Aggregation of Markovian Models
- An Alternating Least Squares Approach -

Peter Buchholz and Jan Kriege
Department of Computer Science,
TU Dortmund,
Dortmund, Germany
{peter.buchholz,jan.kriege}@udo.edu

Author prepared version of a paper published in

Proc. of the 9th International Conference on Quantitative Evaluation of SysTems (QEST 2012), IEEE Computer Society, 2012.

http://dx.doi.org/10.1109/QEST.2012.17

Aggregation of Markovian Models
- An Alternating Least Squares Approach -

Peter Buchholz and Jan Kriege
Department of Computer Science, TU Dortmund, Dortmund, Germany
{peter.buchholz, jan.kriege}@udo.edu

Abstract

To deal with the problem of state space explosion in Markovian models often compositional modeling and the aggregation of components are used. Several approximate aggregation methods exist which are usually based on heuristics. This paper presents a new aggregation approach for Markovian components which computes aggregates that minimize the difference according to some algebraically defined function which describes the difference between the component and the aggregate. If the difference becomes zero, aggregation is exact and component and aggregate are indistinguishable. Approximate aggregates are computed using an alternating least squares approach which tries to minimize the norm-wise difference between the original component and the aggregate. The approach is extended to generate bounding aggregates which allow one to compute bounds on transient or stationary quantities when the aggregate is embedded in an environment.

Keywords: Compositional Modeling, Aggregation, Markov Models, Non-Negative Least Squares, Bounds;

1 Introduction

A major problem when analyzing Markovian models is the state space explosion which makes models intractable because of their size. Often large models that exceed the number of states, which can be solved with modern workstations, result from the modeling of real systems. To deal with this problem, compositional modeling and the reduction of the state space resulting in smaller components that are combined into one complex model are applied. Compositional Markovian models appeared in the literature of the recent decades in various forms like as stochastic automata networks (SANs) [30], compositional variants of stochastic Petri Nets (SPNs) [16], stochastic process algebras [21], hierarchical Markovian models [5] or interactive Markov chains [20]. In contrast to many other variants we consider here models with active output and passive input signals similar to probabilistic or stochastic variants of I/O automata [34].

In the past several approaches have been proposed [16, 20, 21, 30, 29] that reduce the state space of components by finding a smaller component with equivalent behavior and substitute the original component by the smaller one in the compositional model. These approaches have in common that stochastic bisimulation [7, 22] which is based on lumpability [6, 23] is used to reduce the size of the state space. Since the overall state space grows combinatorially in the sizes of the state spaces of the

components, substituting a component with n states by a component with $m < n$ states reduces the overall number of states by a factor m/n.

Approaches using bisimulation for state space reduction exploit the fact that bisimulation is a congruence according to model composition. In [32] a general equivalence definition for components of the same size has been proposed that goes beyond bisimulation and that has been extended to equivalence of components of different sizes in [10, 13]. This equivalence definition, which includes bisimulation and lumpability as special cases, relates Markovian and non-Markovian representations (called Rational Arrival Processes [4]) and has been used in [11] to compute minimal representations for several components. However, the approach from [11] usually results in non-Markovian representations, which lack the intuitive representation as a Markov chain, and experiments show that the resulting models are often only slightly smaller than models resulting from lumpability. Therefore, we present an approach that uses approximate instead of exact reduction by solving non-negative least squares problems and restrict the model class to Markovian models by applying uniformization to the Markovian components.

This paper is structured as follows. In Section 2 the basic model class is introduced. Our approach for an approximate aggregation of Markovian components using a least squares optimization approach is given in Section 3. In Section 4 we present additional examples to show the possibilities of approximate state space reduction for PH distributions and MAPs as one specific application example. The paper ends with the conclusions in Section 5.

2 Compositional Markovian Models

In this section we first introduce Markovian components, describe afterwards their composition and introduce an equivalence relation for Markovian components. The section captures results which have been published in [11] for a more general class of models, denoted as rational (arrival) processes. However, in contrast to [11] we restrict models to Markovian models and define the equivalence relation accordingly. To remain in the class of Markovian models, we apply uniformization to Markovian components which has not been done yet.

2.1 Markov Components

We begin with the definition of Markovian components with input and output transitions denoted as events.

Definition 1 $\mathcal{A} = (\pi_0, \mathbf{G}_0, \mathbf{G}_1, \ldots, \mathbf{G}_K, \mathbf{U}_{K+1}, \ldots, \mathbf{U}_{K+L})$ *is a Markovian component of order* n *(i.e., with state space* $\mathcal{S} = \{0, \ldots, n-1\}$*) with input and output signals, where*

- π_0 *is an* n*-dimensional distribution (row) vector,*
- $\mathbf{G}_0$ *is an* $n \times n$ *matrix with non-negative elements outside the diagonal and row sum* ≤ 0*,*
- $\mathbf{G}_k$ *(*$k = 1, \ldots, K$*) are non-negative* $n \times n$ *matrices such that* $\mathbf{G} = \mathbf{G}_0 + \sum_{k=1}^{K} \mathbf{G}_k$ *is a valid generator matrix of a CTMC,*

- $\mathbf{U}_l$ *($l = K+1, \ldots, K+L$) are stochastic matrices*[1].

The component starts in state $i \in \mathcal{S}$ with probability $\pi_0(i)$ and behaves like a Markov chain with generator matrix $\mathbf{G}$. If a transition defined in matrix $\mathbf{G}_k$ occurs event k is emitted. The component may additionally receive events from its environment. If it receives event l in state i, the state changes immediately to j with probability $\mathbf{U}_l(i,j)$. Events from the environment are emitted by other components which are composed with the component.

If $\mathbf{G}$ is an irreducible generator matrix or contains only single irreducible submatrix, then it has a unique stationary vector which is given by

$$\pi\mathbf{G} = \mathbf{0} \text{ and } \pi\mathbb{1} = 1. \tag{1}$$

Let $\alpha \geq \max_{i\in\mathcal{S}} |\mathbf{G}_0(i,i)|$ and define $\mathbf{A}_0 = \mathbf{G}_0/\alpha + \mathbf{I}$, $\mathbf{A}_k = \mathbf{G}_k/\alpha$. $\mathbf{A}_0, \mathbf{A}_1, \ldots, \mathbf{A}_K$ are non-negative matrices and $\mathbf{A} = \mathbf{A}_0 + \sum_{k=1}^{K} \mathbf{A}_k$ is a stochastic matrix resulting from uniformization [31] of the original matrices. The matrices describe an embedded discrete time process which specifies the behavior of the continuous time process at the event times of a Poisson process. Observe that incoming events are not modified by uniformization since they are already described in the continuous time process by conditional probabilities.

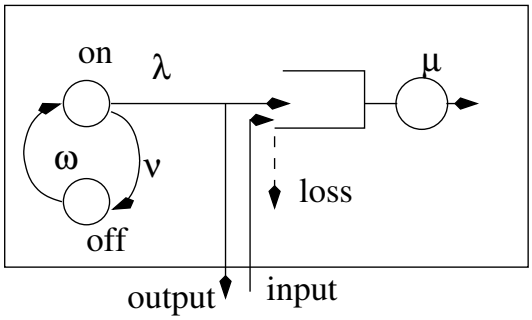

Figure 1: First example model.

Examples: We consider two small running examples to clarify the approach. The first example is a finite capacity queue with IPP input and exponential service times. If the queue is full and an internal customer arrives, an output signal is generated. Furthermore, customers may be routed via an input signal into queue. If a customer arrives from the outside to a full queue, the customer gets lost.

States of the model can be described by tuples (i_1, i_2) where i_1 describes the state of the IPP (0 equals *off* and 1 equals *on*) and i_2 describes the number of customers in the queue. We consider a small configuration with queue capacity 3 resulting in a state space with 8 states: $\mathcal{S} = \{(0,0), (1,0), (0,1), (1,1), (0,2), (1,2), (0,3), (1,3)\}$. According to this state space ordering the following matrices describe the model. Diagonal elements of matrix $\mathbf{G}_0$ are not printed, they are given by the sum of

[1] In this definition signals $1, \ldots, K$ are output signals and signal $K+1, \ldots, K+L$ are input signals. This numbering is not mandatory. We assume that a component has K output and L input signals which are numbered arbitrarily. This generalization of the numbering is required to define composition. Only number 0 is fixed and used for the matrix of internal events and has negative diagonal elements.

non-diagonal elements in the row of $\mathbf{G}_0$ and by the sum of elements in the corresponding row of $\mathbf{G}_1$.

$$\mathbf{G}_0 = \begin{pmatrix} -\Sigma & \omega & 0 & 0 & 0 & 0 & 0 & 0 \\ \nu & -\Sigma & 0 & \lambda & 0 & 0 & 0 & 0 \\ \mu & 0 & -\Sigma & \omega & 0 & 0 & 0 & 0 \\ 0 & \mu & \nu & -\Sigma & 0 & \lambda & 0 & 0 \\ 0 & 0 & \mu & 0 & -\Sigma & \omega & 0 & 0 \\ 0 & 0 & 0 & \mu & \nu & -\Sigma & 0 & \lambda \\ 0 & 0 & 0 & 0 & \mu & 0 & -\Sigma & \omega \\ 0 & 0 & 0 & 0 & 0 & \mu & \nu & -\Sigma \end{pmatrix}$$

$$\mathbf{G}_1 = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \lambda \end{pmatrix} \qquad \mathbf{U}_2 = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

By means of uniformization the matrices $\mathbf{G}_0$ and $\mathbf{G}_1$ can be transformed into stochastic matrices.

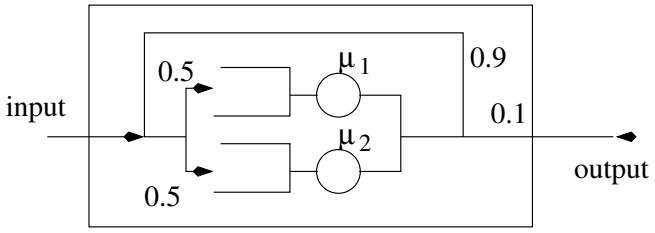

Figure 2: Second example model.

The second example is shown in Figure 2. It is a simple component with 2 parallel queues which receive customers from the outside and emit customers after they have been served. For $\mu_1 = \mu_2$ the model is lumpable and the identity of the queues needs no longer be distinguished. To obtain a finite state space, we consider the system with up to 3 customers, additionally arriving customers get lost. States can be described by tuples (i_1, i_2) where i_j equals the number of customers in queue j. We use the following state space ordering $\mathcal{S} = \{(0,0), (0,1), (1,0), (0,2), (1,1), (2,0), (0,3), (1,2), (2,1), (3,0)\}$. For lack of space, we do not print the whole matrices which can be easily derived from the model description.

2.2 Composition of Components

One of the major advantages of component based modeling is the possibility to compose large models from simple components. We consider here only the composition of two components which is sufficient to show properties of our aggregation approach in a compositional environment. Later we briefly outline how the approach can be extended for the composition of more than two components. The composition approach we use distinguishes input and output events. One output event is mapped onto one input event in the other component. The component that generates the output event determines the transition rate and the component with the input event has to accept the incoming transition. This viewpoint of composition is also used in probabilistic I/O automata [34] or hierarchical queueing networks [5]. It is possible to define more general forms of composition without explicitly distinguishing input and output events but as shown recently in [11] this requires strict conditions to preserve equivalence between components.

Let $\mathcal{A}^{(1)} = (\pi_0^{(1)}, \mathbf{G}_0^{(1)}, \mathbf{G}_1^{(1)}, \ldots, \mathbf{G}_K^{(1)}, \mathbf{U}_{K+1}^{(1)}, \ldots, \mathbf{U}_{K+L}^{(1)})$ and $\mathcal{A}^{(2)} = (\pi_0^{(2)}, \mathbf{G}_0^{(2)}, \mathbf{G}_{K+1}^{(2)}, \ldots, \mathbf{G}_{K+L}^{(2)}, \mathbf{U}_1^{(2)}, \ldots, \mathbf{U}_K^{(2)})$ be two Markovian components of order $n^{(1)}$, $n^{(2)}$ that are composed by assigning output signals from $\mathcal{A}^{(1)}$ to input signals of $\mathcal{A}^{(2)}$ and vice versa. We denote two Markovian components as compatible if they can be composed in this way (i.e., output signals in one component correspond to the input signals in the other component). The descriptor matrix of the composed model is then given by

$$\mathbf{G}^{(12)} = \mathbf{G}_0^{(1)} \oplus \mathbf{G}_0^{(2)} + \sum_{k=1}^{K} \mathbf{G}_k^{(1)} \otimes \mathbf{U}_k^{(2)} + \sum_{k=K+1}^{K+L} \mathbf{U}_k^{(1)} \otimes \mathbf{G}_k^{(2)} \tag{2}$$

and the initial vector equals $\pi_0^{(12)} = \pi_0^{(1)} \otimes \pi_0^{(2)}$. $\oplus$ and $\otimes$ are the Kronecker sum and product, respectively, which are introduced for example in [31] in detail. Then the distribution at time t equals $\pi_t^{(12)} = \pi_0^{(12)} e^{\mathbf{G}t}$ and the stationary distribution equals $\pi^{(12)} = \lim_{t\to\infty} \pi_t^{(12)}$. We use the notation $\mathcal{A}^{(12)} = \mathcal{A}^{(1)} \|_c \mathcal{A}^{(2)}$ for the composition.

The matrix $\mathbf{G}^{(12)}$ describes a CTMC, if we consider additionally the matrices $\mathbf{G}_k^{(12)}$ which equal $\mathbf{G}_k^{(1)} \otimes \mathbf{U}_k^{(2)}$ for $k \leq K$ and $\mathbf{U}_k^{(1)} \otimes \mathbf{G}_k^{(2)}$ for $K < k \leq K + L$, then we obtain a CTMC with labeled transitions. Furthermore, $\mathbf{G}_0^{(12)} = \mathbf{G}_0^{(1)} \oplus \mathbf{G}_0^{(2)}$. Thus, $(\pi_0, \mathbf{G}_0^{(12)}, \mathbf{G}_1^{(12)}, \ldots, \mathbf{G}_{K'}^{(12)})$ ($K' = K + L$) is a CTMC with labeled transitions or in other words a component without input but with output events. This implies that transition rates and therefore the behavior are completely determined. The joint density of observing $(k_1, t_1, \ldots, k_H, t_H)$ with $k_l \in \{1, \ldots, K + L\}$ and $0 < t_1 < \ldots < t_M$ (i.e., observing event k_l at time t_l) for $(\pi_0^{(12)}, \mathbf{G}_0^{(12)}, \mathbf{G}_1^{(12)}, \ldots, \mathbf{G}_{K'}^{(12)})$ is then given by

$$f_{(\pi_0^{(12)}, \mathbf{G}_0^{(12)}, \mathbf{G}_1^{(12)}, \ldots, \mathbf{G}_{K'}^{(12)})}(k_1, t_1, \ldots, k_H, t_H) = \pi_0^{(12)} \left(\prod_{h=1}^{H} e^{t_h \mathbf{G}_0^{(12)}} \mathbf{G}_{k_h}^{(12)} \right) \mathbb{1} \,. \tag{3}$$

Similarly, the stationary throughput of event k is defined as

$$g_{(\pi_0^{(12)}, \mathbf{G}_0^{(12)}, \mathbf{G}_1^{(12)}, \ldots, \mathbf{G}_{K'}^{(12)})}(k) = \pi^{(12)} \mathbf{G}_k^{(12)} \mathbb{1} \,. \tag{4}$$

If we consider the components after uniformization and $\alpha^{(1)}$, $\alpha^{(2)}$ are the uniformization rates, then the transition matrix of the uniformized CTMC of the composed component can be computed directly using the following equation.

$$\mathbf{A}^{(12)} = \frac{1}{\alpha^{(1)} + \alpha^{(2)}} \left(\alpha^{(1)} \mathbf{A}_0^{(1)} \oplus \alpha^{(2)} \mathbf{A}_0^{(2)} + \alpha^{(1)} \sum_{k=1}^{K} \mathbf{A}_k^{(1)} \otimes \mathbf{U}_k^{(2)} + \alpha^{(2)} \sum_{k=K+1}^{K+L} \mathbf{U}_k^{(1)} \otimes \mathbf{A}_k^{(2)} \right) \tag{5}$$

The uniformization rate of the composed model equals $\alpha^{(12)} = \alpha^{(1)} + \alpha^{(2)}$ and it is easy to show that $\mathbf{A}^{(12)} = \mathbf{G}^{(12)}/\alpha^{(12)} + \mathbf{I}$ and $\mathbf{A}_k^{(12)} = \mathbf{G}_k^{(1)}/\alpha^{(12)} \otimes \mathbf{U}_k^{(2)}$ for $1 \leq k \leq K$ and $\mathbf{U}_k^{(1)} \otimes \mathbf{G}_k^{(2)}/\alpha^{(12)}$ for $K < k \leq K + L$. Uniformization [31] can be used to compute the joint densities

$$f_{(\pi_0^{(12)},\mathbf{A}_0^{(12)},\mathbf{A}_1^{(12)},\ldots,\mathbf{A}_K^{(12)})}(k_1, t_1, \ldots, k_H, t_H) = \pi_0^{(12)} \left(\prod_{h=1}^{H} \left(e^{-\alpha^{(12)} t_h} \left(\sum_{j=0}^{\infty} \frac{(\alpha^{(12)} t_h)^j}{j!} \left(\mathbf{A}_0^{(12)} \right)^j \right) \mathbf{A}_{k_h}^{(12)} \right) \right) \mathbb{1}. \quad (6)$$

It should be noted that $\mathbf{G}^{(12)}$ or $\mathbf{A}^{(12)}$ can be reducible and $\mathcal{S}^{(12)}$ may contain states that are unreachable from initial states with non-zero probabilities. In these cases, the matrix structure can be extended using approaches as proposed in [5] or [8]. However, these approaches are beyond the scope of this paper.

The composition can be easily extended to more than two components. If events are sent from one component to another one, the behavior of the composed system is like in queueing networks where customers travel from one subnet to another [5]. In a composed model with J events and X components we have to define for each $j \in \{1, \ldots, J\}$ one $x \in \{1, \ldots, X\}$ such that event j is an output event of x and transition rates are given in matrix $\mathbf{G}_j^{(x)}$. For the remaining components $y \in \{1, \ldots, X\} \setminus \{x\}$ event j is an input event with matrices $\mathbf{U}_j^{(y)}$. If event j does not influence y, then $\mathbf{U}_j^{(y)} = \mathbf{I}_{n^{(y)}}$ where $\mathbf{I}_{n^{(y)}}$ is the identity matrix of order $n^{(y)}$ and $n^{(y)}$ is the number of states in $\mathcal{S}^{(y)}$. With these ingredients the descriptor matrix of a model with X components and J events is given by

$$\mathbf{G}^{(1\ldots X)} = \bigoplus_{x=1}^{X} \mathbf{G}_0^{(x)} + \sum_{j=1}^{J} \bigotimes_{y<x_j} \mathbf{U}_j^{(y)} \bigotimes \mathbf{G}_j^{(x_j)} \bigotimes_{y>x_j} \mathbf{U}_j^{(y)} \quad (7)$$

where x_j is the index of the component where event j is an output event. The model class covers queueing networks [5] and networks of I/O automata [34] with exponential timing.

We have not considered rate based rewards in the definition of components. However, they can be easily integrated as specific events that do not modify the state. Let $\mathbf{r}_k$ a vector of non-negative rewards, then define $\mathbf{G}_k = diag(\mathbf{r}_k)$ which is a diagonal matrix with $\mathbf{r}_k(i)$ in position (i, i). If π_t is the distribution at time t and π is the stationary vector, then $\pi_t \mathbf{G}_k \mathbb{1} = \pi_t \mathbf{r}_k$ and $\pi \mathbf{G}_k \mathbb{1} = \pi \mathbf{r}_k$ are the instantaneous reward at time t and the steady state reward, respectively.

Example: The example models can be combined with other models. Two copies of the first model can be combined by feeding the output of one component to the input of the other one, resulting in a system with two queues and mutual overflow.

2.3 Equivalent Components

The equivalence relation that we consider here is based on an restricted form of the general equivalence for Markovian and non-Markovian systems proposed in [13]. To define equivalence of Markovian components we use the following function

$$ff_{(\pi_0,\mathbf{G}_0,\mathbf{G}_1,\ldots,\mathbf{G}_K,\mathbf{U}_{K+1},\ldots,\mathbf{U}_{K+L})}(k_1, t_1, \ldots, k_H, t_H) = \pi_0 \left(\prod_{h=1}^{H} e^{t_h \mathbf{G}_0} \mathbf{X}_{k_h} \right) \mathbb{1} \quad (8)$$

where $\mathbf{X}_k = \mathbf{G}_k$ if $1 \leq k \leq K$ and $\mathbf{X}_k = \mathbf{U}_k$ if $K < k \leq K + L$. For $L = 0$ $ff_{\cdot}(.)$ equals the density $f_{\cdot}(.)$ but for $L > 0$ $ff_{\cdot}(.)$ is usually not a density, i.e.,

$$\sum_{H=0}^{\infty} \left(\sum_{k_1} \cdots \sum_{k_H} \int_0^{\infty} \int_{\tau_1}^{\infty} \cdots \int_{\tau_{H-1}}^{\infty} f_{(\pi_0,\mathbf{G}_0,\mathbf{G}_1,\ldots,\mathbf{G}_K,\mathbf{U}_{K+1},\ldots,\mathbf{U}_{K+L})} d\tau_1 \ldots d\tau_H \right) \neq 1$$

since input events with unknown rates are included.

Definition 2 *Two Markovian components* $\mathcal{A} = (\pi_0, \mathbf{G}_0, \mathbf{G}_1, \ldots, \mathbf{G}_K, \mathbf{U}_{K+1}, \ldots, \mathbf{U}_{K+L})$ *and* $\mathcal{B} = (\phi_0, \mathbf{H}_0, \mathbf{H}_1, \ldots, \mathbf{H}_K, \mathbf{T}_{K+1}, \ldots, \mathbf{T}_{K+L})$ *are equivalent if and only if*

$$ff_{(\pi_0,\mathbf{G}_0,\mathbf{G}_1,\ldots,\mathbf{G}_K,\mathbf{U}_{K+1},\ldots,\mathbf{U}_{K+L})} = ff_{(\pi_0,\mathbf{H}_0,\mathbf{H}_1,\ldots,\mathbf{H}_K,\mathbf{T}_{K+1},\ldots,\mathbf{T}_{K+L})}$$

for all $(k_1, t_1, \ldots, k_H, t_H)$ *with* $k_h \in \{1, \ldots, K+L)$ *and* $0 \leq t_1 < t_2 < \ldots < t_H$.

The following relation defines an algebraic equivalence relation between components [11, 13].

Definition 3 *Markovian components* $\mathcal{A} = (\pi_0, \mathbf{G}_0, \mathbf{G}_1, \ldots, \mathbf{G}_K, \mathbf{U}_{K+1}, \ldots, \mathbf{U}_{K+L})$ *and* $\mathcal{B} = (\phi_0, \mathbf{H}_0, \mathbf{H}_1, \ldots, \mathbf{H}_K, \mathbf{T}_{K+1}, \ldots, \mathbf{T}_{K+L})$ *of order* m *and* n $(n < m)$*, respectively, are ordinarily related if and only if an* $m \times n$ *matrix* $\mathbf{V}$ *exists such that*

- $\mathbf{V}\mathbb{1} = \mathbb{1}$,
- $\mathbf{G}_k\mathbf{V} = \mathbf{V}\mathbf{H}_k$ *for* $k = 0, \ldots, K$, *and*
- $\mathbf{U}_k\mathbf{V} = \mathbf{V}\mathbf{T}_k$ *for* $k = K+1, \ldots, K+L$.

The following two theorems can be found in [13].

Theorem 1 *If two Markovian components* $\mathcal{A}^{(1)}$ *and* $\mathcal{A}^{(2)}$ *are ordinarily related, then they are equivalent.*

Theorem 2 *If two Markovian components* $\mathcal{A}^{(1)}$ *and* $\mathcal{A}^{(2)}$ *are ordinarily related and* $\mathcal{A}^{(3)}$ *is a compatible component, then* $\mathcal{A}^{(1)}\|_c\mathcal{A}^{(3)}$ *and* $\mathcal{A}^{(2)}\|_c\mathcal{A}^{(3)}$ *as well as* $\mathcal{A}^{(3)}\|_c\mathcal{A}^{(1)}$ *and* $\mathcal{A}^{(3)}\|_c\mathcal{A}^{(2)}$ *are ordinarily related and hence equivalent.*

Observe that the relation may go beyond the equivalence of Markovian components. It allows one to relate Markovian and non-Markovian representations as shown in [11, 13]. The equivalence is also related to lumpability [23] or performance bisimulation [7, 21] since both notations coincide if matrix $\mathbf{V}$ contains only elements from $\{0, 1\}$. In this case, Markovian components can only be related to Markovian components.

Since $\mathbf{G}_0\mathbf{V} = \mathbf{V}\mathbf{H}_0 \Leftrightarrow (\mathbf{G}_0/\alpha + \mathbf{I})\mathbf{V} = \mathbf{V}(\mathbf{H}_0/\alpha + \mathbf{I})$ and $\mathbf{G}_k\mathbf{V} = \mathbf{V}\mathbf{H}_k \Leftrightarrow \mathbf{G}_k/\alpha\mathbf{V} = \mathbf{V}\mathbf{H}_k/\alpha$, the equivalence can as well be defined on the uniformized matrices. In the sequel we use the uniformized representation of a Markovian component denoted as $\mathcal{A} = (\alpha, \pi_0, \mathbf{A}_0, \mathbf{A}_1, \ldots, \mathbf{A}_K, \mathbf{U}_{K+1}, \ldots, \mathbf{U}_{K+L})$ where $\mathbf{A}_0 = \mathbf{G}_0/\alpha + \mathbf{I}$ and $\mathbf{A}_k = \mathbf{G}_k/\alpha$. In this representation all matrices of a Markovian component are non-negative and $\sum_{k=0}^{K} \mathbf{A}_k$ is a stochastic matrix.

[12] includes an algorithm to compute an ordinarily equivalent representation of a minimal size for any Markovian component. This algorithm has also been applied in [11] to compute minimal representations for some example models. However, this algorithm computes in general non-Markovian representations and the computation of minimal Markovian representations seems to be a hard problem which is unsolved yet. Additionally, experimental results show that the minimal representations are often only slightly smaller than the reduced representations resulting from exact or ordinary lumpability [11]. Thus, we consider in the following section approximate rather than exact reduction and restrict the models to Markovian models.

Examples: If in the second example $\mu_1 = \mu_2$, the model is lumpable and the state space is reduced from 10 to 6 states. In other cases, no exact reduction is possible. The first example allows also no exact aggregation.

3 Approximate Aggregation of Markovian Components

Approximate aggregation is a well known approach to deal with the complexity of large Markov models [31]. However, in many cases, the approximation is only based on heuristics such that the size of the approximation error remains unclear. To define an approximate aggregation approach in a more formal way, we define some measure of approximation which is afterward minimized. Based on this measure, aggregation can be seen as a minimization problem, an algorithmic approach can be defined, and, finally, the degree of approximation in a compositional setting can be analyzed.

3.1 A Formal Definition of Approximate Aggregates

The goal of aggregation is to compute for some Markovian component of order m an aggregate which is a Markovian component of order n ($< m$) such that both components behave similarly in any environment. The computation will be performed on the uniformized components since they allow us to consider only non-negative elements in the matrices and from the matrices of the uniformized component the matrices of the original component can be easily derived. The aggregation should be based on ordinary equivalence as defined in Definition 3 which can be used for the original and uniformized components as shown above. To simplify notation, we do not distinguish between the matrices $\mathbf{A}_k$ and $\mathbf{U}_l$ and denote for the following results all matrices by $\mathbf{A}_k$ and if $k > K$ $\mathbf{A}_k = \mathbf{U}_k$. Similarly, we denote the matrices of the second component by $\mathbf{B}_k$. Thus, we start with some Markovian component $\mathcal{A} = (\pi_0, \mathbf{A}_0, \mathbf{A}_1, \ldots, \mathbf{A}_{K+L})$ of order m and compute a Markovian component $\mathcal{B} = (\phi_0, \mathbf{B}_0, \mathbf{B}_1, \ldots, \mathbf{B}_{K+L})$ of order n ($< m$) with a similar behavior. This results in the following minimization problem

$$\min_{(\mathbf{V}, \mathbf{B}_0, \ldots, \mathbf{B}_{K+L})} \left(\sum_{k=0}^{K+L} \|\mathbf{A}_k \mathbf{V} - \mathbf{V}\mathbf{B}_k\|^2 \right) \tag{9}$$

with the constraints $\mathbf{V}\mathbb{1} = \mathbb{1}$, $\sum_{k=0}^{K} \mathbf{B}_k \mathbb{1} = \mathbb{1}$ and $\mathbf{B}_k \mathbb{1} = \mathbb{1}$ for $k = K+1, \ldots, K+L$. Since a Markovian component should be computed, we have the additional non-negativity constraints: $\phi_0 = \pi_0 \mathbf{V} \geq \mathbf{0}$, which always holds if we choose $\mathbf{V} \geq \mathbf{0}$, and $\mathbf{B}_k \geq \mathbf{0}$ for $k = 0, \ldots, K+L$. We denote by $\|.\|$ the entry-wise Frobenius norm and by

$$\Delta_{\mathcal{A},\mathcal{B},\mathbf{V}} = \sum_{k=0}^{K+L} \|\mathbf{A}_k \mathbf{V} - \mathbf{V}\mathbf{B}_k\|^2 \tag{10}$$

the approximation error which is a measure for the degree of approximation. The approximation error becomes 0 if $\mathcal{A}$ and $\mathcal{B}$ are ordinarily related and $\mathbf{V}$ is the corresponding transformation matrix.

Since the matrices $\mathbf{A}_k$ contain for $k = K+1, \ldots, K+L$ conditional probabilities whereas the matrices $\mathbf{A}_k$ for $k = 1, \ldots, K$ contain probabilities of generating an event, input events contribute more to the error than output events, in particular, if output events are generated with a small probability and most transitions are internal. This potential imbalance can be avoided if estimates for the arrival rates of input events can be estimated. Let λ_k ($k = K+1, \ldots, K+L$) be an estimate of the rate with which events k are generated by the environment of the component and let α be the uniformization rate that has been used to build the matrices $\mathbf{A}_k$ ($k = 0, \ldots, K$), then the input matrices $\mathbf{A}_k$ ($k = K+1, \ldots, K+L$) are substituted by $(\lambda_k/\alpha) \cdot \mathbf{A}_k$ for the optimization in (9). With the method proposed in the following paragraphs, matrices $\mathbf{B}_k$ will then be computed that have row sum λ_k/α. For the resulting aggregated Markovian component $\mathcal{B}$ matrices $\mathbf{B}_k$ are premultiplied by α/λ_k to yield stochastic matrices including conditional probabilities for input events.

Obviously, (9) cannot be computed as it is since $\mathbf{V}$ and $\mathbf{B}_k$ are all unknown and the optimization problem is non-linear and hard to solve. In the following we present a decomposition approach which allows us to compute a solution with an iterative approach solving several (non-negative) linear least squares problems in each iteration. Corresponding approaches are denoted as alternating least squares (ALS) and are applied in various areas [24, 25].

3.2 Optimization Problems for Aggregation

The central idea of ALS is to solve (9) by iteratively solving two linear least squares (LLS) problems where each is easier to solve than the entire problem. This approach generates a sequence of approximation models $\mathcal{B}$ and transformation matrices $\mathbf{V}$ with a non-increasing approximation error. In each step it is assumed that either $\mathcal{B}$ or $\mathbf{V}$ is known such that only one matrix in each equation is unknown and we obtain LLS problems that are solved iteratively. We first present the LLS problems that have to be solved, before we introduce solution algorithms in the next paragraph.

Assume now that $\mathcal{B}$ is known. If we consider a single term in (9) we then obtain

$$\min_{\mathbf{V}} \left(\|\mathbf{A}_k \mathbf{V} - \mathbf{V}\mathbf{B}_k\|^2 \right) \tag{11}$$

with the constraints $\mathbf{V}\mathbb{1} = \mathbb{1}$ and $\pi_0 \mathbf{V} \geq \mathbf{0}$ or $\mathbf{V} \geq \mathbf{0}$.

Equation (11) is well known and can be transformed in a standard form as a set of linear equations [18]. This transformation will be described now. Let $vec(V)$ be an operation that transforms the $m \times n$ matrix $\mathbf{V}$ in an $m \cdot n$ column vector by stacking the columns one below the other, i.e.,

$$\mathbf{x} = vec(\mathbf{V}) = (\mathbf{V}(0,0), \ldots \mathbf{V}(m-1,0), \mathbf{V}(0,1), \ldots, \mathbf{V}(m-1,n-1))^T$$

With this notation we have (see [18, eq. 4.2])

$$vec(\mathbf{A}_k \mathbf{V}) = (\mathbf{I}_n \otimes \mathbf{A}_k)\mathbf{x}$$

and

$$vec(\mathbf{V}\mathbf{B}_k) = (\mathbf{B}_k^T \otimes \mathbf{I}_m)\mathbf{x}.$$

Let $\mathbf{C}_k = \mathbf{I}_n \otimes \mathbf{A}_k - \mathbf{B}_k^T \otimes \mathbf{I}_m$ and $\mathbf{C} = (\mathbf{C}_0^T \ldots \mathbf{C}_{K+L}^T)^T$, then the resulting optimization problem equals

$$\min_{\mathbf{x}:\mathbf{F}\mathbf{x}\geq 0, \mathbf{E}\mathbf{x}=\mathbb{1}_n} \left(\|\mathbf{C}\mathbf{x}\|^2 \right) \tag{12}$$

where $\mathbb{1}_n$ is an n-dimensional column vector of 1, $\mathbf{F} = (\mathbf{I}_n \otimes \pi_0)$ and $\mathbf{E} = (\mathbb{1}_n^T \otimes \mathbf{I}_m)$. This optimization problem is denoted as problem LSEI (*linear least squares with equality and inequality constraints*) in [19, 26]. We use for this and the following LLS problems the same variable names for the matrices and vectors in the equations as used in the standard literature [19, 26].

Now consider the second LLS problem which results from the assumption that $\mathbf{V}$ but not $\mathcal{B}$ is known. Since the error results from the sum of errors according to each matrix $\mathbf{B}_k$ ($k = 0, \ldots, K+L$), the optimum can be computed for each matrix separately. For matrix $\mathbf{B}_k$ the following optimization problem has to be solved.

$$\min_{\mathbf{B}_k:\mathbf{B}_k\geq 0} \left(\|\mathbf{A}_k \mathbf{V} - \mathbf{V}\mathbf{B}_k\|^2 \right) \tag{13}$$

The solution of the optimization problem results in matrices $\mathbf{B}_k$ that minimize the squared difference between the original system and the aggregate according to a fixed aggregation defined by matrix $\mathbf{V}$. Additionally, we need to put constraints on the row sums of the matrices $\mathbf{B}_k$ to assure for $k = K+1, \ldots, K+L$ that $\mathbf{B}_k\mathbb{1} = \mathbb{1}$ (or $(\lambda_k/\alpha)\mathbb{1}$ if the matrices $\mathbf{A}_k$ have been scaled) and we need $\sum_{k=0}^{K} \mathbf{B}_k\mathbb{1} = \mathbb{1}$. Both conditions can be formulated as a set of equality constraints $\mathbf{B}_k\mathbb{1} = \mathbf{b}_k$ where $\mathbf{b}_k = \mathbb{1}$ (or $(\lambda_k/\alpha)\mathbb{1}$) for $k = K+1, \ldots, K+L$. The remaining vectors $\mathbf{b}_k$ $(k = 0, \ldots, K)$ will be computed as the solution of an LLS problem. Let $\mathbf{a}_k = \mathbf{A}_k\mathbb{1}$, $\mathbf{c} = (\mathbf{a}_0^T, \ldots, \mathbf{a}_K^T)^T$ and $\mathbf{x} = (\mathbf{b}_0^T, \ldots, \mathbf{b}_K^T)^T$. Then we solve the following LLS problem

$$\min_{\mathbf{x}:\mathbf{x}\geq\mathbf{0},\mathbf{E}\mathbf{x}=\mathbb{1}_n} \left(\|\mathbf{C}\mathbf{x} - \mathbf{c}\|^2\right) \tag{14}$$

where $\mathbf{C} = \mathbf{I}_{K+1} \otimes \mathbf{V}$ and $\mathbf{E} = \mathbb{1}_{K+1}^T \otimes \mathbf{I}_n$. (14) determines for each row of matrix $\mathbf{B}_k$ the row sum such that the sum of the matrices $\mathbf{B}_k$ $(0, \ldots, K)$ results in a stochastic matrix and the quadratic difference between the row sum of the matrix $\mathbf{A}_k$ and $\mathbf{B}_k$ projected via aggregation matrix $\mathbf{V}$ is minimal. The above problem is of the type NNLSE (*non-negative least squares problem with equality constraints*) [19, 26] and results in vectors $\mathbf{b}_k$ including the row-sums of the matrices $\mathbf{B}_k$.

We now use the function $vec(.)$ to bring the LLS problem (13) in standard vector matrix form. Thus,

$$\min_{\mathbf{x}:\mathbf{x}\geq 0,\mathbf{E}\mathbf{x}=\mathbf{f}} \left(\|\mathbf{C}\mathbf{x} - \mathbf{c}\|^2\right) \tag{15}$$

with $\mathbf{x} = vec(\mathbf{B}_k)$, $\mathbf{c} = vec(\mathbf{A}_k\mathbf{V})$, $\mathbf{C} = \mathbf{I}_n \otimes \mathbf{V}$, $\mathbf{E} = \mathbb{1}_n^T \otimes \mathbf{I}_n$ and $\mathbf{f} = \mathbf{b}_k$ is the standard representation. The corresponding problem is again of the type NNLSE.

3.3 An Algorithmic Approach for Aggregation

We now introduce the complete algorithm which consists of algorithms for solving the LLS problems, (12), (14) and (15), and the iterative combination of the solution steps.

For basic results on the solution of LLS problems we refer to the landmark textbook by Lawson and Hanson [26]. We give here only a brief sketch of possible solution techniques. Both LLS problems can be solved with built in solvers in numerical software packages like the solver *PDCO* for MATLAB [1] or the solver *qp* for Octave [2].

For the problem NNLS (i.e., $\min_{\mathbf{x}:\mathbf{x}\geq\mathbf{0}} \|\mathbf{C}\mathbf{x}\|^2)$) a solution algorithm is available from the netlib or as C-implementation [14]. This algorithm is the core procedure to solve (12) and (15) since both problems can be transformed into problem NNLS as shown in the online companion to this paper [9] (see also [19, 26]). With these optimization algorithms the iterative algorithm for the ALS solution looks as follows.

If the two LLS problems are solved exactly, algorithm *ALS_iteration* generates a sequence of components $\mathcal{B}$ and transformation matrices $\mathbf{V}$ with a non-increasing error (10). Since ALS is only a local optimization method such that one usually cannot expect to compute the global optimum for a non-convex problem as in our case. However, we found a very simple approach to improve the convergence of the algorithm that worked fine in many cases we checked so far. The idea is to put the above iteration in an outer iteration which modifies matrix $\mathbf{V}$ after local convergence has been reached. $\mathbf{V}$ is modified by running a hierarchical cluster algorithm to build n clusters according to the rows of $\mathbf{V}$. Afterwards each row of $\mathbf{V}$ is substituted by a row that contains a 1 in the position that belongs to the cluster number of the row and a 0 elsewhere. Similar rows

Algorithm 1 ALS_iteration($\mathcal{A}$, $\mathbf{V}$)

Require: Markov component $\mathcal{A} = (\pi_0, \mathbf{A}_0, \ldots, \mathbf{A}_{K+L})$ of order m and initial matrix $\mathbf{V}$;

return Markov component $\mathcal{B} = (\phi_0, \mathbf{B}_0, \ldots, \mathbf{B}_{K+L})$ of order n and final matrix $\mathbf{V}$;
generate an initial $\mathcal{B}$;
repeat
  $\mathbf{V}^{prev} = \mathbf{V}$;
  $\mathcal{B}^{prev} = \mathcal{B}$;
  compute new matrices for $\mathcal{B}$ by solving (14) and (15);
  compute a new matrix $\mathbf{V}$ by solving (12) ;
  $\epsilon = \max\left(\|\mathbf{V} - \mathbf{V}^{prev}\|, \max_{k=0,\ldots,K+L}\left(\|\mathbf{B}_k - \mathbf{B}_k^{prev}\|\right)\right)$;
until $\epsilon < \epsilon_{\max}$;
$\phi_0 = \pi_0 \mathbf{V}$;

in $\mathbf{V}$ describe similar states which may be aggregated into a single state. The iteration stops if a clustering (up to renumber of clusters) appears that has been computed before. Algorithm *Aggregate* usually performs only a small number (most times

Algorithm 2 Aggregate($\mathcal{A}$, n)

Require: Markov component $\mathcal{A} = (\pi_0, \mathbf{A}_0, \ldots, \mathbf{A}_{K+L})$ of order m and size of the aggregate $n < m$;

return Markov component $\mathcal{B} = (\phi_0, \mathbf{B}_0, \ldots, \mathbf{B}_{K+L})$ of order n and final matrix $\mathbf{V}$;
generate matrix $\mathbf{V}$ by assigning states randomly to aggregated states such that $rank(\mathbf{V}) = n$;
repeat
  $[\mathcal{B}, \mathbf{W}] =$ ALS_iteration($\mathcal{A}$, $\mathbf{V}$) ;
  Hierarchical clustering of the rows of $\mathbf{W}$ to generate matrix $\mathbf{V}$ with $\mathbf{V}(i,j) = 1$ if row i belongs to cluster j and 0 otherwise ;
until the same clusters have been computed before ;

between 2 and 4) outer iterations to find the aggregate. If we applied the algorithm to examples which are lumpable such that an exact aggregate with n states exists, then this is usually found with the outer iterations but not without outer iterations.

Examples: We run the algorithm for the first example (with the parameter values $\omega = 0.02$, $\nu = 0.01$, $\lambda = 0.9$ and $\mu = 1.0$) for different values of n and perform 10 ALS iterations between two clustering steps. The error function $\Delta_{\mathcal{A},\mathcal{B},\mathbf{V}}$ after $1, \ldots, 20$ ALS iterations is shown in Figure 3. Observe that the error function is printed on a logarithmic scale. It can be seen that the clustering is only effective for $n = 5$, in the remaining cases clustering increases the error temporarily, however, after one or two iterations the error before clustering has been reached by the ALS approach. In all cases, except for $n = 4$, the second clustering step generates the same matrix than the first one and the iteration stops. With some additional ALS iterations the error can be further reduced but the possible reduction is small. The best aggregate has 5 states, the error is about two orders of magnitude smaller than with 4 or less states. The following matrix $\mathbf{V}$ where entries have been rounded

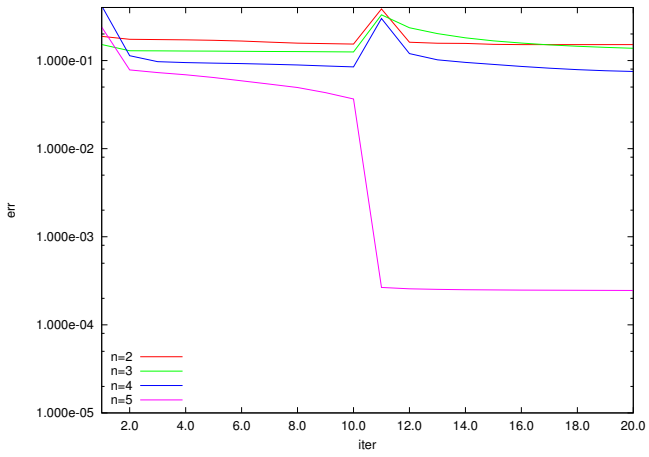

Figure 3: Error $\Delta_{\mathcal{A},\mathcal{B},\mathbf{V}}$ for Example 1.

to 5 digits results form the algorithm with $n = 5$.

$$\begin{pmatrix} 0.99463 & 0.00000 & 0.00219 & 0.00144 & 0.00173 \\ 0.01329 & 0.98479 & 0.00181 & 0.00006 & 0.00005 \\ 0.99221 & 0.00578 & 0.00000 & 0.00034 & 0.00167 \\ 0.01320 & 0.00152 & 0.98355 & 0.00173 & 0.00000 \\ 0.99033 & 0.00272 & 0.00616 & 0.00000 & 0.00079 \\ 0.01309 & 0.00082 & 0.00146 & 0.98305 & 0.00158 \\ 0.98921 & 0.00291 & 0.00347 & 0.00441 & 0.00000 \\ 0.01259 & 0.00037 & 0.00092 & 0.00143 & 0.98468 \end{pmatrix}$$

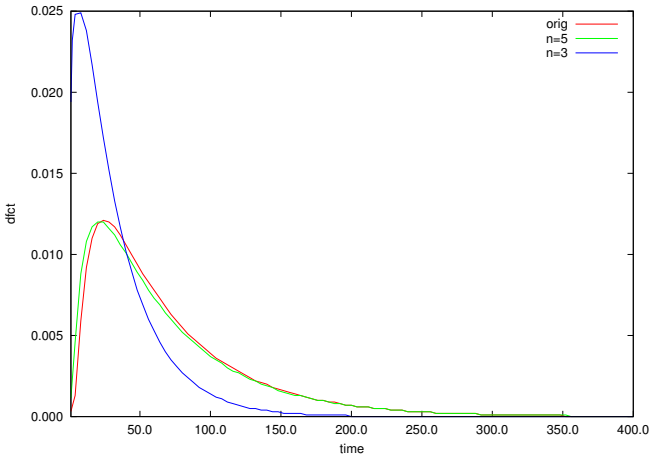

Figure 4: Density of the time until the first customer leaves the system without external arrivals.

It can be seen that the matrix is similar to a matrix where the states $\{1, 3, 5, 7\}$ are aggregated to one state. However, the

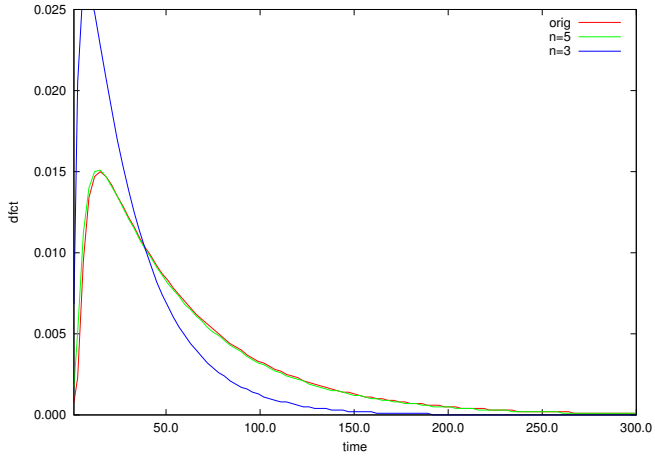

Figure 5: Density of the time until the first customer leaves the system with external Poisson arrivals.

error with the matrix resulting from clustering and an optimal $\mathcal{B}$ computed from (15) equals $3.289e-4$, whereas the error resulting from the above matrix $\mathbf{V}$ equals $2.433e-4$. Figures 4 and 5 show the density of the time until the first internally blocked customers leaves the system (i.e., a signal of type 1 is emitted) when the arrival rate is set to 0 (Fig. 4) and arrivals occur according to a Poisson process with rate 0.5 (Fig. 5). It can be noticed that in both cases, the aggregate with 5 states results in an excellent approximation of the original model with 8 states whereas the aggregate with only 3 states shows a poor approximation quality.

For the second example, the algorithm computes for identical rates μ_i and $n = 6$ the aggregate resulting from lumpability. For similar rates the lumping matrix is slightly modified to yield a smaller approximation error. With the matrix resulting from quasi lumpability [17] the error equals $5.64e-3$ whereas the more general $\mathbf{V}$ matrix results in an error of $6.27e-4$. Again the approximation is small as expected for this kind of model.

3.4 Compositional Properties of Approximate Aggregation

The approximation error depends on the value of $\Delta_{\mathcal{A},\mathcal{B},\mathbf{V}}$ as defined in (10). If the error becomes 0, aggregation is exact and the aggregate can be embedded in every environment yielding exact transient and stationary results that result from observing events [11, 13]. For non-zero approximation errors we develop now a bounding approach.

We begin with the lower bound and consider for each matrix pair $\mathbf{A}_k, \mathbf{B}_k$ a non-negative matrix $\mathbf{D}_k^-$ such that

$$\mathbf{A}_k\mathbf{V} \geq \mathbf{V}\left(\mathbf{B}_k - \mathbf{D}_k^-\right) \;\Rightarrow\; \mathbf{V}\mathbf{D}_k^- \geq \left(\mathbf{V}\mathbf{B}_k - \mathbf{A}_k\mathbf{V}\right), \tag{16}$$

then $\pi_t\mathbf{A}_k\mathbf{V} \geq \pi_t\mathbf{V}\left(\mathbf{B}_k - \mathbf{D}_k^-\right) = \phi_t\left(\mathbf{B}_k - \mathbf{D}_k^-\right)$ for any vector $\pi_t \geq \mathbf{0}$. Since this holds for all k, it is easy to show that matrices $\mathbf{B}_k$ can be substituted by $\mathbf{B}_k - \mathbf{D}_k^-$ in the uniformization approach to compute lower bounds for transition probabilities. It is also easy to show that the property is compositional since for non-negative matrices $\mathbf{A}_k \otimes \mathbf{B}_k \geq \mathbf{A}_k \otimes \left(\mathbf{B}_k - \mathbf{D}_k^-\right)$ and $\mathbf{A}_k \oplus \mathbf{B}_k \geq \mathbf{A}_k \oplus \left(\mathbf{B}_k - \mathbf{D}_k^-\right)$ hold.

Of course, $\sum_{k=0}^{K}\left(\mathbf{B}_k - \mathbf{D}_k^-\right)\mathbb{1} \leq \mathbb{1}$ and if we assure that $\mathbf{B}_k(i,j) \geq \mathbf{D}_k^-(i,j)$, then the new matrix is substochastic.

Consequently, bounds for the stationary vector can be computed using the well known approach by Courtois and Semal [15]. Before we consider a similar approach to compute the upper bound, the optimization problem for the lower bound is brought to normal form and we obtain

$$\min_{\mathbf{x}:\mathbf{x}\geq\mathbf{0},\mathbf{F}\mathbf{x}\geq\mathbf{g}} (\|\mathbf{x}\|) \tag{17}$$

where $\mathbf{x} = vec(\mathbf{D}_k^-)$, $\mathbf{F} = \mathbf{I}_n \otimes \mathbf{V}$ and $\mathbf{g} = vec\left(\mathbf{V}\mathbf{B}_k - \mathbf{A}_k\mathbf{V}\right)$. This problem is again an LLS problem with inequality constraints for which a non-negative solution has to be computed.

It could, of course, happen that the best lower bound matrix $\mathbf{B}_k$ contains several new zero elements which often implies that $\sum_{k=0}^{K} \mathbf{B}_k$ becomes reducible and the bounds become loose even if the approximation is good.

In a similar way an upper bound can be computed by solving

$$\mathbf{A}_k\mathbf{V} \leq \mathbf{V}\left(\mathbf{B}_k + \mathbf{D}_k^+\right) \;\Rightarrow\; \mathbf{V}\mathbf{D}_k^+ \geq \left(\mathbf{A}_k\mathbf{V} - \mathbf{V}\mathbf{B}_k\right), \tag{18}$$

such that $\pi_t\mathbf{A}_k\mathbf{V} \geq \pi_t\mathbf{V}\left(\mathbf{B}_k + \mathbf{D}_k^+\right) = \phi_t\left(\mathbf{B}_k + \mathbf{D}_k^+\right)$ for any vector $\pi_t \geq \mathbf{0}$. Again $\mathbf{B}_k + \mathbf{D}_k^+$ can be used in the uniformization approach to compute an upper bound. Like the lower bound, the upper bound is preserved by composition and the results form [15] can be applied to compute bounds for the stationary vector.

The optimization problem for the upper bound can also be transformed into the standard form, resulting in

$$\min_{\mathbf{x}:\mathbf{x}\geq\mathbf{0},\mathbf{F}\mathbf{x}\geq\mathbf{g}} (\|\mathbf{x}\|) \tag{19}$$

where $\mathbf{x} = vec(\mathbf{D}_k^+)$, $\mathbf{F} = \mathbf{V} \otimes \mathbf{I}_n$ and $\mathbf{g} = vec(\mathbf{A}_k\mathbf{V} - \mathbf{V}\mathbf{B}_k)$.

Examples: The bounds for the first example are loose since the matrix becomes reducible. For the second example results are much better. We combine the component with a single station with exponential service times and rate 0.2, the two parallel stations have rates 0.95 and 1.05, respectively. The combination of the single station and the component results in a closed queueing network with 3 customers inside which is, of course, a product form network. However, we are interested in the quality of the bounding approach. If we apply clustering for the matrix $\mathbf{V}$ we obtain a quasi lumpable partition as in [17] and the bounds for the mean population in the two parallel stations equal $[0.9707, 1.5537]$ if we use the lower bound matrix. The upper bound matrix is in this case not usable since the inverse is not a non-negative matrix (see [15] for details). With the general matrix $\mathbf{V}$ resulting from the optimization step to reduce the aggregation error we compute a lower bound matrix which results in loose bounds but for the upper bound matrix we obtain population bounds $[1.1637, 1.6262]$ which means that at least the lower population bound can be improved.

4 Aggregation of PH distributions and MAPs

We now present some additional examples from a specific application area, namely the aggregation of the state space of Phase Type (PH) distributions and Markovian Arrival processes (MAPs) as one possible application. We consider distributions that are fitted with an Expectation Maximization (EM) algorithm according to some traffic trace. For an efficient fitting the structure of the distribution is usually restricted to an acyclic or a hyper-Erlang form. We use the tool GFIT [33] to fit a Hyper-Erlang distribution $(\pi, \mathbf{G}_0)$ of different orders n according to observations from the benchmark traces LBL-TCP-3

[28] and BC-pAug89 [27]. Since Hyper-Erlang distributions have a very specific form, parameter fitting is more efficient than for general matrices and it is usually not possible to find an exact reduction of the state space according to the conditions given in [11, 12]. Here we try to find a representation with less states that results from approximate rather than exact aggregation. The Hyper-Erlang distribution can be transformed into a MAP $(\mathbf{G}_0, \mathbf{G}_1)$ with $\mathbf{G}_1 = (-\mathbf{G}_0\mathbb{1})\pi$. After uniformization we obtain the matrices $\mathbf{A}_0, \mathbf{A}_1$ and apply the algorithm from section 3.3 to obtain matrix $\mathbf{V}$ and the aggregate $\mathbf{B}_0, \mathbf{B}_1$ of order $m < n$. To assess the quality of the approximation we present plots with the error $\Delta_{\mathcal{A},\mathcal{B},\mathbf{V}}$ and compare the log-likelihood of the original and the aggregated components.

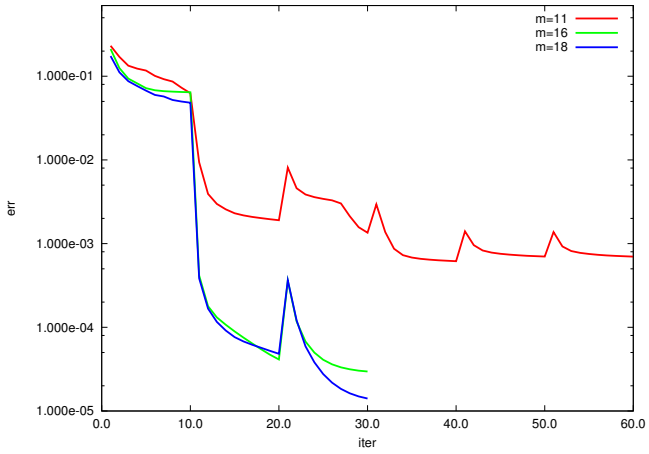

Figure 6: Error $\Delta_{\mathcal{A},\mathcal{B},\mathbf{V}}$ for a component of order 25 and trace LBL-TCP-3

Figs. 6 and 7 show the error $\Delta_{\mathcal{A},\mathcal{B},\mathbf{V}}$ of different aggregates that resulted from state space reduction of components with 25 and 30 states, respectively, which have been fitted to the trace LBL-TCP-3 as described above. As one can see from Fig. 6

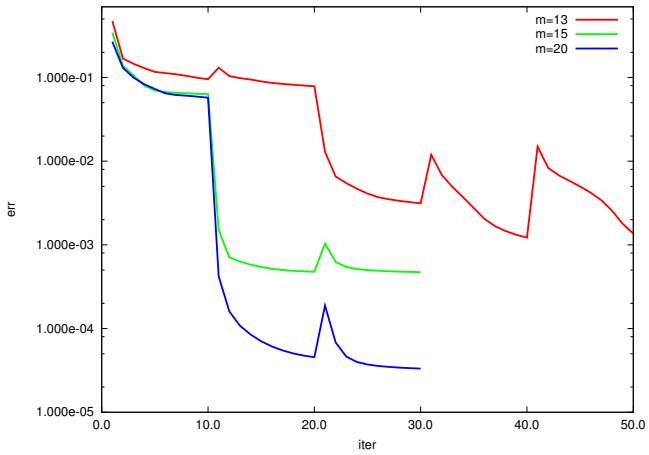

Figure 7: Error $\Delta_{\mathcal{A},\mathcal{B},\mathbf{V}}$ for a component of order 30 and trace LBL-TCP-3

for the component with $n = 25$ states the aggregates with $m = 16$ and $m = 18$ states resulted in a very small approximation error. Decreasing the number of states further increases the error but for the aggregate with $m = 11$ states still a good a

approximation was possible. Similar results have been obtained for a component with $n = 30$ states as shown in Fig. 7. From the figures one can also see that some of the clustering steps temporarily impair the approximation before the ALS iteration steps decrease the error again.

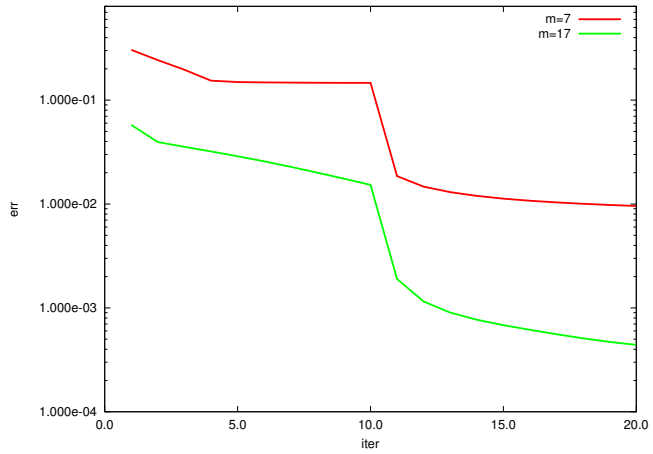

Figure 8: Error $\Delta_{\mathcal{A},\mathcal{B},\mathbf{V}}$ for a component of order 25 and trace BC-pAug89

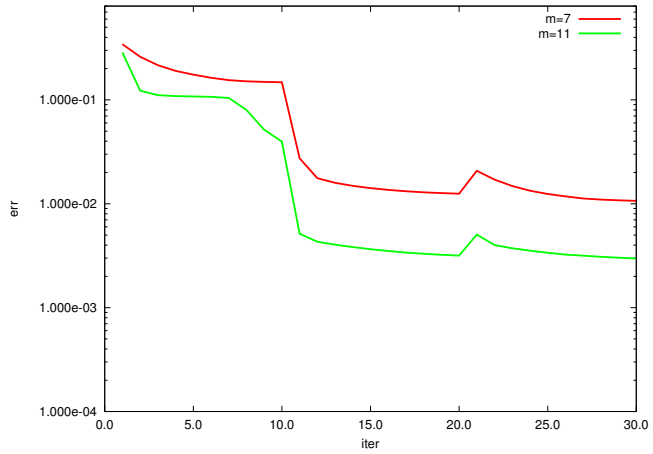

Figure 9: Error $\Delta_{\mathcal{A},\mathcal{B},\mathbf{V}}$ for a component of order 30 and trace BC-pAug89

Figs. 8 and 9 show results for the trace BC-pAug89 which confirm the previous observations for the trace LBL-TCP-3. Again, the the error $\Delta_{\mathcal{A},\mathcal{B},\mathbf{V}}$ for different aggregates of components with 25 and 30 states, respectively, is plotted.

Table 1 shows the log-likelihood values for the original components and the corresponding aggregates. As one can see, the components resulting from state space reduction still provide a large likelihood which is similar to the one of the original Hyper-Erlang distributions and usually better than likelihood we obtain if we fit a Hyper-Erlang distribution with the same number of states as the aggregate.

LBL-TCP-3		BC-pAug89	
$n = 25$	$-1.62451e+06$	$n = 25$	-805613
$m = 11$	$-1.70489e+06$	$m = 7$	-991104
$m = 16$	$-1.70661e+06$	$m = 17$	-860471
$m = 18$	$-1.69917e+06$		
$n = 30$	$-1.60687e+06$	$n = 30$	-800973
$m = 13$	$-1.72577e+06$	$m = 7$	-985632
$m = 15$	$-1.65931e+06$	$m = 11$	-977753
$m = 20$	$-1.60873e+06$		

Table 1: Log-likelihood values for the original and aggregated components

5 Conclusion

We present an approximate aggregation approach based on the minimization of an algebraically defined difference measure between Markov processes with labeled transitions. It is shown that the approach often allows one to compute aggregates with a smaller state space but an almost identical behavior. One possible application area of the proposed approach is the reduction of the state space of PH distributions or MAPs which have been fitted with some algorithm requiring a restricted form of the transition matrices. Our examples show that for typical examples like Hyper-Erlang distributions a sufficiently good approximation with less states is possible. This is important if the distributions are used in models that are solved with numerical methods and are faced with the problem of *state space explosion*.

In its current form the aggregation algorithms is based on alternating least squares approach which solves iteratively least squares problems with up to nm variables where n is the number of states in the original process and m is the number of states in the aggregate. With standard software this restricts the size of computable aggregates to values of nm which are in the range of at most a few thousand or better in the range of less than thousand. With more efficient software it is possible to increase this size. Additionally, it should be possible to adopt approximate reduction methods form linear systems theory [3] which can be applied to much larger problems. This, however, is a subject for future research.

References

[1] PDCO: Prima-dual method for optimization with convex objectives. http://www.stanford.edu/group/SOL/software/pdco.html.

[2] Quadratic programming in octave. http://www.gnu.org/software/octave/doc/interpreter/Quadratic-Programming.html.

[3] A. C. Antoulas and D. C. Sorensen. Approximation of large-scale dynamical systems: An overview. *Int. J. Appl. Comput. Sci.*, 11(5):1093–1121, 2001.

[4] S. Asmussen and M. Bladt. Point processes with finite-dimensional conditional probabilities. *Stochastic Processes and their Application*, 82:127–142, 1999.

[5] P. Buchholz. A class of hierarchical queueing networks and their analysis. *Queueing Syst.*, 15(1-4):59–80, 1994.

[6] P. Buchholz. Exact and ordinary lumpability in finite Markov chains. *Journal of Applied Probability*, 31:59–75, 1994.

[7] P. Buchholz. Equivalence relations for stochastic automata networks. In W. J. Stewart, editor, *Computations with Markov Chains*, pages 197–216. Kluwer Academic Publishers, 1995.

[8] P. Buchholz, G. Ciardo, S. Donatelli, and P. Kemper. Complexity of Kronecker operations and sparse matrices with applications to the solution of Markov models. *INFORMS Journal on Computing*, 12(3):203–222, 2000.

[9] P. Buchholz and J. Kriege. Online companion to the paper Aggregation of Markovian models - an alternating least squares approach -. Available online under http://ls4-www.cs.tu-dortmund.de/cms/de/home/buchholz/publikationen, July 2012.

[10] P. Buchholz and M. Telek. Stochastic Petri nets with matrix exponentially distributed firing times. *Performance Evaluation*, 67(12):1373–1385, 2010.

[11] P. Buchholz and M. Telek. Composition and equivalence of Markovian and non-Markovian models. In *QEST*, pages 213–222, 2011.

[12] P. Buchholz and M. Telek. On minimal representations of rational arrival processes. *Annals of Operations Research*, to appear, 2012. http://www.springerlink.com/content/vkx6163120563324/fulltext.pdf.

[13] P. Buchholz and M. Telek. Rational processes related to communicating Markov processes. *Journal of Applied Probability*, 49, 2012.

[14] J. Cantarella and M. Piatek. tsnnls: A solver for large sparse least squares problems with non-negative variables. http://www.michaelpiatek.com/papers/tsnnls.pdf.

[15] P. J. Courtois and P. Semal. Bounds for the positive eigenvectors of nonnegative matrices and for their approximations by decomposition. *J. ACM*, 31(4):804–825, 1984.

[16] S. Donatelli. Superposed stochastic automata: a class of stochastic Petri nets amenable to parallel solution. *Performance Evaluation*, 18:21–36, 1994.

[17] G. Franceschinis and R. R. Muntz. Bounds for quasi-lumpable markow chains. *Perform. Eval.*, 20(1-3):223–243, 1994.

[18] G. H. Golub, S. Nash, and C. Van Loan. A Hessenberg-Shur method for the problem AX + XB = C. *IEEE Trans. on Automatic Control*, 24(6):909–913, 1979.

[19] K. H. Haskell and R. J. Hanson. An algorithm for linear least squares problems with equality and nonnegativity constraints. *Mathematical Porgramming*, 21:98–118, 1981.

[20] H. Hermanns. *Interactive Markov Chains: The Quest for Quantified Quality*, volume 2428 of *Lecture Notes in Computer Science*. Springer, 2002.

[21] J. Hillston. A compositional approach for performance modelling. Phd thesis, University of Edinburgh, Dep. of Comp. Sc., 1994.

[22] J. Hillston. Compositional Markovian modelling using a process algebra. In W. J. Stewart, editor, *Computations with Markov Chains*, pages 177–196. Kluwer Academic Publishers, 1995.

[23] J. G. Kemeny and J. L. Snell. *Finite Markov Chains*. Springer, 1976.

[24] W. P. Krijnen. Convergence of the sequence of paramters generated by alternating least squares algorithms. *Computational Statistics and Data Analysis*, 51:481–489, 2006.

[25] P. M. Kroonenberg and J. de Leeuw. Principal component analkysis of three mode data by means of alternating least squares algoirthms. *Psychometrica*, 45(1):69–97, 1980.

[26] C. L. Lawson and R. J. Hanson. *Solving Least Squares Problems*. Classics in Applied Mathematics. SIAM, 1995.

[27] W. Leland, M. Taqqu, W. Willinger, and D. Wilson. On the Self-Similar Nature of Ethernet Traffic. *IEEE/ACM Transactions on Networking*, 2(1):1–15, 1994.

[28] V. Paxson and S. Floyd. Wide-Area Traffic: The Failure of Poisson Modeling. *IEEE/ACM Transactions on Networking*, 3:226–244, 1995.

[29] B. Plateau. On the stochastic structure of parallelism and synchronisation models for distributed algorithms. *ACM Performance Evaluation Review*, 13:142–154, 1985.

[30] B. Plateau and J. M. Fourneau. A methodology for solving Markov models of parallel systems. *J. Parallel Distrib. Comput.*, 12(4):370–387, 1991.

[31] W. J. Stewart. *Introduction to the numerical solution of Markov chains*. Princeton University Press, 1994.

[32] M. Telek and G. Horváth. A minimal representation of Markov arrival processes and a moments matching method. *Performance Evaluation*, 64(9-12):1153–1168, 2007.

[33] A. Thümmler, P. Buchholz, and M. Telek. A Novel Approach for Phase-Type Fitting with the EM Algorithm. *IEEE Transactions on Dependable and Secure Computing*, 3(3):245–258, 2006.

[34] S.-H. Wu, S. A. Smolka, and E. W. Stark. Composition and behaviors of probabilistic i/o automata. *Theor. Comput. Sci.*, 176(1-2):1–38, 1997.