Stochastic Evaluation of Large Interdependent Composed Models Through Kronecker Algebra and Exponential Sums

Giulio Masetti[1,3,✉], Leonardo Robol[2,3], Silvano Chiaradonna[3], and Felicita Di Giandomenico[3]

[1] Department of Computer Science, Largo B. Pontecorvo 3, 56125, Pisa, Italy
[2] Department of Mathematics, Largo B. Pontecorvo 1, 56125, Pisa, Italy
[3] Institute of Science and Technology "A. Faedo", 56124, Pisa, Italy,
✉giulio.masetti@isti.cnr.it

Abstract. The KAES methodology for efficient evaluation of dependability-related properties is proposed. KAES targets systems representable by Stochastic Petri Nets-based models, composed by a large number of submodels where interconnections are managed through synchronization at action level. The core of KAES is a new numerical solution of the underlying CTMC process, based on powerful mathematical techniques, including Kronecker algebra, Tensor Trains and Exponential Sums. Specifically, advancing on existing literature, KAES addresses efficient evaluation of the Mean-Time-To-Absorption in CTMC with absorbing states, exploiting the basic idea to further pursue the symbolic representation of the elements involved in the evaluation process, so to better cope with the problem of state explosion. As a result, computation efficiency is improved, especially when the submodels are loosely interconnected and have small number of states. An instrumental case study is adopted, to show the feasibility of KAES, in particular from memory consumption point of view.

Keywords: Stochastic Petri Nets, Stochastic Automata Networks, Markov chains, Mean Time To Absorption, Kronecker algebra, Exponential sums, Tensor Train.

1 Introduction

Stochastic modeling and analysis is a popular approach to assess a variety of non-functional system properties, depending on the specific application domain the system is employed in.

Given the increasing complexity and sophistication of modern and future contexts where cyber systems are called to operate, their modeling and analysis becomes on one side more and more relevant to pursue, and on the other side more and more difficult to achieve (especially when high accuracy of analysis outcomes is requested due to criticality concerns). Modularity and composition are widely recognized as foundational principles to manage system complexity and largeness when applying model-based analysis. Sub-models, tailored to represent specific system components at the desired level of abstraction, are first defined, then composed to derive the overall model, representative of the totality of the system under analysis. However, in order to be effective and scalable, such compositional approach needs to be efficient not only at modeling level, but also at model evaluation level. This topic has been addressed by a plethora of studies. When dependability, performance and performability related measures are of interest, a variety of modeling and solution approaches and automated supporting tools have been proposed, typically adopting high level modeling formalisms (among which the Stochastic Petri Nets family is a major category) and either simulation-based or analytical solution techniques [14, 27].

In this paper, we focus on state-based analytical numerical evaluation and propose a new approach to address the problem of the state explosion in the quantitative assessment of dependability and performability related indicators of large, interconnected systems modeled using Stochastic Petri Net (SPN). The reference picture is an overall system model, resulting from the composition of a set of relatively small models (e.g. expressed through the Superposed GSPN (SGSPN) formalism [13]), each one representing individual system component(s) at a desired level of abstraction, then composed through transition-based synchronization.

Specifically, the paper addresses the solution of the Continuous Time Markov Chain (CTMC) underlying the SPN, whose evolution represents the behavior of systems under analysis at a reasonable level of detail. The focus here is on those CTMCs that present absorbing states, and on the evaluation of the Mean Time To Absorption (MTTA), i.e., the expected time needed to arrive into an irreversible state. To the best of our knowledge, this kind of CTMCs has received low attention in past studies in terms of efficient solutions when dealing with large interconnected systems. However, addressing this context is relevant, since it is meaningful in a variety of modeling scenarios of the system under analysis. For example, depending on the performance or dependability measures under analysis, absorbing states represent system conditions directly involved in the computation of the measure, such as:

- in a reliability model [27], absorbing states can be those representing the system failure,
- in a security model, absorbing states are those representing the fact that a certain level of confidentiality has been violated or a part of the system is under the attacker's control,
- in a safety model, absorbing states are states where the system is considered unsafe.

Resorting to well known symbolic representation of the CTMC to gain in efficiency, the new approach, called Kronecker Algebra Exponential Sums (KAES), advances on existing solutions by exploiting powerful mathematical technologies such as Kronecker algebra [6], Tensor Trains [24] and Exponential sums [3].

The rest of the paper is organized as follows. In Section 2, related work is discussed. In Section 3 an overview of the proposed contribution is presented. Basic concepts and model design principles are introduced in Section 4. A detailed description of the MTTA is offered in Section 5. Then, the proposed KAES method is described in Section 6. In order to demonstrate the benefits of the new method, KAES has been implemented in the MATLAB evaluation environment and applied to a case study, detailed in Section 7. Obtained numerical results, discussed in Section 8, show the feasibility of KAES when the MTTA is evaluated, at increasing the size of the system under analysis, while standard numerical approaches fail due to the state-explosion problem. Finally, in Section 9 conclusions are drawn and future work is briefly discussed.

2 Related Work

It is well known that state-space analysis of discrete event systems has to cope with the problem of state space largeness, which in many cases makes unaffordable the analysis of realistic systems. Therefore, many studies have appeared in the literature, all attempting to alleviate the state space explosion problem.

Among them, a well established strategy consists in promoting state space reduction through a symbolic representation of the CTMC. Proposals in this direction were already formulated a few decades ago (e.g., in [12,25]), and there was active research for several years, as documented in the survey in [6]. The overall system model, resulting from composition of a number of system component models, is typically expressed through a SPN-like formalism (e.g., Generalized SPN (GSPN) [8,16]). The component models are orchestrated by the synchronization of a distinguished set of transitions, called synchronization transitions, that implement interdependencies among components. Such "high-level" model is then automatically translated into a "lower level" representation (such as in a Stochastic Automata Network (SAN) [4]). Moreover, the implicit representation of the CTMC is not obtained through constructing the Infinitesimal generator matrix (Q), but through a symbolic representation of Q, the Descriptor matrix ($\tilde{Q}$), that is the sum of two parts: one is the composition of the independent behaviors of the component automata (all the transitions of each submodel are not synchronized with transitions of other submodels), called here Local matrix (R), and the other one takes into account only the interdependencies, typically called Synchronization matrix (W) [12]. The matrix-vector product, a key mathematical operation common to all numerical methods, is then performed through the descriptor matrix-vector product, as in the shuffle, slice and split algorithms [10]. In these studies, since an irreducible CTMC [27] is assumed, it is required that the *reachability graphs* of all component models are fully connected [16].

Research on how to manipulate symbolically $\tilde{Q}$ in order to efficiently extract information needed to generate the relevant part of the Reachable state-space ($\mathcal{RS}$) of the system model, as well as fast implementation of the descriptor matrix-vector product, has been the subject of many investigations in the last twenty five years. A concise survey can be found in [6].

Although the relevant benefits obtained from the symbolic representation and manipulation of $\tilde{Q}$, when the state-space becomes so large that even storing in memory vectors of size $|\mathcal{RS}|$ is unfeasible, symbolic representations of the vectors, called descriptor vectors, would be desirable. This is the research area where we concentrate in this paper. To the best of our knowledge, only two other papers address symbolic representation of descriptor vectors: Kressner et al. [17] and Buchholz et al. [5]. In [17] the same symbolic vector representation as in KAES, i.e., the *Tensor Train* (TT) format [24], is employed together with standard numerical solvers, such as Alternating Minimal ENergy (AMEN) [11], for the evaluation of the steady-state probability vector, meaningful when the Markov chain is irreducible and finite. In [5] a different representation, the Hierarchical Tucker Decomposition, is employed again for the evaluation of the steady-state probability vector in the irreducible context. However, these solutions cannot be easily generalized to address wider measures of interest, such as the evaluation of transient properties, or adapt to analyze Markov chains with absorbing states, which is the target of KAES.

Finally, although not relevant for the developments in this paper, but for completeness on the literature on efficient management of the generated state space, we recall that an alternative approach to the symbolic representation and manipulation of the $\tilde{Q}$ is to exploit a symbolic state-space exploration with multi-valued decision diagrams (MDDs) [2, 7].

3 Overview of the Novel Contribution

As already introduced, the contribution offered by the KAES approach is an efficient solution to evaluate MTTA when the CTMC is large and has absorbing states, working on symbolic representation of the descriptor vectors. First of all, the KAES approach builds upon the following assumptions, which are also common to most of the research studies from the literature review:

- the state space generated from each submodel has to be bounded;
- the marking dependencies of synchronization transition rates have strict rules (see Section 4);
- the Descriptor matrix $\tilde{Q}$ is obtained in two consecutive steps, deriving: i) first the matrix R, that describes the CTMC generated from each submodel when all the transitions of the submodel are not synchronized with the transitions of the other submodels; ii) then the matrix W, that describes only the interactions among the CTMC generated from the submodels when the synchronized transitions are considered.

In this paper, in order to ease the notation, no instantaneous transition is considered, even if both instantaneous local and synchronization transitions can be tackled, as shown in [6]. The logical view and reasoning behind the contribution offered in this paper is now outlined:

- The standard representation of the vectors involved in the computations would require a storage exponential in the number of interconnected systems.

To overcome this difficulty, a compressed representation is employed. Under suitable assumptions, this only requires a storage linear in the number of interconnected systems. A vector or matrix which can be compressed in this format is said to have low tensor train rank.

- Unfortunately, arithmetic operations performed using this representation degrade the low tensor train rank property, which can be restored by recompression.
- The evaluation of the MTTA is recast into solving a linear system with a modified descriptor matrix $\tilde{Q} - S$, where S is a rank 1 correction – efficiently representable in TT form. Linear system solvers are available in the TT format, but are ineffective for the problem under consideration.
- Therefore, a new splitting of $\tilde{Q}$ as

$$\tilde{Q} = Q_1 + Q_2, \tag{1}$$

  is considered, where Q_1 is represented in terms of Kronecker sums and Q_2 in TT form.
- The inverse of Q_1 can be easily applied to a vector (in TT form) using *exponential sums* [3], since the exponential of Kronecker sums is the Kronecker product of exponentials. This property is exploited to efficiently solve the linear system through an iterative method.
- The way the MTTA is computed guarantees a conservative assessment.

4 System Architecture and Model Design

The systems category we address comprises n components $C_1,\ldots,C_n$. These components are interconnected, according to a specific topology that depends on the application domain the system operates in. Such interconnections, also called dependencies, allow inter-operability among system components, but they also represent formidable vehicles through which potential malfunctions or attacks propagate, possibly leading to cascading or escalating failure effects. The analysis of such systems needs to account for the impact of error/failure propagation due to dependences, especially when focusing on dependability-critical systems. This requires cautiousness in building models for such systems, to properly master the resulting complexity, both at model representation and model solution levels.

At the current stage of development, we target loosely interconnected systems. Although this might appear a significant limitation of the proposed approach, loosely interconnection is actually encountered in realistic contexts, such as the electric infrastructure where grid topologies of hundreds of buses have number of dependencies around 2-3 on average. On the other side, we aim at alleviating the problem of state explosion in analytical modeling, that the KAES approach fulfills at some extent.

Exploiting the modular modeling approach of the SGSPN formalism [13], each system component C_i is modeled through a GSPN extended with synchronization transitions, and the model that corresponds to C_i is called M_i. The overall SGSPN model, called M^{sync}, is a set of submodels M_i which interact only through synchronized transitions.

To fix the notation, a GSPN [1] can be defined as an 8-tuple

$$M = (P, T, I, O, H, pri, w, m_{init}),$$

where P is the set of *places* and T is the set of (timed and immediate) *transitions* with $P \cap T = \emptyset$. The functions I: $P \times T \rightarrow \mathbf{N}$, O: $T \times P \rightarrow \mathbf{N}$ and H: $P \times T \rightarrow \mathbf{N}$ are respectively the input, output and inhibition functions that map arcs (p, t) or (t, p) onto *multiplicity* values. In the graphical representation, the multiplicity is written as a number next to the arc (when grater than 1). The function pri: $T \rightarrow \mathbf{N}$ specifies the priority level associated to each transition, that is 0 for timed transitions and a value greater than 0 for immediate transitions. The weight function w: $T \rightarrow \mathbf{R}^+$ assigns rates to timed transitions and weights to immediate transitions. A marking m of M is a function $m : \ P \rightarrow \mathbf{N}$. A place p has n *tokens* if $m(p) = n$. The initial marking of the GSPN is denoted by m_{init}. GSPN formalism considered in this paper is extended to allow marking-dependent rates and weights, and marking-dependent multiplicities of arcs. Transition t is *enabled* in a marking m, written $m \overset{t}{\rightarrow}$, if t has concession (to fire), i.e., $m(p) \geq I(p, t)$ and $m(p) < H(p, t)$, and if no other transition t' exists that has concession in m, with $pri(t') > pri(t)$. The firing delay, i.e., the time that must elapse before the enabled transition can fire, is an exponentially distributed random variable for timed transitions and is zero for immediate transitions. Firing of a transition t enabled in a marking m yielding a new marking m' is denoted by $m \overset{t}{\rightarrow} m'$, with $m'(p) = m(p) - I(p, t) + O(t, p)$. The set of markings that are reachable from m_{init} (reachability set) is denoted by $\mathcal{RS}$. A GSPN is called *bounded* if for all $p \in P$ and $m \in \mathcal{RS}$ the value of $m(p)$ is bounded. A GSPN is called *structurally bounded* if it is bounded for every initial marking [22]. Following the reasoning briefly outlined in [6], in order to guarantee that every M_i will have a finite state-space, in this paper all the component submodels M_i will be assumed structurally bounded.

In this paper, the standard definition of synchronized transitions is restricted to timed transitions.

Definition 1 (Synchronization transitions). *Let be T^{sync} and T_i the sets of transitions defined respectively in M^{sync} and M_i. Let $\mathcal{ST} \subseteq T^{sync}$ the set of synchronization transitions of M^{sync}. A timed transition t is a* synchronization *(or superposed)* transition*, i.e., $t \in \mathcal{ST}$, if there is an occurrence of t in two or more submodels, i.e., $t \in T_{i_1} \cap \ldots \cap T_{i_k}$, with $k \geq 2$. A synchronized transition t is enabled in a marking of M^{sync} if all the occurrences of t within submodels are enabled in the same marking restricted to the submodels. Formally, calling m a marking of M^{sync} and m_i its projection on M_i, $m \overset{t}{\rightarrow}$ if $m_i \overset{t}{\rightarrow}$ for all i such that $t \in T_i$. In the overall model M^{sync}, all the occurrences of t are enabled at the same time and a unique exponentially distributed firing delay is defined for all them, thus all of them fire at the same instant of time. The overall SGSPN model M^{sync} is equivalent to the whole GSPN model M^{sys} obtained joining all* *the submodels M_i where all the occurrences of t are merged into one transition, also named t. Firing of t in M^{sys} corresponds to the firing of all the occurrences of t within the submodels, i.e., formally*

$$m \overset{t}{\rightarrow} m' \iff m_i \overset{t}{\rightarrow} m'_i \textit{ for all } i \textit{ such that } t \in T_i.$$

All the transitions t that are not synchronization transitions, i.e., those for which there exists a unique i such that $t \in T_i$, are called *local transitions.*

Allowing general marking-dependent rates and weights for the design of M_i can lead to inconsistent components models and this issue is strictly related to the granularity of the model and the tensor algebra of choice (see [4, 6, 9]). In this paper, as in [8], rates and weights of all the local transitions that belong to M_i and multiplicities of the corresponding arcs are allowed to depend on the marking of M_i, whereas rates and weights of the synchronization transitions and multiplicities of the corresponding arcs should be constant.

As described in [6], the system model SGSPN can be translated into a SAN and then the state space of M^{sync}, called $\mathcal{RS}$, is not fully explored, and the CTMC associated to M^{sync} is not assembled. Instead of working with Q, the SAN provides an implicit representation, called descriptor matrix $\tilde{Q}$, of Q. In particular, calling $\mathcal{RS}^{(i)}$ the state-space of M_i when each occurrence of the synchronization transitions is considered local and $N_i = |\mathcal{RS}^{(i)}|$, $\tilde{Q}$ is defined as

$$\tilde{Q} = R + W + \Delta, \tag{2}$$

i.e., the sum of local contributions, called R, and synchronization contributions, called W, where

$$R = \bigoplus_{i=1}^{n} R^{(i)}, \tag{3}$$

$$W = \sum_{t \in \mathcal{ST}} \bigotimes_{i=1}^{n} W^{(t,i)}, \tag{4}$$

$R^{(i)}$ and $W^{(t,i)}$ are $N_i \times N_i$ matrices, the diagonal matrix Δ is defined as $\Delta = -\text{diag}\left((R + W)e\right)$ and the operators $\oplus$ and $\otimes$ are the *Kronecker sum* and *Kronecker product*, respectively. The matrices $R^{(i)}$ and $W^{(t,i)}$ are assembled exploring $\mathcal{RS}^{(i)}$. Specifically, $W^{(t,i)} = \lambda_t \tilde{W}^{(t,i)}$ where λ_t is the constant rate associated to t, equal in every M_i, and $\tilde{W}^{(t,i)}$ is a $\{0, 1\}$-matrix defined as follows:

$$\tilde{W}^{(t,i)}_{m_i, m'_i} = \begin{cases} 1 & \text{if } t \text{ is enabled in } m_i \text{ inside } M_i \text{ and } m_i \overset{t}{\rightarrow} m'_i, \\ 0 & \text{otherwise.} \end{cases} \tag{5}$$

In particular, if the transition t has no effect on the component M_i, we have $\tilde{W}^{(t,i)} = I$. The potential state-space of M^{sync}, called $\mathcal{PS}$, is defined as

$$\mathcal{PS} = \mathcal{RS}^{(1)} \times \cdots \times \mathcal{RS}^{(n)},$$

and $|\mathcal{PS}| = N_1 \cdot \ldots \cdot N_n$ will be indicated as N in the following. Using this notation, R, W and Δ are $N \times N$ matrices.

Performance, dependability and performability properties can be defined in terms of *reward structures* [26,27] at the level of the SGSPN model. These reward structures are automatically translated to reward structures at the Stochastic Activity Network (SAN) level and represented by symbolic reward structures at the CTMC level.

5 Mean Time To Absorption

For simplicity, in the rest of the paper it is assumed that, fixed m_{init}, there exists a unique[4] absorbing state in $\mathcal{PS}$ that is the last of the chain defined by $\tilde{Q}$. This is not restrictive because the problem can be always reduced to this situation by collapsing all the absorbing states of M_i into a single one and reordering the CTMC of M_i so that the absorbing state has index N_i. This guarantees, as a consequence of the lexicographic ordering defined by the Kronecker product, that the last state of $\mathcal{PS}$ is absorbing and corresponds to the last state of $\mathcal{RS}$, where all the component models are in their absorbing state. Thus, in the following N will indicate the absorbing state of $\mathcal{PS}$.

Calling $X(\tau) \in \mathcal{PS}$ the stochastic process defined by $\tilde{Q}$, the MTTA is defined as the expected time for transitioning into the absorbing state, which can be formalized as

$$\text{MTTA} = \int_0^\infty \mathbb{P}\{X(\tau) \neq N\}\ d\tau. \tag{6}$$

Given the unique absorbing state assumption, $\tilde{Q}$ can be replaced by $\hat{Q}$, the submatrix of $\tilde{Q}$ obtained by removing the last row and column (as shown in [27]), that is

$$\tilde{Q} = \left[\begin{array}{c|c} & v_1 \\ \hat{Q} & \vdots \\ & v_{N-1} \\ \hline 0 \ldots 0 & 0 \end{array}\right] \tag{7}$$

Then the MTTA can be expressed as

$$\text{MTTA} = -\hat{\pi}_0^T \hat{Q}^{-1} \mathbb{1}, \tag{8}$$

where $\hat{\pi}_0$ contains the first $N-1$ entries of π_0, and therefore the problem has been recast into the solution of a linear system.

[4] Notice that this assumption does not imply that $\tilde{Q}$ has an unique row of zeros, as for the case of the stochastic process defined by Q.

6 The KAES Approach

Targeting the efficient evaluation of the MTTA as in (8), the KAES approach develops solutions to treat both the descriptor matrix and the descriptor vector in a symbolic representation. Specifically, KAES is an iterative method, and relies on the following steps:

- A compressed representation scheme for the descriptor vector $\tilde{v}$ is devised by using *tensor trains.* This representation will be used throughout the iterations, and is described in Section 6.1.
- The linear system (8) is solved by a Neumann iteration obtained by splitting the descriptor matrix $\tilde{Q}$ as in (1), and analyzed in Section 6.2.
- The core of the iteration is the inversion of Q_1, which can be efficiently performed in the compressed format using exponential sums; this technique is described in Section 6.3.
- Some further remarks on the efficient computation of the Neumann iteration are reported in Section 6.4.

6.1 Symbolic Representation of the Descriptor Vector

As already discussed when presenting the related work, studies on the symbolic representation of the descriptor matrix in the Kronecker algebra are already well consolidated.

Concerning the descriptor vector, a few approaches have recently appeared on compact representations, as already reviewed in Section 2, but in the context of irreducible CTMC. Here, we exploit the Tensor Train (TT)-representation as in [17], since it is a convenient low-rank tensor format, but addressing CTMC with absorbing states.

We refer the reader to [24] for an overview of the philosophy and the theory of TT tensors, including an accurate description of the truncation procedure.

In a nutshell, a TT-representation of a tensor $\mathcal{X}$ can be given by a tuple $(G_1, \dots, G_n)$ of arrays, where G_1 and G_n are matrices (so they have two indices), and G_j for $j = 2, \dots, n-1$ are order 3 tensors (that is, arrays with 3 indices) such that

$$\mathcal{X}(i_1, \dots, i_n) = G_1(i_1, :)G_2(:, i_2, :) \dots G_{n-1}(:, i_{n-1}, :)G_n(:, i_n),$$

where we have used the MATLAB notation : to denote “slices” of the tensors, and the products are the usual matrix-matrix or matrix-vector products. More precisely, given an array with two indices $G(\alpha, \beta)$, we define $G(:, \beta)$ as the column vector with entry in position α equal to $G(\alpha, \beta)$, and $G(\alpha, :)$ is a row vector with entry in position β equal to $G(\alpha, \beta)$. Similarly, given an array with three indices $G(\alpha, \beta, \gamma)$, we define $G(:, \beta, :)$ as the matrix whose entry (α, γ) is equal to $G(\alpha, \beta, \gamma)$.

The G_j, often called *carriage*, are tensors of dimension $\nu_{j-1} \times N_j \times \nu_j$, where we fix $\nu_0 = \nu_n = 1$ (and thus G_1 and G_n are matrices).

The vector $(\nu_0, \dots, \nu_n)$ is called the TT-rank of the tensor $\mathcal{X}$. In our context, the initial probability vector π_0 and vector $\mathbb{1}$ can be easily expressed in the Kronecker form

$$\pi_0 = \pi_0^{(1)} \otimes \dots \otimes \pi_0^{(n)}, \tag{9}$$

$$\mathbb{1} = \mathbb{1}^{(1)} \otimes \dots \otimes \mathbb{1}^{(n)}, \tag{10}$$

and in TT-format as:

$$\pi_0(i_1, \dots, i_n) = \pi_0^{(1)}(i_1) \cdot \dots \cdot \pi_0^{(n)}(i_n),$$
$$\mathbb{1}(i_1, \dots, i_n) = \mathbb{1}^{(1)}(i_1) \cdot \dots \cdot \mathbb{1}^{(n)}(i_n).$$

Similarly, also the auxiliary vectors necessary to perform the iterative computation of KAES are expressed in TT-format.

The matrix Q and the other auxiliary matrices used in the following have low TT-ranks (and so are expressed in TT-format) when the CTMC is obtained from a loosely interconnected system model, as discussed in Section 4. We refer the reader to [20] for further details on the justification for the presence of such low-rank structures.

TT-format representation is convenient, since it employs $\mathcal{O}(N_{\max} \cdot n \cdot \nu_{\text{eff}}^2)$ flops for each matrix-vector product, instead of the generally larger $\mathcal{O}(N_{\max}^n)$ flops of the corresponding standard representation, where $N_{\max} = \max\{N_1, \dots, N_n\}$ and ν_{eff} is the *effective rank*[5].

When two tensors are added or other matrix operations are performed, the result is still represented in the TT format, but usually with a suboptimal value of the ranks ν_j. For this reason, it is advisable to recompress the result using a rounding procedure, available in the TT-format, that has a complexity $\mathcal{O}(N_{\max} \cdot n \cdot \nu_{\text{eff}}^2 + n \cdot \nu_{\text{eff}}^4)$. When the rank r is low, this number is still very small compared to the number of states, which is $N_{\max}^n$.

Although this unavoidably leads to rounding errors, the accuracy can be chosen by the user. Note that, differently from the floating point arithmetic, the trade-off between the rounding error parameter and the required number of correct digits is more complex to devise, since the computational effort is not an increasing function of the accuracy level.

Often, in the following, TT-tensor will be treated as first-order objects, assuming that the arithmetic on these objects has been overloaded. When this happens, it is assumed that truncation is performed after each operation, to restore an optimal representation of the data.

6.2 Matrix Splitting and Neumann Expansion

In order to exploit the low-rank format, it is necessary to avoid the extraction of the submatrix $\hat{Q}$, since it cannot be directly expressed in the language of

[5] The effective ranks have been obtained through the `erank` function provided by the TT-toolbox [24].

Kronecker algebra. Therefore, an auxiliary rank 1 matrix S that satisfies

$$\hat{\pi}_0^T \hat{Q}^{-1} \mathbb{1} = \pi_0^T (\tilde{Q} - S)^{-1} \mathbb{1}, \tag{11}$$

where $\mathbb{1}$ is the vector of all ones of appropriate dimension, is defined as

$$S = (\tilde{Q}u)u^T - uu^T, \qquad u \in \mathbb{C}^N, \qquad u_j = \begin{cases} 0 & \text{if } j < N \\ 1 & \text{if } j = N \end{cases},$$

where $N = |\mathcal{PS}|$ is the dimension of $\tilde{Q}$. If $\tilde{Q}$ has a low TT-rank, the same holds for $\tilde{Q} - S$, and therefore it can be expected that exploiting an existing TT-enabled system solver to compute the MTTA would maintain the TT-ranks low.

The solvers AMEN [11] and DMRG [23], used in [17] where $\tilde{Q}$ is irreducible, have been tested to solve Equation (11). Unfortunately, there was not always convergence, thus making the measure of interest not assessable in many cases.

For this reason, a different approach has been designed to compute the MTTA. The idea is to make use of the so-called *Neumann expansion*:

$$(I - M)^{-1} = \sum_{j=0}^{\infty} M^j, \tag{12}$$

valid for each matrix M that has spectral radius[6] $\rho(M) < 1$.

The crucial point in KAES is the definition of the splitting of Equation (1) such that $M = -Q_1^{-1}(Q_2 - S)$ verifies the necessary condition for the Neumann expansion applicability and promotes fast evaluation of Q_1^{-1}. This is done in two steps: first a diagonal matrix Δ' is chosen such that $\Delta' \leq \Delta$, and $\Delta' = \Delta'_1 \oplus \ldots \oplus \Delta'_n$ and then Q_1, Q_2 are defined as $Q_1 = \Delta' + R$, and $Q_2 = W + \Delta - \Delta'$. From the definition of Q_1 and Q_2 follows that

$$(\tilde{Q} - S)^{-1} = (I + Q_1^{-1}(Q_2 - S))^{-1} Q_1^{-1}, \tag{13}$$

and it is possible to prove [20] that $\rho(M) < 1$. Using (12) one can approximate the row vector $y = \pi_0^T (\tilde{Q} - S)^{-1}$ by truncating the infinite sum to k terms:

$$y_k = \sum_{j=0}^{k} (-1)^j \pi_0^T (Q_1^{-1}(Q_2 - S))^j Q_1^{-1}. \tag{14}$$

and then compute

$$\text{MTTA} = -y_k \cdot \mathbb{1} + \mathcal{O}(\rho(M)^{k+1}) \tag{15}$$

with a straightforward dot product. The notation $\mathcal{O}(\rho(M)^{k+1})$ is used to indicate that the error is bounded by a constant times $\rho(M)^{k+1}$. The choice of Δ' can be tuned to choose a tradeoff between the speed of convergence and the memory consumption, determined by the rank growth in the iterations.

[6] The spectral radius is defined as the maximum of the modulis of the eigenvalues.

Notice that, defining $z_{k+1} = Q_1^{-1}(Q_2 - S)z_k$ and $z_0 = Q_1^{-1}(\mathbb{1} - e_N^T Q_1^{-1}\mathbb{1} \cdot e_N)$, it is possible to re-write Equation (15) as MTTA $= -\pi_0^T \cdot z_k + \mathcal{O}(\rho(M)^{k+1})$, where $z_{k+1} \geq z_k$ for all $k = 0, 1, \ldots$ because $e_N^T z_0 = 0$ and both Q_1^{-1} and Q_2 are non-negative matrices. This means that the MTTA can be computed in a conservative way, being the approximation $-\pi_0^T \cdot z_k$ a lower bound.

In this paper, a variation of (14) is employed; this modification yields a method with quadratic convergence, overcoming difficulties encountered when $\rho(M)$ gets close to 1. It is based on refactoring $(I - M)^{-1}$ as

$$(I - M)^{-1} = (I + M)(I + M^2) \cdots (I + M^{2^k}) \cdots$$

The downside is that this variation requires to store powers of the matrix $Q_1^{-1}(Q_2 - S)$ in place of just results of matrix vector products and system solves. This has higher memory requirements – but all these matrices are stored in the TT-format, ensuring linear memory storage in the number of subsystems when the TT-ranks (measuring the level of interaction between components) are low.

6.3 Inversion Through Exponential Sums

The main ingredient for implementing KAES is to efficiently evaluate the action of the inverse of Q_1 on a TT-vector and on a TT-matrix. To this aim, in this paper a well-known *exponential sums* construction is adopted. This construction has been used in a variety of contexts (see, for instance, [15,18,19] and the references therein), often being rediscovered by different authors. The construction is built upon a few important observations. The first one is that in Section 6.2 all the addends are expressed as Kronecker sums, namely

$$Q_1 = Q_1^{(1)} \oplus \ldots \oplus Q_1^{(n)}. \tag{16}$$

Thus, a very important property of the standard splitting in Equation (2) is maintaned in the new splitting: all the Kronecker products belong to only one of the splitters, i.e., Q_2, and the Kronecker sums to the other one, namely Q_1.

The second consideration is that, given a TT-tensor $\mathcal{X}$, it is possible to efficiently evaluate the product $\mathcal{Y} = (M_1 \otimes \ldots \otimes M_n)\mathcal{X}$, as this can be performed in $\mathcal{O}(n)$ flops, assuming a low TT-rank for $\mathcal{X}$. Moreover, the result is still a TT-tensor with the same rank.

All the Kronecker products are in Q_2 and the assumption of dealing with loosely interconnected components implies that there are only a few non-identity matrices in W, and then in Q_2. Thus, in this setting $\mathcal{X}$ is the sum of a few terms with TT-rank 1, and consequently has low TT-rank.

The third observation is that, from Equation (16) follows that

$$e^{Q_1} = e^{Q_1^{(1)}} \otimes \ldots \otimes e^{Q_1^{(n)}}. \tag{17}$$

This can be easily proved using the addends defining the Kronecker sum in Equation (16) commute, and that $e^{A+B} = e^A e^B$ whenever $AB = BA$. Then, the conclusion follows by $(I \otimes A)(B \otimes I) = A \otimes B$.

Taking this remarks into account, let to consider the approximated expansion

$$\frac{1}{x} \approx \sum_{j=1}^{\ell} \alpha_j e^{-\beta_j x}, \tag{18}$$

which can be obtained truncating the expansion of $1/x$ to ℓ terms; the error in the approximation on $[1, \infty]$ performed when truncating to ℓ terms can be controlled with a priori estimates. Several constructions are available, we refer the reader to [3] which provides the optimal result, and can guarantee an error term that converges to zero exponentially in ℓ. According to the construction in [20], one can choose the decomposition Q_1 in a way that the eigenvalues of Q_1 are the ones of R shifted to be in the left half of the complex plane. For simplicity, here the case where the eigenvalues of R are real is considered[7] — the general case can be handled with minimal modifications [20].

In particular, the spectrum of Q_1 is contained in $(-\infty, \sigma_{\min}]$, and the action of the inverse can be approximated, applying Equation (17) and Equation (18), as

$$Q_1^{-1} \approx \sum_{j=1}^{\ell} \alpha_j e^{-\beta_j Q_1^{(1)}} \otimes \ldots \otimes e^{-\beta_j Q_1^{(n)}} \tag{19}$$

where $\sigma_{\min}$ is the eigenvalue with minimum modulus of Q_1 and α_j, β_j are computed working on $-\frac{1}{\sigma_{\min}} Q_1$, that has eigenvalues enclosed in $[1, \infty)$.

Since Q_1 is a Kronecker sum, the computation of its eigenvalues can be performed almost for free; in fact, if $Q_1 = Q_1^{(1)} \oplus \ldots \oplus Q_1^{(n)}$ and we denote by $\sigma(Q_1)$ its spectrum,

$$\sigma(Q_1) = \left\{ \sigma_{i_1}^{(1)} + \ldots + \sigma_{i_n}^{(n)} \mid \sigma_{i_k}^{(k)} \in \sigma\left(Q_1^{(k)}\right) \right\}$$

In particular, computing the minimum and maximum eigenvalue just requires to compute the extremal eigenvalues of each factor $Q_1^{(k)}$.

Consequently, the action of the right-side expression in Equation (19) is cheap to evaluate, being the sum of l actions of Kronecker products.

6.4 Efficient Computation of the Neumann Iterations

In computing MTTA, one has to evaluate $-\pi_0^T (Q - S)^{-1} \mathbb{1}$. To accomplish this, it is possible to either evaluate $\pi_0^T (Q - S)^{-1}$ and then compute the dot product with $\mathbb{1}$, or to compute $(Q - S)^{-1} \mathbb{1}$ instead, and take the dot product with π_0.

It can be seen that the former strategy is more convenient. In fact, the graph with M^T as adjacency matrix is a subgraph of the one induced by $\tilde{Q}^T$. In particular, states in $\mathcal{PS} \setminus \mathcal{RS}$ have no impact on the evaluation of MTTA because they correspond to zero entries in π_0, and these entries will remain zero in $\pi_0^T M^k$ for

[7] This assumption is verified in the cases considered in the numerical experiments.

any $k > 0$. This guarantees that this part of the chain has no effect on the computation: there is no need to have an explicit algorithm to detect the reachable states as in [6], because these are implicitly ignored.

Moreover, this strategy is seen to provide lower TT-ranks during the Neumann iterations, compared to computing $(Q - S)^{-1}\mathbb{1}$ first.

This choice has another beneficial effect: the addends in the series (14) are non-negative, and therefore the MTTA is approximated from below — and at every step the partial result is effectively a lower bound [20].

7 Case Study

To illustrate the effectiveness of the proposed approach, we consider a complex computer system composed by n interconnected components C_1,..., C_n, properly functioning at time 0. Each C_i fails after an exponentially distributed time with rate λ_i. With probability p the failed component C_i can be repaired and restarted as properly functioning after an exponentially distributed time with rate μ_i. Instead, with probability $1 - p$ the failure of C_i propagates instantaneously to all the components directly interconnected to it. In this case, all the failed components cannot be repaired. The list of the $\bar{d}_i$ indexes of the components where the failure of C_i can propagate is $\bar{D}_i = \{h_1, h_2, \ldots, h_{\bar{d}_i}\}$. The list of the d_i indexes of the components whose failure can propagate to C_i is $D_i = \{j_1, j_2, \ldots, j_{d_i}\}$. The topology of interactions among components is given by the $n \times n$ adjacency matrix $\mathcal{T} = [\mathcal{T}_{i,j}]$, where $\mathcal{T}_{i,j} = 1$ if $j \in \bar{D}_i$, else $\mathcal{T}_{i,j} = 0$. Thus, $\mathcal{T}$ defines an oriented graph that represents how the n components depend on each other and how they are connected to form the overall system. Although different topologies $\mathcal{T}$ can be defined, for example when different access rights to components are defined for different types of service or customers, for the sake of simplicity only one topology $\mathcal{T}$ is considered in the following.

7.1 Model of the Case Study

The SGSPN model representing the overall system of the case study is obtained defining a submodel for each single component C_i, with $i = 1, \ldots, n$, and composing all such submodels through a transition-synchronization approach, as described in Section 4. The model of the component C_i is depicted in Figure 1. The places On_i (initialized with one token), $Down_i$ and F_i are local to the model and represent the states where, respectively, C_i works properly (one token in On_i), is under repair (one token in $Down_i$), and is failed and cannot be repaired. The transitions $TDown_i$ and TOn_i are local to the model and represent respectively the exponentially distributed time with rate $p_i\lambda_i$ to the occurrence of a failure, when the failed component can be repaired, and the exponentially distributed time with rate μ_i after which the component returns to operate properly. The transitions $TFail_i$ and $TFail_{j_k}$ with $k = 1, \ldots, d_i$ are synchronization transitions used to synchronize the models representing each component of the

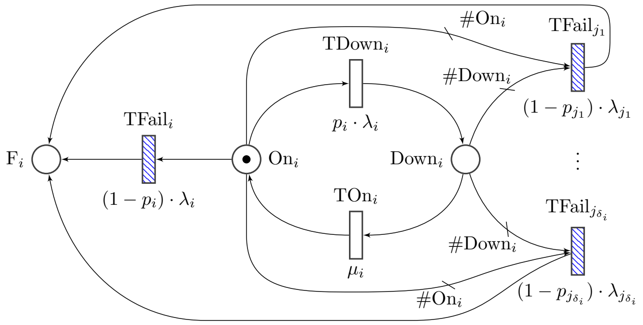

Fig. 1. Model of the component C_i of the case study. The shaded transitions are synchronization transitions.

system, i.e., to propagate the failure that affects C_i to its neighbors with probability $1-p_i$. $TFail_i$ represents the exponentially distributed time with rate $(1-p_i)\lambda_i$ to the occurrence of a failure on C_i, that instantaneously propagates to C_h, with $h \in \bar{D}_i$ (without the possibility to repair the failed components). $TFail_i$ is replicated in the models of C_i and C_h, for each $h \in \bar{D}_i$. In each C_h model, it exists a transition $TFail_{j_k}$ with $j_k = i$, synchronized with $TFail_i$, that propagates the failure occurred in C_i. The transitions $TFail_{j_k}$ for each $k = 1,\ldots,d_i$ in Figure 1 represent the time to the occurrence of a failure on C_{j_k} that instantaneously propagates to C_i (without the possibility to repair the failed components). Each transition $TFail_{j_k}$ is replicated in the models of C_i and C_h with $j_k \in D_h$. In each model C_h exists a transition $TFail_h$ with $j_k = h$, synchronized with $TFail_{j_k}$, that represents the occurrence of the failure in C_h that propagates to C_i.

In absence of immediate transitions, a synchronized transition is enabled when it, and all the transitions synchronized with it, have concession. As shown in Figure 1, the transition $TFail_i$ has concession when one token is in the place On_i. All the transitions $TFail_{j_k}$, for $k = 1,\ldots,d_i$, have always concession, being the multiplicity of each input arc equal to the number of tokens in the corresponding input place On_i and $Down_i$, as shown in Figure 1. Thus, $TFail_i$ is enabled when there is one token in On_i. The firing of $TFail_i$ occurs simultaneously in the model of C_i where $TFail_i$ removes the token from On_i and adds one token to F_i (the component is failed and cannot be repaired), and in the model of C_h, for each $h \in \bar{D}_i$, where, as shown in Figure 1 replacing i with h, $TFail_{j_k}$, with $j_k = i$, removes one token from On_h and $Down_h$ (if any) and adds one token to F_h (the failure of C_i propagated to C_h that cannot be repaired).

On the model the following reward structure is considered

$$r = \begin{cases} 1 & \text{if } \#\mathrm{F}_i = 1 \text{ for all } i, \\ 0 & \text{otherwise,} \end{cases}$$

so that the mean time to system failure τ_F corresponds to the cumulative measure defined by r on the interval of time $[0, \infty)$.

The model depicted in Figure 1 can be classified as a reliability model [27] and produces a CTMC with a unique absorbing state. The model of Figure 1 is structurally bounded, being a stochastic finite state machine and the corresponding stochastic automaton is characterized by:

$$R^{(i)} = \begin{pmatrix} 0 & p_i\lambda_i & 0 \\ \mu_i & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \qquad W^{(t,i)} = \begin{cases} \begin{pmatrix} 0 & 0 & (1-p_i)\lambda_i \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} & \text{if } j = i \\ \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{pmatrix} & \text{if } j \neq i \text{ and } \mathcal{T}(i,j) = 1 \\ \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} & \text{otherwise} \end{cases} \tag{20}$$

The mean time to system failure τ_F is then evaluated through the MTTA, because of the correspondence between the unique absorbing state of the CTMC and the system failure state.

8 Evaluation Results

In this section, details on how the case study described in Section 7 has been evaluated through KAES are discussed, and the obtained results in terms of time and memory consumption are presented in Table 1. The analysis is carried out for different numbers of components $n = 10, 20, \ldots, 50$ and generating random topologies of interactions following a predefined template. Results are obtained implementing[8] the case study SAN, i.e., Equation (20), and the KAES method in MATLAB [21].

As a form of validation of KAES, it has been verified that for $n \leq 10$ the values obtained for the mean time to system failure τ_F with KAES coincide with the values obtained with the standard technique (full exploration of $\mathcal{RS}$ followed by the linear system of Equation (8) solution). However, since here the analysis focuses on assessing the efficiency of the proposed method, the results obtained for τ_F are out of the scope of this paper, and then are not shown.

In order to demonstrate the ability of KAES in improving on current limitations suffered by standard techniques, analyzed scenarios are characterized by: 1) both $\mathcal{PS}$ and $\mathcal{RS}$ are large; 2) the model parameters define a *stiff* [27] CTMC. In particular, $|\mathcal{PS}| = 3^n$, being $|\mathcal{RS}^{(i)}| = 3$, and $|\mathcal{RS}|$ depends on $\mathcal{T}$.

[8] `https://github.com/numpi/kaes`

Note that if $\mathcal{T}$ represents the complete graph of interdependencies then $|\mathcal{RS}|$ is trivially small. In fact, the initial state is the one with 1 token in On_i for all $i = 1, \ldots, n$, and when the first $TFail_j$ fires, all the tokens are removed from On_i for all i; thus, the system only has two reachable states.

Therefore, to have large $|\mathcal{RS}|$, the topology $\mathcal{T}$ of interactions is obtained as follows: first, a star topology is constructed, where, labeling the nodes from 1 to n, there exist $n-1$ edges connecting 1 to j, for $j = 2, \ldots, n$. Then, for each node with index greater than 1, another edge connecting it to a random node is added with probability 0.2. Although artificially generated, such topologies are good representatives of topologies addressed by KAES (large number of components, loosely interconnected), and are suitable for the adopted case study.

The parameters for each M_i have been randomly selected, but aiming at obtaining a stiff CTMC. Specifically, in the performed evaluations they are:

$$\lambda_i \in [0.5, 1.5], \quad \mu_i \in [2000, 3000], \quad p_i \in [0.95, 1],$$

so that there are 4 orders of magnitude among the parameters. The tests have been repeated 100 times for each value of n, using the randomized topology described above. For large n, not all the cases could be solved using the available system memory. The percentage of cases exceeding the available memory is reported in Table 1 for each n.

The average amount of CPU (user and system) time (in seconds) and the average amount of the RAM memory (in GB) consumed by KAES have been quantified. The averages are computed only on those cases where KAES was successful. Computations were performed on a Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20GHz, where each experiment had 12 CPUs and 120GB of RAM at its disposal. As shown in Table 1, the actual memory consumption for all the values of n is much lower than the maximum available.

Note that, although not reported in the table, the standard approach was not able to complete the state space exploration for $n \geq 20$.

Table 1. Potential spaces dimensions, memory consumption, time and number of cases where the KAES approach was successful, where μ reports the average over the 100 runs and σ is the standard deviation.

n	$\lvert\mathcal{PS}\rvert$	memory (Gb)		time (s)		% solved
		μ	σ	μ	σ	cases
10	59049	0.90	0.08	1.17	0.81	100%
20	$3.49 \cdot 10^9$	3.07	9.68	65.83	346.24	100%
30	$2.06 \cdot 10^{14}$	8.31	19.40	193.29	619.63	91%
40	$1.22 \cdot 10^{19}$	4.42	9.97	140.89	477.67	91%
50	$7.18 \cdot 10^{23}$	7.79	17.27	299.44	840.78	84%

The method is able to solve the great majority of cases, although the rate of success decreases as the number of components increases. For n equal to 10 and 20, all cases are solved, and the lowest percentage is 84 for the most populated scenario ($n = 50$). An important observation is that time and memory consumptions seem to depend on the adopted topology, and in fact they can vary significantly for the different topologies generated for a given n, as confirmed by the values of the standard deviations reported in Table 1. However, it is not straightforward to understand the phenomena leading to this result, namely whether it is strictly related to the theoretical definition of the KAES method or to its implementation (especially, how the rounding is performed since the adopted toolbox for this procedure is a general one), or to both of them. Further investigations are necessary to shed light on this aspect, so to promote refinements in the KAES methodology and/or implementation.

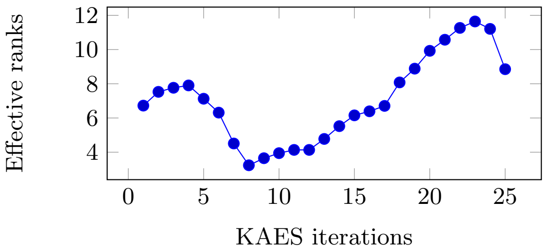

Fig. 2. Evolution of the effective ranks, representing an average of the TT-ranks of the carriages, for each iteration of KAES, with $n = 20$ and a specific topology.

To confirm the low memory consumption, in Figure 2 the evolution of the ranks (represented using the "effective ranks", a single number that measures an average of the TT-ranks in the various modes) is reported. The ranks are considerably small, compared to $|\mathcal{PS}|$, and for all the experiments evolve in a similar way. This is a strong indicator that KAES has been well conceived as an efficient solution method.

9 Conclusions and Future Work

This paper addressed analytical modeling of large, interconnected systems by developing a new numerical evaluation approach, called KAES, to efficiently evaluate the Mean-Time-To-Absorption in CTMC with absorbing states. Resorting to powerful mathematical theories, properly combined, the symbolic representation of both the descriptor matrix and the descriptor vector is pursued to mitigate the explosion of the state space when evaluating the stochastic model. Although symbolic representation has been already applied in existing studies, such previous works focus on steady-state analysis while KAES targets limiting analysis in presence of absorbing states.

KAES has been implemented in the MATLAB evaluation environment and compared with traditional numerical solution when applied to a representative case study for the evaluation of the MTTA. Although preliminary and restricted to the studied scenario, obtained results clearly show the feasibility of KAES at increasing the size of the system under analysis, while standard numerical approaches fail due to the generated state space being too large. Moreover, the way the measure is computed guarantees a conservative assessment, which is relevant when dealing with dependability critical applications.

Of course, more experiments are needed to better understand strengths and limitations of this new technique in a variety of system scenarios, at varying both the system topology and the parameters setting. In particular, a deeper understanding of the link between TT-ranks and the topology of interactions among system components would be desirable, since the memory consumption is strictly related to TT-ranks. This extended evaluation campaign is already in progress. The obtained outcomes are expected to trigger improvements at methodology and/or implementation level.

Further and most important, the powerfulness of the adopted techniques and the conceived organization of the KAES steps make this method not restricted to the evaluation of the MTTA measure only, but adaptable to evaluate general performability related indicators. In fact, a straightforward generalization of KAES is represented by the substitution of the all-ones-vector $\mathbb{1}$ in Equation (8) with a more general reward vector r, to promote the evaluation of other performance and dependability properties of single absorbing state CTMC, expressed as cumulative measures over the interval $[0, \infty)$. Whenever r can be expressed in terms of AND and OR conditions based on $r^{(i)}$, defined on M_i, it is possible to write r in terms of Kronecker products and sums, and apply KAES as presented in this paper.

References

1. M. Ajmone Marsan, G. Balbo, G. Chiola, G. Conte, S. Donatelli, and G. Franceschinis. An introduction to generalized stochastic Petri nets. *Microelectronics and Reliability*, 31(4):699–725, 1991.
2. J. Babar, M. Beccuti, S. Donatelli, and A. Miner. Greatspn enhanced with decision diagram data structures. *Applications and Theory of Petri Nets*, pages 308–317, 2010.
3. D. Braess and W. Hackbusch. Approximation of $1/x$ by exponential sums in $[1, \infty)$. *IMA journal of numerical analysis*, 25(4):685–697, 2005.
4. L. Brenner, P. Fernandes, A. Sales, and T. Webber. A framework to decompose GSPN models. In *International Conference on Application and Theory of Petri Nets*, pages 128–147, 2005.
5. P. Buchholz, T. Dayar, J. Kriege, and M. C. Orhan. On compact solution vectors in Kronecker-based Markovian analysis. *Performance Evaluation*, 115:132–149, 2017.
6. P. Buchholz and P. Kemper. Kronecker based matrix representations for large Markov models. *Validation of Stochastic Systems: A Guide to Current Research*, pages 256–295, 2004.
7. G. Ciardo. Data representation and efficient solution: A decision diagram approach. In *Formal Methods for Performance Evaluation: 7th International School on Formal Methods for the Design of Computer, Communication, and Software Systems*, pages 371–394. Springer Berlin Heidelberg, 2007.

8. G. Ciardo and A. S. Miner. A data structure for the efficient Kronecker solution of GSPNs. In *Proceedings 8th International Workshop on Petri Nets and Performance Models (Cat. No.PR00331)*, pages 22–31, 1999.
9. G. Ciardo and M. Tilgner. On the use of Kronecker operators for the solution of generalized stochastic Petri nets. 1996. Nasa Technical Report Server 20040110963.
10. R. M. Czekster, P. Fernandes, J.-M. Vincent, and T. Webber. Split: A flexible and efficient algorithm to vector-descriptor product. In *VALUETOOLS*, pages 83:1–83:8, 2007.
11. S. V. Dolgov and D. V. Savostyanov. Alternating minimal energy methods for linear systems in higher dimensions. *SIAM Journal on Scientific Computing*, 36(5):A2248–A2271, 2014.
12. S. Donatelli. Superposed stochastic automata: A class of stochastic Petri nets with parallel solution and distributed state space. *Performance Evaluation*, 18(1):21–36, 1993.
13. S. Donatelli. Superposed generalized stochastic Petri nets: Definition and efficient solution. In *Application and Theory of Petri Nets 1994*, pages 258–277, 1994.
14. K. Goševa-Popstojanova and K. Trivedi. Stochastic modeling formalisms for dependability, performance and performability. In *Performance Evaluation: Origins and Directions*, pages 403–422, 2000.
15. L. Grasedyck. Existence and computation of low Kronecker-rank approximations for large linear systems of tensor product structure. *Computing*, 72(3-4):247–265, 2004.
16. P. Kemper. Numerical analysis of superposed GSPNs. *IEEE Transaction Software Engineering*, 22(9):615–628, Sept. 1996.
17. D. Kressner and F. Macedo. Low-rank tensor methods for communicating Markov processes. *Quantitative Evaluation of Systems*, pages 25–40, 2014.
18. D. Kressner and C. Tobler. Krylov subspace methods for linear systems with tensor product structure. *SIAM journal on matrix analysis and applications*, 31(4):1688–1714, 2010.
19. W. Kutzelnigg. Theory of the expansion of wave functions in a Gaussian basis. *International journal of quantum chemistry*, 51(6):447–463, 1994.
20. G. Masetti and L. Robol. Tensor methods for the computation of MTTF in large systems of loosely interconnected components. 2019. Technical report, ISTI-CNR Open Portal.
21. MathWorks. *MATLAB R2018a*. The Mathworks, Inc., 2018.
22. T. Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580, 1989.
23. I. Oseledets. DMRG approach to fast linear algebra in the TT-format. *Computational Methods Applied Mathematics*, 11(3):382–393, 2011.
24. I. V. Oseledets. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011.
25. B. Plateau, J.-M. Fourneau, and K.-H. Lee. Peps: A package for solving complex Markov models of parallel systems. In *Modeling Techniques and Tools for Computer Performance Evaluation*, pages 291–305, 1989.
26. W. H. Sanders and J. F. Meyer. A unified approach for specifying measures of performance, dependability and performability. *Dependable Computing for Critical Applications, Vol. 4 of Dependable Computing and Fault-Tolerant Systems*, pages 215–237, 1991.
27. K. S. Trivedi and A. Bobbio. *Reliability and Availability Engineering: Modeling, Analysis, and Applications*. 2017.