence RG generation is usable only for relatively small nets, much smaller than most practically relevant examples are.

Consequently, a large number of approaches has been published to increase the size of RGs which can be handled. A straightforward idea is to increase the available computing power and memory to increase the size of RGs. This is done by using powerful parallel or distributed computer architectures. Examples for this approach can be found in [1, 9] describing implementations on various parallel architectures and [14, 25], where workstation clusters are used for RG generation. These approaches describe RG exploration for Generalized Stochastic Petri Nets (GSPNs), however, they apply for RG exploration of PNs as well. The general problem of parallel/distributed state space generation is still, that an exponentially growing problem is attacked by increasing the available resources at most linearly. Additionally, the problem of an efficient parallelization of RG generation arises. Efficient realization of the RG generation algorithm in a distributed way is non-trivial since the different distributed tasks are dependent and require synchronization introducing additional overhead. In particular, the speedup that can be reached by a parallel implementation is model dependent which makes the problem of an efficient general purpose realization of parallel RG generation even harder.

An alternative to handle large RGs is to reduce their size without loosing relevant information. This idea can be exploited at two different levels. First, the net can be simplified by reducing the number of places and transitions. The corresponding approaches are denoted as reduction rules, published for uncolored PNs in [4] and subsequently for colored PNs (CPNs) in [16]. Reduction rules are defined with respect to the properties of interest. Thus, first properties need to be defined and then reduction rules which preserve these properties can be introduced, which yields a set of predefined rules for a set of predefined properties as in [4, 16]. The main drawback of reduction rules is that their applicability is restricted to relatively specific structures. Consequently, the gain obtained by reduction rules is for most nets relatively small and reduction rules can most times only be used as an a priori step which does not solve the problem of large RGs. The second approach to reduce the size of RGs is to perform the reduction at the level of reachable markings. Such an approach requires a compositional state space generation such that generation and reduction can be interleaved. Different techniques exploiting this idea exist. The usual way is to define the complete PN as a collection of interacting components. Usually component RGs are much smaller than the complete RG. Thus, RGs for the components are generated efficiently and are reduced according to some reduction rules which preserve relevant properties. Subsequently, reduced component state spaces are composed. In most approaches in this context, components interact via synchronized transitions. In [13], an approach for CPNs is introduced where RGs of components are generated in parallel by considering only local transitions. Additionally, a synchronization graph describing synchronized transitions is defined. By interleaving local and synchronized transition firing the complete RG can be generated or properties holding on the complete RG can be proved. Similarly in [29], complete component RGs are generated first, which are finally combined and reduced such that important properties like deadlocks or boundedness are preserved. In [34], a compositional analysis method for place-bordered subnets is presented. It is also based on the interleaving of composition and behavior preserving reduction. In [23], a different approach for components composed via synchronized transitions is proposed. The approach introduces a compact representation of the complete RG and an efficient way to characterize RS. The idea is that the incidence matrix characterizing RG can be composed via Kronecker operations from incidence matrices of component RGs and RS is a subset of the cross product of component reachability sets. Knowing RS and the RGs of the components, RS and RG of the CPN are completely characterized. In [5], an approach for hierarchical RG generation is proposed for hierarchically structured CPNs. Similar to the previous approach the RG is described using component RGs by composing incidence matrices via Kronecker operations. The approach requires that the complete net is structured in an appropriate way. The disadvantage of all these methods for efficient RG generation is that the component structure has to be defined by the modeler and all methods are very sensitive for the component structure. Techniques which reduce the size of RG by behavior preserving reduction depend on the required results. If relatively detailed results are required, most reduction fail or have only a small effect on the size of RG.

Other methods for efficient RG computation include the stubborn set method [35], which eliminates unnecessary interleavings from RG during generation, and the exploitation of symmetries to reduce RG [10, 22]. In both methods some additional computation is necessary during RG generation and the CPN has to observe several structural conditions that the methods can be used in an efficient way (i.e., to exploit symmetries, the CPN has to contain symmetric parts, otherwise the reduction has no effect). The idea of symmetry exploitation combined with a compact representation of RG by composing component RGs is described in [17] for quantitative analysis. Techniques based on ordered binary decision diagrams (OBDDs) rely on symmetries as well, in [30] Pastor et. al. describe OBDD-algorithms mainly for 1-bounded PNs.

Apart from techniques to characterize the complete or reduced RG in an efficient and compact way, several approaches to derive results without generating RS and RG exist. Usually these approaches yield only partial results in the sense that we can not formally prove results, we can only disprove some by finding failure states. These techniques include simulation and invariant analysis [22].

In this paper, we introduce an approach which is related to the work presented in [23] and [5]. RS and

RG are handled in a compositional way, which allows the representation and generation of large RSs/RGs. In contrast to other known methods performing compositional analysis, our approach represents the complete RG. Behavior preserving reduction is not applied. Consequently, arbitrary properties can be be checked on the resulting RG. However, it is also possible to combine the approach with behavior preserving reduction, although this is not considered in this paper. The proposed technique can be completely automated for a large class of PNs including all PNs which are covered by P-invariants. We present the approach here for uncolored PNs to simplify notation. Keeping in mind that every CPN with finite color sets can be unfolded to a uncolored PN [22], it obvious that the approach can be applied for a large class of CPNs too.

The structure of the paper is as follows. In Sect. 2, the PN class is defined, reachability and invariant analysis are introduced. Sect 3 describes the definition of regions which divide a PN into subnets. In Sect. 4 an abstraction operator is described which allows us to abstract from details in the net description to reduce the size of RS. Afterwards, Sect. 5, introduces a hierarchical and compositional representation of RS and RG. Then, different analysis approaches are proposed which exploit the hierarchical representation of RS and RG. Sect. 7 contains a non-trivial example to clarify the advantages of the new approach compared to conventional RG generation.

2 Basic Definitions and Known Results

We assume that the reader is familiar with PNs and the related basic concepts. For details about these fundamentals we refer to [21, 22, 28].

Definition 1 *A Petri net is a 5 tuple* $PN = (P, T, I^-, I^+, M^0)$, *where*

- $P = \{p_1, \ldots, p_n\}$ *is a finite and non-empty set of places,*
- $T = \{t_1, \ldots, t_m\}$ *is a finite and non-empty set of transitions* ($P \cap T = \emptyset$),
- $I^-, I^+ : P \times T \to \mathbb{N}$ *are the backward and forward incidence functions, and*
- $M^0 : P \to \mathbb{N}$ *is the initial marking.*

The initial marking is a special case of a marking $M : P \to \mathbb{N}$. A marking M can be interpreted as an integer (row) vector which includes per place p one element which describes the number of tokens on place p.

$\bullet t = \{p \in P | I^-(p,t) > 0\}$ gives the set of input places for a transition t, and $t\bullet = \{p \in P | I^+(p,t) > 0\}$ gives the set of output places. Analogously we define $\bullet p = \{t \in T | I^+(p,t) > 0\}$, $p\bullet = \{t \in T | I^-(p,t) > 0\}$. The notion can directly be extended to sets. In the sequel we consider connected nets, i.e. each place, transition has at least one incoming and one outgoing arc. Transition $t \in T$ is enabled in marking M, iff $\forall p \in P : M(p) \geq I^-(p,t)$. Any transition enabled in M can fire, changing the marking of any $p \in P$ to marking $M'(p) = M(p) + I^+(p,t) - I^-(p,t)$. This will be indicated by $M[t{>}\,M'$, $M[t{>}$ denotes that t is enabled in M and $M[\triangleright$ describes the set of enabled transitions in M. Considering firing sequences $s \in T^*$ yields to the definition of the language $L(PN) = \{s \in T^* | M^0[s{>}\}$ and $RS(PN) = \{M | \exists s \in T^* : M^0[s{>}\,M\}$, which is the set of all reachable markings for PN. The reachability graph $RG(PN)$ contains nodes for every $M \in RS(PN)$ and an arc $M \to M'$, if $M[t{>}\,M'$ for some $t \in T$. If necessary, arcs can be labeled with the corresponding transition and/or a transition rate as for stochastic Petri nets (SPNs). SPNs [27] extend the above class slightly by the association of exponentially distributed firing times with transitions. We define a function $W : T \times M \to \mathbb{R}_+$, where $\mathbb{R}_+$ is the set of non-negative numbers. $W(t,M)$ is the rate of an exponential distribution associated with transition t in marking M. We assume that $W(t,M) > 0.0$ if t is enabled in M. The RG of a SPN results from the RG of the corresponding PN by adding transition rates to the edges. RS is identical in both cases. SPNs can be used for performance analysis by analyzing the continuous time Markov chain described by the SPN [27].

The incidence matrix $\mathbf{C}$ is a matrix which contains for each place $p \in P$ a row and for each transition $t \in T$ a column such that $\mathbf{C}(p,t) = I^+(p,t) - I^-(p,t)$. It can be used to define net-level properties of a net PN.

Definition 2 *A vector* $\mathbf{x} \in \mathbb{Z}^n$ $\mathbf{x} \neq \mathbf{0}$ *is a P-invariant, if* $\mathbf{xC} = \mathbf{0}$. *A PN is covered by positive P-invariants, if for each place* $p \in P$, *a P-invariant* $\mathbf{x} \geq \mathbf{0}$ *with* $\mathbf{x}(p) > 0$ *exists.*

A vector $\mathbf{y} \in \mathbb{Z}^m$ $\mathbf{y} \neq \mathbf{0}$ *is a T-invariant, if* $\mathbf{Cy}^T = \mathbf{0}$. *A PN is covered by positive T-invariants, if for each transition* $t \in T$ *a T-invariant* $\mathbf{y} \geq \mathbf{0}$ *with* $\mathbf{y}(t) > 0$ *exists.*

An algorithm for computation of invariants is given in [26], although its time complexity is exponential for a worst case, usually invariant computation is much easier than generation of RS and RG. Incidence matrix and invariants ensure certain properties, however they do not completely characterize RS. The following theorem summarizes some classical results.

Theorem 1 *For a PN with a set of P-invariants X and a set of T-invariants Y the following results hold.*

- *If marking* M' *is reachable from marking* M, *then an integer vector* $\mathbf{z}$ *exists such that* $M' = M + \mathbf{Cz}^T$. *This implies that for every* $M \in RS(PN)$ *an integer vector* $\mathbf{z}_M$ *with* $M = M^0 + \mathbf{C}(\mathbf{z}_M)^T$ *exists.*
- *If* $x, x' \in X$ *and* $c, c' \in \mathbb{Z}$ *then* $c \cdot x + c' \cdot x' \in X$. *Analogously for* Y.
- *For each reachable marking* M *the relation* $M\mathbf{x}^T = M^0\mathbf{x}^T$ *has to hold for all* $\mathbf{x} \in X$.
- *If PN is covered by positive P-invariants, then it is bounded.*
- *If PN is bounded and live, then it is covered by positive T-invariants.*

Proof: Proofs can be found in standard books on PNs. ○

Although invariants offer some insight in the dynamic behavior of the modeled system, they are most times not sufficient to obtain the required results. Thus, RS and RG have to be generated for a detailed analysis. Usually, first RS is generated and the arcs of RG are computed in a second step. The following algorithm computes RS for a *PN*, it terminates if RS contains a finite number of markings[1].

generate_RS (PN)
    $\mathcal{RS} = \mathcal{U} = \{M^0\}$;
    while $(\mathcal{U} \neq \emptyset)$ *do*
        remove M *from* $\mathcal{U}$;
        for all $t \in M[\triangleright$ *do*
            generate M' *with* $M[t > M'$;
            if $(M' \notin \mathcal{RS})$ *then*
                $\mathcal{U} = \mathcal{U} \cup M'$;
                $\mathcal{RS} = \mathcal{RS} \cup M'$;
            fi
        od
    od
end

Set $\mathcal{U}$ contains markings for which successors have not been generated, whereas $\mathcal{RS}$ contains all generated markings. For $\mathcal{U}$ a simple data structure like a queue or stack is sufficient since elements only have to be added and removed. For $\mathcal{RS}$ a data structure allowing an efficient membership test is necessary. Consequently, $\mathcal{RS}$ can be realized using an appropriate hash function or a tree like structure allowing a membership test with an effort logarithmic in the number of elements. The problem with hashing are possible collisions. It is usually very hard to avoid collisions for general PNs. Therefore, most software tools use binary trees for the generation of RS.

We briefly analyze the effort required for the generation of RG, when a binary tree is used to store $\mathcal{RS}$. Let n be the number of markings in RS and let $n \cdot d$ the number of arcs in RG. Hence the mean

[1] A slightly extended version catches infinite RS, see coverability graph construction in PN literature.

number of successors per marking is d. The time required for the generation of RS is in the order of $d\sum_{i=1}^{n}\log_2(i) = d\cdot\log_2(n!)$ which is approximately $d\cdot n\cdot\log_2(n)$. Additionally, memory limitations have to be taken into account. Even if more sophisticated data structures are used for RS, the number of markings which can be generated on a standard workstation lies between $150,000$ and $1,500,000$. For PNs including a large number of places, the value can be much smaller. In case of certain symmetries, ordered binary decision diagrams (OBDDs) are able to handle extremely large sizes of RS and RG (see [8] among others). Pastor et al [30] describe how OBDD-techniques can be applied for PNs. However, the use of OBDDs requires the existence of symmetries to yield a compact representation.

After RS has been generated, the arcs in RG are generated in a second step. RG can be represented by a $n\times n$ incidence matrix $\mathbf{Q}$. If transition identities and rates are not relevant, $\mathbf{Q}$ can be stored as a Boolean matrix. Otherwise $\mathbf{Q}$ has to include the required information.

3 Autonomous Regions in PNS

In this section, we define parts of a PN which will be latter substituted by a less detailed representation in an abstraction operation. These parts, which are denoted as regions, have a place border at the input and a transition output border. This is different from other hierarchical constructs in the PN area [22], where places or transitions are refined. However, the definition is natural from a behaviorally oriented point of view (see also [11]), because a region describes a part acting for its own. Communication is performed by receiving tokens from the environment (place bordered input) and sending tokens to the environment (transition bordered output).

Definition 3 *A subset $T_r\subseteq T$ of a set of transitions of $PN=(P,T,I^-,I^+,M^0)$ defines a subnet $PN_r=(P_r,T_r,I_r^-,I_r^+,M_r^0)$, where $P_r=\bullet T_r$ and I_r^-,I_r^+,M_r^0 are the corresponding functions of PN restricted to P_r and T_r, respectively. PN_r defines a region iff the input bags of transitions in T_r and in $T_x=T\setminus T_r$ are disjoint, i.e., $P_r\bullet=T_r$. For a region PN_r, the set of output transitions $T_r^{out}\subseteq T_r$ consists of all $t\in T_r$ such that $I^+(p,t)>0$ for some $p\notin P_r$. Analogously the set of input transition is $T_r^{in}=\{t\in T\backslash T_r|\exists p\in P_r: I^+(p,t)>0\}$.*

A region describes an autonomous part of a PN, which will be used to define a hierarchical structure. A region is minimal if it contains no region as a proper subset[2]. The concept is illustrated by the following example, which will serve as a running example to accomplish the line of argumentation.

Example 1 *We consider a producer/consumer model where a producer A successively fills two buffers B1 and B2. Fig. 1 shows the corresponding PN, where places $\{p1,p2,p3\}$ describe the state of the producer. Places $\{p5,p7\}$ are buffer places, whose capacity is limited by places $\{p4,p6\}$. Buffer B2 is always filled with two items/tokens at once, while B1 obtains single tokens. The model contains two consumers of equal behavior. A consumer non-deterministically takes tokens from each buffer, but the first buffer is only considered if two consumers are willing to consume. Places $\{p8,p9,p10\}$ give the state of both consumers. The model is clearly artificial and just intended to illustrate our concepts. Minimal regions in this model are shown in Fig. 1 by shaded polygons.*

Proposition 1 *For a PN with regions $N_{r1},N_{r2},\dots$*

1. *minimal regions are disjoint, i.e. if N_{r1}, N_{r2} are minimal and $N_{r1}\neq N_{r2}$ then $P_{r1}\cap P_{r2}=\emptyset$ and $T_{r1}\cap T_{r2}=\emptyset$,*
2. *minimal regions define a partition, i.e. for each $p\in P$ ($t\in T$) exists exactly one minimal region N_{ri} with $p\in P_{ri}$ ($t\in T_{ri}$),*
3. *regions are closed under union, i.e. if N_{r1}, N_{r2} are regions then $T_{r1}\cup T_{r2}$ defines a subnet N_r which is a region.*

[2]Minimal regions coincide with the equivalence relation of the conflict relation [33].

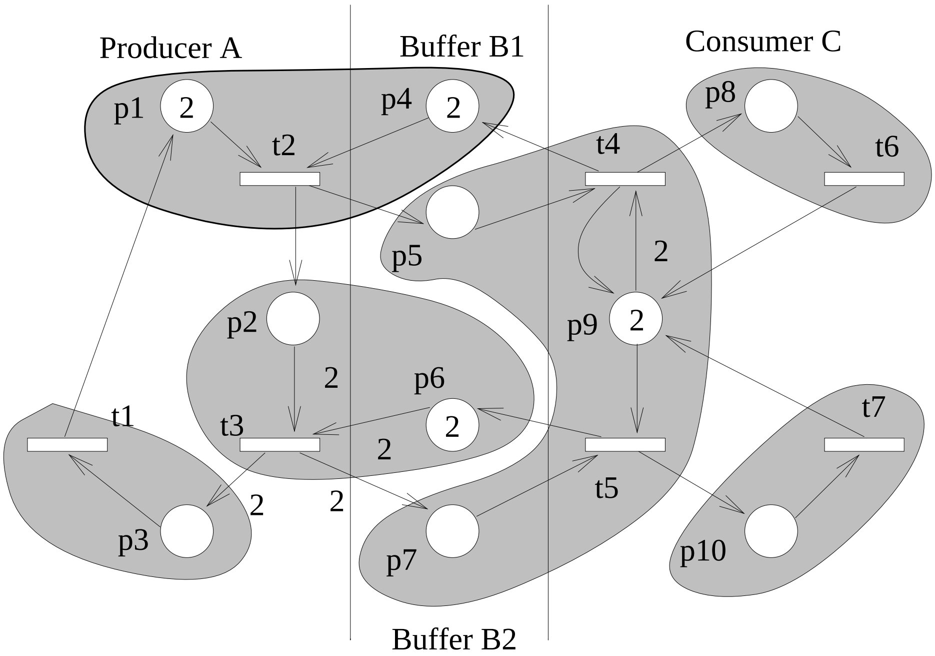

Figure 1: Producer/Consumer Model and its partition into minimal regions

Proof. Straightforward for nets, where each place has at least one outgoing arc. If this is not the case, one need to define an additional region, which consists of places with empty set of output transitions.

Minimal regions can be generated using the simple algorithm shown below.

$UT = T$;
$i = 0$;
while $UT \neq \emptyset$ do
    remove t' from UT ;
    $T_i = \{t'\}$;
    while $\bullet UT \cap \bullet T_i \neq \emptyset$ do
        remove t with $\bullet t \cap \bullet T_i \neq \emptyset$ from UT ;
        $T_i = T_i \cup \{t\}$;
    od ;
    $i = i + 1$;
od ;

Once the algorithm terminated, sets T_i contains transitions which are used to define regions according to Def. 3 (i.e., $(\bullet T_i)\bullet = T_i$).

4 Generation of Abstract Views

Let PN be a Petri net. We want to enhance information associated to a place by the following kind of vector:

Definition 4 *A p-vector $\mathbf{v}_p$ for a place $p \in P$ is a vector $\mathbf{v}_p \in \mathbb{Z}^{n+m}$ with index $\mathbf{v}_p = (x_1, x_2, \ldots, x_n, x_{n+1}, \ldots, x_{n+m})$. Entries in $\mathbf{v}_p$ are referenced by $\mathbf{v}_p(x)$ for $x \in P \cup T$, places obtain lower index values.*

An aggregation function $AG : \mathbb{Z}^{n+m} \times \mathbb{N}^n \longrightarrow \mathbb{N}$ for p-vectors and markings is defined as $AG(\mathbf{v}_p, M) = \sum_{i=1}^{n} \mathbf{v}_p(i) \cdot M(i)$.

A linear combination $LC : \mathbb{Z}^{n+m} \times \mathbb{Z}^{n+m} \times T \longrightarrow \mathbb{Z}^{n+m}$ of p-vectors $\mathbf{v}_a$, $\mathbf{v}_b$ is defined for a $t \in T$, if $\mathbf{v}_a(t) > 0$ and $\mathbf{v}_b(t) < 0$ as $LC(\mathbf{v}_a, \mathbf{v}_b, t) = \mathbf{v}_c / gcd(\mathbf{v}_c)$, where $\mathbf{v}_c = c_a \cdot \mathbf{v}_a + c_b \cdot \mathbf{v}_b$ with $c_a = lcm(|\mathbf{v}_a(t)|, |\mathbf{v}_b(t)|)/|\mathbf{v}_a(t)|$ and $c_b = lcm(|\mathbf{v}_a(t)|, |\mathbf{v}_b(t)|)/|\mathbf{v}_b(t)|$.

lcm gives the least common multiple of two integers and gcd of an integer vector is the greatest common divisor of the elements. Note that $\mathbf{v}_c(t) = 0$ by construction.

We inductively define extended nets which result from a sequence of net transformations based on linear combinations:

Definition 5 *Let $s \in T^*$ be a sequence of transitions of a PN, and ϵ denote the empty sequence. An extended net is a tuple (N_s, V_s, A_s) inductively defined as follows: $(N_\epsilon, V_\epsilon, A_\epsilon)$ is an extended net where $N_\epsilon = PN$, $V_\epsilon = \cup_{p \in P} \mathbf{v}_p$, $A_\epsilon = V_\epsilon$ and*

$$\mathbf{v}_p(x) = \begin{cases} 1 & : \text{ if } x = p \\ -I^-(p,t) + I^+(p,t) & : \text{ if } x = t \\ 0 & : \text{ otherwise} \end{cases}$$

Let for better readability $New = \{\mathbf{v} | \mathbf{v} = LC(\mathbf{v}_a, \mathbf{v}_b, t), \mathbf{v}_a, \mathbf{v}_b \in A_s\}$ be an abbreviation for the resulting new vectors of a linear combination w.r.t. transition t, and $Used = \{\mathbf{v}_a, \mathbf{v}_b | \mathbf{v} = LC(\mathbf{v}_a, \mathbf{v}_b, t), \mathbf{v}_a, \mathbf{v}_b \in A_s\}$ denote the vectors used to generate New for an extended net (N_s, V_s, A_s).

(N_{st}, V_{st}, A_{st}) is an extended net if (N_s, V_s, A_s) is an extended net, $s \in (T \backslash \{t\})^*$ and
$V_{st} = V_s \cup New$
$A_{st} = (A_s \backslash Used) \cup New$
$P_{st} = P_s \cup \{p | \exists \mathbf{v}_p \in New\}$
$T_{st} = T_s$

$$I^-_{st}(x,y) = \begin{cases} I^-_s(x,y) & : \text{ if } x \in P_s \\ \sum_{z \in P_\epsilon} \mathbf{v}_x(z) I^-_s(z,y) & : \text{ if } x \notin P_s \end{cases}$$

$$I^+_{st}(x,y) = \begin{cases} I^+_s(x,y) & : \text{ if } x \in P_s \\ \sum_{z \in P_\epsilon} \mathbf{v}_x(z) I^+_s(z,y) & : \text{ if } x \notin P_s \end{cases}$$

$M^0_{st}(p) = AG(\mathbf{v}_p, M^0)$

Note that $c_a, c_b \geq 0$ by construction.

We additionally distinguish ordinary places $P^{ord}_s = P_\epsilon$ from those generated in the extension sequence denoted by $P^{agg}_s = P_s \backslash P^{ord}$, such that $P_s = P^{ord}_s \cup P^{agg}_s$.

The definition separates available vectors A_s from the total set of vectors V_s in order to ensure that vectors are used in at most one step of a sequence s. This restriction is made in order to focus on those linear combinations which are relevant in the following. The net transformation basically mimicks the computation of P-invariants according to [26]. (N_s, V_s, A_s), where $s \in T^*$ contains all transitions $t \in T$ exactly once describes an extended net where each P-invariant is realized by a place $p \in P^{agg}$. For an aggregated place p representing a P-invariant, the marking is constant (i.e., $I^+(p,t) = I^-(p,t)$ for all $t \in T$). We can interpret the marking of an aggregated place as a macro marking which includes an abstract view of the detailed marking. Since the complete net does not exchange tokens with the environment, macro markings representing P-invariants are invariant. However, if sequence s contains only a subset of transitions, then the marking of an aggregated place represents possibly only a macro marking for a subset of places belonging to a P-invariant. In this case, the marking of the aggregated place changes whenever tokens are added to or removed from the partial P-invariant it represents. Since the net transformation follows the computation of P-invariants, the effort is limited to the effort for computing P-invariants. Often the effort is much smaller, since only a subset of transitions is used in s.

Example 2 *Before we go into further details we come back on our running example: it contains P-invariants (described as formal sums):*

$$p1 + p2 + p3 = 2 \quad (1)$$

$$p4 + p5 = 2 \quad (2)$$

$$p6 + p7 = 2 \quad (3)$$

$$p8 + p9 + p10 = 2 \quad (4)$$

and a T-invariant $(2\,t1, 2\,t2, t3, 2\,t4, 2\,t5, 2\,t6, 2\,t7)$*. Fig. 2 shows the extended net for sequence* $s = t1t6t7$, $P_s^{agg} = \{p11, p12, p13\}$ *are hatched and arcs are dotted to indicate the differences from* N_ϵ*. The minimal regions which are connected via t1, t6, and t7 have been merged; shaded polygons denote the new and larger regions in* N_s*. p-vectors and corresponding linear combinations are given in the tabular below:*

	p1	*p2*	*p3*	*p4*	*p5*	*p6*	*p7*	*p8*	*p9*	*p10*	*t1*	*t2*	*t3*	*t4*	*t5*	*t6*	*t7*
v1	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*1*	*-1*	*0*	*0*	*0*	*0*	*0*
v2	*0*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*1*	*-2*	*0*	*0*	*0*	*0*
v3	*0*	*0*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*-1*	*0*	*2*	*0*	*0*	*0*	*0*
v4	*0*	*0*	*0*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*-1*	*0*	*1*	*0*	*0*	*0*
v5	*0*	*0*	*0*	*0*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*1*	*0*	*-1*	*0*	*0*	*0*
v6	*0*	*0*	*0*	*0*	*0*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*-2*	*0*	*1*	*0*	*0*
v7	*0*	*0*	*0*	*0*	*0*	*0*	*1*	*0*	*0*	*0*	*0*	*0*	*2*	*0*	*-1*	*0*	*0*
v8	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*1*	*0*	*0*	*0*	*0*	*0*	*1*	*0*	*-1*	*0*
v9	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*1*	*0*	*0*	*0*	*0*	*-1*	*-1*	*1*	*1*
v10	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*1*	*0*	*0*	*0*	*0*	*1*	*0*	*-1*
v11 = v1 + v3	*1*	*0*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*-1*	*2*	*0*	*0*	*0*	*0*
v12 = v8 + v9	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*1*	*1*	*0*	*0*	*0*	*0*	*0*	*-1*	*0*	*1*
v13 = v10 + v12	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*1*	*1*	*1*	*0*	*0*	*0*	*0*	*0*	*0*	*0*

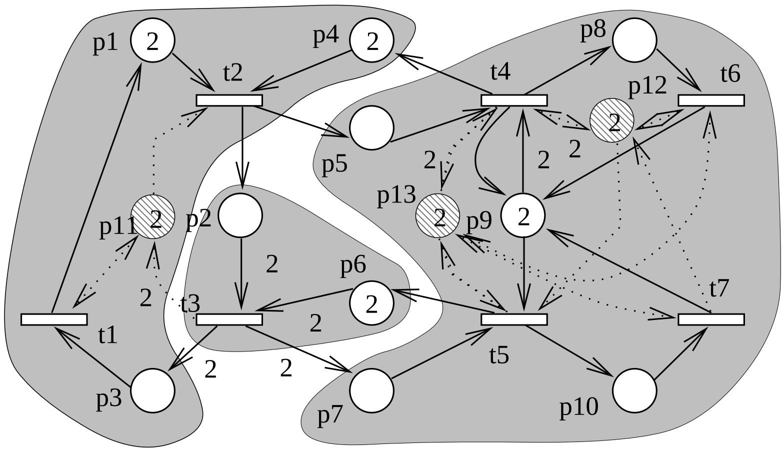

Figure 2: Extended net

Note that the definition of extended net defines arc weights for new arcs connected to new places as weighted arcs of the original net. This allows to consider bidirectional arcs (self-loops) appropriately, as the arcs connected to places $p12$ *and* $p13$ *in Fig. 2 illustrate.*

A place $p \in P^{agg}$ represents a set of places in P^{ord}, and $M(p) = AG(\mathbf{v}_p, M) = \sum_{x \in P_\epsilon} \mathbf{v}_p(x) M(x)$ gives an aggregated marking for the marking of this set. Aggregating information is the crucial point in deducing a hierarchy. Before we describe a way to split an extended net into a high level net and a set of low level nets to obtain the desired hierarchy, we formalize the aggregation and subsequently consider reachability and language invariance of the net extension.

Lemma 1 $M(p) = \sum_{z \in P_\epsilon} \mathbf{v}_p(z) M(z)$

Proof. We consider an induction over transition sequences s, initially all $p \in P_\epsilon^{ord}$ trivially fulfill the lemma and $P_\epsilon^{agg} = \emptyset$. For the induction step we consider a $p \in P_{st}$ which results from $\mathbf{v}_p = c_a \cdot \mathbf{v}_a + c_b \cdot \mathbf{v}_b$, where $\mathbf{v}_a, \mathbf{v}_b \in New$ at step s (the case $p \in P_s$ is trivial in N_{st} since for these places M^0, I^- and $I+$ remain unchanged). We further consider an induction over firing sequences σ, initially $M^0(p) = \sum_{z \in P_\epsilon} \mathbf{v}_p(z) M^0(z)$ holds by definition. For the induction step, we consider $M^0[\sigma{>}\, M[\kappa{>}\, M'$ where the induction assumption ensures $M(p) = \sum_{z \in P_\epsilon} \mathbf{v}_p(z) M(z)$ and we have to show $M'(p) = \sum_{z \in P_\epsilon} \mathbf{v}_p(z) M'(z)$ after extending σ by a transition κ.

$$\begin{aligned} M'(p) &= M(p) - I^-(p,\kappa) + I^+(p,\kappa) \\ &= M(p) - \sum_{z \in P_\epsilon} \mathbf{v}_p(z) I^-(z,\kappa) + \sum_{z \in P_\epsilon} \mathbf{v}_p(z) I^+(z,\kappa) \\ &= M(p) + \sum_{z \in P_\epsilon} (c_a \cdot \mathbf{v}_a(z) + c_b \cdot \mathbf{v}_b(z))(-I^-(z,\kappa) + I^+(z,\kappa)) \end{aligned}$$

according to the definition of successor marking and extended net. By induction assumption

$$M(p) = \sum_{z \in P_\epsilon} \mathbf{v}_p(z) M(z) = \sum_{z \in P_\epsilon} (c_a \cdot \mathbf{v}_a(z) + c_b \cdot \mathbf{v}_b(z)) M(z)$$

such that we can replace $M(p)$ in the equation above. Observe that I^-,I^+ remain invariant for $z \in P_\epsilon$, such that we obtain:

$$\begin{aligned} M'(p) &= \sum_{z \in P_\epsilon} (c_a \cdot \mathbf{v}_a(z) + c_b \cdot \mathbf{v}_b(z))(M(z) - I_s^-(z,\kappa) + I_s^+(z,\kappa)) \\ &= \sum_{z \in P_\epsilon} (c_a \cdot \mathbf{v}_a(z) + c_b \cdot \mathbf{v}_b(z))(M(z) - I_{st}^-(z,\kappa) + I_{st}^+(z,\kappa)) \\ &= \sum_{z \in P_\epsilon} \mathbf{v}_p(z) M'(z) \end{aligned}$$

□

The way in which places are added to a net in an extension sequence, ensures that the reachability set and the language remain the same.

Lemma 2 *For all $s \in T^*$ for which an extended net (N_s, V_s, A_s) is defined, holds*

$$\begin{aligned} RS(PN) &= RS(PN_s)|_P \\ L(PN) &= L(PN_s) \end{aligned}$$

Proof. by induction over sequences s, initially $s = \epsilon$ is trivially fulfilled. For the induction step we start with the special case $New = Used = \emptyset$ which directly implies equality. This case can occur e.g. if $A_s = \emptyset$ or if $\not\exists \mathbf{v}_a, \mathbf{v}_b \in A_s : \mathbf{v}_a(t) > 0, \mathbf{v}_b(t) < 0$. For the general case we give a proof by contradiction:
case $RS(PN_\epsilon) \subset RS(PN_s)|_P$
$\exists \sigma \in T^* : M_{st}^0[\sigma > M_{st}$ but σ not possible in PN_s. Hence $\exists t' \in \sigma$ which is not enabled in PN_s. i.e. there are less tokens in a place $p \in P_s$. Contradiction to definition of extended nets, because M^0, I^-, I^+ changes only with respect to new places.
case $RS(PN_\epsilon) \supset RS(PN_s)|_P$
$\exists \sigma \in T^* : M_s^0[\sigma > M_s$ but σ not possible in PN_{st}. Hence $\exists t' \in \sigma$ which is not enabled in PN_{st}. i.e. there are less tokens in a place $p \in P_{st} \backslash P_s$. According to Lemma 1, $M(p) = \sum_{z \in P_\epsilon} \mathbf{v}_p(z) M(z)$. Since $I_{st}^-(p,t) = \sum_{z \in P_\epsilon} \mathbf{v}_p(z) I_{z,t}^-$ and obviously for each $z \in P_\epsilon : M(z) \geq I_{z,t}^-$, we obtain a contradiction.
In summary equality holds. Equivalence of languages follows by the same line of argumentation. □

A direct consequence of Lemma 2 is that invariants remain valid, T-invariants due to language equivalence and P-invariants due to additivity of invariants, cf. Theorem 1. Furthermore we can decide for places like $p13$ in our example, where $I^+(p13,t) = I^-(p13,t)$ for all transitions t, whether a given initial marking $M^0(p13)$ ensures that a transition is dead due to $M^0(p13) < I^-(p13,t)$ or whether the place can safely by omitted since $M(p13) \geq I^-(p13,t)$ for all $M \in RS$. If the former is the case, it is clear that the net is not live.

So far we have described a way to add places to a net without changing its reachability set or language. The notion of an extended net is only a formal prop to introduce a hierarchical net; it simplifies argumentation why a hierarchical net indeed includes the reachability set or language of its N_ϵ. The key issue for a hierarchy is abstraction: at a higher level the state of a subsystem must be represented in less detail than at a lower level. We use aggregated places to obtain an aggregated state representation and the notion of subsystem is build on the concept of region.

Let $R(N_\epsilon)$ denote the set of minimal regions w.r.t. to an extended net $(N_\epsilon, V_\epsilon, A_\epsilon)$. When we extend this net for a transition $t \in T_r^{out}$ of a region $N_r \in R(N_\epsilon)$, then the new places connect N_r with regions that contain $t\bullet$. Consequently we merge all these regions with N_r, which yields a new region N_r' according to Prop. 1. Since we start from a partition into regions, the resulting set of regions is a partition again, but this partition is less fine. t becomes internal in N_r', because $\bullet t \cup t\bullet \subseteq P_r'$, and new places give an aggregated description of the internal behavior of N_r' w.r.t. transition t. Following this procedure over a sequence s of transitions yields (N_s, V_s, A_s) and a partition into regions, where some regions have internal transitions and aggregated places. In this situation a decomposition of an extended net into a high level net using the aggregated description and a set of low level nets resulting from regions with an internal behavior gives the two-level hierarchy we aim for.

More formally, a high level net for a given extended net (N_s, V_s, A_s) results from a projection with respect to A_s.

Definition 6 *Let (N_s, V_s, A_s) be an extended net, its corresponding high level net $HN = (P^H, T^H, I^{H-}, I^{H+}, M^{0H})$ is defined by:*

$$P^H = \{p | \mathbf{v}_p \in A_s\} \tag{5}$$
$$T^H = \bullet P^H \cup P^H \bullet \tag{6}$$

and I^{H-}, I^{H+}, M^{0H} are the corresponding projections of I^-, I^+, and M^0 w.r.t. $P^H \cup T^H$.

Example 3 *Fig. 3 shows the high level net for the extended net of our running example in Fig. 2.*

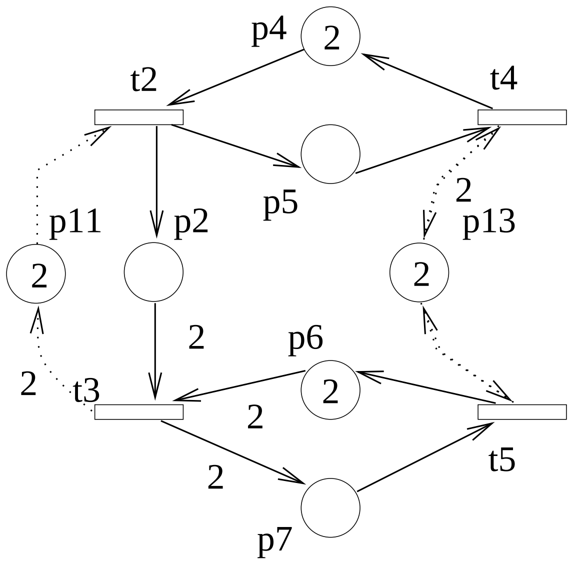

Figure 3: High level net for $s = t1t6t7$

Lemma 3 *Let $RS'(N_\epsilon) = \{M' | \exists M \in RS(N_\epsilon) : \forall p \in P_s : M'(p) = AG(\mathbf{v}_p, M)\}$ for (N_s, V_s, A_s), then*

$$RS'(N_\epsilon) \subseteq RS(HN) \tag{7}$$

Proof. uses previous lemma about equality for N_ϵ and N_s, and that HN is deduced by omitting places (releases enabling conditions thus increases RS) and by omitting transitions, which are isolated (since all elements in the pre- and postset of such transitions are used in linear combinations and thus not contained in A_s anymore). Isolated transitions have no effect on RS. $\square$

P-invariants of HN are linear combinations of P-invariants of N_ϵ, hence if N_ϵ is covered by P-invariants, so is HN. Consequently, we can guarantee finiteness of $RS(HN)$ if N_ϵ is covered by P-invariants.

Lemma 3 states that the HN indeed considers a more abstract net, such that the detailed net can only behave in a way which is consistent with this abstraction/aggregation. If a region in the extended net contains all places of the set $Used$ for a transition t, it shows an internal behavior, which allows to define a non-trivial isolated low-level net:

Definition 7 *A low level net for a region r in an extended net* (N_s, V_s, A_s) *is a net* $LN = (P^L, T^L, I^{L-}, I^{L+}, M^{0L})$ *given* $P^L = (P_r \cap P^H) \cup (P_r \cap P_\epsilon)$, $T^L = P^L\bullet$, *(*$\bullet$ *with respect to* N_ϵ*).*

$$I^{L-}(p)(t) = \begin{cases} I^{H-}(p)(t) & \textit{if } p \in P^H \backslash P_\epsilon \\ I^{-}(p)(t) & \textit{if } p \in P_\epsilon \\ 0 & \textit{otherwise} \end{cases}$$

$$I^{L+}(p)(t) = \begin{cases} I^{H+}(p)(t) & \textit{if } p \in P^H \backslash P_\epsilon \\ I^{+}(p)(t) & \textit{if } p \in P_\epsilon \\ 0 & \textit{otherwise} \end{cases}$$

If LN and the corresponding region r^H *in* HN *do not differ in their transitions,* LN *is trivial and can be neglected. Otherwise* LN *is non-trivial.*

Observe that both, HN and LNs do only pick aggregated places from A_s, elements in $V_s \backslash A_s$ are neglected, because their information is sufficiently represented by the linear combinations they contributed to.

A LN for a region r and its HN share places $P_r \cap P^H \neq \emptyset$ and transitions T_r^{out}. These common net elements form an interface for the HN to communicate with the LN in an asynchronous manner. The HN puts tokens via transitions $t \in T_r^{in}$ onto $P_r \cap P^H$ – sends signals to the LN – and experiences output behavior through firing of $t \in T_r^{out}$. The notion of hierarchy is justified, since the HN abstracts from the details inside the LN, tokens on $P_r \cap P^H \neq \emptyset$ describe a so-called macro marking of the LN, and T_r^{out} represent the aggregated behaviour of the LN. However an LN can be merged with the HN to observe the detailed behavior. Formally we describe this as an extended LN (EN) as $(P^E, T^E, I^{E-}, I^{E+}, M^{0E})$, where $P^E = P^H \cup P^L$, $T^E = T^H \cup T^L$, and I^{E-}, I^{E+}, M^{0E} are defined analogously. An EN is an ordinary PN, with RS(EN) and L(EN) defined as before. The relationship between HN and LN is not symmetric, because the HN has an aggregated description of the LN by $P_r \cap P^H \neq \emptyset$ and T_r^{out}, but not vice versa. As seen from the LN, a HN provides an environment, with whom the LN interacts in an asynchronous manner, but the LN lacks the aggregated description of the HN behavior. Hence the reachability of a LN cannot be seen independently of a HN, such that the reachability set $RS(LN)$ of the LN for a given environment HN needs the notion of EN and results from the projection of $RS(EN)|_{LN}$. In the following section we will use these concepts to give a hierarchical/compositional representation of $RS(N)$ based on $RS(HN)$ and $RS(i)|_i$ for each LN i.

Example 4 *Figs. 4,5 show the extended low level nets for the two non-trivial low level nets – indicated by shaded polygons – of our running example. They result from the producer part, where transition* $t1$ *becomes internal, and from the consumer part, where transitions* $t6$ *and* $t7$ *become internal. For the consumer region, only* $p13$ *becomes part of* A_{t1t6t7} *because* $v12$ *has been used for its construction. The region with transition* $t3$ *has no internal transition.*

Obviously selection of s has a massive impact on the resulting hierarchical net description. Consideration of "optimal" sequences is subject to further investigations. At this state we can only formulate goals and rules of thumb to follow:

1. it is clear, that a non-trivial LN results from merging adjacent regions, i.e. if for N_s with regions A, B and $t \in T_A^{out}, t\bullet \cap B \neq \emptyset$ such that t in s, then all $t' \in T_A^{out}, t'\bullet \cap B \neq \emptyset$ should become element of s.

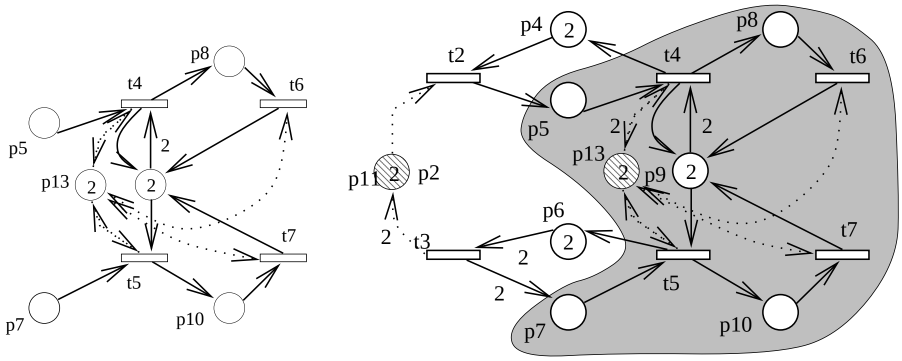

Figure 4: Low level net and extended low level net for the consumer region

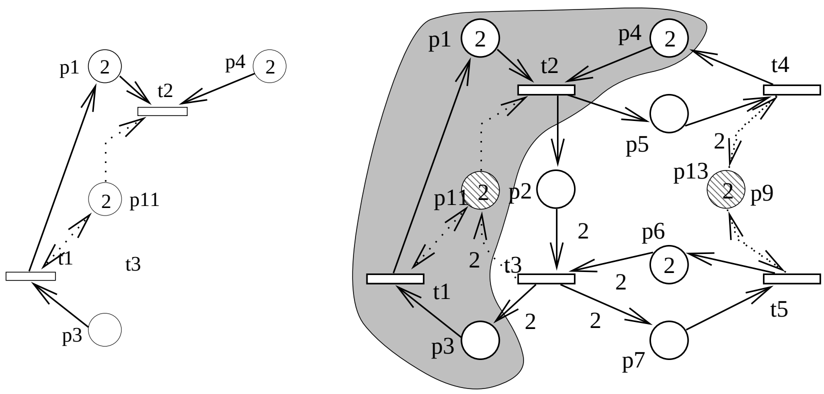

Figure 5: Low level net and extended low level net for the producer region

2. deriving a hierarchy is a divide and conquer strategy, so a sequence should yield a set of non-trivial LNs, such that complexity is equally distributed over this set. This means that only those regions are merged, whose result will not cover a majority of the net.

3. aggregated places introduce overhead, especially building all linear combinations can impose an unacceptable increase of net elements. This is the reason for an exponential worst case time complexity of invariant computation. In our case we have the freedom to select transitions and to consider only a subset of transitions. Hence those transitions t are preferred where $|\bullet t| \cdot |t \bullet|$ w.r.t. N_s is relatively small.

In our implementation of the proposed approach, we have integrated these heuristic rules to generate appropriate transition sequences. First experiences with several examples (e.g., the example presented in Sect. 7) are very encouraging. The program automatically chooses a sequence of transitions which partions complex nets into non-trivial parts and a non-trivial HPN.

5 Hierarchical Representations of RS and RG

Dividing PN into HN and LNs allows us to generate and represent $RS(PN)$ and $RG(PN)$ in a space and time efficient way. For notational convenience we assume that PN is decomposed into one HN and J LNs, which are consecutively numbered 1 through J. Furthermore we assume in the sequel that all reachability sets for the HN and the extended LNs are finite. Consequently, reachability sets are isomorphic to finite sets of consecutive integers. Thus let $RS(HN) = \{0, \ldots, n_H - 1\}$. An M_x corresponds to the x-th marking in $RS(HN)$, and we use x and M_x interchangeably. We can represent RG(HN) by a $n_H \times n_H$

matrix $\mathbf{Q}^H$ with $\mathbf{Q}^H(x,y) = t$ iff transition $t \in T^H$ fulfills $M_x[t> M_y$, provided that between any two markings in RG(HN) at most one transition exists. If more than one transition between M_x and M_y exists, $\mathbf{Q}^H(x,y)$ describes a list of transition indexes. We use for generality the notation $t \in \mathbf{Q}^H(x,y)$ for all t that fulfill $M_x[t> M_y$.

The reachability set $RS(j)$ of a LN j depends on the environment given by HN. Hence, we consider the EN e that corresponds to j and define $RS(j)$ as a projection of $RS(e)$ on the places of j. Since any LN j and the HN share some places $P^j \cap P^H$, we also define $\widetilde{RS}(j)$ as a projection of $RS(j)$ onto places from $P^j \cap P^H$. Markings from $\widetilde{RS}(j)$ are macro markings and allow to partition $RS(j)$. Macro markings are useful for $RS(j)$ generation since full details of an HN are irrelevant at the LN level, one can redefine transitions of T_j^{in}, such that their firing is marking dependent with respect to $RS(HN)|_{P^j \cap P^h} \supseteq \widetilde{RS}(j)$. So EN is only a formal prop to obtain a clear notion of $RS(j)$, in practice however, computation of $RS(j)$ can be performed more efficiently by using only macro marking dependent transitions of T_j^{in} since transitions local to HN are ignored. The resulting set $RS(j)$ might contain markings which are not in $RS(PN)$, but these can be eliminated in a subsequent step, cf. Sect. 6.

Let $\widetilde{RS}(j) = \{0,\ldots,\tilde{n}_j - 1\}$ and denote by $RS(j,\tilde{x})$ with $\tilde{x} \in \{0,\ldots,\tilde{n}_j - 1\}$ the set of markings from $RS(j)$ which belong to marking $\tilde{x}$ in $\widetilde{RS}(j)$. Markings from a set $RS(j,\tilde{x})$ are indistinguishable in the HN, i.e., the marking of the places from $P^H \cap P^j$ is the same. Since reachability sets are assumed to be finite, each set $RS(j,\tilde{x})$ can be represented by a set of integers $\{0,\ldots,n_j(\tilde{x}) - 1\}$. A marking $M_x \in RS(HN)$ uniquely determines the macro markings for all LNs. We denote by x^j the macro marking of LN j belonging to marking x and obtain $M_{x^j} = M_x|_{P^H \cap P^j}$.

Markings of PN_s can be characterized using $J+1$-dimensional integer vectors $(x_H, x_1, \ldots, x_J)$, where $0 \leq x_H < n_H$ and $0 \leq x_j < n_j(x_H^j)$ for $j \in \{1,\ldots,J\}$. x_H describes a marking from $RS(HN)$ and x_j a marking from $RS(j, x_H^j)$, where x_H^j describes the macro marking of LNj when the marking of HN equals x_H. This implies $M_{x_H}|_{P^H \cap P^j} = M_{x_j}|_{P^H \cap P^j}$. Since the previous relation holds, each integer vector of the previously introduced form determines a marking of the extended net. We define a hierarchically generated reachability set

$$RS^H(PN_s) = \bigcup_{x_H \in RS(HN)} x^H \times_{j=1}^{J} RS(j, x_H^j) \tag{8}$$

Observe that the number of markings in $RS^H(PN_s)$ equals

$$\sum_{x=0}^{n_H - 1} \prod_{j=1}^{J} n_j(x^j)$$

Lemma 4 *The hierarchically generated reachability set and the reachability set are related as follows.*

$$RS(PN_s) \subseteq RS^H(PN_s) \textit{ and } RS(PN) \subseteq RS^H(PN_s)|_P$$

Proof. The previous lemmas imply that $RS(PN_s)|_{P^H} \subseteq RS(HN) = RS^H(PN_s)|_{P^H}$, $RS(PN_s)|_{P^j} \subseteq RS(j) = RS^H(PN_s)|_{P^j}$ for all $j \in \{1,\ldots,J\}$, and for each $M \in RS(PN_s)$: $M|_{P^H} \in RS(HN), M|_{P^j} \in RS(j)$. By construction of the hierarchically generated reachability set $M \in RS^H(PN_s)$ follows. The second relation follows since $RS(PN) = RS(PN_s)|_P$. □

Equation (8) describes a very compact way to represent huge reachability sets by composing a few smaller sets. Observe that a few reachability sets with some hundred of markings are enough to describe sets with several millions or billions of markings. To keep the representation compact, it has to be assured that reachability sets of the LNs are roughly of the same size. Of course, this is hard to assure a priori, but it is possible to generate regions in a way that they include a similar number of places and transitions which is often sufficient to yield reachability sets of a similar size for the different regions. However, the reachability set of the original net is not equal to $RS^H(PN_s)$, it is only included in the hierarchically generated reachability set. Before we compute $RS(PN_s)$ as part of $RS^H(PN_s)$, the reachability graph is represented in a compact form similarly to the compact representation of the reachability set.

First of all, we define the effect of transitions locally for LNs. Two different classes of transitions have to be distinguished with respect to LN j.

- $LT^j = T^j \setminus T^H$ the set of local transitions in LN j.
- $ST^j = T_j^{in} \cup T_j^{out}$ the set of transitions, which describe the communication between LN j and HN.

The effect of transitions at marking level is defined using Boolean matrices. As usual we assume that multiplication of Boolean values is defined as Boolean *and* and summation as Boolean *or*. Thus let $\mathbf{Q}_t^j[\tilde{x}, \tilde{y}]$ be a $n_j(\tilde{x}) \times n_j(\tilde{y})$ matrix describing transitions in the reachability graph of LN j due to firing transition t. $\mathbf{Q}_t^j[\tilde{x}, \tilde{y}](x, y) = 1$, iff transition t is enabled in marking $x \in RS^j[\tilde{x}]$ and firing of t yields successor marking $y \in RS^j[\tilde{y}]$. All remaining elements in the matrices are 0. Since transitions $t \in LT^j$ do not modify the marking of the HN, $\mathbf{Q}_t^j[\tilde{x}, \tilde{y}] = \mathbf{0}$ for $\tilde{x} \neq \tilde{y}$ and $t \in LT^j$. Furthermore, we define for $t \notin LT^j \cup ST^j$: $\mathbf{Q}_t^j[\tilde{x}, \tilde{y}] = \mathbf{I}_{n_j(\tilde{x})}$ if $\tilde{x} = \tilde{y}$ and $\mathbf{0}_{n_j(\tilde{x}), n_j(\tilde{y})}$ otherwise. $\mathbf{I}_n$ is a $n \times n$ matrix with 1 at the diagonal and 0 elsewhere. $\mathbf{0}_{n,m}$ is a $n \times m$ matrix with all elements equal to 0. The reason for this definition is that a transition $t \notin LT^j \cup ST^j$ does not modify the marking of LN j and cannot be disabled by LN j. This is exactly described by matrices $\mathbf{I}$ and $\mathbf{0}$. Define for $t \in T$, $\tilde{x} \in \widetilde{RS}(j)$ and $x \in RS(j, \tilde{x})$

$$q_t^j(\tilde{x}, x) = \sum_{\tilde{y} \in \widetilde{RS}(j)} \sum_{y \in RS(j, \tilde{y})} \mathbf{Q}_t^i[\tilde{x}, \tilde{y}](x, y),$$

If $q_t^j(\tilde{x}, x) = 1$, then t is enabled by LN j in marking x. For the HN we define $q_t^H(x) = 1$ if $t \notin T^H$ or $t \in Q^H(x, y)$ for some $y \in RS(HN)$. In all other cases $q_t^H(x) = 0$.

The matrices describe the effect of transitions with respect to the HN or a single LN. The next step is to consider the effect of a transition with respect to the global net. Transition t is enabled in marking $x \in RS^H(PN_s)$ iff

$$q_t(x_H, x_1, \ldots, x_J) = q_t^H(x_H) \prod_{j=1}^{J} q_t^j(x_H^j, x_j) = 1 \ .$$

It is straightforward to prove this enabling condition. Since $q_t^j(.) \equiv 1$ for $t \notin LT^j \cup ST^j$ and $q_t^H(.) \equiv 1$ for $t \notin T^H$, enabling depends only on the marking of parts where the transition belongs to. A transition is enabled if it is enabled in all parts simultaneously. In a similar way we can characterize transitions between markings. Transition t is enabled in marking $(x_H, x_1, \ldots, x_J)$ and its firing yields successor marking $(y_H, y_1, \ldots, y_J)$ iff $t \in \mathbf{Q}^H(x_H, y_H)$ and

$$\prod_{j=1}^{J} \mathbf{Q}_t^j(x_j, y_j) = 1 \ .$$

This relation allows us to characterize the reachability graph completely. To do this in a more elegant way, we define Kronecker operations for matrices.

Definition 8 *The Kronecker product* $\mathbf{A} \otimes \mathbf{B}$ *of a* $n_A \times m_A$ *matrix* $\mathbf{A}$ *and a* $n_B \times m_B$ *matrix* $\mathbf{B}$ *is defined as a* $n_A n_B \times m_A m_B$ *matrix*

$$\mathbf{C} = \begin{pmatrix} \mathbf{A}(0,0)\mathbf{B} & \cdots & \mathbf{A}(0, m_A - 1)\mathbf{B} \\ \vdots & \ddots & \vdots \\ \mathbf{A}(n_A - 1, 0)\mathbf{B} & \cdots & \mathbf{A}(n_A - 1, m_A - 1)\mathbf{B} \end{pmatrix}$$

The Kronecker sum $\mathbf{A} \oplus \mathbf{B}$ *is defined for square matrices only as*

$$\mathbf{D} = \mathbf{A} \otimes \mathbf{I}_{n_B \times n_B} + \mathbf{I}_{n_A \times n_A} \otimes \mathbf{B} \ .$$

The definition of Kronecker sums/products does not include the data type of the matrix elements. Indeed all kinds of algebraic rings can be used. In particular we consider here Boolean or real values. Since the Kronecker product is associative we can define a generalization for J matrices $\mathbf{A}_j$ of dimension $n_j \times m_j$.

$$\mathbf{C} = \bigotimes_{j=1}^{J} \mathbf{A}_j = \prod_{j=1}^{J} \mathbf{I}_{l_j} \bigotimes \mathbf{A}_j \bigotimes \mathbf{I}_{u_j} \ ,$$

where $l_j = \prod_{i=1}^{j-1} m_i$ and $u_j = \prod_{i=j+1}^{J} n_i$. In the same way the Kronecker sum can be defined for $n_j = m_j$ as

$$\mathbf{D} = \bigoplus_{j=1}^{J} \mathbf{A}_j = \sum_{j=1}^{J} \mathbf{I}_{l_j} \bigotimes \mathbf{A}_j \bigotimes \mathbf{I}_{u_j} \ .$$

Observe that $\mathbf{C}$ is a matrix with $\prod_{j=1}^{J} n_j$ rows and $\prod_{j=1}^{J} m_j$ columns. If we consider the number of non-zero elements in $\mathbf{C}$ in terms of the number of non-zero element in $\mathbf{A}_j$ and denote the number of non-zero elements in a matrix $\mathbf{A}$ as $nz(\mathbf{A})$, then we obtain

$$nz(\mathbf{C}) = \prod_{j=1}^{J} nz(\mathbf{A}_j) \ .$$

Kronecker sums and product are a very compact way to represent huge matrices. Implicitly Kronecker operations realize a linearization of a J dimensional number. Row indices of matrix $\mathbf{C}$ or $\mathbf{D}$ are computed from the row indices of the matrices $\mathbf{A}_j$ using the relation

$$x = \sum_{j=1}^{J} x_j \prod_{i=j+1}^{J} n_i \ ,$$

where x is the row index in $\mathbf{C}$ or $\mathbf{D}$, x_j is the row index in $\mathbf{A}_j$ and n_j is the number of rows of $\mathbf{A}_j$. In the same way column indices are computed from the relation

$$y = \sum_{j=1}^{J} y_j \prod_{i=j+1}^{J} m_i \ ,$$

y is the column index in $\mathbf{C}$ or $\mathbf{D}$, y_j is the column index in $\mathbf{A}_j$ and m_j is the number of columns of $\mathbf{A}_j$. These representations are denoted as mixed radix number representations. Obviously x (y) determines all x_j (y_j) and vice versa. For complementary information about Kronecker operations and mixed radix number schemes we refer to [15], considering an example with $J = 3$ and $n_j = m_j = 10$ for all $j \in \{1, 2, 3\}$ is recommended.

Mixed radix numbering schemes can as well be applied to number markings in $RS^H(PN_s)$. However, we use a two level scheme, where the first number describes the HN marking and the second number is computed from the numbers of LN markings. Thus marking $(x_H, x_1, \ldots, x_J)$ receives number (x_H, x_L), where

$$x_L = \sum_{j=1}^{J} x_j \prod_{i=j+1}^{J} n_i(x_H^i) \ .$$

Using this numbering scheme, $RG^H(PN_s)$ can be represented using Kronecker products of Boolean matrices. We define $\mathbf{Q}_t^H$ as the incidence matrix of the reachability graph considering only transition t. Using the two level marking number, $\mathbf{Q}_t^H$ has a block structure with n_H^2 block matrices.

$$\mathbf{Q}_t^H = \begin{pmatrix} \mathbf{Q}_t^H[0,0] & \cdots & \mathbf{Q}_t^H[0, n_H - 1] \\ \vdots & \ddots & \vdots \\ \mathbf{Q}_t^H[n_H - 1, 0] & \cdots & \mathbf{Q}_t^H[n_H - 1, n_H - 1] \end{pmatrix}$$

Submatrix $\mathbf{Q}_t^H[x, y]$ includes all transitions between markings belonging to HN marking x and markings belonging to HN marking y due to transition t in the net. Each submatrix can be represented as a Kronecker product of LN matrices.

$$\mathbf{Q}_t^H[x, y] = \bigotimes_{j=1}^{J} \mathbf{Q}_t^j[x^j, y^j] \tag{9}$$

This form describes a very compact representation of a huge matrix.

Assume that $\mathbf{Q}_t$ is a Boolean matrix describing transition t at $RS(PN)$, then $\mathbf{Q}_t^H$ equals

$$\begin{pmatrix} \mathbf{Q}_t & \mathbf{0} \\ \mathbf{A} & \mathbf{B} \end{pmatrix}$$

after appropriate ordering of markings (i.e., markings from $RG(PN) \cap RG^H(PN)$ are followed by markings from $RG^H(PN) \setminus RG(PN)$). If $RS(PN) = RS^H(PN_s)$, then matrices $\mathbf{A}$ and $\mathbf{B}$ disappear. Since the initial marking is part of $RG(PN) \cap RG^H(PN)$, the above representation implies that successors of reachable markings can be computed using matrices $\mathbf{Q}_t^H$ and, consequently, also reachability analysis can be performed using these matrices.

The incidence matrix of $RG^H(PN)$ can be represented as

$$\mathbf{Q}^H = \sum_{t \in T} \mathbf{Q}_t^H \ .$$

For a compact representation, the Kronecker representation is definitely preferable. It can be applied in various analysis algorithms as shown in the Sect. 6. Local transitions cause a specific matrix pattern of nonzero elements. Since $\mathbf{Q}_t^i[x,x]$ equals an identity matrix for $t \in LT^j$, $j \neq i$ and $\mathbf{Q}_t^j[x,y] = \mathbf{0}$ for $x \neq y$,

$$\mathbf{Q}_t^H[x,y] = \begin{cases} \mathbf{I}_{l_j(x)} \otimes \mathbf{Q}_t^j[x^j, x^j] \otimes \mathbf{I}_{u_j(x)} & \text{if } x = y \\ \\ \mathbf{0} & \text{otherwise} \end{cases}$$

where $l_j(x) = \prod_{i=1}^{j-1} n_i(x^i)$ and $u_j(x) = \prod_{i=j+1}^{J} n_i(x^i)$. By collecting local transitions in one matrix

$$\mathbf{Q}_l^j[x_j, y_j] = \sum_{t \in LT^j} \mathbf{Q}_t^j[x_j, y_j] \ ,$$

we obtain the following representation for a submatrix of $\mathbf{Q}^H$

$$\mathbf{Q}^H[x,y] = \begin{cases} \bigoplus_{j=1}^{J} \mathbf{Q}_l^j[x^j, y^j] + \sum_{t \in T \setminus \cup_{k=1}^{J} LT^k} \bigotimes_{j=1}^{J} \mathbf{Q}_t^j[x^j, y^j] & \text{if } x = y \\ \\ \sum_{t \in T \setminus \cup_{k=1}^{J} LT^k} \bigotimes_{j=1}^{J} \mathbf{Q}_t^j[x^j, y^j] & \text{otherwise} \end{cases} \tag{10}$$

Eq. 10 does not distinguish between different local transitions of the same LN. If such a distinction is necessary, transitions which have to be visible can be excluded from the sets LT^j. In this way it is possible to keep all relevant information in the representation of $RG^H(PN_s)$.

If we consider SPNs, transitions are enhanced by a transition rate. Thus $\mathbf{Q}$ contains real instead of Boolean values. However, the Kronecker representation of the matrix is very similar. If all transitions have marking independent transition rates μ_t, matrix $\mathbf{Q}^H$ is given by

$$\sum_{t \in T} \mu_t \mathbf{Q}_t$$

In this case the elements of $\mathbf{Q}_t$ are interpreted as real values 1.0 and 0.0, respectively.

Example 5 *The running example is rather small so that we can not expect practical gain from representing RS or RG in a compositional way as proposed in this section. However, even for this simple example the representation becomes more compact and the example allows us to clarify the general concepts.*

The following table summarizes the number of markings in column RS and the number of transitions in the RG in the corresponding column for the various nets considered here. Obviously the HN has a RG which is significantly reduced compared to PN.

	Marking description							Successor markings					
no	ap1	ap2	p2	p4	p5	p6	p7	tnr	no	tnr	no	tnr	no
0	2	2	0	2	0	2	0	t2	1				
1	1	2	1	1	1	2	0	t2	2	t4	3		
2	0	2	2	0	2	2	0	t3	4	t4	5		
3	1	2	1	2	0	2	0	t2	5				
4	2	2	0	0	2	0	2	t4	6	t5	7		
5	0	2	2	1	1	2	0	t3	6	t4	8		
6	2	2	0	1	1	0	2	t2	9	t4	10	t5	11
7	2	2	0	0	2	1	1	t4	11	t5	12		
8	0	2	2	2	0	2	0	t3	10				
9	1	2	1	0	2	0	2	t4	13	t5	14		
10	2	2	0	2	0	0	2	t2	13	t5	15		
11	2	2	0	1	1	1	1	t2	14	t4	15	t5	16
12	2	2	0	0	2	2	0	t4	16				
13	1	2	1	1	1	0	2	t2	17	t4	18	t5	19
14	1	2	1	0	2	1	1	t4	19	t5	20		
15	2	2	0	2	0	1	1	t2	19	t5	0		
16	2	2	0	1	1	2	0	t2	20	t4	0		
17	0	2	2	0	2	0	2	t4	21	t5	22		
18	1	2	1	2	0	0	2	t2	21	t5	23		
19	1	2	1	1	1	1	1	t2	22	t4	23	t5	1
20	1	2	1	0	2	2	0	t4	1				
21	0	2	2	1	1	0	2	t4	26	t5	25		
22	0	2	2	0	2	1	1	t4	25	t5	2		
23	1	2	1	2	0	1	1	t2	25	t5	3		
24	0	2	2	2	0	0	2	t5	26				
25	0	2	2	1	1	1	1	t4	26	t5	5		
26	0	2	2	2	0	1	1	t5	8				

Table 1: Reachable markings and possible transitions of the HN.

	RS	*RG transitions*
PN	*254*	*622*
HN	*27*	*51*
LN 1	*18*	*30*
LN 2	*45*	*114*

All markings and possible transitions of the HN are shown in Tab. 1. Macro markings with respect to LN 1 *are defined by projection of the HN marking on the places* $ap1, p4$*; macro markings for LN* 2 *are defined by projection of the HN marking on* $ap2, p5, p7$*. In both cases* 9 *macro markings are generated. From the extended nets,* RS^j *and the matrices* $\mathbf{Q}_t^j$ *are computed. For LN* 1 *a macro marking represents in the average 2 markings. This small number is not surprising since LN* 1 *consists internally of two places connected via single transition, such that macro markings only abstract from the internal place where tokens reside. For the second LN more internal details are hidden by the aggregated description used in the HN. Consequently, a macro marking of LN* 2 *represents in the average* 9 *detailed markings. The Kronecker representation requires* 195 *transitions to represent the complete reachability graph with* 622 *transitions. Of course, this comparison does not consider overhead to store different matrices in the Kronecker representation. However, the overhead depends on the number of transitions in* $\cup ST^i$ *and the number of LNs. Both quantities are negligible compared to the number of markings if we consider large nets. The hierarchically generated reachability set* RS^h *includes* 270 *markings, which means that* 16 *markings are unreachable. We consider this point in the subsequent section.*

6 Hierarchical Analysis Approaches

We now introduce analysis approaches which rely on the Kronecker representation of $RG(PN)$. In particular it is necessary to introduce a method to characterize $RS(PN)$ and not only a superset in form of $RS^h(PN_s)$. The central idea of reachability analysis is that the numbering of markings in $RS^h(PN_s)$ is a perfect hash function for markings in $RS(PN)$. This has first been exploited for efficient reachability analysis of SGSPNs, a class of generalized SPNs consisting of components synchronized via transitions, in the work of Kemper [23]. We can use a similar approach here, but do not necessarily rely on it, see e.g., [12] as an alternative. Let $n(x) = \prod_{i=1}^{J} n_i(x^i)$ the number of markings in RS^h when the marking of the HN is M_x. Let $\mathbf{r}[x]$ be a Boolean vector of length $n(x)$ which is used to store results of the reachability analysis. Thus $\mathbf{r}[x_H](x_L) = 1$ indicates that marking $(x_H, x_L) \in RS(PN)$ and $\mathbf{r}[x_H](x_L) = 0$ indicates after termination that $(x_H, x_L) \notin RS(PN)$. Formally we use here one Boolean vector per HN marking, but it is obviously possible to store all these vectors consecutively in a single Boolean vector of appropriate length. Reachability analysis requires, apart from the vectors $\mathbf{r}[x]$ and the different matrices introduced in the previous section, a set $\mathcal{U}$ to store unexplored markings, similar to the set $\mathcal{U}$ used in *generate_RS*. However, now $\mathcal{U}$ only has to store integer pairs instead of complete marking vectors.

Let (x_{0H}, x_{0L}) be the number of the initial marking, then $\mathbf{r}[x_{0H}](x_{0L})$ is initialized with 1, all remaining vector components are zero. Additionally, $\mathcal{U}$ is initialized with (x_{0H}, x_{0L}). Then the following algorithm is used to determine reachable markings.

```
generate_structured_RS (PN)
  while (U ≠ ∅) do
    remove (x_H, x_L) from U ;
    for j = 1 to J do
    for all y^j with Q^j_l(x^j, y^j) = 1 do              // compute successor in subnet j
      y_L = x_L + (y^j − x^j) · l_j(x_H)
      if r[x_H](y_L) = 0 then
        r[x_H](y_L) = 1 ;
        U = U ∪ (x_H, y_L) ;
    for all y_H with Q^H(x_H, y_H) ≠ ∅ do               // compute successor in subnet HN
    for all t ∈ Q^H(x_H, y_H) do
      y_L = x_L ;
      for all j with t ∈ T^j do
        if y^j_L with Q^j_t[x^j_H, y^j_H](x^j_L, y^j_L) = 1 exists then      (*)
          y_L = y_L + (y^j_L − x^j_L) · l_j(y_H) ;
        else
          y_L = −1 and break ;
      if y_L ≥ 0 then
        if r[x_H](y_L) = 0 then
          r[x_H](y_L) = 1 ;
          U = U ∪ (x_H, y_L) ;
end
```

In the step indicated by $(*)$, the algorithm exploits the fact that firing of transition t always yields a unique successor marking. Therefore each row of a matrix $\mathbf{Q}_t^i$ can include at most one element. The approach can be easily extended for PNs where different successor markings are possible. This situation occurs in nets where probabilistic output bags for transitions are allowed. Since the algorithm computes all successor markings of reachable markings, it is straightforward to prove that *generate_structured_RS* generates $RS(PN)$ and terminates when $RS(PN)$ is finite which is the case here, since $RS^h(PN_s)$ has assumed to be finite.

The remaining point is the comparison of *generate_structured_RS* and *generate_RS*. As before we assume that the reachability set contains n markings and in the average d transitions are possible in each marking. The theoretical time complexity of *generate_RS* is $O(nd \log_2 n)$ if insert and member functions on RS use $\log_2 n$ operations. The complexity of *generate_structured_RS* is in $O(nd)$, since the Boolean vectors allow us to test in $O(1)$ whether a marking has been reached before. The reduction by a logarithmic factor seems to be not too much on a first glance. However, the approach is used for large reachability sets such that this implies a reduction by at least an order of magnitude. Additionally the constants behind the asymptotic complexity are much lower for *generate_structured_RS*. The reason is that all operations are performed with simple integer operations, while several operations of *generate_RS* are time consuming. For example, if a new marking M is found in *generate_RS*; a data structure to hold M has to be allocated and inserted into the data structure storing the already generated markings. Since this data structure is usually a tree, pointers have to be modified. In *generate_structured_RS* the same operation only requires to set a bit in vector $\mathbf{r}$. Thus usually we can expect an improvement of run times which is around two orders of magnitude for large reachability sets. However, to apply *generate_structured_RS*, PN has to be decomposed first and the reachability sets and matrices for the subnets have to be generated. The complexity of both problems is for large nets much lower than reachability analysis. This can also be seen in the example presented below.

Apart from time complexity, we also have to compare space complexity. Of course, also the difference in memory requirements depends on the concrete example. However, if the net has been decomposed into LNs with roughly identical reachability set sizes and the size of $RS^h(PN_s)$ and $RN(PN)$ does not differ too much (i.e., not by several orders of magnitude), then (8) assures that the size of the LN/HN reachability sets and matrices is negligible compared with the size of the complete reachability set and graph. Experiences show that the approach allows us to handle much larger reachability sets. An additional advantage of *generate_structured_RS* is that we make use of secondary memory in a very efficient way. Since vector $\mathbf{r}$ is structured into subvectors and successor markings are computed consecutively for subvectors, it is possible to preload required subvectors from secondary memory.

After the reachability set has been computed by setting the values in vector $\mathbf{r}$, it can be decided in $O(1)$ whether a marking is reachable or not. Furthermore, all successor markings for a marking can be computed from the Kronecker representation for local transitions in a constant time and for others in a time at most linear in the number of subnets. Since the Kronecker representation includes information about transitions yielding to successors, even successors reachable by specific transitions can be computed. Based on these basic steps, standard algorithms for model checking can be applied for the nets.

In a similar way the Kronecker representation can be exploited for the quantitative analysis of SPNs. The basic step here is to realize the product of a vector with a sum of Kronecker products of matrices. However, this step is already known in numerical analysis and can be combined with various iterative numerical analysis techniques [3]. Thus the Kronecker representation allows the analysis of large SPN models, which cannot be handled using standard means. For further details we refer to the literature [7, 32].

Example 6 *The size of the running example is so small that is useless to compare runtimes for reachability graph generation. Instead we briefly consider unreachable markings appearing in the hierarchical representation. As already mentioned RS^h contains 270 markings, but only 254 of them are reachable. As an example for unreachable markings we consider markings of the form $(0,1,1,2,0,0,2,*,*,*)$, where the vector includes the number of tokens on the places $p1-p7$. For the places $p8, p9, p10$, we now consider possible markings. Obviously all three places are part of a P-invariant such that the sum of tokens on these places has to equal 2. In the hierarchical generated reachability set all possible distributions of 2 tokens over the places $p8, p9, p10$ are included. However, reachability analysis shows that only markings are reachable where place $p10$ is empty.*

The reason for this restriction can be explained by considering the behavior of the net in some more detail. A token on $p10$ implies that $t5$ has fired after $t4$. But since $p6$ is empty, $t3$ fired after $t5$ and, since $p2$ is non-empty $t1$ and $t2$ fired also after $t5$. Now after firing $t2$ a token resides on $p5$ which has to be transferred to $p4$ by firing $t4$. However, this means that $t4$ fired after $t5$ and $p10$ has to be empty.

The restriction which assures that $p10$ has to be empty when the marking of the places $p1-p7$ is as shown above, is a global restriction which depends on the whole net. So it is not visible in an isolated part and the above mentioned markings belong to RS^h, but reachability analysis shows that they are not reachable and are not part of RS.

Two other optimizations can be used to improve *generate_structured_RS*.

First optimization As noticed in [23] certain unnecessary interleavings due to internal transitions can be eliminated. The idea is that local transitions in different LNs do not interfere. Thus if $t_1 \in LT^i$ and $t_2 \in LT^j$ are both enabled in some marking, then the sequences $t_1 t_2$ and $t_2 t_1$ are both possible and yield an identical successor marking. Consequently it is only necessary to consider one sequence. More general, for a set $\mathcal{T} = \{t_1, \ldots, t_q\}$ of local transitions which belong all to different LNs and which are enabled in some marking, only those transition sequences which are described by a subset of $\mathcal{T}$ and where transitions occur in the order as described in $\mathcal{T}$ need to be considered. This reduces the number of possible sequences from $\sum_{l=1}^{q} \binom{q}{l} l!$ to $\sum_{l=1}^{q} \binom{q}{l}$. In this way the time complexity of reachability analysis can be reduced.

Second optimization In [6] an approach is discussed which reduces time and space complexity. The idea is to reduce a priori the marking sets of LN by combining some markings which are always together reachable or not. As a simple example consider two markings $x, y \in RS(i)$ and a pair of transitions $t, t' \in LT^i$. If t is enabled in x and its firing yields y and t' is enabled in y yielding successor marking x, then x is reachable whenever y is reachable and vice versa. We denote this as identical reachability of markings. Obviously identical reachability holds for all markings in an irreducible subset of a matrix $\mathbf{Q}_l^i[x, y]$. In [6] it is shown that this condition can be further relaxed. However, this extension is beyond the scope of this paper. Markings which are identically reachable can be aggregated a priori. Aggregation in this case means that a set of identically reachable markings is substituted by a single aggregate marking such that all transitions entering or leaving one marking in the subset are substituted by transitions entering/leaving the aggregate marking and transitions between markings in the subset are substituted by transitions starting and ending in the aggregate marking. These transformations are easily performed by adding in the matrices $\mathbf{Q}_t^i$ all rows and columns belonging to markings in the subset to be aggregated. The size of $RS(i)$ and $RG(i)$ is reduced by this aggregation which implies that the size of $RS^h(PN_s)$ and also the effort for reachability analysis are reduced too. After reachability analysis the reachability of an aggregated marking implies that all markings represented by this aggregated marking are also reachable and vice versa; if the aggregated marking is not reachable, then the detailed markings are also not reachable.

Both optimizations depend on the net which is considered. However, for most nets the effort for reachability analysis can be reduced significantly.

7 An Application Example

$\lvert s \rvert$	reg, total	reg, non-tr	P^{agg}	RS^h	RS(HN)	max RS(j)	percent $RS(PN)$
108	10	9	137	10389356	3298	631	15.95
109	9	8	148	10389356	3225	631	15.95
110	8	8	396	8182196	2109	631	20.25
111	7	7	397	4860218	1414	631	34.10
112	6	6	430	4860218	1361	631	34.10
113	5	5	511	4860218	1303	631	34.10

Table 2: Hierarchical representation for sequence s

The running example we considered so far is only useful to illustrate formal concepts, in order to demonstrate applicability of our approach we consider the production cell of [24], which has been subject to modeling and analysis by a variety of tools and which is known to be non-trivial. The production cell model originates from an existing production cell in an industrial setting, which physically consists of six components: a elevating rotary table, a rotable robot with two extendable arms, a traveling crane

	gen_hierarchy		gen_struct_RS	
$\|s\|$	CPU	user	CPU	user
108	0.5	1.0	93.8	94.0
109	0.6	1.0	88.2	88.0
110	2.5	3.0	83.0	84.0
111	2.5	3.0	73.3	74.0
112	2.6	3.0	64.9	65.0
113	3.0	3.0	58.9	60.0

Table 3: Computation characteristics for sequence s

and two conveyor belts. The production cell performs transportation and processing metal plates in a (cyclic) pipeline. A feeding conveyor belt transports metal plates to the elevating table, the table lifts plates for the robot, the robot inserts plates into the press and takes them after pressing from the press onto the second conveyor belt. Originally plates leave the system by the second belt, but in order to have a closed system, the crane is installed to put plates from the second belt onto the feeding belt, such that the number of plates within the system is constant. Thanks to the work of Heiner et al. [18, 19] a Petri net model exists, which considers processing of 5 plates. Refinement is used to organize a model of this size, however the dynamic behavior of the model is not defined unless all refined subnets are available in full detail. This kind of hierarchy is very common for modeling purposes, but useless in terms of analysis. Hence our analysis starts from a flat Place/Transition net with 231 places and 202 transitions[3]. From [19] it is known that the net is live and 1-bounded. The reachability set contains 1,657,242 markings and the reachability graph 6,746,379 transitions.

The algorithm to derive a hierarchy starts from a partition into minimal regions and considers a sequence s, which starts with transitions being internal in minimal regions (which is the case for 74 transitions in our example), subsequently it considers small regions first. Fig. 6 shows how the total number of regions decreases once the internal regions have been considered. On the other hand, the number of non-trivial regions increases in an initial phase since the algorithm prefers small regions and finally decreases when there are no trivial regions left and non-trivial regions are merged. Table 2 indicates the influence of s on the hierarchical representation of RS, it gives the number of regions (non-trivial and in total), the number of aggregated places P^{agg}, cardinalities of the hierarchical reachability set RS^h, the reachability set of the high level net $RS(HN)$, and the maximal number of markings observed among the low level nets. The quality of the whole construction is shown in column "percent $RS(PN)$", which gives the reachable fraction of RS^h. Table 3 gives corresponding computation times in seconds for the computation of the two-level hierarchy and the subsequent computation of the reachability set $RS(PN)$ contained in $RS^h(PN)$, times are given as CPU time and user (wall clock) time. These times have been observed on a SPARCstation 4 with 64 MB main memory, 890 MB virtual memory, and 110 MHz CPU. Obviously computation times are uncritical if the number of aggregated places does not explode.

It is worth mentioning that it takes slightly more than a minute to generate the complete reachability set and reachability graph and represent them in a very space efficient way. About 6 Megabyte memory space are necessary to generate and represent RG and RS. These values are excellent compared to conventional RG generation algorithms. In [18] the same model has been analyzed on a similar workstation using different PN analysis tools. RG generation with the tool PROD needs about 14 hours (see [18]). The small runtimes and storage requirements show that much larger systems can be handled with the approach. We have also analyzed an open version of the production cell for which other tools where not able to generate RS (see [18]). For this version our method needs about 3 minutes real time to generate RS with 2,776,936 markings and RG with 13,152,132 arcs.

As already noticed in [19], computation of a generating set of semi-positive P-invariants is difficult for this net. Our approach is closely related to invariant computation: if we compute an extended net for a sequence covering all transitions T, we obtain a generating set of P-invariants as well. However this

[3] We thank J. Spranger for translating the model into the APNN format [2] used in our implementation.

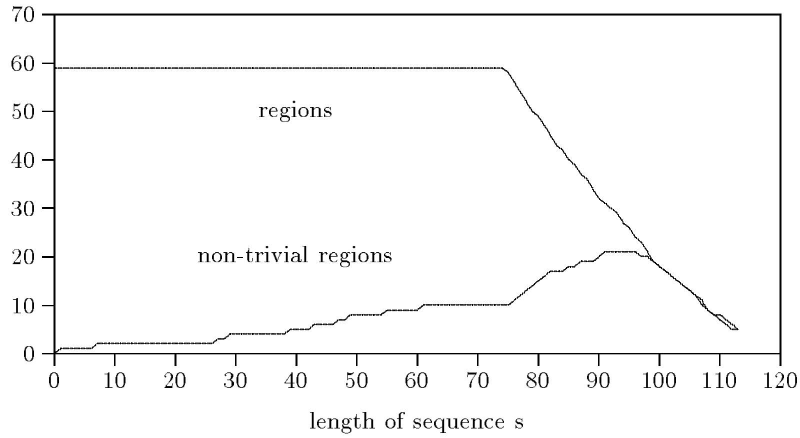

Figure 6: Total number of regions, number of non-trivial regions

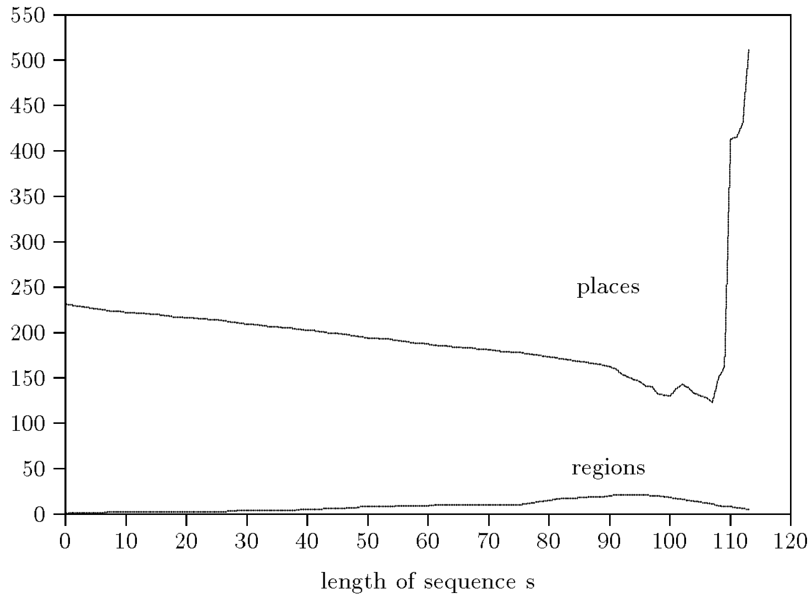

Figure 7: Number of aggregated places, number of non-trivial regions

extreme is not suitable and we consider only a subset of transitions, in order to remain some activity in the HN. From a pragmatic point of view the approach allows us to consider those transitions which can be handled with acceptable computational costs and stop the derivation of a hierarchy if it becomes too expensive. Fig. 7 clearly indicates that a careful selection of transitions can avoid high computational costs. However there is a sharp increase after 108 steps, and the hierarchy derivation stops after 113 steps. For a P-invariant computation 202 steps are necessary, hence Fig. 7 also illustrates the difficulties for invariant computation observed in [19]. According to the results in Table 2, the number of regions and a limit for the number of aggregated places give suitable parameters to stop the automatic hierarchy generation where it makes sense.

8 Conclusions

We have proposed a new approach for the efficient generation and compact representation of reachability sets and graphs of large PNs. In contrast to other approaches, the technique can be applied to general nets without definition of a hierarchical structure and without inherent symmetries. The structuring of the PN into asynchronously interacting regions is done automatically by an algorithm which uses a basic step related to invariant computation to make a transition internal to a region. The algorithm considers a sequence of distinct transitions which can be arbitrary in principle. For our implementation we use some heuristic rules in order to structure a net into regions of approximately the same size. The algorithm stops once a user given number of regions has been obtained. Usually the number of regions should not be chosen too large, to avoid a too complex HN. For nets covered by P-invariants termination is guaranteed, however we cannot ensure termination for general PNs. The problem is that reachability sets of some part, HN or a LN, can become unbounded, even if the reachability set of the complete net is bounded. This problem can not occur for nets which are covered by P-invariants.

The non-trivial example considered in this paper illustrates our experience with the algorithm exercised on a set of examples: the new approach allows the time and space efficient generation and representation of huge reachability sets and graphs. This is, of course, a step towards the analysis of complex PNs. Our current research aims at the integration of the algorithms for model-checking with the Kronecker representation of the reachability graph. First results indicate that this approach allows to analyze much larger nets than conventional means. Additionally, the Kronecker representation can be used for the efficient analysis of SPNs using numerical analysis techniques. For an overview of these techniques we refer to [7].

References

[1] S. Allmaier, M. Kowarschik, G. Horton; State space construction and steady state solution of GSPNs on a shared-memory multiprocessor; *Proc. of the 7th Int. Workshop on Petri Nets and Performance Models (PNPM'97), IEEE CS-Press, 1997, 112-121.*

[2] F. Bause, P. Kemper, P. Kritzinger; Abstract Petri net notation; *Petri Net Newsletter 49 (1995) 9-27.*

[3] P. Buchholz, G. Ciardo, S. Donatelli, P. Kemper; Complexity of Kronecker operations and sparse matrices with applications to the solution of Markov models; *submitted for publication*

[4] G. Berthelot; Transformation and decomposition of nets; In: *G. Rozenberg (ed.), Advances in Petri Nets 85, Springer LNCS 254 (1986).*

[5] P. Buchholz; Hierarchical high level Petri nets for complex system analysis; In: *R. Valette (ed.), Application and Theory of Petri Nets 1994, Springer LNCS 815 (1994) 119-138.*

[6] P. Buchholz; Hierarchical Structuring of Superposed GSPNs; In: *Proc. of the 7th Int. Workshop on Petri Nets and Performance Models (PNPM'97), IEEE CS-Press, 1997, 81-90.*

[7] P. Buchholz; Structured Analysis Approaches for Large Markov Chains; In: *Performance 96 Tutorials; Technical Report, Laboratoire de Reseaux de Communication, Ecole Polytechnique Federal de Lausanne (1996).*

[8] J.R. Burch, E.M. Clarke, K.L. McMillan, D.L. Dill, J. Hwang; Symbolic Model Checking: 10^{20} States and Beyond; In: *Proc. 5th Annual Symposium on Logic in Computer Science, June 1990.*

[9] S. Caselli, G. Conte, P. Marenzoni; Parallel state space exploration for GSPN models; In: *G. De Michelis, M. Diaz (eds.); Application and Theory of Petri Nets 1995, Springer LNCS 935 (1995) 181-200.*

[10] G. Chiola, C. Dutheillet, G. Franceschini, S. Haddad; On well-formed coloured nets and their symbolic reachability graph; In: *K. Jensen, G. Rozenberg (eds.), High Level Petri Nets, Theory and Application, Springer (1991) 373-396.*

[11] G. Chiola, A. Ferscha; Distributed simulation of Petri nets; *IEEE Parallel and Distributed Technology 1 (1993) 33-50.*

[12] G. Ciardo, A.S. Miner; Storage alternatives for large structured state spaces; In: *R. Marie, B. Plateau, M. Calzarossa, G. Rubino (eds.); Proc. 9th Int. Conf. Modelling Techniques and Tools for Computer Performance Evaluation, St. Malo, France, 1997, Springer LNCS 1245 (1997) 44-57.*

[13] S. Christensen, L. Petrucci; Modular state space analysis of coloured Petri nets; In: *G. De Michelis, M. Diaz (eds.); Application and Theory of Petri Nets, Springer LNCS 935 (1995) 201-217.*

[14] G. Ciardo, J. Gluckman, D. Nicol; Distributed state-space generation of discrete-state stochastic models; *to appear in Informs Journal on Computing*

[15] M. Davio; Kronecker products and shuffle algebra; *IEEE Trans. on Comp. 30 (1981) 116-125.*

[16] S. Haddad; A reduction theory for coloured Petri nets; In: *G. Rozenberg (ed.), Advances in Petri Nets 1989, Springer LNCS 424 (1990) 209-235.*

[17] S. Haddad, P. Moreaux; Asynchronous composition of high level Petri nets: a quantitative approach; In: *J. Billington, W. Reisig (eds.), Application and Theory of Petri Nets 1996, Springer LNCS 1091 (1996) 192-211.*

[18] M. Heiner, P. Deusen; Petri net based design and analysis of reactive systems; In: *Proc. 3rd Workshop on Discrete Evenet Systems (WoDES'96), pp. 308-313.*

[19] M. Heiner, P. Deussen, J. Spranger ; A case study in developing control software on manufacturing systems; In: *M. Silva, R. Valette and K. Takahashi (eds.), Workshop Manufacturing and Petri nets, within 17th int. Conf. Application and Theory of Petri nets, Osaka, Japan, 1996.*

[20] P. Huber, A.M. Jensen, L.O. Jepsen, K. Jensen; Reachability trees for high-level Petri nets; *Theor. Comp. Sci. 45 (1986) 261-292.*

[21] K. Jensen; Coloured Petri nets and the invariant method; *Theor. Comp. Sci. 14 (1981) 317-336.*

[22] K. Jensen; Coloured Petri nets vol. 1; *Springer EATCS Monographs (1992).*

[23] P. Kemper; Reachability analysis based on structured representations; In: *J. Billington, W. Reisig (eds.), Application and Theory of Petri Nets 1996, Springer LNCS 1091 (1996) 269-288.*

[24] C. Lewerentz, T. Lindner (eds.); Formal development of reactive systems; *Springer LNCS 891 (1995).*

[25] P. Marenzoni, S. Caselli, G. Conte; Analysis of large GSPN models: a distributed solution tool; In: *Proc. 7th Int. Workshop Petri Nets and Performance Models (PNPM'97), IEEE CS-Press, 1997, 122-131.*

[26] J. Martinez, M. Silva; A simple and fast algorithm to obtain all invariants of a generalized Petri net; In: *Application and Theory of Petri Nets 1981, Springer IFB 52 (1981).*

[27] M. Molloy; Performance analysis using stochastic Petri nets; *IEEE Trans. on Softw. Eng. 31 (1982).*

[28] T. Murata; Petri nets: properties, analysis and applications; *Proc. of the IEEE 77 (1989) 541-580.*

[29] M. Notomi, T. Murata; Hierarchical reachability graph generation of bounded Petri nets for concurrent-software analysis; *IEEE Trans. on Softw. Eng. 20 (1994) 325-336.*

[30] E. Pastor, O. Roig, J. Cortadella, R.M. Badia; Petri net analysis using Boolean manipulation; In: *Application and Theory of Petri nets 1994, Springer LNCS (1994).*

[31] M. Rauhamaa; A comparative study of methods for efficient reachability analysis; *Techn. Rep. 14 (1990), Digital Systems Laboratory, Helsinki University of Technology.*

[32] W.J. Stewart, K. Atif, B. Plateau; The numerical solution of stochastic automata networks; *European Jour. of Oper. Res. 86 (1995) 503-525.*

[33] L. Recalde, E. Teruel, M. Silva; $\{SC\}^*$ ECS: A class of modular and hierarchical cooperating systems; In: *J. Billington, W. Reisig (eds.), Application and Theory of Petri Nets 1996, Springer LNCS 1091 (1996) 440-459.*

[34] A. Valmari; Compositional analysis with place bordered subnets; In: *R. Valette (ed.), Application and Theory of Petri Nets 1994, Springer LNCS 815 (1994) 531-547.*

[35] A. Valmari; State of the art report: stubborn sets; *Petri Net Newsletter 46 (1994) 6-14.*