Representation transformations for finding Markovian representations

András Mészáros[1], Gábor Horváth[1,3], and Miklós Telek[1,2]

[1] Budapest University of Technology and Economics,
[2] MTA-BME Information systems research group,
[3] Inter-University Center of Telecommunications and Informatics
{meszarosa,ghorvath,telek}@hit.bme.hu

Abstract. In this paper we consider existing and new representation transformation methods for non-Markovian generalizations of Markov chain driven stochastic models which intend transforming non-Markovian representations into Markovian ones and evaluate their efficiency through numerical experiments. One of the new features of the considered methods is the ability to obtain a Markovian representation of larger size.

Keywords: Markov arrival process, Rational arrival process, representation transformation, Markovian representation

1 Introduction

Background Markov chain driven stochastic models, like PH distributions [1], MAPs [2], and MMAPs [3], are efficiently used for describing practical systems of various application fields, including computer networks and telecommunication systems. These models are described with a set of vectors and/or matrices, which are referred to as representation. In some cases, e.g., when the coefficient of variation of inter-event times is low, the non-Markovian generalizations of these Markov chain based models are more efficient with respect to the size of the representation. Another motivation for dealing with non-Markovian generalizations comes from the fact that there are moments based fitting methods which generate non-Markovian representations based on a set of moments and joint moments.

Stochastic processes with Markovian representations have essentially important nice features. There is always a valid stochastic process associated with a Markovian representation, and it is easy to check if a representation is Markovian or not. The main drawback of using non-Markovian representations is that non-Markovian representations might or might not represent valid stochastic processes. A non-Markovian representation is non-valid the joint density function of consecutive inter-arrival times defined by the representation, see eg. (1), is negative at some point. In this case there is no stochastic process associated with the non-Markovian representation. Additionally, it is not obvious how to determine if a matrix exponential distribution (defined by a non-Markovian representation) is a valid distribution (e.g. its density function is non-negative) and

for more complex processes, like RAPs and MRAPs, we do not have the methodology for checking if a non-Markovian representation defines a valid process or not.

In this paper we cannot present a general solution for checking if a non-Markovian representation is associated with a stochastic process or not, we only propose an elementary step. If a non-Markovian representation can be transformed to a Markovian representation such that the representation defines the same process, then the non-Markovian representation surely defines a valid stochastic process. Our proposed elementary step is through numerical procedures for transforming non-Markovian representations to Markovian ones. We survey previously available methods and present new ones. The main new feature in this work is the representation transformation procedure which searches for a Markovian representation of larger size.

The rest of the paper is organized as follows. Section 2 surveys the considered set of stochastic models and summarizes the properties which are utilized later on. Existing and new representation transformation methods are introduced for same size in Section 3 and for extended size in Section 4. Section 5 is devoted to the numerical experimentation with the different representation methods. The paper is concluded in Section 6.

2 Markov chain driven stochastic models and their non-Markovian generalizations

2.1 Phase type and matrix exponential distributions

We start with the basic definition of PH [1] and ME [4] distributions.

Definition 1. *Let $\mathcal{X}$ be a random variable with cumulative distribution function (cdf)*

$$F_X(x) = Pr(\mathcal{X} < x) = 1 - \alpha e^{\boldsymbol{A}x}\mathbb{1},$$

where α is an initial row vector of size n, $\boldsymbol{A}$ is a square matrix of size $n \times n$, and $\mathbb{1}$ is the column vector of ones of size n. In this case, we say that $\mathcal{X}$ is matrix exponentially distributed with representation $\alpha, \boldsymbol{A}$, or shortly, ME($\alpha, \boldsymbol{A}$) distributed.

Definition 2. *If $\mathcal{X}$ is an ME($\alpha, \boldsymbol{A}$) distributed random variable, where α and $\boldsymbol{A}$ have the following properties:*

- $\alpha_i \geq 0$, $\alpha\mathbb{1} = 1$ *(there is no probability mass at $x = 0$),*
- $A_{ii} < 0$, $A_{ij} \geq 0$ *for $i \neq j$,* $\boldsymbol{A}\mathbb{1} \leq 0$,

then we say that $\mathcal{X}$ is phase type distributed with representation $\alpha, \boldsymbol{A}$, or shortly, PH($\alpha, \boldsymbol{A}$) distributed.

The vector-matrix representations satisfying the conditions of Definition 2 are called Markovian.

2.2 Marked MAPs and marked RAPs

MAPs and RAPs model point processes with a single type of events. Extension of these processes to multiple event types are referred to as marked MAPs (MMAPs) [3] and marked RAPs (MRAPs) [5]. Due to space limitations we summarize the properties of these more general processes only.

Let $\{X(t), Y(t)\}$ be a point process supplemented with the type of the events $\{1, \ldots, K\}$, with joint probability density function of inter-event times and associated types $f(t_1, k_1, \ldots, t_j, k_j)$ for $j = 1, 2, \ldots$, and $k_j \in \{1, \ldots, K\}$.

Definition 3. *$\{\mathcal{X}(t), \mathcal{Y}(t)\}$ is called a marked rational arrival process if there exists a set of $K+1$ finite matrices $(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K})$, such that*

$$f_{(\boldsymbol{H_0},\ldots,\boldsymbol{H_K})}(t_1, k_1, \ldots, t_j, k_j) = \underline{\pi} e^{\boldsymbol{H_0} t_1} \boldsymbol{H}_{k_1} e^{\boldsymbol{H_0} t_2} \boldsymbol{H}_{k_2} \ldots e^{\boldsymbol{H_0} t_j} \boldsymbol{H}_{k_j} \mathbb{1} \quad (1)$$

where $\sum_{i=0}^{K} \boldsymbol{H_k} \mathbb{1} = 0$, and π is the solution of

$$\underline{\pi}(-\boldsymbol{H_0})^{-1} \sum_{i=1}^{K} \boldsymbol{H_k} = \underline{\pi}, \quad \underline{\pi}\mathbb{1} = \mathbb{1} \; . \quad (2)$$

In this case, we say that $\{\mathcal{X}(t), \mathcal{Y}(t)\}$ is a marked rational arrival process with representation $(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K})$, or shortly, MRAP$(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K})$.

Definition 4. *If $\{\mathcal{X}(t), \mathcal{Y}(t)\}$ is a MRAP$(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K})$, such that*

- *$\boldsymbol{H_k}_{ij} \geq 0$ for $k \geq 1$,*
- *$\boldsymbol{H_0}_{ii} < 0$, $\boldsymbol{H_0}_{ij} \geq 0$ for $i \neq j$, $\boldsymbol{H_0}\mathbb{1} \leq 0$,*

then we say that $\{\mathcal{X}(t), \mathcal{Y}(t)\}$ is a marked MAP with representation $(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K})$, that is, MMAP$(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K})$.

The representations satisfying the conditions of Definition 4 are called Markovian. Later we are going to use the following property of MRAPs from [6]

Definition 5. *The rank of the $(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K})$ representation, with respect to the initial vector, is the number of linear independent vectors of the form $\underline{\pi}\boldsymbol{H_0}^{a_1}\boldsymbol{H_{k_1}}\boldsymbol{H_0}^{a_2}\boldsymbol{H_{k_2}}\ldots$, with $a_i \in \{1, 2, \ldots\}$ and $k_i \in \{1, 2, \ldots, K\}$.*

An efficient computational method for computing this rank is provided in [6].

Obviously, MRAPs and MMAPs with $K = 1$ are RAPs and MAPs. Based on this property in the rest of the paper we are going to follow a unified treatment of MRAPs/MMAPs and RAPs/MAPs with K event types. This way we are going to present the existing representation transformation methods of the literature in a more general environment than it is in the original publications.

procedure MRAP2MMAP-[7]$(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K})$
  $a = 0.5$;
  while $a > \epsilon$ **do**
    for $n = 1$ **to** $maxIter$ **do**
      $(i^*, j^*) = \underset{i,j \in [1,\ldots,n], i \neq j}{\arg\min} G(Tr(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K}, \boldsymbol{T}(i, j, a)))$
      $(\boldsymbol{H_0}^{(new)}, \ldots, \boldsymbol{H_K}^{(new)}) = Tr(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K}, \boldsymbol{T}(i^*, j^*, a))$
      if $G(\boldsymbol{H_0}^{(new)}, \ldots, \boldsymbol{H_K}^{(new)}) < G(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K})$ **then**
        $(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K}) = (\boldsymbol{H_0}^{(new)}, \ldots, \boldsymbol{H_K}^{(new)})$
      end if
      if $\boldsymbol{H_0}, \ldots, \boldsymbol{H_K}$ is Markovian **then**
        return $(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K})$
      end if
    end for
    $a = a/2$
  end while
return $(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K})$
end procedure

Fig. 1. The representation transformation method of [7]

3 Representation transformation methods with the same size

In this section we first present the previous research related to RAP/MRAP to MAP/MMAP transformation. After that we introduce the new algorithms. The past work only dealt with transformation between representations of the same size. In this section we also introduce a representation transformation method with size extension.

3.1 Previous works with single element modifications

The development of efficient numerical representation transformation methods for finding a Markovian representation from a non-Markovian one dates back to 2007 [7]. This algorithm, generalized to MRAPs, is shown in Figure 1.

The algorithm applies a series of elementary transformations improving the representation in each step until it reaches a Markovian representation or a local optimum of the goal function. The matrix representing the elementary transformations is given by

$$\boldsymbol{T}(i, j, x) = \boldsymbol{I} + x\boldsymbol{E_{ij}} - x\boldsymbol{E_{ii}}, \qquad (x \neq 1), \tag{3}$$

where $\boldsymbol{I}$ is the identity matrix and $\boldsymbol{E_{ij}}$ is the matrix whose only nonzero entry is the i, j element which equals to 1. The procedure determines the best possible elementary transformation matrix and transforms the representation in each step. The transformation step in the algorithm is represented by $Tr(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K}, \boldsymbol{T}) = (\boldsymbol{T}^{-1}\boldsymbol{H_0}\boldsymbol{T}, \ldots, \boldsymbol{T}^{-1}\boldsymbol{H_K}\boldsymbol{T})$.

procedure MRAP2MMAP-[8]($\boldsymbol{H_0}, \ldots, \boldsymbol{H_K}$)
  $r \leftarrow 1$
  repeat
    $(i^*, j^*, x^*) = \underset{\substack{i,j \in [1,\ldots,n], i \neq j, \\ x \in [0,1]}}{\arg\min} G_4(Tr(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K}, \boldsymbol{T}(i,j,x)), r)$
    $(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K}) = Tr(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K}, \boldsymbol{T}(i^*, j^*, x^*))$
    if $\boldsymbol{H_0}, \ldots, \boldsymbol{H_K}$ is Markovian **then**
      return $(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K})$
    end if
    $r \leftarrow 1.05r$
  until $x^* < \epsilon$
return $(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K})$
end procedure

Fig. 2. The representation transformation method of [8]

In [7] the following three goal functions are considered (denoted by $G_1(\cdot), G_2(\cdot)$ and $G_3(\cdot)$):

$$G_1(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K}) = \sum_{i,j,i \neq j} \exp(-2[\boldsymbol{H_0}]_{i,j}) + \sum_{k=1}^{K} \sum_{i,j} \exp(-2[\boldsymbol{H_k}]_{i,j}),$$

$$G_2(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K}) = \sum_{i,j,i \neq j} \exp(-1000[\boldsymbol{H_0}]_{i,j}) + \sum_{k=1}^{K} \sum_{i,j} \exp(-1000[\boldsymbol{H_k}]_{i,j}),$$

$$G_3(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K}) = \sum_{i,j,i \neq j} \exp(-([\boldsymbol{H_0}]_{i,j} - 1)^3) + \sum_{k=1}^{K} \sum_{i,j} \exp(-([\boldsymbol{H_k}]_{i,j} - 1)^3).$$

The algorithm switches goal functions from time to time, which, for simplicity, is not reflected in Figure 1. The multiple goal functions help to leave a local optimum and let the optimization go for a better solution.

While the algorithm is successful in finding the Markovian representation in many cases, our current intuition is that the rigidity of the transformation matrix (3) might pose a problem in some cases. Buchholz et al. proposed a modified version of this method in [8] by applying a different goal function and parameter x in (3) is included in the optimization as well. The proposed procedure is summarized in Figure 2. The goal function used in this algorithm is given by

$$\begin{aligned} G_4(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K}, r) = {} & r \sum_{\substack{i,j \\ i \neq j}} ((-[\boldsymbol{H_0}]_{i,j})^+)^2 - \sum_{\substack{i,j \\ i \neq j}} (([\boldsymbol{H_0}]_{i,j})^+)^2 \\ & + r \sum_{k=1}^{K} \sum_{i,j} ((-[\boldsymbol{H_k}]_{i,j})^+)^2 - \sum_{k=1}^{K} \sum_{i,j} (([\boldsymbol{H_k}]_{i,j})^+)^2, \end{aligned} \tag{4}$$

where $(x)^+$ is the positive part of x, i.e., $(x)^+ = \max\{0, x\}$.

The intuitive motivation of such a goal function is the following. If we do not reward the higher non-negative elements, the algorithm will only try to increase the negative elements, while possibly decreasing the other elements to zero. When these elements reach zero, the algorithm can get stuck in a non-Markovian representation. Therefore, this algorithm tries to make a "provision" for the non-negative elements in the beginning. Later the weight of these elements decreases, and procedure focuses more and more on increasing the negative elements of the representation.

Apart from the different goal function, the main difference between `MRAP2MMAP-[7]` and `MRAP2MMAP-[8]` is that in each iteration the former one performs an exhaustive search for the optimal i, j parameters of the elementary transformation matrix, while the latter one incorporates a continuous variable x into the optimization as well, needing a more involved solution method.

3.2 A new family of representation transformation methods

Based on our experience with the past algorithms discussed in Section 3.1 we found that the overly simple structure of the elementary transformation matrix they are using limits their efficiency considerably.

The structure of the elementary matrix plays an important role during the representation transformation. A too simple elementary transformation matrix can restrict the basic movements of the optimization method. Along the restricted path potentially no Markovian representations may be reachable. On the other hand, a too general elementary transformation matrix introduces too many variables to optimize. This makes the solution of the corresponding nonlinear optimization problem less effective. It will be slow and might give a local optimum that is far from the global one.

In this section we introduce a family of representation transformation procedures. All these procedures follow the same basic idea as Algorithms 1 and 2, that is, they still use consecutive transformations to find a Markovian representation of a RAP. However, we propose enhancements in two principal details of the algorithm.

1. *Elementary modifications of the transformation matrix* The simple elementary transformation matrix given by (3) is basically an identity matrix (acting as an initial transformation matrix) modified such that a single entry is put in an off-diagonal position. It is possible to generalize this idea by allowing more general modifications of the initial transformation matrix.
   In the family of algorithms introduced in this paper we are making use of three kinds of such elementary modifications.
   - *Single scalar modification.* The initial transformation matrix $\boldsymbol{T_0}$ is modified by a single scalar x as

     $$\boldsymbol{M_S}(\boldsymbol{T_0}, i, j, x) = \boldsymbol{T_0} + x\boldsymbol{E_{ij}} - x\boldsymbol{E_{ii}}. \tag{5}$$

     In this case, only a single variable has to be optimized.

- *Single vector modification.* The initial transformation matrix is increased by a vector as

$$\boldsymbol{M_V}(\boldsymbol{T_0}, i, \underline{x}) = \boldsymbol{T_0} + \underline{e}_i^T \cdot \underline{x}, \tag{6}$$

  where $\underline{e}_i$ is the ith unit row vector. Vector $\underline{x}$ is such that $\boldsymbol{M_V}(\boldsymbol{T_0}, i, \underline{x})\mathbb{1} = \mathbb{1}$, thus number of elements to be optimized is $n-1$, where n is the size of the representaiton.
- *Dyadic product modification.* The most general modification of the initial transformation matrix used in this paper is given by a dyadic product of two vectors (where the i, j element of matrix $\underline{u}^T \cdot \underline{v}$ is $u_i v_j$), yielding

$$\boldsymbol{M_D}(\boldsymbol{T_0}, \underline{u}, \underline{v}) = \boldsymbol{T_0} + \underline{u}^T \cdot \underline{v}, \tag{7}$$

  with $\boldsymbol{M_D}(\boldsymbol{T_0}, \underline{u}, \underline{v})\mathbb{1} = \mathbb{1}$, which means that $2n-1$ elements need to be optimized.

2. *The way the transformation matrix is constructed.* We investigate two cases on how the transformation matrix is obtained in each step of the algorithm.
   - *Transformation with a single elementary modification.* In this case, the transformation matrix used to transform the current representation in each step of the algorithm is an identity matrix with an elementary modification applied. Thus, in each step the optimal matrix $\boldsymbol{T} = \boldsymbol{M_S}(\boldsymbol{I}, i, j, x), \boldsymbol{T} = \boldsymbol{M_V}(\boldsymbol{I}, i, \underline{x})$ or $\boldsymbol{T} = \boldsymbol{M_D}(\boldsymbol{I}, \underline{u}, \underline{v})$ is determined (depending on which variant of the algorithm we are using) and the corresponding transformation is applied.
   - *Transformation with cumulative elementary modifications.* According to this approach the transformation is not applied in each step of the algorithm. The algorithm starts with $\boldsymbol{T} = \boldsymbol{I}$, and makes several elementary modifications one after the other in a cumulative way (thus $\boldsymbol{T} = \boldsymbol{M_S}(\boldsymbol{T}, i, j, x), \boldsymbol{T} = \boldsymbol{M_V}(\boldsymbol{T}, i, \underline{x})$ or $\boldsymbol{T} = \boldsymbol{M_D}(\boldsymbol{T}, \underline{u}, \underline{v})$), as long as there is improvement according to the goal function. The transformation is then applied with matrix $\boldsymbol{T}$. Observe that the matrix $\boldsymbol{T}$ obtained this way is much more general then the one obtained by a single modification.

The three options according to the elementary transformations and the two options according to the way the transformation matrix is obtained gives 6 possible algorithms in total. These algorithms will be identified as Algorithm `MRAP2MMAP-xy`, where `x` can be

- `x`=`S`, if the algorithm uses single scalar modifications,
- `x`=`V`, if it uses single vector modifications,
- and `x`=`D`, if dyadic product modifications are used.

Similarly, the letter at position `y` determines the way the transformation matrix is constructed

- `y`=`S`, if the algorithm uses single elementary modifications,
- `y`=`C`, if it uses cumulative elementary modifications.

procedure MRAP2MMAP-DC $(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K})$
  $iter \leftarrow 1$
  while $\boldsymbol{H_0}, \ldots, \boldsymbol{H_K}$ **not** Markovian **and** $iter < maxIter$ **do**
    $\boldsymbol{T} \leftarrow \boldsymbol{I}$
    for $i = 1$ **to** N **do**
      $(\underline{u}^*, \underline{v}^*) \leftarrow \arg\min_{\underline{u},\underline{v}} G(Tr(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K}, \boldsymbol{M_D}(\boldsymbol{T}, \underline{u}, \underline{v})))$
      $\boldsymbol{T} \leftarrow \boldsymbol{M_D}(\boldsymbol{T}, \underline{u}^*, \underline{v}^*)$
    end for
    $(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K}) \leftarrow Tr(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K}, \boldsymbol{T})$
    $iter \leftarrow iter + 1$
  end while
return $(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K})$
end procedure

Fig. 3. Representation transformation method with single ($N = 1$) and cumulative ($N > 1$) dyadic elementary modifications

Figure 3 with $N = 1$ and $N > 1$ depicts two possible variants of this family of methods. The goal function is the sum of the square of the negative elements of the representation, thus

$$G(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K}) = \sum_{\substack{i,j \\ i \neq j}} ((-[\boldsymbol{H_0}]_{i,j})^+)^2 + \sum_{k=1}^{K} \sum_{i,j} ((-[\boldsymbol{H_k}]_{i,j})^+)^2. \tag{8}$$

4 Representation transformation methods with size extension

If a size n non-Markovian MRAP has no Markovian representation of size n, then the above discussed procedures are not applicable. There are cases, however, when a MRAP of order n has a Markovian representation of order $m > n$. With a simple modification, our algorithms can be applied to the problem of finding a larger Markovian representation as well. The idea is based on the results of [6].

Theorem 1. *([6], Theorem 4) If, for an MRAP $(\boldsymbol{D_0}, \ldots, \boldsymbol{D_K})$ of size m and initial vector $\underline{\pi}$ (where $\underline{\pi}$ is the solution of $\underline{\pi} = \underline{\pi}(\boldsymbol{D_0})^{-1} \sum_{k=1}^{k} \boldsymbol{D_k}$ and $\underline{\pi}\mathbb{1} = 1$), the rank of the representation with respect to the initial vector (see Definition 5) is n, and $n < m$, then there exists a non singular $m \times m$ transformation matrix $\boldsymbol{T}$, such that the transformed representation has the following block structure (with block sizes n and $m - n$)*

$$\boldsymbol{T}\boldsymbol{D_k}\boldsymbol{T}^{-1} = \begin{bmatrix} \boldsymbol{H_k} & \boldsymbol{0} \\ \star & \star \end{bmatrix}, \text{ for } k = 0, 1, \ldots, K \text{ and } \underline{\pi}\boldsymbol{T} = \begin{bmatrix} \underline{\gamma} & 0 \end{bmatrix}, \tag{9}$$

where $\star$ denotes irrelevant matrix blocks with arbitrary elements, and $(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K})$ is an equivalent representation of the same process with size n and initial vector $\underline{\gamma}$.

This theorem together with the further results of [6] provide a way to obtain a smaller representation of a RAP, if it is possible. In this paper, however, we have the opposite problem. We are investigating RAPs not having a Markovian representation of the same size, thus we need to find a way to extend the size of the representation.

Theorem 2. *If representation* $(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K})$ *of size* n *is non-Markovian, but the process has a* $(\boldsymbol{D_0}, \ldots, \boldsymbol{D_K})$ *Markovian representation of size* $n+1$ *whose rank with respect to the initial vector is* n*, then such a representation can be obtained in the form* $\boldsymbol{D_k} = \boldsymbol{T}^{-1}\boldsymbol{H_k}'\boldsymbol{T}$*,* $k = 0, 1, \ldots, K$*, where*

$$\boldsymbol{H_k}' = \left[\begin{array}{c|c} \boldsymbol{H_k} & \begin{matrix}0\\ \vdots \\ 0\end{matrix} \\ \hline [\underline{x_k}]_1 \ \cdots \ \cdots \ [\underline{x_k}]_{n+1} & \end{array}\right],$$

and matrix $\boldsymbol{T}$*, vectors* $\underline{x_k}$*,* $k = 0, 1, \ldots, K$*, are such that* $\boldsymbol{T}\mathbb{1} = \mathbb{1}$*,* $\sum_{k=0}^{K} \underline{x_k}\mathbb{1} = 0$*, and* $[\underline{x_0}]_{n+1} / \sum_{k=1}^{K} [\underline{x_k}]_{n+1} \neq -1$*.*

Proof. By construction we have that $\boldsymbol{T}$ transforms matrices $\boldsymbol{D_k}$, $k = 0, 1, \ldots, K$, into the required block structure of Theorem 1.

It remains to be proven that the initial vector also has the required block structure. That is, the solution of $\underline{\gamma'} = \underline{\gamma'}(-\boldsymbol{H_0}')^{-1} \sum_{k=1}^{K} \boldsymbol{H_k}'$ has the form $\underline{\gamma'} = \begin{bmatrix}\underline{\gamma} & 0\end{bmatrix}$. Focusing on the last element of the vector equation and utilizing the block structure of matrices $\boldsymbol{H_k}'$ we have $[\underline{\gamma'}]_{n+1} = [\underline{\gamma'}]_{n+1} \frac{-1}{[\underline{x_0}]_{n+1}} \sum_{k=1}^{K} [\underline{x_k}]_{n+1}$. From this scalar equation we have that $[\underline{\gamma'}]_{n+1} = 0$ if $[\underline{x_0}]_{n+1} / \sum_{k=1}^{K} [\underline{x_k}]_{n+1} \neq -1$. □

Based on Theorem 2 we search for a Markovian representation of size $n+1$ with the algorithm in Figure 4. For simplicity we omit the constraint $[\underline{x_0}]_{n+1} / \sum_{k=1}^{K} [\underline{x_k}]_{n+1} \neq -1$ in the algorithm description, as, in practice, the procedure never generates vector elements which violate it. This algorithm is based on the alternating optimization of the elementary transformation matrix and vectors $\underline{x_k}$, $k = 0, 1, \ldots, K$.

The $Tr()$ operator that performs the similarity transformation with size extension is defined by

$$Tr'(\boldsymbol{H_0}, \ldots, \boldsymbol{H_K}, \underline{x_0}, \ldots, \underline{x_K}, \boldsymbol{T}) =$$
$$\left(\boldsymbol{T}^{-1}\left[\begin{array}{c|c} \boldsymbol{H_0} & \begin{matrix}0\\ \vdots \\ 0\end{matrix} \\ \hline [\underline{x_0}]_1 \ \cdots \ \cdots \ [\underline{x_0}]_{n+1} & \end{array}\right]\boldsymbol{T}, \ldots, \boldsymbol{T}^{-1}\left[\begin{array}{c|c} \boldsymbol{H_K} & \begin{matrix}0\\ \vdots \\ 0\end{matrix} \\ \hline [\underline{x_K}]_1 \ \cdots \ \cdots \ [\underline{x_K}]_{n+1} & \end{array}\right]\boldsymbol{T}\right) \tag{10}$$

procedure MRAP2MMAP-SizeExt($\boldsymbol{H_0}, \dots, \boldsymbol{H_K}$)
  $\underline{x_k} \leftarrow$ random vector, $k = 0, \dots, K$, such that $\sum_{k=0}^{K} \underline{x_k}\mathbb{1} = 0$
  $\boldsymbol{T} \leftarrow \boldsymbol{I}$
  while $\boldsymbol{D_0}, \dots, \boldsymbol{D_K}$ **not** Markovian **and** $iter < maxIter$ **do**
    $\boldsymbol{B} \leftarrow \boldsymbol{I}$
    for $i = 1$ **to** N **do**
      $(\underline{u}^*, \underline{v}^*) \leftarrow \arg\min\limits_{\underline{u},\underline{v}} G(Tr'(\boldsymbol{H_0}, \dots, \boldsymbol{H_K}, \underline{x_0}, \dots, \underline{x_K}, \boldsymbol{T} \cdot \boldsymbol{M_D}(\boldsymbol{B}, \underline{u}, \underline{v})))$
      $\boldsymbol{B} \leftarrow \boldsymbol{M_D}(\boldsymbol{B}, \underline{u}^*, \underline{v}^*)$
    end for
    $\boldsymbol{T} \leftarrow \boldsymbol{T} \cdot \boldsymbol{B}$;
    $(\underline{x_0}^*, \dots, \underline{x_K}^*) \leftarrow \underset{\substack{\underline{x_0},\dots,\underline{x_k} \in \mathbb{R}^{(n+1)} \\ \sum_{k=0}^{K} \underline{x_k}\mathbb{1}=0}}{\arg\min} G(Tr'(\boldsymbol{H_0}, \dots, \boldsymbol{H_K}, \underline{x_0}, \dots, \underline{x_K}, \boldsymbol{T}))$
    $(\underline{x_0}, \dots, \underline{x_k}) \leftarrow (\underline{x_0}^*, \dots, \underline{x_k}^*)$
    $(\boldsymbol{D_0} \dots, \boldsymbol{D_K}) = Tr'(\boldsymbol{H_0}, \dots, \boldsymbol{H_K}, \underline{x_0}, \dots, \underline{x_K}, \boldsymbol{T})$
    $iter \leftarrow iter + 1$
  end while
return $(\boldsymbol{D_0}, \dots, \boldsymbol{D_K})$
end procedure

Fig. 4. Representation transformation with size extension using cumulative dyadic product modifications

Based on the duality of redundant representations in [6], one might miss a size extension approach with redundancy according to the closing vector. It is possible to construct a counterpart procedure of similar nature that generates representation with redundancy according to the closing vector, but we have two reasons for not recommending such procedures.

- Each ME representation satisfying simple eigenvalue and density conditions (namely, the dominant eigenvalue is real and has the largest real part; the density is strictly positive on $(0, \infty)$) has a PH representation with redundant initial vector.
- We have the following conjecture:
  If $(\boldsymbol{H_0}, \dots, \boldsymbol{H_K})$ of size n is non-Markovian and it has a Markovian representation of size $n + 1$, then it has a Markovian representation of size $n + 1$ with a redundant initial vector and an other one with redundant closing vector.

5 Numerical results

In this section we will provide some examples to demonstrate the behavior of the algorithms introduced.

5.1 Finding a Markovian representation of the same size

To investigate how efficient the various representation transformation methods are, we define a kind of benchmark that consists of a large number of non-Markovian representations of MAPs. We are interested in how many times the various methods succeed to obtain a Markovian representation.

The MAPs belonging to the benchmark are generated as follows.

I *"Balanced" problems.* In this problem class we generate MAPs with matrices having random integer entries falling into $[0, 10]$. This random, but Markovian representation is then transformed to a non-Markovian one with a random transformation matrix. The benchmark includes
   a.) 10 representations of size 3,
   b.) 10 representations of size 4,
   c.) and 10 representations of size 5.

II *"Stiff" problems.* MAPs belonging to this class have random entries as in case of "Balanced" MAPs, but a particular matrix row or an entry is several orders of magnitudes smaller or larger than the other ones. Again, a random transformation matrix is applied to obtain a non-Markovian representation. We expect that it will be more difficult to obtain a Markovian representation for these problems. The benchmark includes
   a.) 10 representations of size 3 with random integer entries falling into $[0, 100000]$ except for the first row, where the random entries are from $[0, 10]$.
   b.) 10 representations with similar construction, but the size is 4
   c.) 10 representations of size 3 with random integer entries falling into $[0, 100000]$ except for the first row, where the random entries are from $[0, 10]$. One entry in the first row is from $[0, 100000]$.
   d.) 10 representations with similar construction, but the size is 4.
   e.) 10 representations of size 3 with random integer entries falling into $[0, 10]$ except for the first row, where the random entries are from $[0, 100000]$.
   f.) 10 representations with similar construction, but the size is 4
   g.) 10 representations of size 3 with random integer entries falling into $[0, 10]$ except for the first row, where the random entries are from $[0, 100000]$. One entry in the first row is from $[0, 10]$.
   h.) 10 representations with similar construction, but the size is 4.

III *"Sparse" problems.* In this case the matrices of the MAPs have a large number of 0 entries. The non-zero entries are random integers from $[0, 10]$, and the positions of the non-zero entries are randomly chosen as well (such that the irreducibility of the background process is ensured). The non-Markovian input for the algorithms is obtained by a random similarity transformation again. The more zero entries the matrices have, the smaller the space of Markovian representations for that particular process is, thus we expect that it is difficult to find a Markovian solution for these problems as well. The following sparse problems are included in the benchmark
   a.) 10 representations of size 3 with 1 non-zero entry in each row (including the rows of $\boldsymbol{H_k}, k = 0, \ldots, K$, except the diagonal of $\boldsymbol{H_0}$),

b.) 10 representations of size 3 with 2 non-zero entry in each row,
c.) 10 representations of size 3 with 3 non-zero entry in each row,
d.) 10 representations of size 4 with 1 non-zero entry in each row,
e.) 10 representations of size 4 with 2 non-zero entry in each row,
f.) 10 representations of size 4 with 3 non-zero entry in each row.

IV *"Near-bound" problems.* These problems are based on the numerical example of [7]. Matrices $\boldsymbol{H_0}$ and $\boldsymbol{H_1}$ are given by

$$\boldsymbol{H_0} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -3 & h \\ 0 & -h & -3 \end{bmatrix}, \quad \boldsymbol{H_1} = \begin{bmatrix} 1 \\ 3-h \\ 3+h \end{bmatrix} \cdot \begin{bmatrix} 0.2 & 0.3 & 0.5 \end{bmatrix}. \tag{11}$$

It is proven in [7] that this RAP has a Markovian representation if $h \leq 0.552748375$. In order to quantify the efficiency of the algorithms, we are trying to approach this bound as much as possible and check which algorithm finds the Markovian representation. Thus, the benchmark includes 12 MAPs defined by (11) with h values running from 0 to 0.55.

In total, this benchmark consists of 182 MAPs given by non-Markovian representations. The algorithms involved in the comparison are the `MRAP2MMAP-[7]` (see Figure 1) and the 6 variants of our algorithm introduced in Section 3.2.

We implemented all the algorithms in MATLAB environment. To solve the arising nonlinear optimization problems we used exhaustive search for the discrete and the built-in `fminsearch` function for the continuous variables. The `fminsearch` function is based on the Nelder-Mead simplex algorithm that can be found in most scientific toolboxes.

We are interested in the following measures:

- The number of times the algorithm finds a Markovian representation with a tolerance of -10^{-5}, denoted by N_1. All results are considered as Markovian if the smallest element (except the diagonal of $\boldsymbol{H_0}$) is greater than the tolerance.
- The number of times the algorithm finds a Markovian representation with a tolerance of -10^{-7}, denoted by N_2.
- The average execution time of the algorithm denoted by T.

The results are summarized in Table 1. For the "balanced" problems all algorithms performed well. There are some cases when an algorithm stucks in a local optimum and fails to find a solution, but it does not occur frequently. For the "stiff" problems the results are quite mixed. Our `VC` variant wins in this case, with the `DC` and `DS` variants being the second. `MRAP2MMAP-[7]` failed to find a solution several times. The algorithms had a hard time obtaining a solution for "sparse" problems as well. Our `DC` variant managed to solve more problems than the other algorithms.

In general, we can conclude that the cumulative transformation matrix modifications perform much better than the single ones, and that the "dyadic product" and the "single vector" modifications give the best overall performance. Regarding the execution times, it is surprising how fast `MRAP2MMAP-[7]` is, it is clearly the fastest method in the comparison.

Table 1. Results of the comparison of the algorithms

Problem	MRAP2MMAP-[7]			MRAP2MMAP-DS			MRAP2MMAP-DC			MRAP2MMAP-VS			MRAP2MMAP-VC			MRAP2MMAP-SS			MRAP2MMAP-SC		
	N_1	N_2	T[s]	N_1	N_2	T[s]	N_1	N_2	T[s]	N_1	N_2	T[s]	N_1	N_2	T[s]	N_1	N_2	T[s]	N_1	N_2	T[s]
I. "Balanced" problems																					
I.a.)	10	10	0.028	10	10	1.63	10	10	1.92	9	9	20.63	10	10	4.81	10	10	2.63	10	10	2.66
I.b.)	10	10	0.68	10	10	6.74	10	10	4.69	10	10	49.9	10	10	17.9	10	9	40.8	10	10	11.4
I.c.)	10	10	0.36	10	10	11.9	10	10	8.69	9	8	354	9	9	166	9	9	57.7	10	9	96.7
II. "Stiff" problems																					
II.a.)	10	10	2.07	10	10	5.59	9	9	13.8	10	10	57.6	10	10	10.2	9	9	10.8	10	10	6.94
II.b.)	10	10	3.85	9	9	48.6	10	10	8.8	7	7	219	9	9	72.9	9	9	48.8	8	8	70.8
II.c.)	10	10	0.11	9	9	15.8	10	10	2.63	7	7	76.9	10	10	10.6	8	7	20.4	9	9	12.2
II.d.)	10	10	0.44	10	10	21.3	10	10	6.73	8	8	182	9	9	81.5	7	7	70.3	10	10	31.6
II.e.)	5	5	7.66	10	10	8.16	10	10	4.5	9	9	47.3	9	9	28.8	8	8	14.6	10	10	7.66
II.f.)	2	2	23.6	9	9	82.3	8	8	79.2	7	7	270	10	10	51.5	10	9	48.9	9	8	57.2
II.g.)	4	4	9	10	10	11.3	10	10	4.65	9	9	34.6	9	9	37.4	6	6	21.1	10	10	8.48
II.h.)	2	2	23.6	6	6	165	7	7	107	8	7	255	10	10	80.8	5	5	86.1	8	7	65.5
III. "Sparse" problems																					
III.a.)	9	9	8.06	10	10	42.6	10	10	7.07	8	8	83.8	10	10	14.9	8	7	25.4	9	9	19
III.b.)	5	5	6.42	10	10	6.16	10	10	4.69	7	6	72.3	10	9	36.3	5	4	31.9	7	6	28.9
III.c.)	8	8	1.69	9	9	20.8	9	9	20.4	5	5	69.6	8	8	32.8	5	5	29.4	7	7	20.8
III.d.)	8	7	20	9	6	192	10	9	71.5	3	2	383	8	8	175	7	5	122	7	7	93.5
III.e.)	4	2	16.4	4	2	287	8	8	88.3	2	1	364	7	6	269	2	1	151	3	2	131
III.f.)	4	4	10.5	6	6	154	6	6	114	2	2	311	5	5	239	1	1	140	3	3	132
IV. "Near bound" problem																					
IV.	6	5	3.58	12	12	6.23	12	12	2.56	1	1	127	12	12	11.4	12	12	36.9	10	10	25.2
Summary																					
Average	70%	68%	7.64	90%	87%	60.4	93%	92%	30.6	66%	64%	165	91%	90%	74.5	72%	68%	53.3	82%	80%	45.6

5.2 Finding a Markovian representation with size extension

There is no method available to determine the MAP order of a RAP in the general case. Therefore the examples discussed here will be special RAPs: we will only examine renewal processes with ME(3) distributed inter-event times as there are results for the order of these.

Our first example is an order 3 RAP. The inter-arrival time distribution is a matrix exponential distribution discussed by Harris in [9].

$$\boldsymbol{H_0} = \begin{bmatrix} -1 & 1 & 0 \\ 0 & -2 & 2 \\ 0 & 0 & -3 \end{bmatrix}, \quad \boldsymbol{H_1} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ a & b & a \end{bmatrix},$$

where $b = 3 - 2a$.

With $a > 1.5$ the process does not have an order 3 Markovian representation. With $a = 2$ the process is on the border of the representation space of order 4 MAPs because with parameter $a > 2$ it will not have a size 4 Markovian representation. All our size extension based methods find a size 4 Markovian representation for $a = 2$ with a tolerance of 10^{-7}. The fastest are the dyadic optimization methods, needing only a few minutes, while the slowest `SC` scalar optimization method needs more than an hour. As an example we present the output of the `VS` algorithm

$$\boldsymbol{D_0} = \begin{bmatrix} -1 & 0 & 0 & 1 \\ 0 & -2.12 & 2.12 & 0 \\ 0 & 0 & -4 & 0.03 \\ 0 & 2.79 & 0 & -2.79 \end{bmatrix}, \quad \boldsymbol{D_1} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1.97 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

In the second example we examine an order 3 RAP with a structure similar to the previous one.

$$\boldsymbol{H_0} = \begin{bmatrix} -1 & 1 & 0 \\ 0 & -1 & 1 \\ 0 & 0 & -1 \end{bmatrix}, \quad \boldsymbol{H_1} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ a & b & a \end{bmatrix},$$

with $b = 1 - 2a$. By applying Theorem 7.5 in [10] one can get that this process has a size $n = 3 + \lceil \frac{\xi}{2-\xi} \rceil$ Markovian representation with triangular $\boldsymbol{D_0}$, where $\xi = \frac{(1-2a)^2}{a^2}$ if and only if $a < \frac{1}{2-\sqrt{2}}$. According to Conjecture 7.6 in the same paper this is also the order of the MAP. If $a > 0.5$, this order is at least 4. (The confirmation can be done similar to the first example.) If $0.5 < a \leq 1$ the order is exactly 4, as can be seen from the previous formula for n. For order 5, we get the $1 < a \leq 1 + (0.25(3 + \sqrt{3}))$ boundaries. We choose $a = 1$, thus the process will be on the border of order 4 and order 5 MAP class. The results are similar to that of the first example. Each method finds a Markovian representation with 10^{-7} tolerance. The dyadic and vector based optimizations need a couple of minutes, the scalar based optimizations finish in around an hour. We give the output of

the `VS` method in this case as an example.

$$\boldsymbol{D_0} = \begin{bmatrix} -1 & 0.048 & 0.005 & 0.802 \\ 0 & -1.07 & 0.48 & 0 \\ 0 & 0.119 & -1.83 & 0 \\ 0 & 0.972 & 0 & -1 \end{bmatrix}, \quad \boldsymbol{D_1} = \begin{bmatrix} 0.104 & 0 & 0.041 & 0 \\ 0.38 & 0.015 & 0.194 & 0.002 \\ 1.029 & 0.069 & 0.607 & 0 \\ 0 & 0.008 & 0.02 & 0 \end{bmatrix}.$$

6 Conclusion

We have presented a set of representation transformation methods for finding Markovian representation based on general (non-Markovian) matrix representations of various stochastic models including ME distributions, RAPs and MRAPs. The presented new methods relax two limitations of the previously applied ones: the single element based similarity transformation of the representation in each iteration cycle and the fixed size of the representations. The price of the more complex methods is the potentially increased computational complexity and the dependence on non-linear optimization tools.

Acknowledgement This work was supported by the Hungarian Government through the TÁMOP-4.2.2C-11/1/KONV-2012-0001 and the OTKA K101150 projects, and by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences.

References

1. Neuts, M.F.: Matrix-Geometric Solutions in Stochastic Models: An Algorithmic Approach. Dover (1981)
2. Neuts, M.F.: A versatile Markovian point process. Journal of Applied Probability **16**(4) (1979) 764–779
3. He, Q.M., Neuts, M.: Markov arrival processes with marked transitions. Stochastic Processes and their Applications **74** (1998) 37–52
4. Bladt, M., Neuts, M.F.: Matrix-exponential distributions: Calculus and interpretations via flows. Stochastic Models **19**(1) (2003) 113–124
5. Bean, N., Nielsen, B.: Quasi-birth-and-death processes with rational arrival process components. Stochastic Models **26**(3) (2010) 309–334
6. Buchholz, P., Telek, M.: On minimal representations of rational arrival processes. Annals of Operations Research (2011) 1–24
7. Telek, M., Horváth, G.: A minimal representation of Markov arrival processes and a moments matching method. Performance Evaluation **64**(9) (2007) 1153–1168
8. Buchholz, P., Kemper, P., Kriege, J.: Multi-class Markovian arrival processes and their parameter fitting. Performance Evaluation **67**(11) (2010) 1092–1106
9. Harris, C.M., Marchal, W.G., Botta, R.F.: A note on generalized hyperexponential distributions. Stochastic Models **8**(1) (1992) 179–191
10. O'Cinneide, C.A.: Triangular order of triangular phase-type distributions. Stochastic Models **9**(4) (1993) 507–529