Tensor methods for the computation of MTTF in large systems of loosely interconnected components

Giulio Masetti, Leonardo Robol

February 1, 2019

Abstract

We are concerned with the computation of the mean-time-to-failure (MTTF) for a large system of loosely interconnected components, modeled as continuous time Markov chains. In particular, we show that splitting the local and synchronization transitions of the smaller subsystems allows to formulate an algorithm for the computation of the MTTF which is proven to be linearly convergent. Then, we show how to modify the method to make it quadratically convergent, thus overcoming the difficulties for problems with convergent rate close to 1.

In addition, it is shown that this decoupling of local and synchronization transitions allows to easily represent all the matrices and vectors involved in the method in the tensor-train (TT) format — and we provide numerical evidence showing that this allows to treat large problems with up to billions of states — which would otherwise be unfeasible.

1 Introduction

2 Stochastic Automata Networks

As described in [4], it is possible to address the study of large CTMC defining symbolically the infinitesimal generator matrix Q, so that avoiding a complete state-space exploration. In particular, the Stochastic Automata Network (SAN) formalism [22] allows to represent the CTMC as a set of stochastic automata $M_1, \ldots, M_k$, each having a (small) reachable set of state $\mathcal{RS}_i$, where transitions among states are of two kinds: *local* and of *synchronization*. Transitions t that are local to M_i, written $t \in \mathcal{LT}_i$, have impact only on $\mathcal{RS}_i$ and indicate the switch from a state $s \in \mathcal{RS}_i$ to a state $s' \in \mathcal{RS}_i$, in the following written $s \xrightarrow{t} s'$. Synchronization transitions $t \in \mathcal{ST}$, instead, can appear in more than one automaton. In particular, if $t \in M_{i_1}$, $t \in M_{i_2}$, ..., and $t \in M_{i_h}$ then $s_{i_j} \xrightarrow{t} s'_{i_j}$ can fire only if the automaton M_{i_1} is in state s_{i_1}, and the automaton M_{i_2} is in state s_{i_2}, ..., and the automaton M_{i_h} is in state s_{i_h}, at the same time.

The overall CTMC is then the orchestration of local stochastic automata where the director is represented by $\mathcal{ST}$. The infinitesimal generator matrix Q is not assembled, its representation is called *descriptor matrix* and is formally defined by

$$Q = R + W + \Delta, \tag{1}$$ {{eq:descriptorQ}}

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

$$R = \bigoplus_{i=1}^{k} R^{(i)}, \tag{2}$$

$$W = \sum_{t_j \in \mathcal{ST}} \bigotimes_{i=1}^{k} W^{(t_j,i)}, \tag{3}$$

$R^{(i)}$ and $W^{(t_j,i)}$ are $|\mathcal{RS}^{(i)}| \times |\mathcal{RS}^{(i)}|$ matrices, and the diagonal matrix Δ is defined as $\Delta = -\text{diag}\,((R+W)e)$ and the operator $\oplus$ is the *Kronecker sum*, as formally described in Section 3.1. The matrices $R^{(i)}$ and $W^{(t_j,i)}$ are assembled exploring $\mathcal{RS}^{(i)}$ and can be specified through an high level formalism such as GSPN [11, 5] or PEPA [12]. In particular, $W^{(t_j,i)} = \lambda_{t_j} \tilde{W}^{(t_j,i)}$ where $\tilde{W}^{(t_j,i)}$ is a $\{0,1\}$-matrix defined as follows:

$$\tilde{W}^{(t_j,i)}_{s_i,s_i'} = \begin{cases} 1 & \text{if } t_j \text{ is enabled in } s_i \text{ inside } M_i \text{ and } s_i \xrightarrow{t_j} s_i' \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where λ_{t_j} is the constant rate associated to t_j, equal in every M_i. In particular, if the transition t_j has no effect on the component M_i, we have $\tilde{W}^{(t_j,i)} = I$. In the following we will call

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

the *potential state space* and the $|\mathcal{PS}| \times |\mathcal{PS}|$ descriptor matrix Q will be treated implicitly.

3 Low-rank tensors

3.1 Kronecker sums

Often, high-dimensional problems are formulated as the sum of 1D (or, more generally, low-dimensional) operators acting on a subset of the dimensions. The most classical example For instance, one may consider the Poisson problem on the d-dimensional box $[0,1]^d$ defined by the PDE

$$\Delta u = f, \qquad u : [0,1]^d \to \mathbb{R}, \qquad \Delta u := \sum_{i=1}^{d} \frac{\partial^2 u}{\partial x_i^2}.$$

If we denote by ∂^2 the 1-dimensional second derivative operator on $[0,1]$, then we can write

$$\Delta = \underbrace{\partial^2 \otimes I \otimes \ldots \otimes I}_{\frac{\partial^2}{\partial x_1^2}} + \underbrace{I \otimes \partial^2 \otimes \ldots \otimes I}_{\frac{\partial^2}{\partial x_2^2}} + \ldots \underbrace{I \otimes \ldots \otimes I \otimes \partial^2}_{\frac{\partial^2}{\partial x_d^2}}.$$

We shall use a special notation to denote these operators, and we give it the name *Kronecker sum*; we write $A_1 \oplus \ldots \oplus A_d$ to mean the operator defined by the sum of A_1 operating on the first dimension, A_2 on the second, up to A_d on the last dimension. In particular, we have

$$\Delta = \frac{\partial^2}{\partial x_1^2} \oplus \cdots \oplus \frac{\partial^2}{\partial x_d^2}.$$

In our setting, we will not deal directly with such differential problems, yet the same structure will appear in the definition of the infinitesimal generator Q of the Markov chains. This is due to the modelization of large and complex systems as a union of smaller (and simpler) subsystems, which only have "weak" interactions. In this case, the infinitesimal generator is not exactly in the form of a Kronecker sum, but it has a low-rank tensorial structure.

The term "tensor rank" does not have a single universally accepted meaning. In fact, unlike in the matrix case (that is, $d = 2$), several different ranks can be defined — and they have different computational properties. The most classical definition is the CPD rank, linked to the Canonical Polyadic Decomposition (also sometimes called PARAFAC — see [14] and the references therein for more details); this is linked to the definition of rank as sum of rank 1 terms, which are in turn defined as outer product $v_1 \otimes \cdots \otimes v_d$. However, using this low-rank format is inherently difficult and unstable. For instance, the set of rank R tensors is not closed if $d > 2$, and this makes the low-rank approximation problem ill-posed [8]. Moreover, the computation of the best rank r approximation of a tensor is a difficult problem, and the solution can only be approximated by carefully adapted optimization algorithms, see [15] for a review.

For this reason, there has been interest in finding alternative low-rank representation of tensors. A very robust possibility that is well-understood is the Tucker decomposition, linked to the Higher Order SVD (HOSVD) [7]. However, this approach requires to store a d-dimensional tensor (even though of smaller sizes), and so is only suited for small values of d.

When one is faced with the task of working with high values of d (say, $d > 5$), and a small number of entries for each mode – a natural choice are instead tensor-train [20] or the hierarchical tucker decomposition [16].

We shall concentrate on the former choice, and in the next section we briefly recall the main tools that we use in the rest of the paper.

3.2 Tensor-train format

When a Markov chain is modeled combining the states spaces of smaller stochastic process, the set of potential states is obtaining combining the possible states

of the smaller components; therefore, the growth in the number of states is *exponential* in the number of subsystems. This phenomenon is often known under the name of "curse of dimensionality", and several approaches have been considered in the literature to address the difficulties that arise in the analysis of such systems.

One technology that allows to treat these problems, and has already been applied to Markov chains (see [2, 15]), is the one of *tensor trains* (TT). This tools has been used successfully to deal with high-dimensional PDEs such as the one given as representative example in the previous section [10, 13] and for integration in time [17]. These problems are strictly related to the setting of Markov chain, where the probability distribution at a certain time t solves a linear ODE described by the infinitesimal generator Q, as we now describe more in detail.

Let us consider a large system composed by k smaller subsystems, each with n_i states, $i = 1, \dots, k$. The potential state space $\mathcal{PS}$ can then be written as

$$\mathcal{PS} := \{1, \dots, n_1\} \times \dots \times \{1, \dots, n_k\}.$$

At each time t, the probability vector $\pi(t)^T = \pi_0^T e^{tQ}$ can be expressed in tensor form, as an array with k indices $\pi(t) = \pi(i_1, \dots, i_k)$. A tensor train representation of a tensor v is a collection of order 3 tensors M_i of size $r_i \times n_i \times r_{i+1}$ such that $r_1 = r_k = 1$, and

$$v_{i_1,\dots,i_k} = \sum_{t_1,\dots,t_{k-1}} M_{1,i_1,t_1} M_{t_1,n_2,t_2} \dots M_{t_{j-1},n_j,t_j} \dots M_{t_{k-1},n_k,1}.$$

The core matrices M_i are called *carriages*, hence the name *tensor train* [20]. The tuple $(r_2, \dots, r_{k-1})$ is called the TT-rank of the tensor v. Similarly, matrices $m \times n$ can be represented as tensors by subdividing the row and column indices. The TT format has been very successful in overcoming dimensionality problems in several frameworks, such as high-dimensional PDEs []. We shall demonstrate that their use is beneficial in the the computation of performance and reliability measures as well.

On the software side, a well-established framework [21] is available for Python and MATLAB[19] We rely on the latter for our numerical experiments.

3.3 Exponential sums

Given a Kronecker sum $\mathcal{A} := A_1 \oplus \dots A_d$, we are interested in efficiently computing its inverse, or the solution of a linear system $\mathcal{A}x = b$. Given the relevance of this problem in high-dimensional PDEs and various other settings of applied mathematics, several approaches have been devised over the years. In this section we briefly recall the one known after the name of *exponential sums* [3].

For the sake of self-completeness, we briefly recall the main important facts related to this topic. The idea behind exponential sums is to rephrase the inverse as a combination of matrix exponentials. The latter are much easier to compute for a Kroencker sum, as stated by the next Lemma.

Lemma 3.1. *Let $\mathcal{A} = A_1 \oplus \ldots \oplus A_d$ be a Kronecker sum. Then, the matrix exponential $e^{\mathcal{A}}$ is given by*

$$e^{\mathcal{A}} = e^{A_1} \otimes e^{A_2} \otimes \cdots \otimes e^{A_d}.$$

Proof. It suffices to recall that $e^{A+B} = e^A e^B$ whenever A and B commute; clearly, all the addends in the sum defining $\mathcal{A}$ commute, and the result follows by $e^{A \otimes B} = e^A \otimes e^B$. □

In view of the previous result, assume we know an expansion of $\frac{1}{x}$ of the following form:

$$\frac{1}{x} = \sum_{j=1}^{\infty} \alpha_j e^{-\beta_j x}, \qquad \forall x \in \Lambda(\mathcal{A}),$$

where $\Lambda(\cdot)$ denote the spectrum of the operator. Then,

$$\mathcal{A}^{-1} = \sum_{j=1}^{\infty} \alpha_j e^{-\beta_j \mathcal{A}}.$$

Truncating the above series yields an approximation of the inverse, and the matrix exponential are very cheap to compute if one knows the factors A_i. It remains to construct a method to efficiently compute α_j and β_j of such an expansion. We say that an exponential sum has accuracy $\epsilon > 0$ on the interval $[a, b]$ if, for any $x \in [a, b]$, we have $\left| \frac{1}{x} - \sum_{j=1}^{k} \alpha_j e^{-\beta_j x} \right| \leq \epsilon$.

We shall make two assumptions. First, we ask that the operator $\mathcal{A}$ has only real and positive eigenvalues; then, we ask that all the matrices A_i (and therefore $\mathcal{A}$ as well) are *normal* matrices. As we will see later, both assumption can be relaxed, but not removed completely. However, for now they make the analysis much easier.

Lemma 3.2 (Crouzeix–Palencia [6])**.** *Let A be any matrix, $f(z)$ a function, and consider its* field of values *defined as $\mathcal{W}(A) := \{x^H A x \in \mathbb{C} \mid \|x\|_2 = 1\}$. Then,*

$$\|f(A)\|_2 \leq (1 + \sqrt{2}) \max_{z \in \mathcal{W}(A)} |f(z)|.$$

Moreover, if A is normal, then $\|f(A)\| = \max_{z \in \Lambda(A)} |f(z)|$, where $\Lambda(A)$ is the set of eigenvalues of A.

Proof. The results for normal matrices is easily provable since A can be diagonalized by an orthogonal (or unitary) transformation, and then $f(A)$ can be recasted to computing $f(D)$, with D diagonal containing the eigenvalues of A.

On the other hand, the result for general matrices needs to consider the field of values, and is far from being trivial. We refer to the work of Crouzeix and Palencia for the proof [6]. □

The strength of Lemma 3.2 is that it allows to relate the quality of a good approximation of a function with its evaluation as a matrix function. In particular, we have the following.

Lemma 3.3. *Let α_j, β_j be the coefficients of an exponential sum with accuracy ϵ over $[1, R]$. Then, if A is a Hermitian matrix with spectrum contained in $[1, R]$, we have*

$$\left\| A^{-1} - \sum_{j=1}^{k} \alpha_j e^{-\beta_j A} \right\|_2 \leq \epsilon$$

Proof. The result is a straightforward application of Lemma 3.2 to the function $f(z) = \frac{1}{z} - \sum_{j=1}^{k} \alpha_j e^{-\beta_j z}$ over the domain $[1, R]$. □

The previous result implies that, given a (normal) matrix in Kronecker sum form $\mathcal{A} = A_1 \oplus \ldots \oplus A_k$, to achieve an accuracy ϵ in the computation of $\mathcal{A}^{-1}$ or, equivalently, in the solution of the linear system $\mathcal{A}x = b$, we shall obtain an exponential sum with coefficients α_j, β_j achieving that accuracy ϵ over the eigenvalues of $\mathcal{A}$. The construction of the coefficients α_j, β_j is beyond the scope of this paper; our construction relies on [3].

Lemma 3.2 can be used instead when the matrix is non-normal and/or has complex eigenvalues. However, one needs to rely on approximation methods and it more difficult to provide explicit bounds. We refer to [3] and the references therein for further information.

4 Computing the MTTF

In [18] it has been shown the computation of several perfomability measures can be recasted as the evaluation of a matrix function. In most cases, one has to compute $w^T f(Q)v$ for appropriate vectors v, w, and a certain function $f(z)$. In this work, we focus on the computation of the *mean-time-to-absorption*. We assume that the Markov process has a single absorbing state, and without loss of generality we assume it to have index N. Then, we compute the quantity

$$\text{MTTA} = \int_0^\infty \mathbb{P}\{X(\tau) < N\}\ d\tau = \mathbb{E}\left[\int_0^\infty \mathbb{1}_{\{1,\ldots,N-1\}}(X(\tau))\ d\tau \right].$$

Often, we are interested in the case where the absorbing state corresponds to the failure state of the system. In this case, we use the name *mean-time-to-failure* and the notation MTTF. We assume that the matrix Q is partitioned as follows:

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

where the last row is forced to be zero because the state N is absorbing. Following [23], we know that

$$\text{MTTA} = -\hat{\pi}_0^T \hat{Q}^{-1} e = \pi_0^T f(Q) e, \qquad f(z) = \begin{cases} -\frac{1}{z} & z \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$ {{eq:mtta}}

where e is the vector of all ones, and $\hat{\pi}_0$ contain the first $N-1$ entries of π_0. In this work, we are interested in the case where Q can be efficiently represented in the TT format. The TT format gives many computational advantages, but does not provide an easy way to extract a submatrix. In fact, the matrix $\hat{Q}$ might not have any particular tensor structure. To overcome this drawback, we introduce an auxiliary matrix S which allows to rephrase the measure using the inverse of a low-rank perturbation of Q.

Lemma 4.1. *Let Q the infinitesimal generator of a continuous time Markov chain with exponential rates with N states; assume that the state N is the only failure state, and let S be the rank 1 matrix defined as*

$$S = (Qe_N)e_N^T - e_N e_N^T.$$

Then, if π_0 has the N-th component equal to 0, we have $\mathrm{MTTA} = -\pi_0^T(Q - S)^{-1}e$*, where e is the vector of all ones.*

Proof. By construction, we have that $Q - S$ is block diagonal and therefore

$$(Q-S)^{-1} = \begin{bmatrix} \hat{Q} & \\ & 1 \end{bmatrix}^{-1} = \begin{bmatrix} \hat{Q}^{-1} & \\ & 1 \end{bmatrix},$$

and $-\pi_0^T(Q-S)^{-1}e = -\hat{\pi}_0^T\hat{Q}^{-1}e - [\pi_0]_N = \mathrm{MTTA} - [\pi_0]_N$. We conclude noting that $[\pi_0]_N = 0$. □

Remark 4.2. Note that the TT-rank of S is $(1, \dots, 1)$, since it is a matrix of rank 1, and if Q has a low TT-rank the same holds for $Q - S$.

The important consequence of the previous Lemma is that, even though we cannot extract a submatrix from Q to compute the MTTF, we can make a rank 1 (in the TT sense) perturbation that makes Q invertible, and still delivers the same result.

Lemma 4.3. *Let $v = e + \gamma e_N$, for any $\gamma \in \mathbb{R}$. Then, with the notation of Lemma 4.1, we have*

$$\mathrm{MTTA} = -\pi_0^T(Q-S)^{-1}e = -\pi_0^T(Q-S)^{-1}v.$$

Proof. Note that, $[\pi_0]_N = 0$, and therefore

$$-\pi_0^T(Q-S)^{-1}v = -\begin{bmatrix} \hat{\pi}_0^T\hat{Q}^{-1} & 0 \end{bmatrix} \begin{bmatrix} 1 \\ \vdots \\ 1 \\ 1+\gamma \end{bmatrix} = -\hat{\pi}_0^T\hat{Q}^{-1}e = \mathrm{MTTA}.$$

□

We have recasted the problem in solving a linear system $(Q-S)x = e$, where the matrix $Q - S$ is expressed in TT format. Unfortunately, as we will see

later on, a few problem of interest for us do not play very well together with the more widespread tensor train system solvers (such as AMEN [] or DMRG []). For this reason, we propose a different solution scheme based on the *Neumann expansion*. In particular, let M be any matrix with spectral radius strictly smaller than 1. Then,

$$(I-M)^{-1} = I + M + M^2 + M^3 + \ldots = \sum_{j=0}^{\infty} M^j \tag{6}$$ {{eq:neumann}}

If we partition Q as $Q = Q_1 + Q_2$, we can write

$$(Q-S)^{-1} = (Q_1 + Q_2 - S)^{-1} = (I + Q_1^{-1}(Q_2 - S))^{-1} Q_1^{-1}.$$

Setting $M = -Q_1^{-1}(Q_2 - S)$, assuming that $\rho(M) < 1$ and using the Neumann expansion (6) we obtain

$$(Q-S)^{-1} = \sum_{j=0}^{\infty} (-1)^j (Q_1^{-1}(Q_2 - S))^j Q_1^{-1}. \tag{7}$$

The above formula can be used to approximate $x = (Q-S)^{-1}e$ as needed for (5) by truncating the infinite sum to k terms:

$$x_k = \sum_{j=0}^{k} (-1)^j (Q_1^{-1}(Q_2 - S))^j Q_1^{-1} e, \qquad \|x - x_k\|_\infty \approx \mathcal{O}(\rho(M)^{k+1}).$$

Our main contribution is to give an explicit method to construct the additive splitting $Q = Q_1 + Q_2$ so that M is guaranteed to have spectral radius less than 1. This will be achieved in Theorem 4.7. The pseudocode describing the resulting method is presented in Algorithm 1.

Algorithm 1 Neumann series (6) to approximate $x = (Q-S)^{-1}e$

```
procedure NeumanSeries(Q_1, Q_2, k)
    y ← Q_1^{-1} e
    x ← y
    for j = 1, ..., k do
        y ← −Q_1^{-1}(Q_2 − S)y
        x ← x + y
    end for
    return x
end procedure
```

This method has a linear convergence rate [9], which is given by $\rho(M)$. However, it can be accelerated to obtain a quadratically convergent method by a simple modification. Note that we can refactor (6) as follows:

$$(I-M)^{-1} = (I+M)(I+M^2)(I+M^4)\cdots(I+M^{2^j})\cdots \tag{8}$$ {{eq:neumann2}}

Truncating the above equation and permuting the factors $(I + M^{2^j})$ yields another method to approximate $x = (Q - S)^{-1}e$, which has a much faster convergence, and is described by the equation:

$$(I - M)^{-1}Q_1^{-1}e = (I + M^{2^k})(I + M^{2^{k-1}}) \cdots (I + M^2)(I + M)Q_1^{-1}e$$

The pseudocode for this approach is given in Algorithm 2. As visible on line 5, this method required to store the repeated squares of the matrix M.

Algorithm 2 Neumann series (8) to approximate $x = (Q - S)^{-1}e$

```
1: procedure NeumanSeries(Q_1, Q_2, k)
2:     M ← −Q_1^{-1}(Q_2 − S)
3:     x ← Q_1^{-1}e + MQ_1^{-1}e
4:     for j = 2, ..., k do
5:         M ← M^2
6:         x ← x + Mx
7:     end for
8:     return x
9: end procedure
```

Remark 4.4. A favorable property of both approaches is that the convergence of x_k to x is monothonically decreasing and non-positive. That is, for each $k' \leq k$ we have $x_{k'} \geq x_k$. Since the MTTA is equal to $-\pi_0^T x$, the estimates $-\pi_0^T x_k$ of the MTTA obtained in the intermediate steps are guaranteed lower bounds.

We note that Algorithm 1 can be slightly modified to compute $\pi_0^T(Q - S)^{-1}$ instead of $(Q - S)^{-1}e$. Both vectors can then be used to compute the MTTF through a dot product. However, the former choice has the advantage that $\pi_0(s) = 0$ implies that $x(s) = 0$ throughout the iterations for every $s \in \mathcal{PS} \setminus \mathcal{RS}$. In particular, the non-reachable part of the chain has no effect on the computed tensor, and this helps to keep the TT-rank low during the iterations. The modified algorithm is reported for completeness in Algorithm 3, where now x, y are row vectors.

Algorithm 3 Neumann series (6) to approximate $\pi^T = \pi_0^T(Q - S)^{-1}$

```
1: procedure NeumanSeries(Q_1, Q_2, k)
2:     y ← π_0^T
3:     x ← y
4:     for j = 1, ..., k do
5:         y ← −yQ_1^{-1}(Q_2 − S)
6:         x ← x + y
7:     end for
8:     x ← xQ_1^{-1}
9:     return x
10: end procedure
```

Lemma 4.5. *Let $A \geq 0$ be a non-negative $N \times N$ matrix, and e the vector of all ones. Then,*

$$\rho(A) \leq \|A\|_\infty = \max_{1 \leq i \leq N} (Ae)$$

Moreover, if there is at least one component of Ae strictly smaller than $\|A\|_\infty$, then $\rho(A) < \|A\|_\infty$.

Proof. The first statement is the definition of infinity norm, whereas the second follows directly by the first Gerschgorin theorem. □

The next result provides a technique for splitting an infinitesimal general Q (i.e., up to a change of sign, any M-matrix with zero row sums) in a way that allow to perform the Neumann expansion. Let us first recall a few properties of diagonally dominant matrices.

Lemma 4.6. *Let $A = D - B$, with $D < 0$ and diagonal, $B \geq 0$, and $Be < -De$, where e is the vector with all components equal to 1. Then, A is invertible and $A^{-1} \leq 0$.*

Proof. This fact can be easily prove using the tools from theory of nonnegative matrices, see for instance [1]. Since $D < 0$, the condition $A^{-1} \leq 0$ is equivalent to $(D^{-1}A)^{-1} \geq 0$. Moreover, the strict row diagonal dominance implies that $\|D^{-1}A\|_\infty < 1$, so we have

$$(D^{-1}A)^{-1} = (I - (-D^{-1}A))^{-1} = \sum_{j \geq 0} (-D^{-1}A)^j \geq 0,$$

where we have used that $(-D^{-1}A) \geq 0$. This concludes the proof. □

Theorem 4.7. *Let $A = D + A_1 + A_2$ any $N \times N$ matrix such as D is diagonal and non-positive, A_1, A_2 are non negative, $e_N^T(D + A_1 + A_2) = e_N^T A_1 = 0$, and $(D + A_1 + A_2)e = 0$. Then, if we define $S = (A_1 + A_2)e_N e_N^T$, $(D + A_1)$ is invertible and $\min_{i=1,\dots,N-1} A_{iN} > 0$, we have that $\|(D+A_1)^{-1}(A_2 - S)\|_\infty < 1$.*

Proof. Let us denote with $M := (D + A_1)^{-1}(A_2 - S)$. All the columns of this matrix are non-positive (see Lemma 4.6), with the only exception of the last one, which is non-negative. This is a consequence of the fact that $(D + A_1)^{-1}$ is non-positive, and $(A_2 - S)$ has the first $N - 1$ columns with positive entries, and the last one with negative ones.

Therefore, it is clear that we have $\|M\|_\infty = \|M(e - 2e_N)\|_\infty$, and the vector $M(e - 2e_N)$ is element-wise non-positive by construction. We have

$$M(e - 2e_N) = (D + A_1)^{-1}(A_2 - S)(e - 2e_N).$$

Using the relations $(D + A_1)^{-1}A_2 e = (D + A_1)^{-1}(D + A_1 + A_2)e - e = -e$ and $Se = Se_N = (A_1 + A_2)e_N$ we get

$$\begin{aligned} M(e - 2e_N) &= -e - 2(D + A_1)^{-1}A_2 e_N + (D + A_1)^{-1}(A_1 + A_2)e_N \\ &= -e + (D + A_1)^{-1}(A_1 - A_2)e_N \\ &= -e + e_N - (D + A_1)^{-1}(D + A_2)e_N. \end{aligned}$$

By construction, we know that the last row of $D + A_2$ is equal to $-e_N^T A_1$ and is therefore zero. The first $N-1$ entries in $(D+A_2)e_N$ are taken from A_2 and therefore they are (strictly) positive. Since $(D+A_1)^{-1}$ is entry-wise strictly negative, we have that $v = -(D+A_1)^{-1}(D+A_2)e_N$ has the first $N-1$ components strictly positive. Therefore, we have that $M(e-2e_N) > -1$ element-wise, and on the other hand we knew that $M(e-2e_N)$ is non-positive. This implies that $\|M\|_\infty < 1$, as claimed. □

Lemma 4.8. *Let y be any vector. Then, using the notation of Theorem 4.7, $(D+A_1)^{-1}(A_2-S)y$ has the last component equal to zero. Moreover, let y_0 be any vector, and define*

$$y_{k+1} = -(D+A_1)^{-1}(A_2-S)y_k, \qquad k \geq 1.$$

Then, if y_k for $k > 0$ is non-negative we have $y_{k'} \geq 0$ for any $k' \geq k$.

Proof. We start showing that $e_N^T(D+A_1)^{-1}(A_2-S) = 0$, which proves the first claim. We have

$$\begin{aligned} e_N^T(D+A_1)^{-1}(A_2-S) = D_{NN}^{-1} e_N^T(A_2-S) &= e_N^T A_2 - e_N^T A_1 e_N e_N^T - e_N^T A_2 e_N e_N^T \\ &= e_N^T A_2 - e_N^T A_2 e_N e_N^T = e_N^T A_2(I - e_N e_N^T) \\ &= -e_N^T D(I - e_N e_N^T) = 0, \end{aligned}$$

where we have used the properties $e_N^T(D + A_1 + A_2) = e_N^T + A_1 = 0$, and the definition of $S = (A_1 + A_2)e_N e_N^T$.

Assume now that $e_N^T y = 0$. Then, $z := (A_2 - S)y = A_2 y \geq 0$, since $Sy = 0$. Moreover, $(D+A_1)^{-1}$ is non-positive in view of Lemma 4.6, and therefore $-(D+A_1)^{-1}z \geq 0$, concluding the proof. □

Remark 4.9. Note that choosing $\gamma = -1$ in the notation of Lemma 4.3 provides a starting vector for the Neumann iteration Equation (6) that satisfies the hypotheses of Lemma 4.8. Therefore, in this case the iteration to approximate the MTTA is monothonically increasing, and at the step k gives a lower bound for the final value of the MTTA.

Theorem 4.10. *Let $Q = \Delta + R + W$ be an infinitesimal generator of a Markov chain as described in* (1)*, with*

$$R = R^{(1)} \oplus \ldots \oplus R^{(k)}, \qquad \Delta = -\mathrm{diag}((W+R)e),$$

and $\gamma \geq \|\Delta\|_\infty$. Then, if we define $D := -\gamma I$, $A_1 = R$, and $A_2 = W + (\Delta - \gamma I)$, these matrices satisfy the hypotheses of Theorem 4.7 and there exists α_j, β_j such that

$$D + A_1 = \left(R^{(1)} - \frac{\gamma}{k} I\right) \oplus \left(R^{(2)} - \frac{\gamma}{k} I\right) \oplus \cdots \oplus \left(R^{(k)} - \frac{\gamma}{k} I\right),$$

and therefore

$$(D+A_1)^{-1}(A_2-S) \approx \sum_{j=1}^{\ell} \alpha_j \left(e^{\beta_j(R_1 - \frac{\gamma}{k} I)} \otimes \ldots \otimes e^{\beta_j(R_k - \frac{\gamma}{k} I)}\right)(A_2 - S).$$

Remark 4.11. We note that the choice of γ allows to control the condition number of the matrix $D + A_1$; our experience shows that larger values for γ (which give lower condition numbers), provide slower convergence, but also lower TT-ranks during the Neumann iteration.

5 Computational remarks

In this section we report a few computational remarks concerning our implementation. We have written a function `computeMTTF` that, given a collection of R and W matrices describing the probability transition inside the small subsystems and their interaction, constructs the matrices Q_1 and Q_2 in order to perform the Neumann iteration.

Then, we have implemented the linearly convergent (6) and the quadratically convergence (8) iterations. A few considerations can be helpful in trying to obtain maximum performances from the implementation.

5.1 Ordering of the subsystems

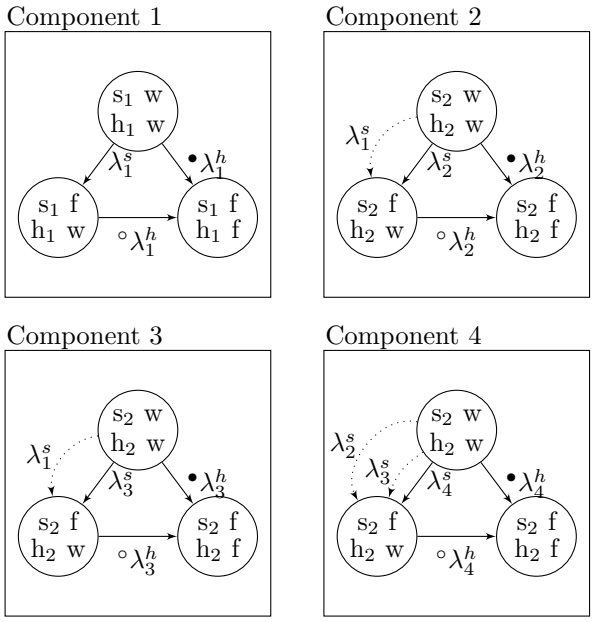
Since the TT representation represents the interaction between between the subsystem i and $i + 1$ in each carriage, we have found that it is beneficial to reorder the topology so that few nodes are linked to far ones.

In particular, given the adjacency matrix $\mathcal{T}$ that represents the connection graph (i.e., $\mathcal{T}_{ij}) = 1$ if and only if there is an edge in graph from the node i to the node j), it can be helpful to reorder the subsystems to make this matrix as banded as possible. To this end, we have employed the reverse Cuthill-McKee ordering implemented in MATLAB in the function `symrcm`.

5.2 The choice of γ

The choice of the parameter γ in Theorem 4.10 can have important effects on the performance of the algorithm. We have verified that choosing γ relatively large, for instance $\gamma = 2\|\Delta\|_\infty$, can be helpful. This reduces the conditioning of the matrix to invert to a small number (smaller than 2, in fact), and thus very few exponential sums are needed to achieve a very high accuracy. This helps to keep the TT-ranks low during the iteration, especially when applying (8), which in turn suffers very mildly from having the spectral radius close to 1. On the other hand, when applying the linearly convergence iteration (6), a choice closer to zero can be more convenient.

We do not have a "universal recipe" for these choices, so it might be helpful to do some preliminary parameter tuning on small problems of a given class before tackling the large scale cases.



Topology of interdependencies:

$$\mathcal{T} = \begin{matrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{matrix}$$

Figure 1: Example SAN for the case study where there are 4 components. Dotted arrows are synchronization transitions.

6 Case study

Consider a cyber-physical system comprising k components, consisting each of a mechanical object and a Monitoring and Control Unit (MCU). The mechanical objects are independent one from the other whereas the working status of the MCU software on component j depends on data produced by local sensors and can depend also on data coming from the MCU of component i. Thus, it is possible to define a topology of interactions among component MCUs: define $\mathcal{T}$ the $k \times k$ matrix as $\mathcal{T}(i,j) = 1$ if $i = j$ or the j-th MCU consumes data produced by the i-th MCU. At every time instant, the MCU and the mechanical object on each component can be *working* or *failed*. If the mechanical object on component i fails then instantaneously also the MCU on component i fails. If $\mathcal{T}(i,j) = 1$ then the failure of the i-th MCU implies an instantaneous failure of the j-th MCU. The failure time of the software running on the i-th MCU is assumed to be exponentially distributed with rate λ_i^s.

The MCU on component i can modify the behaviour of the mechanical object on component i, so the failure time of the mechanical object on component i is exponentially distributed with rate $^\bullet\lambda_i^h$ if the MCU on component i is working, and $^\circ\lambda_i^h$ if the MPC is already failed. No repair is considered.

We are interested in evaluating the Mean Time to System Failure, where the system is considered failed when all the mechanical objects are failed. Figure 1 depicts the SAN model for a simple case where there are 4 components. In

particular, the state of component i, represented with a circle, is defined by software status (s_i is `w` if working, `f` if failed) and the hardware status (h_i is `w` if working, `f` if failed). Transitions can be local or synchronized, represented as arrows and dotted arrows, respectively, and labelled by their rate. Each component model has 3 states, so that $|\mathcal{PS}| = 3^k$, and the cardinality of $\mathcal{RS}$ depends on $\mathcal{T}$: to a large number on non-zeros entries in $\mathcal{T}$ corresponds a small $\mathcal{RS}$. Notice that, if $\mathcal{T} = I$ then $\mathcal{RS} = \mathcal{PS}$. The local and synchronization contribution matrices are then obtained as in Equation (9) and Equation (10), respectively.

$$R^{(i)} = \begin{pmatrix} 0 & 0 & {}^{\bullet}\lambda_i^h \\ 0 & 0 & {}^{\circ}\lambda_i^h \\ 0 & 0 & 0 \end{pmatrix} \tag{9}$$ {{eq:R}}

$$W^{(j,i)} = \begin{cases} \begin{pmatrix} 0 & \lambda_i^s & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} & \text{if } j = i \\ \begin{pmatrix} 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} & \text{if } j \neq i \text{ and } \mathcal{T}(i,j) = 1 \\ \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} & \text{otherwise} \end{cases} \tag{10}$$ {{eq:W}}

7 Experimental results

The case study model presented in Section 6 has been implemented in MATLAB [19] and studied applying the method discussed so far. In particular, we consider the following set of parameters:

$$ {}^{\bullet}\lambda_i^h = \frac{i}{10}, \quad {}^{\circ}\lambda_i^h = i, \quad \lambda_i^s = i, $$

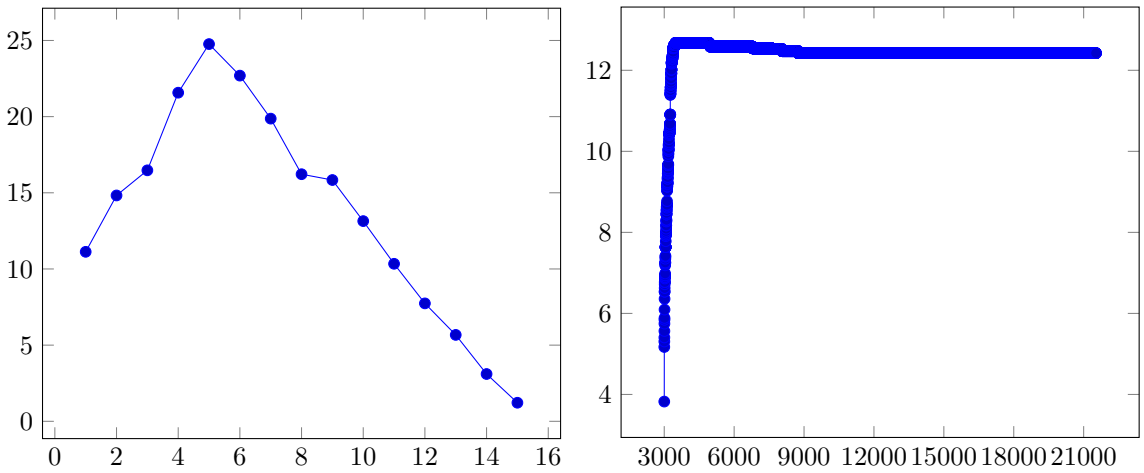
and the topology $\mathcal{T}$ has been chosen at random with the following constraints:

- each component has impact on itself, i.e., $\mathcal{T}(i,i) = 1$,
- the sparsity of $\mathcal{T}$ is about $\frac{1}{2k}$,

Table 1 collects, for both Algorithms 2 and 3, the time spent by the presented method in computing the MTTF and the maximum amount of memory it has consumed. We can notice that, at increasing of k, both time and memory do not follow a regular pattern, due to the randomly generated $\mathcal{T}$. In all the considered experiments, Algorithm 3 is slower than Algorithm 2 but it has a reduced memory footprint, confirming the theoretical prediction.

k	$\lvert\mathcal{RS}\rvert$	$\lvert\mathcal{PS}\rvert$	time(s) Alg 3	mem (Gb) Alg 3	time (s) Alg 2	mem (Gb) Alg 2
10	2400	59049	57.12	$6.25 \cdot 10^{-1}$	28.18	2.23
12	9760	531441	87.95	0.66	37.87	3.01
14		$4.78 \cdot 10^{6}$	145.74	0.68	39.97	2.42
16		$4.3 \cdot 10^{7}$	193.16	0.64	26.48	2.78
18		$3.87 \cdot 10^{8}$	288.49	0.64	117.2	7.33
20		$3.49 \cdot 10^{9}$	423.07	0.8	128.28	6.19
22		$3.14 \cdot 10^{10}$	524.83	0.69	101.02	6.82
24		$2.82 \cdot 10^{11}$	787.72	0.89	275.34	14.2
26		$2.54 \cdot 10^{12}$	885.34	0.68	75.95	2.64
28		$2.29 \cdot 10^{13}$	1095.48	0.88	48.94	2.87
30		$2.06 \cdot 10^{14}$	1337.4	0.85	54.64	3.15
32		$1.85 \cdot 10^{15}$	1547.46	0.88	59.16	3.41

Table 1: Reachable and potential spaces dimensions, time and memory consumption for Algorithms 2 and 3.



8 Conclusions

We have shown that tensor trains are a powerful tool for the analysis of performance and reliability measures (in this case, the mean time to failure) of large systems, when the interconnection between the smaller subsystems that compose them is sufficiently weak.

We have presented a theoretical analysis of an iteration that is easily applicable in the tensorized format, and with guaranteed convergence. A quadratically convergent variation has been shown as well, and the performances have been tested on a representative set of examples.

Several lines of research remain open: the connection between the weak connections and the TT-rank in the iteration needs to be studied further, in order to understand the relation more in depth. Moreover, several more measures are of interest in this context, and the application of tensor techniques for this task could lead to faster and reliable methods for their computation.

References

[1] Abraham Berman and Robert J. Plemmons. *Nonnegative matrices in the mathematical sciences*, volume 9 of *Classics in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1994. Revised reprint of the 1979 original.

[2] Matthias Bolten, Karsten Kahl, Daniel Kressner, Francisco Macedo, and Sonja Sokolović. Multigrid methods combined with low-rank approximation for tensor structured markov chains. *arXiv preprint arXiv:1605.06246*, 2016.

[3] Dietrich Braess and Wolfgang Hackbusch. Approximation of $1/x$ by exponential sums in $[1, \infty)$. *IMA journal of numerical analysis*, 25(4):685–697, 2005.

[4] Peter Buchholz and Peter Kemper. *Kronecker based matrix representations for large Markov models*, pages 256–295. Springer, 2004.

[5] 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.

[6] Michel Crouzeix and César Palencia. The numerical range as a spectral set. *arXiv preprint arXiv:1702.00668*, 2017.

[7] Lieven De Lathauwer, Bart De Moor, and Joos Vandewalle. A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl.*, 21(4):1253–1278, 2000.

[8] Vin de Silva and Lek-Heng Lim. Tensor rank and the ill-posedness of the best low-rank approximation problem. *SIAM J. Matrix Anal. Appl.*, 30(3):1084–1127, 2008.

[9] James W. Demmel. *Applied numerical linear algebra*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1997.

[10] S. V. Dolgov, B. N. Khoromskij, and I. V. Oseledets. Fast solution of parabolic problems in the tensor train/quantized tensor train format with initial application to the Fokker-Planck equation. *SIAM J. Sci. Comput.*, 34(6):A3016–A3038, 2012.

[11] S. Donatelli. Superposed stochastic automata: A class of stochastic petri nets with parallel solution and distributed state space. *Perform. Eval.*, 18(1):21–36, 1993.

[12] Jane Hillston. *A Compositional Approach to Performance Modelling*. Cambridge University Press, New York, NY, USA, 1996.

[13] Vladimir A. Kazeev and Boris N. Khoromskij. Low-rank explicit QTT representation of the Laplace operator and its inverse. *SIAM J. Matrix Anal. Appl.*, 33(3):742–758, 2012.

[14] Tamara G. Kolda and Brett W. Bader. Tensor decompositions and applications. *SIAM Rev.*, 51(3):455–500, 2009.

[15] Daniel Kressner and Francisco Macedo. Low-rank tensor methods for communicating markov processes. In *International Conference on Quantitative Evaluation of Systems*, pages 25–40. Springer, 2014.

[16] Daniel Kressner and Christine Tobler. Algorithm 941: `htucker`–a Matlab toolbox for tensors in hierarchical Tucker format. *ACM Trans. Math. Software*, 40(3):Art. 22, 22, 2014.

[17] Christian Lubich, Ivan V. Oseledets, and Bart Vandereycken. Time integration of tensor trains. *SIAM J. Numer. Anal.*, 53(2):917–941, 2015.

[18] Giulio Masetti and Leonardo Robol. Computing performability measures in markov chains by means of matrix functions. *arXiv preprint arXiv:1803.06322*, 2018.

[19] MathWorks. *MATLAB R2018a*. The Mathworks, Inc., Natick, Massachusetts, 2018.

[20] I. V. Oseledets. Tensor-train decomposition. *SIAM J. Sci. Comput.*, 33(5):2295–2317, 2011.

[21] Ivan Oseledets, Sergey Dolgov, Vladimir Kazeev, Olga Lebedeva, and Thomas Mach. MATLAB TT-Toolbox. `https://github.com/oseledets/TT-Toolbox`, 2019.

[22] Brigitte Plateau and William J. Stewart. *Stochastic Automata Networks*, pages 113–151. Springer US, Boston, MA, 2000.

[23] Kishor S Trivedi and Andrea Bobbio. *Reliability and Availability Engineering: Modeling, Analysis, and Applications*. Cambridge University Press, 2017.