AN APPROACH FOR BOUNDING REWARD MEASURES IN MARKOV MODELS USING AGGREGATION

DAVID DALY,* *University of Illinois*

PETER BUCHHOLZ,** *Universität Dortmund*

WILLIAM H. SANDERS,* *University of Illinois*

Abstract

An ongoing challenge in Markovian analysis is the so-called "state-space explosion," which is the exponential growth of the size of the state space of a model as the model size increases. We introduce a partial order on the states of a model to facilitate aggregation of states in order to reduce model size while bounding the error introduced by the aggregation. The partial order implies that the current and future behavior of the model is better in one state than another. We develop the theory of the partial order and its properties, show how it is related to monotonicity of matrices and to lumpability, and show how it can be efficiently applied to certain compositionally defined models.

Keywords: Aggregation; Markovian Processes; Bounding Technique; Transient Analysis; Partial Order

AMS 2000 Subject Classification: Primary 60J22

Secondary 68M15; 68M20

* Postal address: Department of Electrical and Computer Engineering and Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, 1308 W. Main St., Urbana, IL, U.S.A.
This work was supported in part by the National Science Foundation under Grant No. 0086096
* Email address: {ddaly, whs}@crhc.uiuc.edu

** Postal address: Informatik IV, Universität Dortmund, D-44221 Dortmund, Germany

** Email address: peter.buchholz@udo.edu

1. Introduction

An ongoing challenge in Markovian analysis is the so-called "state-space explosion," which is the exponential growth of the size of the state space of a model when the model's size grows. A lot of work has been done to decrease the size of the state space of a model (lumpability [9]) and to increase the size of the state space that can be solved (disk based methods [6], Kronecker [2], etc.).

Aggregation is one technique to address the state-space explosion problem; however, aggregation can introduce approximation error into the solution, which may or may not be bounded. Courtois and Semal bounds [3, 4] are the best-known bounded aggregation techniques for steady-state analysis, but are costly to compute and are tight only if the matrix is nearly completely decomposable (NCD) or nearly lumpable. The Courtois and Semal bounds can be applied to specific classes of models in order to develop more efficient bounding methods for those classes of models, such as Muntz et al. [10], Semal [13], and others have done.

However, the Semal and Courtois bounds are valid only for steady-state analysis, and different techniques must be used for transient analysis. The approach of stochastic orders and monotone matrices [8] is one such technique, and has been known for a long time. More recently, stochastic ordering has been extended by Forneau, Pekergin [7], and others to allow algorithmic generation of bounding aggregates. However, all of those techniques have very strict required conditions and require that states be ordered ahead of time and usually have loose bounds.

We have developed an aggregation technique that extends the stochastic ordering technique. We define a new partial order that is a more general relation between states than the total order required for monotone matrices. The partial order allows us to prove that the transient and steady-state solutions of an aggregated system yield bounds, and the partial order may also be used to construct bounding aggregates. Additionally, the partial order can be used with structured models to develop aggregates and to prove global properties, based upon a local analysis of the model's components. The partial order is also shown to be a generalization of monotonicity of the transition matrix and lumpability, in the sense that both properties can be proven and exploited using the partial order. The partial order was first introduced in [5] for a more limited

class of model and analysis. We extend that work by generalizing the model class and properties considered, as well as proving additional properties of the relation.

2. Background

We first define some basic notation, introduce a class of Markovian models for which the approach can be applied, and describe the basic analysis steps for this kind of models. We choose a class of asynchronously communicating Markov models [1] that are especially useful for the description of queuing networks [2] as the model class. In the examples, we specifically consider open-capacity restricted queuing networks with losses. However, the approach can applied to more general models.

2.1. Notation

Vectors and matrices are named using small or capital boldface letters like $\mathbf{p}$ or $\mathbf{P}$. We describe elements by setting their indices in brackets. For example, $\mathbf{P}(x, y)$ indicates the element in row x, column y of matrix $\mathbf{P}$. Furthermore, $\mathbf{P}(x\bullet)$ denotes row x of $\mathbf{P}$. If we have different matrices or vectors of the same type, then they are distinguished by sub- or superscripts. Thus, $\mathbf{P}_1, \ldots, \mathbf{P}_K$ denotes a set of matrices. $\mathbf{I}$ is the identity matrix of an appropriate dimension. If the dimension does not follow from the context, then it is added as a subscript, i.e., $\mathbf{I}_n$ is the identity matrix of order n. $\mathbf{e}$ is a row vector with all elements equal to 1, and $\mathbf{e}_x$ is a row vector with 1 in position x and 0 elsewhere. Again, the vector dimensions follow from the context. $\mathbf{0}$ is the zero matrix or vector. $\mathbf{p}^T$ and $\mathbf{P}^T$ are used for the transposed vector and matrix.

Except for the set of real numbers $\mathbb{R}$ and non-negative real number $\mathbb{R}_+$, sets are described by calligraphical letters (e.g., $\mathcal{S}$, $\mathcal{LS}$). Letters like i, j, k, l, x, y are used as running indices.

2.2. Stochastic Comparison

We briefly review two partial orders, one for random variables and the other for vectors. The two partial orders are "usual stochastic order" for random variables and "dominance" for probability vectors. We say a random variable X is less than a random variable Y in the *usual stochastic order* ($X \leq_{st} Y$) if $P(X > a) \leq P(Y > a), \forall\ a$ [11]. $X \leq_{st} Y$ means that the inverse cumulative distribution function for X will be less

than Y for all values, and thus the mean of X will be less than or equal to the mean of Y. The set of functions that preserve stochastic order is the set of all non-decreasing real functions. Additionally, if $a \geq_{st} c, b \geq_{st} d$, a and b are independent, and c and d are also independent, then $a + b \geq_{st} c + d$.

Given two probability vectors $\mathbf{p}_0$, $\mathbf{p}_1$ of length N, $\mathbf{p}_0$ *dominates* $\mathbf{p}_1$ ($\mathbf{p}_0 \succcurlyeq \mathbf{p}_1$ if $\sum_{m=n}^{N} \mathbf{p}_0(m) \geq \sum_{m=n}^{N} \mathbf{p}_1(m)$, $\forall n \leq N$ [8]. A random variable X_0 can be associated with the probability vector $\mathbf{p}_0$, by association of each possible value x of the random variable X_0 with an index i in the probability vector, where the entry $\mathbf{p}_0(i) = P(X_0 = x)$. If the value associated with each state in the probability vector is increasing in the state index, it is obvious that $\mathbf{p}_0 \succcurlyeq \mathbf{p}_1 \Leftrightarrow X_0 \geq_{st} X_1$.

2.3. Model Class

The models we consider in this paper are Markovian, i.e., the behavior depends only on the current state and the holding time in a state is exponentially distributed. However, in contrast to the usual Markov chains, we consider systems of interacting components that can be combined to form more complex systems. The composition of these components can be formally described using Kronecker sums and products of component matrices, e.g., [2], and will be briefly considered in section 6 in the context of bounding. Here we introduce the specification of a single component by means of vectors and matrices.

Let $\mathcal{S} = \{0, \ldots, n-1\}$ be the finite state space of the component. Most of the presented steps apply also to infinite state space, but in this paper the model class is restricted to the finite case. A component can perform two kinds of transitions: local transitions that are initiated by the component and are quantified by a transition rate, and reactions to input that are initiated by some environment and are quantified by probabilities.

We start with local transitions. If we assume that the component generates no output, then all transitions are collected in a matrix $\mathbf{Q}_0 \in \mathbb{R}_+^{n,n}$ such that $\mathbf{Q}_0(x, y)$ is the rate of an internal transition changing the state from x to y. If the component generates output, the outputs are described by matrices $\mathbf{Q}_1, \ldots, \mathbf{Q}_L$ where L is the number of disjoint outputs and $\mathbf{Q}_l(x, y)$ describes the rate of a transition from state x to y that simultaneously generated an output of type l. It is implicitly assumed that

outputs cannot be directly influenced by the environment (i.e., leaving entities cannot be blocked to remain in the component).

The dynamics initiated by the component are described by a continuous-time Markov chain (CTMC) with generator matrix

$$\mathbf{Q} = \sum_{l=0}^{L} \mathbf{Q}_l - diag\left(\sum_{l=0}^{L} \mathbf{Q}_l \mathbf{e}^T\right) \tag{1}$$

where $diag(\mathbf{a})$ for column vector $\mathbf{a}$ is a diagonal matrix with element $\mathbf{a}(i)$ in position (i, i). For a later analysis we define additionally

$$q = \max_{s \in \mathcal{S}} \left(\sum_{l=0}^{L} |\mathbf{Q}_l(s, s)|\right) \tag{2}$$

the matrix

$$\hat{\mathbf{Q}} = \mathbf{Q} - \sum_{l=1}^{L} \mathbf{Q}_l \tag{3}$$

and the matrices

$$\mathbf{P}_l = \begin{cases} \mathbf{Q}_l/q & \text{for } l = 1, \ldots, L \\ \\ \hat{\mathbf{Q}}/q + \mathbf{I} & \text{for } l = 0 \end{cases} \tag{4}$$

A component must react not only to outputs and internal transitions, but also to inputs. We assume that K different inputs numbered 1 through K may occur and that an input cannot be refused. Consequently, each state has to accept each input, and successor states are chosen probabilistically. Input probabilities are collected in matrices $\mathbf{U}_k \in \mathbb{R}_+^{n,n}$ $(k = 1, \ldots, K)$. Each $\mathbf{U}_k$ is a stochastic matrix, and $\mathbf{U}_k(x, y)$ includes the probability of moving from state x to y upon an arrival of type k.

To measure results of a system, rate-based reward values are used. Rewards are non-negative values assigned to states, and the interpretation is that the component gains a reward when staying in a state. R different rewards are defined, and each reward r is described by a column vector $\mathbf{r}_r \in \mathbb{R}_+^{n,1}$ such that $\mathbf{r}_r(x)$ is the reward value of state x and measure r.

The instantaneous reward gained at some point in time is a random variable defined by the state probability vector, and the reward for each state. This random variable

can be represented by another probability vector, in which each element in the vector is the probability of a specific value of the reward. The vector can be generated by a simple transformation of the state probability vector. We create the matrix $\mathbf{R}_r$ to be this transformation. Let M be the number of discrete reward values possible. We define a reward mapping function $rm_r(s)$, which maps a specific reward value to an index in the vector, with the restriction that $t > s \Rightarrow rm_r(t) > rm_r(s)$. We define the transformation $\mathbf{R}_r \in \mathbb{R}^{N,M}$

$$\mathbf{R}_r(x,y) = \begin{cases} 1 & \text{if } rm_r(x) = y \\ \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

Then $\mathbf{pR}_r$ is a probability vector describing the probability distribution of reward r corresponding to the state probability distribution $\mathbf{p}$. Further, rewards are arranged in increasing order, meaning that if there is a dominance between two such vectors, there will be a stochastic order between the rewards.

Finally, the initial distribution of the component must be defined. This is done in a row vector $\mathbf{p}_0 \in \mathbf{R}_+^{n,1}$ such that $\mathbf{p}_0$ describes a distribution and $\mathbf{p}_0(x)$ is the initial probability of state x.

2.4. Basic Behavior and Analysis

We consider two different kinds of analysis, the first kind is based on the observation of inputs and rewards without consideration of the outputs of a component. This viewpoint is sufficient for computation of results for the component in a random environment that is not affected by the outputs of the component. Examples include feed-forward networks for which the input of a queue is generated by upstream queues and the output only affects downstream queues. Additionally, by setting K to 0, CTMCs that do not have interactions with their environments are included. The second kind of analysis considers inputs, rewards, and outputs of the system. For both types of analysis we apply the so-called "uniformization approach" (see, e.g., [14]).

The first kind of analysis is based on the observation of the rewards at some time $t > 0$ or during the interval $[0, t)$. Both measures are conditioned on sequences of external inputs $(t_1, k_1), \ldots, (t_M, k_M)$ with $0 \leq t_1 \leq \ldots \leq t_M \leq t$ and $k_m \in \{1, \ldots, K\}$. For notational convenience let $t_0 = 0$, $t_{M+1} = t$, $\mathbf{t} = (t_0, \ldots, t_{M+1})$, and $\mathbf{k} = (k_1, \ldots, k_M)$

for $m = 1, \ldots, M$. Vectors $\mathbf{k}$ and $\mathbf{t}$ are valid, if $\mathbf{t}(m) \geq \mathbf{t}(m-1)$ for $m = 1, \ldots, M+1$ and $\mathbf{k}(m) \in \{1, \ldots, K\}$. The notation $\forall \mathbf{t}, \mathbf{k}$ is used for the (infinite) set of all possible valid vectors.

Define $R^r_{\mathbf{t},\mathbf{k}}$ as the instantaneous reward of type r gained by the component at time $\mathbf{t}(M+1)$, if the the arrival sequence defined by $\mathbf{t}$ and $\mathbf{k}$ is observed. The value M is implicitly defined by the vectors and might be zero, which would mean that no interaction with the environment occurred and that the reward should be considered at time t_1. The distribution of $R^r_{\mathbf{t},\mathbf{k}}$ ($\mathbf{f}^r_{\mathbf{t},\mathbf{k}}$) can be computed as a probability vector as

$$\mathbf{f}^r_{\mathbf{t},\mathbf{k}} = \mathbf{p}_0 \left(\prod_{m=1}^{M} \mathbf{Q}[\mathbf{t}(m) - \mathbf{t}(m-1)] \mathbf{U}_{\mathbf{k}(m)} \right) \mathbf{Q}[\mathbf{t}(M+1) - \mathbf{t}(M)] \mathbf{R}_r \tag{6}$$

where matrices $\mathbf{Q}[t]$ are computed using uniformization

$$\mathbf{Q}[t] = \sum_{i=0}^{\infty} po(i, qt) \mathbf{P}^k \tag{7}$$

where $q \geq \max_{x \in \mathcal{S}} |\mathbf{Q}(x,x)|$, $\mathbf{P} = \mathbf{Q}/q + \mathbf{I}$, and $po(i, qt)$ is the probability that a Poisson process with rate q performs i jumps in the interval $[0, t)$ that is given by

$$po(i, qt) = e^{-qt} \frac{(qt)^i}{i!} \tag{8}$$

and $E[R^r_{\mathbf{t},\mathbf{k}}]$ is the expectation of the reward value and can be computed as

$$E[R^r_{\mathbf{t},\mathbf{k}}] = \mathbf{p}_0 \left(\prod_{m=1}^{M} \mathbf{Q}[\mathbf{t}(m) - \mathbf{t}(m-1)] \mathbf{U}_{\mathbf{k}(m)} \right) \mathbf{Q}[\mathbf{t}(M+1) - \mathbf{t}(M)] \mathbf{r}_r \ . \tag{9}$$

The infinite summation can be truncated at a finite point for a given error bound using standard means [14].

Apart from the reward at some time t, we can consider the accumulated reward gained during the interval $[0, t)$. Define the corresponding measure as $AR^r_{\mathbf{t},\mathbf{k}}$. For the computation of the expectation of this measure we define vectors

$$\mathbf{p}^+_{\mathbf{t},\mathbf{k},m} = \mathbf{p}_0 \left(\prod_{i=1}^{m} \mathbf{Q}[\mathbf{t}(i) - \mathbf{t}(i-1)] \mathbf{U}_{\mathbf{k}(i)} \right) \tag{10}$$

that describe the state immediately after the m^{th} arrival. Then we obtain

$$E[AR^r_{\mathbf{t},\mathbf{k}}] = \sum_{m=0}^{M} \mathbf{p}^+_{\mathbf{t},\mathbf{k},m} \left(\int_{\mathbf{t}(m)}^{\mathbf{t}(m+1)} \mathbf{Q}[\tau] d\tau \right) \mathbf{r}_r \tag{11}$$

where the integral can be computed using uniformization [15] as

$$\int_{t'}^{t} \mathbf{Q}[\tau]d\tau = \int_{0}^{t-t'} \mathbf{Q}[\tau]d\tau = \frac{1}{q}\sum_{i=0}^{\infty} \mathbf{P}^i(1 - po(i, q(t-t')) \ . \tag{12}$$

The analysis conditioned on the outputs will be considered later in Section 6.1.1. However, the analysis will be based upon an extra property. Once that property has been checked, the analysis of the system with outputs will be the same as for the system without outputs.

3. Comparison of State

The previous section introduced the basic model class and its analysis according to the expectation of the reward at some time or during an interval. Here we consider the results conditioned on the initial state. The goal is to compare states in the following sense: State x is greater than state y if it yields a greater reward for all possible sets of external arrivals ($\mathbf{k}$) and arrival times ($\mathbf{t}$). If we can assure that starting in state x always yields larger rewards than starting in state y, then that fact can be used to define aggregates that bound reward values and have a smaller state space than the original component. The generation of aggregates is considered in the next section; here we explain the comparison of states intuitively, before introducing the idea of covering. We then develop an inductive definition of a partial order that relates states based upon the idea of covering. Following the definition, we prove several properties of the partial order, and compare the partial order with two established concepts for the comparison of states in CTMCs: monotonicity and strong stochastic ordering.

3.1. Intuitive Explanation

To compare states, we define $R^r_{\mathbf{t},\mathbf{k},\mathbf{p}}$ as the reward that is gained by computing the value $R^r_{\mathbf{t},\mathbf{k}}$ in (9) using $\mathbf{p}_0 = \mathbf{p}$. Intuitively, we say state x is greater than state y (denoted as $x \succeq y$) if and only if

$$\forall \mathbf{t}, \mathbf{k}, r : R^r_{\mathbf{t},\mathbf{k},\mathbf{e}_x} \geq_{st} R^r_{\mathbf{t},\mathbf{k},\mathbf{e}_y}. \tag{13}$$

Relation $\succeq$ defines a partial order among the states of $\mathcal{S}$. Observe that $x \succeq y$ implies also $AR^r_{\mathbf{t},\mathbf{k},x} \geq_{st} AR^r_{\mathbf{t},\mathbf{k},y}$ for all vectors $\mathbf{t}, \mathbf{k}$. This can be shown by the combination of

a simple inductive argument with the fact that the accumulated reward at some point and the reward at the next moment both depend on the current state but are otherwise independent, so that it is possible to add them together and preserve stochastic order.

Since (13) considers all possible sequences of allowed vectors $\mathbf{t}$ and $\mathbf{k}$, we cannot expect to find a general method to determine $\succeq$ exactly for arbitrary components. Instead, we will define a partial order $\succeq_\sim$ that is weaker than $\succeq$ (such that $x \succeq_\sim y \Rightarrow x \succeq y$), but that can be efficiently computed. Intuitively, $\succeq_\sim$ is defined as the fixed point of a refinement algorithm such that $x \succeq_\sim y$ implies 1) that the reward for x is at least the reward for y for all the measures and 2) that each transition out of y to some state z (internal or by some input) is *covered* by a set of transitions of the same type out of x, such that the set of transitions has the same probability as the transition from y to z, and each transition in the set goes to a state greater than z according to $\succeq_\sim$. This concept of covering will be introduced in the following subsection in some more detail. If the mentioned conditions can be assured, then it is rather obvious that x cannot yield a smaller reward than y. This will be proved more formally below.

3.2. Covering of Transitions

The basic step in the following inductive definition of a partial order among states is the so-called "covering" of states according to transitions. Let $\succeq_*$ be some partial order among the states of $\mathcal{S}$. We say that *state x covers state y* according to transitions in some matrix $\mathbf{D} \in \mathbb{R}_+^{n,n}$ and according to some partial order $\succeq_*$ if and only if

1. $x \succeq_* y$,
2. for each element $\mathbf{D}(y,z)$ there exists some vector $\mathbf{d}_{x,y}^z \in \mathbb{R}_+^{1,n}$ such that
   (a) $\mathbf{d}_{x,y}^z = \mathbf{0}$ for $\mathbf{D}(y,z) = 0$,
   (b) $\mathbf{d}_{x,y}^z * \mathbf{e}^T \geq \mathbf{D}(y,z)$,[1] and
   (c) $\forall u,\ \mathbf{d}_{x,y}^z(u) > 0 \Rightarrow u \succeq_* z$.
3. $\sum_{z=0}^{n-1} \mathbf{d}_{x,y}^z = \mathbf{D}(x\bullet)$.

[1] If matrix $\mathbf{D}$ has fixed row sums, as it does for stochastic matrices, then $\geq$ is substituted by $=$.

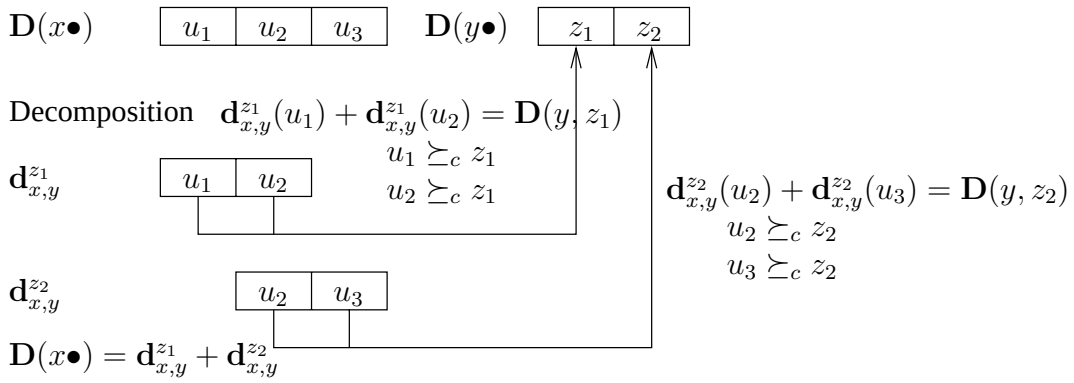

FIGURE 1: Example of a Covering. Row y is Covered by Row x.

A covering implies that for each transition originating in y and ending in z, a set of transitions with the same value originates in x, and each of these transitions ends in a state that is greater than z according to partial order $\succeq_*$.

A simple example of the idea of covering is shown in Fig. 1. In the example, state y (with transitions described by row $\mathbf{D}(y\bullet)$) is covered by state x (with transitions described by row $\mathbf{D}(x\bullet)$), as will be argued in the following. To see this covering, note that row $\mathbf{D}(y\bullet)$ contains two non-zero elements in the positions z_1 and z_2, and row $\mathbf{D}(x\bullet)$ contains three non-zero elements in the positions u_1, u_2, and u_3. The probability in row x is divided into two vectors to cover the two elements of row y. The first vector contains element u_1 and a fraction of element u_2, and the second vector contains the rest of u_2 and u_3. For a covering, the following conditions have to be met:

1. The sum of elements in the vector $\mathbf{d}^{z_1}_{x,y}$ has to be greater than or equal to $\mathbf{D}(y, z_1)$ and the sum of elements in the vector $\mathbf{d}^{z_2}_{x,y}$ has to be greater than or equal to element $\mathbf{D}(y, z_2)$ in row y. Equality has to hold in both cases if we assume that the matrix has equal row sum (as is the case in the figure).
2. The sum of vectors $\mathbf{d}^{z_1}_{x,y}$ and $\mathbf{d}^{z_2}_{x,y}$ has to be equal to row $\mathbf{D}(x\bullet)$.
3. Each non-zero element in the vector $\mathbf{d}^{z_1}_{x,y}$ has to be greater than z_1 (i.e., $u_1 \succeq_* z_1$ and $u_2 \succeq_* z_1$), and each non-zero element in the vector $\mathbf{d}^{z_2}_{x,y}$ has to be greater than z_2 (i.e., $u_2 \succeq_* z_2$ and $u_3 \succeq_* z_2$)

We say that a partial order $\succeq_*$ is *completely covered* for the matrices $\mathbf{D}_1, \ldots, \mathbf{D}_N$ if and only if condition 2 is observed for all pairs of states $x \succeq_* y$.

3.3. Inductive Definition

A useful partial order has to assure that $x \succeq y$ implies that x yields a greater reward than y according to (13). We now introduce an iterative approach to compute such a partial order as a fixed point of a refinement approach.

Define a family of partial orders $\succeq_c$ ($c = 0, 1, 2, \dots$) on $\mathcal{S}$ as follows:

1. $x \succeq_0 y$ if and only if $\mathbf{r}_r(x) \geq \mathbf{r}_r(y)$ for all $r = 1, \dots, R$,
2. for $c > 0$, $x \succeq_c y$ if and only if x covers y according to the matrices $\mathbf{P}, \mathbf{P}_1, \dots, \mathbf{P}_L$ and $\mathbf{U}_1, \dots, \mathbf{U}_K$, and the partial order $\succeq_{c+1}$.

Condition 1 of the definition of covering assures that partial order $\succeq_{c+1}$ contains no relations that are not contained by $\succeq_c$. We say that $\succeq_{c+1}$ is *finer* than $\succeq_c$, or that $\succeq_c$ is *coarser* than $\succeq_{c+1}$. The following theorem introduces two basic properties of the partial order.

Theorem 1. *For a sequence of partial orders $\succeq_c$ defined as above, the following properties hold:*

1. *If $\succeq_c = \succeq_{c+1}$, then $\succeq_c = \succeq_{c+i}$ for all $i > 0$.*
2. *All relations $\succeq_c$ are transitive and reflexive.*

Proof. To show 1 it is sufficient to consider step 2, in which relations that are present in $\succeq_c$ are removed. Assume that $\succeq_c = \succeq_{c+1}$. Consider states x, y such that $x \succeq_{c+1} y$. We show that $x \succeq_{c+2} y$. Condition 1 of the definition of covering is true by assumption. Also from assumption, we know that x covers y with respect to the transition matrices for the relations in $\succeq_c$, so it also covers y for the relations in $\succeq_{c+1}$, since the relations are the same. Therefore, condition 2 is also true, $x \succeq_{c+2} y$, and further, $\succeq_c = \succeq_{c+2} = \succeq_{c+i}$.

Transitivity of the relations is proved inductively. Obviously, $\succeq_0$ is transitive, since $x \succeq_0 y$ implies $\mathbf{r}_r(x) \geq \mathbf{r}_r(y)$ for all $r = 1, \dots, R$. Consequently, $x \succeq_0 y$ and $y \succeq_0 z$ implies $x \succeq_0 z$.

We show that if transitivity has been proved for $\succeq_c$, it also holds for $\succeq_{c+1}$. To show that $x \succeq_{c+1} y$ and $y \succeq_{c+1} z$ imply $x \succeq_{c+1} z$, we have to show that a covering of y by x and of z by y implies a covering of z by x. Assume that the covering is built according

to matrix $\mathbf{D}$, and consider element $\mathbf{D}(z,w) > 0$ that should be covered. Let $\mathbf{d}^w_{y,z}$ be the vector of the covering of z by y. For each value $\mathbf{d}^w_{y,z}(v) > 0$ there exists a vector $\mathbf{d}^v_{x,y}$. Now construct a vector

$$\mathbf{d}^w_{x,z} = \sum_{v:\mathbf{d}^w_{y,z}(v)>0} \frac{\mathbf{d}^w_{y,z}(v)}{\mathbf{D}(y,v)} \mathbf{d}^v_{x,y} \ .$$

It is easy to show that the resulting vectors describe the covering of z by x for matrix $\mathbf{D}$. Since the result holds for arbitrary matrices, it can be applied for all matrices of a component. The proof for reflexivity is trivial.

The first point in the above theorem shows that after finitely many steps a partial order is reached that remains constant in further iterations. This holds since the number of relations between states in $\succeq_0$ is finite, and each step from $\succeq_c$ to $\succeq_{c+1} \neq \succeq_c$ removes relations between states that can happen only finitely many times. $\succeq_c$ is denoted by $min_c : \succeq_c = \succeq_{c+1}$ as $\succeq_\sim$. Below we will consider some properties of that partial order. The second point in the previous theorem shows that $\succeq_\sim$ is a partial order.

Theorem 2. *If $\succeq_*$ is a partial order that is a refinement of $\succeq_0$ and that is completely covered for the matrices $\mathbf{P}, \mathbf{P}_1, \ldots, \mathbf{P}_L, \mathbf{U}_1, \ldots, \mathbf{U}_K$, then $x \succeq_* y \Rightarrow x \succeq_\sim y$.*

Proof. The proof is done inductively over the sequences $\succeq_c$ $c = 1, 2, \ldots$. First notice that for $x \succeq_* y$ the relation $x \succeq_0 y$ is necessary by assumption. Furthermore it is obvious that if no covering exists that refines a coarse relation, then also no covering exists that refines any refinement of the coarse relation.

Assume that $x \succeq_* y \Rightarrow x \succeq_c y$. We show that $x \succeq_* y \Rightarrow x \succeq_{c+1} y$. If x covers y according to the relations in $\succeq_*$ then it must also cover y according to the coarser order $\succeq_c$. Therefore, $x \succeq_* y \Rightarrow x \succeq_{c+1} y$.

Since the result holds for all c, $x \succeq_* y \Rightarrow x \succeq_\sim y$ holds and that proves the theorem.

The theorem shows that $\succeq_\sim$ is the coarsest order that can be found through consideration of coverings. However, $\succeq$ might be coarser than $\succeq_\sim$, since the concept of covering considers only sufficient conditions to prove $x \succeq y$.

3.4. Basic Properties

We now prove the basic properties of the partial order $\succeq_\sim$, namely that $x \succeq_\sim y$ implies $x \succeq y$.

Theorem 3. *If $x \succeq_\sim y$, then $R^r_{\mathbf{t},\mathbf{k},\mathbf{e}_x} \geq_{st} R^r_{\mathbf{t},\mathbf{k},\mathbf{e}_y}$ for all valid vectors $\mathbf{t},\mathbf{k}$.*

Proof. According to (6) and (7) the probability vector $\mathbf{f}^r_{\mathbf{t},\mathbf{k},\mathbf{e}_z}$ for $R^r_{\mathbf{t},\mathbf{k},\mathbf{e}_z}$ is computed as an infinite sum of the form

$$\mathbf{f}^r_{\mathbf{t},\mathbf{k},\mathbf{e}_z} = \sum_{i=0}^{\infty} \sum_{\mathbf{a}\in\{0,..,K\}^i} prob_{\mathbf{a}} \mathbf{e}_z \left(\prod_{j=1}^{i} \mathbf{D}_{\mathbf{a}(i)} \right) \mathbf{R}_r$$

where $prob_{\mathbf{a}}$ are appropriate probabilities, $\mathbf{D}_0 = \mathbf{P}$ and $\mathbf{D}_k = \mathbf{U}_k$ for $k > 0$. We prove that for an arbitrary sequence of matrices of length $i = 0, 1, 2, \ldots$ the vector dominance relation

$$\mathbf{e}_x \left(\prod_{j=1}^{i} \mathbf{D}_{\mathbf{a}(i)} \right) \mathbf{R}_r \succcurlyeq \mathbf{e}_y \left(\prod_{j=1}^{i} \mathbf{D}_{\mathbf{a}(i)} \right) \mathbf{R}_r$$

holds. that implies that the above summation is greater for $z = x$ than for $z = y$, and that there is a stochastic order between the corresponding random variables. The proof is done by induction.

For $i = 0$ we have $\mathbf{e}_x\mathbf{R}_r = \mathbf{e}_{rm_r(\mathbf{r}_r(x))} \succcurlyeq \mathbf{e}_{rm(\mathbf{r}_r(y))} = \mathbf{e}_y\mathbf{R}_r$, which holds since $x \succeq_0 y$, and the probability vector entries are arranged in order of increasing reward value.

Assuming that the result holds for a sequence of length i, we show that it then also holds for sequences of length $i+1$. Let $\mathbf{a}$ describe a sequence of length $i+1$ that can be decomposed into a sequence of length i defined by the subvector $\mathbf{a}'$ that consists of the elements 2 through $i+1$ of $\mathbf{a}$ and a sequence of length 1 with matrix $\mathbf{D}_{\mathbf{a}(1)}$. The value for y can be computed as

$$\mathbf{e}_y \prod_{j=1}^{i+1} \mathbf{D}_{\mathbf{a}(j)}\mathbf{R}_r = \mathbf{D}_{\mathbf{a}(1)}(y\bullet) \prod_{j=2}^{i+1} \mathbf{D}_{\mathbf{a}(j)}\mathbf{R}_r = \sum_{z=0}^{n-1} \mathbf{D}_{\mathbf{a}(1)}(y,z)\mathbf{e}_z \prod_{j=2}^{i+1} \mathbf{D}_{\mathbf{a}(j)}\mathbf{R}_r.$$

For x we obtain

$$\mathbf{e}_x \prod_{j=1}^{i+1} \mathbf{D}_{\mathbf{a}(j)}\mathbf{R}_r = \mathbf{D}_{\mathbf{a}(1)}(x\bullet) \prod_{j=2}^{i+1} \mathbf{D}_{\mathbf{a}(j)}\mathbf{R}_r = \sum_{z=0}^{n-1} \sum_{w=0}^{n-1} \mathbf{d}^z_{x,y}(w)\mathbf{e}_w \prod_{j=2}^{i+1} \mathbf{D}_{\mathbf{a}(j)}\mathbf{R}_r$$

where vectors $\mathbf{d}^z_{x,y}$ are the vectors describing the covering $\mathbf{D}_{\mathbf{a}(1)}(y,z)$. It follows that $\mathbf{d}^z_{x,y}(w) > 0$ implies $w \succeq_\sim z$ and $\sum_{w=0}^{n-1} \mathbf{d}^z_{x,y}(w) = \mathbf{D}_{\mathbf{a}(1)}(y,z)$[2]. Since the assumption holds for sequences of length i we have

$$\forall w:\ w \succeq_\sim z:\quad \mathbf{e}_w \prod_{j=2}^{i+1} \mathbf{D}_{\mathbf{a}(j)}\mathbf{R}_r \succcurlyeq \mathbf{e}_z \prod_{j=2}^{i+1} \mathbf{D}_{\mathbf{a}(j)}\mathbf{R}_r \quad \Rightarrow$$
$$\sum_{w:w\succeq_\sim z} \mathbf{d}^z_{x,y}(w)\mathbf{e}_w \prod_{j=2}^{i+1} \mathbf{D}_{\mathbf{a}(j)}\mathbf{R}_r \succcurlyeq \mathbf{D}_{\mathbf{a}(1)}(y,z)\mathbf{e}_z \prod_{j=2}^{i+1} \mathbf{D}_{\mathbf{a}(j)}\mathbf{R}_r,$$

which completes the proof.

4. Monotonicity and Lumping

The partial order $\succeq_\sim$ is related to st-monotonicity for matrices [8] and to lumping [9]. We show that a generator matrix is st-monotone if and only if all the states are totally ordered according to $\succeq_\sim$ by the state index, and two states x and y are lumpable if and only if $x \succeq_\sim y$ and $y \succeq_\sim x$. We prove both of these relations below.

Theorem 4. *$x \succeq_\sim y$ and $y \succeq_\sim x$ if and only if x and y are lumpable and have the same reward vector.*

Proof. If states x and y are lumpable with the same reward vector, they will trivially cover each other, and therefore $x \succeq_\sim y$ and $y \succeq_\sim x$.

We show that the converse is also true for $\succeq_\sim$. Assume that $x \succeq_\sim y$ and $y \succeq_\sim x$. The state x must cover the state y with respect to $\succeq_\sim$, and vice versa.

Let $R \subseteq \mathcal{S}$ be a subset of the state space such that for $v \in R$, $D(y,v) > 0$ or $D(x,v) > 0$. So long as R is non-empty and finite, there exists a (not necessarily unique) state $z \in R$ such that $\forall v \in R, z \succeq_\sim v$ or $v \not\succeq_\sim z$.

Define the set of states equivalent to z:

$$L(z) \subseteq S \text{ s.t. } v \in L(z) \implies v \succeq_\sim z \text{ and } z \succeq_\sim v \tag{14}$$

By the definition of covering we have:

$$d^z_{x,y}(u) > 0 \implies u \succeq_\sim z \tag{15}$$

But by assumption $z \succeq_\sim u$, therefore $u \in L(z)$.

[2] Observe that equality has to hold here since all involved matrices are stochastic.

$$\sum_{v \in L(z)} d^v_{x,y} \leq \sum_{v=0}^{n-1} d^v_{x,y} = D(x\bullet) \tag{16}$$

Therefore:

$$\sum_{v \in L(z)} D(y,v) \leq \sum_{v \in L(z)} D(x,v) \tag{17}$$

But the argument is symmetric, so:

$$\sum_{v \in L(z)} D(y,v) = \sum_{v \in L(z)} D(x,v) \tag{18}$$

Considering the remaining states in R, there exists a state $w \in R$ such that $\forall u \in R/L(z), \text{either} w \succeq_\sim u$, or $u \not\succeq_\sim w$. We conclude that $d^w_{x,y}(v) = 0,\ \forall v \in L(z)$, since we have already shown that states x and y have the same probability mass of going to a state in $L(z)$. Therefore, the above argument can be applied to $L(w)$, and inductively on all remaining states in R. The result is that for all states x and y such that $x \succeq_\sim y \succeq_\sim x$:

$$\sum_{v \in L(z)} D(x,v) = \sum_{v \in L(z)} D(y,v), \forall z \in S \tag{19}$$

Therefore, all states x, y such that $x \succeq_\sim y$ and $y \succeq_\sim x$ proceed with the same probability to states that are also equivalent according to $\succeq_\sim$. Therefore, we conclude that the states that are equivalent according to $\succeq_\sim$ form the sets of lumped states, which means that x and y are lumpable.

Theorem 5. *If* $\mathbf{D}$ *is monotonic and* $x > y$ *implies* $\mathbf{r}(x) \geq \mathbf{r}(y),\ \forall x, y$, *then* $x > y$ *implies* $x \succeq_\sim y, \forall x, y$.

Proof. Since $x > y$ implies $\mathbf{r}(x) \geq \mathbf{r}(y)$ we have $x \geq_0 y$ for $x > y$.

Assume $x > y$ implies $x \geq_k y, \forall x, y$. Consider two arbitrary states x, y such that $x > y$. By assumption $\mathbf{D}(x\bullet) \succcurlyeq \mathbf{D}(y\bullet)$. It can be shown the x covers y. The cover can be constructed by greedily selecting the largest subvectors for $d^z_{x,y}$ starting with the largest reachable state and proceeding through lesser states to the smallest states. Therefore, $x > y$ implies $x \geq_{k+1} y, \forall x, y$. Therefore, by induction, $x > y$ implies $x \succeq_\sim y, \forall x, y$.

Theorem 6. *If $x > y$ implies $x \succeq_\sim y$ and $y \not\succeq_\sim x$ then* $\mathbf{D}$ *is monotonic and $x > y$ implies $\mathbf{r}(x) \geq \mathbf{r}(y)$.*

Proof. $x > y$ implies $x \succeq_\sim y$, which directly implies that $x > y$ implies $\mathbf{r}(x) \geq \mathbf{r}(y)$. $x \succeq_\sim y$ implies that x covers y, so there exist vectors $d^z_{x,y}$ for all z. Since $d^z_{x,y}(u) > 0$ implies $u \succeq_\sim z$, and $u \succeq_\sim z$ only if $u > z$, $\sum_{z=0}^{n-1} \mathbf{d}^z_{x,y} \succcurlyeq \mathbf{D}(y\bullet)$. But $\sum_{z=0}^{n-1} \mathbf{d}^z_{x,y} = \mathbf{D}(x\bullet)$, implying that $\mathbf{D}(x\bullet) \succcurlyeq \mathbf{D}(y\bullet)$ for all $x > y$, which means that $\mathbf{D}$ is monotone.

If the state space is maximally lumped so that there are no states in the state space that are lumpably equivalent, then the requirement in Theorems 5 and 6 that $y \not\succeq_\sim x$ becomes trivially true. If that is the case, the Theorems 5 and 6 are the converses of each other.

5. Aggregation and Comparison

The concept introduced so far compares states from one state space $\mathcal{S}$. However, to build aggregates, states from different state spaces have to be compared. Here we first present the general concept of aggregation and then show how states belonging to different state spaces are compared.

5.1. Aggregation by State Mapping

The basic idea of aggregation is to combine several states of the original component and represent them with one state of an aggregate. Afterwards, the aggregate can be integrated instead of the component in an arbitrary environment. Different aggregation methods exist and in some cases yield exact results. However, aggregation of components usually introduces an approximation error of an unknown size. Although approximation errors are often relatively small, at least for stationary analysis, it is important to bound the error. Unfortunately, not much is known about bound computation in compositional analysis.

Let $\mathcal{S}_c = \{0, \ldots, n_c - 1\}$ be the state space of the entire component and $\mathcal{S}_a = \{0, \ldots, n_a - 1\}$ be the state space of the aggregate. We assume here $n_c \geq n_a$ for a useful aggregation, although there might be some interesting cases with $n_a > n_c$, e.g., if the aggregate is a product form queuing network and stationary results are required. The aggregation is described by a function $map : \mathcal{S}_c \to \mathcal{S}_a$. "$map^{-1}(x')$ for $x' \in \mathcal{S}_a$"

denotes the set $\{x|x' \in map(x)\}$. We assume that all sets $map^{-1}(x')$ are non-empty; if they are not, state x' can be removed from the state space of the aggregate. A mapping can be described by a matrix $\mathbf{M} \in \mathbb{R}_{+}^{n_o,n_a}$ such that $\mathbf{M}(x,x') = 1$ for $x' = map(x)$ and 0 otherwise.

5.2. Comparison of Different Models

The comparison of states from different state spaces according to a stochastic ordering was first presented in [12]. Here we use similar ideas, but we use the partial order among states developed in the previous section and compare CTMCs that can interact with their environments, whereas [12] compares DTMCs without interaction with the environment.

For the comparison of two models we use the superscript (o) for the matrices and vectors of the first (original) model and superscript (a) for the second (aggregated) model. Then we build the union of the two models, yielding matrices and vectors of the form

$$\mathbf{D} = \begin{pmatrix} \mathbf{D}^{(o)} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}^{(a)} \end{pmatrix} \text{ and } \mathbf{r}_r = \begin{pmatrix} \mathbf{r}_r^{(o)} \\ \mathbf{r}_r^{(a)} \end{pmatrix}.$$

For the resulting system, partial order $\succeq_\sim$ can be computed as before. We say that (a) is larger than (o) if and only if

$$\forall y \in \mathcal{S}^{(a)}, \forall x \in map^{-1}(y) : \; y \succeq_\sim x. \tag{20}$$

Similarly we define (a) as being smaller than (o) if and only if

$$\forall x \in \mathcal{S}^{(o)}, \forall y \in map(x) : \; x \succeq_\sim y. \tag{21}$$

The following theorem shows that the terms "smaller" and "larger" have a direct implication for the reward measures of the system.

Theorem 7. *If aggregate (a) is larger than system (o), then*

$$\forall \mathbf{t}, \mathbf{k}, r : R^{r,(a)}_{\mathbf{t},\mathbf{k},\mathbf{p}_0^{(a)}} \geq_{st} R^{r,(o)}_{\mathbf{t},\mathbf{k},\mathbf{p}_0^{(o)}}$$

and if aggregate (a) is smaller than system (o), then

$$\forall \mathbf{t}, \mathbf{k}, r : R^{r,(a)}_{\mathbf{t},\mathbf{k},\mathbf{p}_0^{(a)}} \leq_{st} R^{r,(o)}_{\mathbf{t},\mathbf{k},\mathbf{p}_0^{(o)}}$$

when

$$\mathbf{p}_o^{(a)}(x') = \sum_{x \in map^{-1}(x)} \mathbf{p}_0^{(o)}(x)$$

Proof. From Theorem 3 it is known that $x \succeq_\sim y$ implies $R^r_{\mathbf{t},\mathbf{k},\mathbf{e}_x} \geq_{st} R^r_{\mathbf{t},\mathbf{k},\mathbf{e}_y}$. If the aggregate is larger than the original system, then all states x' of the aggregate are larger than the states x of the original system that are mapped on x'. Consequently, $R^r_{\mathbf{t},\mathbf{k},\mathbf{e}_{x'}} \geq_{st} R^r_{\mathbf{t},\mathbf{k},\mathbf{e}_x}$ for $x' \in \mathcal{S}^{(a)}$ and $x \in map^{-1}(x')$. Furthermore, $\mathbf{p}_o^{(a)}(x') = \sum_{x \in map^{-1}(x)} \mathbf{p}_0^{(o)}(x)$ such that the following relation holds:

$$R^{r,(a)}_{\mathbf{t},\mathbf{k},\mathbf{p}_0^{(a)}} = \sum_{x' \in \mathcal{S}^{(a)}} \mathbf{p}_o^{(a)}(x') R^{r,(a)}_{\mathbf{t},\mathbf{k},\mathbf{e}_{x'}}$$

$$\geq_{st} \sum_{x' \in \mathcal{S}^{(a)}} \sum_{x \in map^{-1}(x')} \mathbf{p}_o^{(o)}(x) R^{r,(o)}_{\mathbf{t},\mathbf{k},\mathbf{e}_x} = R^{r,(a)}_{\mathbf{t},\mathbf{k},\mathbf{p}_0^{(o)}}.$$

The proof for smaller aggregates is completely analogous.

Since the above theorem holds for all valid vectors $\mathbf{t}$ and $\mathbf{k}$, the same result also holds for accumulated rewards $AR^{r,(a)}_{\mathbf{t},\mathbf{k},\mathbf{p}_0^{(a)}}$ and $AR^{r,(o)}_{\mathbf{t},\mathbf{k},\mathbf{p}_0^{(o)}}$. The theorem shows that under the same arrival process, a larger aggregate determines an upper bound and a smaller aggregate determines a lower bound of the reward measures of the original model.

For a complete system (i.e., a CTMC without inputs and outputs but with rewards), the only thing that needs to be checked is whether condition (21) or (20) holds with respect to matrix $\mathbf{P}$. If either of them does, it is assured that the smaller/larger aggregate bounds the reward of the complete system from below/above, due to theorem 7. It is obviously not necessary to compute the whole order $\succeq_\sim$; only (21) or (20) must to be verified. That nevertheless implies that we either know some order among the states to be aggregated or impose such an order for the states of the aggregate.

6. Analysis of Components in Composition

Having analyzed components and simple systems, we now investigate more complex systems that are compositions of systems. The composition of systems relies on the composition of Markov chains. The resulting structures are well-established, and it is known that the generator matrix of the composed model can be described as a sum of

Kronecker products of small component matrices, which is a nice structure with which to prove compositional results and realize algorithms. We first consider the acyclic composition of two components, and extend our approach to the cyclic composition of two components. Observe that the composition can easily be extended to multiple components. We assume that the reward structures for all of the components are combined by a function that is non-decreasing in all of the component rewards, such as addition, multiplication (if rewards are positive), min, and max.

6.1. A Few Common Components

Before we analyze the acyclic models, we briefly introduce and describe some common types of components that arise in compositional systems. In this section, we introduce some properties of the components, and in the following section we will analyze compositions of those types of components in more detail.

6.1.1. *Output model* An output process may have no rewards and no input events such that $R = K = 0$. However, the departure rates of the outputs provide an order, as one state cannot cover another state if it has a lower output rate. Therefore, $x \succeq_\sim y$ implies that the departure rate of each class l is greater if we start in x than if we start in y, and this holds for all times.

The class of output processes that can be described with the proposed concepts include MAPs and BMAPs and can additionally be used to specify correlated arrivals of different classes.

6.1.2. *Input model* For a system with inputs, but without outputs, $L = 0$ and the results from Section 5 apply. We have already shown that a system with inputs can be replaced with a smaller or larger aggregate to bound results for a given arrival process. To classify the behavior with respect to different arrival processes, the reaction of the system according arrivals has to be classified. We say a system is *input-increasing* if $\mathbf{U}_k(x, y) > 0$ implies $y \succeq_\sim x$. Thus, in an input-increasing system, each input improves the situation according to all rewards. Similarly, a system is *input-decreasing* if $\mathbf{U}_k(x, y) > 0$ implies $x \succeq_\sim y$. Consequently, the situation becomes worse with an arrival.

Input-increasing and -decreasing systems show some form of a monotonicity property

such that with an increasing/decreasing arrival rate, the reward grows/shrinks. In compositional analysis, such behavior is important when input processes or other components are replaced with larger or smaller aggregates and bounds for the component should be computed. Without such a monotonic behavior, it would be unclear whether the reward value increased or decreased under a larger or smaller aggregate for an input process.

6.1.3. *Input/Output model* A model with inputs and outputs is a simple extension of the input model and the output model. The analysis of the component will depend upon whether the component is input-increasing/decreasing with respect to the matrices $\mathbf{U}_1, \ldots, \mathbf{U}_k$.

6.2. Acyclic Composition of Two Models

We now consider a system composed from an output model (1), and an input model (2). Since the first component is an output model, and the second is an input model, the first component has no inputs (i.e., $K = 0$) and the second system has no outputs. Furthermore, we assume that the number of outputs of the first system equals the number of inputs of the second system, such that inputs and outputs with the same number can be connected. It is well-known that the generator matrix of the composed system can be represented as

$$\mathbf{Q} = \hat{\mathbf{Q}}^{(1)} \bigoplus \hat{\mathbf{Q}}^{(2)} + \sum_{k=1}^{K} \mathbf{Q}_k^{(1)} \bigotimes \mathbf{U}_k^{(2)} \tag{22}$$

and the reward and initial distribution vectors are given by

$$\mathbf{r}_r = \mathbf{r}_r^{(1)} \odot \mathbf{r}_r^{(2)} \text{ and } \mathbf{p}_0 = \mathbf{p}_0^{(1)} \bigotimes \mathbf{p}_0^{(2)} \tag{23}$$

where (i) denotes the component index, $n^{(i)}$ is the size of the state space for component i, $\mathbf{I}_n$ is the identity matrix of order n, and $\odot$ is any non-decreasing function. The following theorem gives some results about bounds on the complete model that exist when one component in the model is replaced with a larger or smaller aggregated version, based upon the preservation of $\succeq_\sim$ by matrix operations.

Theorem 8. *Consider the composition of two components* (1) *and* (2) *as described above, then the following relations hold.*

1. *If component* (2) *is input-increasing (decreasing and component* (1) *has no rewards) and* $(a+)$ *is a larger aggregate for* (1)*, then the composition of* $(a+)$ *with* (2) *yields a system that is larger (smaller) than the original system.*

2. *If component* (2) *is input-increasing (decreasing and component* (1) *has no rewards) and* $(a-)$ *is a smaller aggregate for* (1)*, then the composition of* $(a-)$ *with* (2) *yields a system that is smaller (larger) than the original system.*

3. *If* $(a+)$ *is a larger aggregate for* (2)*, then the composition of* $(a+)$ *with* (1) *yields a system that is larger than the original system.*

4. *If* $(a-)$ *is a smaller aggregate for* (2)*, then the composition of* $(a-)$ *with* (1) *yields a system that is smaller (larger) than the original system.*

Proof. Parts 3 and 4 follow directly from Theorem 7, since the changes have no effect on the first component and are guaranteed to increase the second component, and the reward is a non-decreasing function of the rewards of each component.

To prove part 1 of the theorem, we need to show that the complete model with larger aggregate for the first component has states that are greater than the states of the complete model with unaggregated components. First we show that if $x_1 \succeq_{\sim}^{(1)} x_2$ and $y_1 \succeq_{\sim}^{(2)} y_2$, then $(x_1, y_1) \succeq_{\sim} (x_2, y_2)$.

Consider arbitrary states $(x_1, y_1), (x_2, y_2)$ such that $x_1 \succeq_{\sim}^{(1)} x_2$ and $y_1 \succeq_{\sim}^{(2)} y_2$. $(x_1, y_1) \succeq_0 (x_2, y_2)$ since the rewards are combined in a non-decreasing fashion.

Assume that if $x_1 \succeq_{\sim}^{(1)} x_2$ and $y_1 \succeq_{\sim}^{(2)} y_2$, then $(x_1, y_1) \succeq_c (x_2, y_2)$. None of the local transitions for the second component have any effect on the first component, so all the transitions from y_1 will cover those from state y_2. Transitions in the first component can change the state of the second component, but may only change it to a greater state. All transitions from state x_1 will cover those from state x_2 with respect to the first component. However, since component 1 is an output model and $x_1 \succeq_{\sim}^{(1)} x_2$, x_1 will also be more likely to improve the state of the second component than x_2. Therefore, all of the transitions in the first component for state x_1 will cover those for x_2 in the complete model. Therefore, state (x_1, y_1) covers state (x_2, y_2), and $(x_1, y_1) \succeq_{c+1} (x_2, y_2)$. Therefore, $(x_1, y_1) \succeq_{\sim} (x_2, y_2)$ for $x_1 \succeq_{\sim}^{(1)} x_2$ and $y_1 \succeq_{\sim}^{(2)} y_2$.

Given that if $x_1 \succeq_{\sim}^{(1)} x_2$ and $y_1 \succeq_{\sim}^{(2)} y_2$, then $(x_1, y_1) \succeq_{\sim} (x_2, y_2)$, it becomes

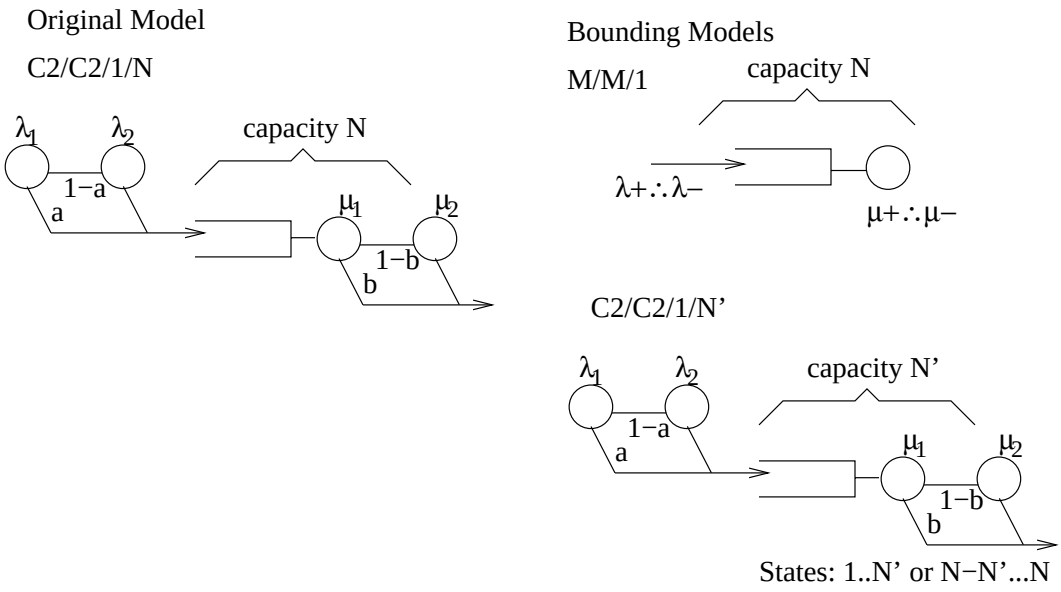

FIGURE 2: $C_2/C_2/1/N$ Model and Possible Aggregates.

trivial to prove part 1 of the theorem. State (x, y) in the unaggregated complete model will be mapped to state $(map(x), y)$ when the first component is aggregated. But $map(x) \succeq_{\sim}^{(1)} x$, and $y \succeq_{\sim}^{(2)} y$, so therefore $(map(x), y) \succeq_{\sim} (x, y)$ for all states x, y.

If component (2) is input-decreasing, the analysis is similar, except that more arrivals lead to lesser states in the second component. Therefore, a larger aggregate for component (1) will decrease the reward of the second component. If there are no rewards on the first model, then the composed system will also be smaller.

The second part of the theorem is just the reverse of the first part, and can be proved with a similar argument.

Of course, the different ways to build bounding aggregates can be combined. For example, the input process could be replaced with some smaller or larger aggregate according to 3 or 4, and afterwards a larger or smaller aggregate for component (2) could be used to build a second aggregated model that bounds the results of the first one.

To illustrate the possibilities, we consider a queue with capacity N and a 2-phase Coxian arrival and service process with parameters μ_1, μ_2, a and λ_1, λ_2, b, respectively. States of the model can be described by triples (x, y, l) where $l \geq 0$ describes

the population or level, and x and y describe the phase of the service and arrival distribution. We consider the population, the probability that the buffer is completely filled, and the throughput as the rewards. The rewards for the population and the probability that the buffer is completely filled depend only on the population in the queue, whereas the reward for the throughput depends on the phase of the service time distribution. We begin with aggregation of the arrival process and represent the aggregate with an exponential distribution with rate $\lambda^- = \min(a\lambda_1, \lambda_2)$ for a smaller aggregate and $\lambda^+ = \max(a\lambda_1, \lambda_2)$ for a larger aggregate. Similarly, we can define aggregates for the service process by using rates $\mu^- = \min(b\mu_1, \mu_2)$ for a larger aggregate and $\mu^+ = \max(a\mu_1, \mu_2)$ for a smaller aggregate. These rates define the matrix $\mathbf{Q}_1^{(a)}$ as the aggregate. Matrix $\mathbf{U}_1^{(a)} = (1)$ in both cases. Obviously, both aggregates are input-increasing. Observe that due to the specific structure of the aggregate (i.e., a birth-death process) and the specific reward structure (i.e., increasing with the state index) the relation $x \geq y \Leftrightarrow x \succeq_\sim y$ holds.

A smaller aggregate for the system is generated by combining the input process with rate λ^- and the service process with rate μ^+. To show that the resulting system is smaller, we have to prove that every state $x \in \mathcal{S}^{(a)}$ is smaller than all the states x represents. Such a proof is usually easier than the complete computation of $\succeq_\sim$ if an order among the states of the aggregate is known, as it is here. Since the example has a very specific structure and states are aggregated according to the first two components in their description, we can identify each aggregated state by one population. Thus, aggregated state l represents all detailed states (x, y, l) where x is the phase of the arrival process and, if $l > 0$, y is the phase of the service process. By the choice of the reward values it is assured that $(x, y, l) \succeq_0 (l)$. By induction we have to show that $(x, y, l) \succeq_c (l)$ holds for all $c \geq 0$. First consider matrix $\mathbf{U}$. In the detailed state space, an arrival brings the system with probability 1.0 to state $(x, y, l + 1)$ if $l < K$ and to state (x, y, l) if $l = K$. In the aggregated system an arrival causes a transition to state $l+1$ or to state l if $l = K$. Additionally, we know $(l) \succeq_\sim (l-1)$ in the aggregated state space. Together these imply that if $(x, y, l) \succeq_c (l)$, then $(x, y, l) \succeq_{c+1} (l)$ according to the matrices $\mathbf{U}$. Now consider matrix $\mathbf{P}$ and compare the values in the matrix of the aggregated system $\mathbf{P}^{(a)}$ and the original system $\mathbf{P}^{(o)}$. Observe that both matrices have to be generated according to the same value α that is chosen here as $\max(\mu_1, \mu_2)$.

For $l = 0$, the state remains the same with probability 1.0 in both the detailed and aggregated systems. In all other states, the aggregated system goes with probability $p = \mu^+/\alpha$ to states $l-1$ and remains with probability $1-p$ in state l. In the original system, a transition in matrix $\mathbf{P}$ that originated in state (x, y, l) $(l > 0)$ ends with some probability q in a state at level $l-1$ and remains with probability $1-q$ in a state at level l. Due to the construction of the rates for the aggregated system, $p \geq q$ holds in all cases. Consequently, we can repeat the argumentation used for matrix $\mathbf{U}$. If $(x, y, l) \succeq_c (l)$, then $(x, y, l) \succeq_{c+1} (l)$ after the refinement has been done with respect to matrix $\mathbf{P}$. Since the relation is not modified by any of the matrices and $(x, y, l) \succeq_0 (l)$ holds initially, it is clear that $(x, y, l) \succeq_\sim (l)$ also holds. The proof for the larger aggregate is very similar and is not given here.

There are several other options in building aggregates. A natural idea is to restrict the buffer size to some value $N^- < N$. For a smaller aggregate, all states (x, y, l) with $l > N^-$ are mapped onto state (x, y, N^-). The proof that this aggregate is smaller than the original system is similar to that for the previous case. It has to be shown that $(x, y, l) \succeq_\sim (x, y, l')$ where $(x, y, l) \in \mathcal{S}^{(o)}$, $(x, y, l') \in \mathcal{S}^{(a)}$ and $l = l'$ for $l \leq N$ and $l' = N^-$ otherwise. Each transition in the original system from (x, y, l) to $(x, y, l+1)$ is replaced with the same transition in the aggregate if $l < N^-$. For $l \geq N^-$, the transition is replaced with a transition remaining in state (x, y, l). A transition downwards from (x, y, l) to $(v, w, l-1)$ in the original system either remains the same in the aggregate (if $l \leq N^-$) or is replaced with a transition from (x, y, N^-) to $(v, w, N^- - 1)$. Together, with the appropriate definition of rewards, we have $(x, y, l) \succeq_0 (x, y, l')$ and can show inductively that this relation holds for all partial orders $\succeq_c$. A larger aggregate is generated by replacing all states (x, y, l) with $l \leq N - N^-$ onto $(x, l, 0)$, and replacing the remaining states with $l > N - N^-$ onto state $(x, y, l - N + N^-)$.

6.3. Cyclic Composition

A cyclic composition can be created with models that are input and output models. A chain of such models can be created by connecting inputs and outputs appropriately. The cyclic case is merely a simple extension of the acyclic case, and all models may be compared pairwise to determine the net effect. We analyze the three possible combinations of two components that are input-increasing or input-decreasing. Therefore,

consider two input and output models A and B, such that the output of A goes to B, and the output of B goes to A.

If A and B are both input-increasing, replacing A with a larger aggregate will lead to a larger global model, as A will experience greater states locally and it will have increased departures to B. The increase in arrivals to B will result in greater states in B, and the greater states in B will increase the departures back to A, further improving the state of A. The models will behave similarly if a smaller aggregate is used.

If one component (A) is input-increasing and the other (B) is input-decreasing, no bounding results can be ensured. Replacing A with a larger aggregate will lead to greater local states and more departures. However, more departures will lead to lesser local states in B, and therefore fewer departures from B. A will receive fewer arrivals, and thus have lesser local states. This negative feedback makes it impossible for us to guarantee any bounds at this level of analysis. Similar behavior will also be observed if A is replaced with a smaller aggregate, or if B is replaced with a larger or smaller aggregate.

If both models are input-decreasing, the behavior is slightly more complicated, and no global ordering results on the states can be assured. However, if both components are input-decreasing, and component A is replaced with a larger aggregate, the states of component A will be increased, leading to more arrivals to component B. The increase in arrivals to B will lead to lesser local states in B, and fewer departures to component A. However, the decrease in arrivals to A will further improve its local behavior. Therefore, component A will always be in greater states that it would have been, and component B will be in lesser states. This behavior therefore will lead to bounds for rewards defined on one component, but not for global rewards that are defined as a function of both components.

7. Conclusion

In this paper we introduced a new partial order for proving that the results of solving aggregated models bound the results of the original model. The partial order is applied to a general class of Markovian models that may interact asynchronously with their environments. In order to define the new partial order, we developed the concept of

"covering", which specifies that one state is greater than another state, and proceeds to greater states than does the other state. We then used the concept of covering to define the partial order, and proved that the partial order has the property that one state is greater than another state if the model will have a greater reward at any point in time, provided that the model starts in the first state instead of the second state. Additionally, we showed how the partial order was related to the stochastic ordering techniques of monotone matrices and to the concept of lumpability in Markovian models. Finally, we developed sufficient conditions for the existence of relations of the partial order in a structured model based upon analysis of the components of the model, which facilitates the substitution of aggregated components for the constituent components of the model.

Acknowledgement

The authors would like to thank Jenny Applequist for her help in editing this paper.

References

[1] Buchholz, P. (1992). Numerical solution methods based on structured descriptions of Markovian models. In *Computer Performance Evaluation - Modelling Techniques and Tools*. ed. G. Balbo and G. Serazzi. Elsevier. pp. 251–267.

[2] Buchholz, P. (1994). A class of hierarchical queueing networks and their analysis. *Queueing Systems* **15,** 59–80.

[3] Courtois, P. J. (1977). *Decomposability*. Academic Press, New York.

[4] Courtois, P.-J. and Semal, P. (1986). Computable bounds for conditional steady-state probabilities in large Markov chains and queueing models. *IEEE Journal on Selected Areas in Communications* **SAC-4,** 926–937.

[5] Daly, D., Buchholz, P. and Sanders, W. H. An approach for bounding dependability measures of Markov models. Submitted for publication.

[6] Deavours, D. D. and Sanders, W. H. (1998). An efficient disk-based tool for solving very large Markov models. *Performance Evaluation* **33,** 67–84.

[7] Fourneau, J. M. and Pekergin, N. (2002). An algorithmic approach to stochastic bounds. In *Performance 2002: Tutorial Lectures*. ed. M. Calzarossa and S. Tucci. vol. 2459 of *LNCS*. Springer. pp. 64–88.

[8] Keilson, J. and Kester, A. (1977). Monotone matrices and monotone Markov processes. *Stochastic Processes and Their Applications* **5,** 231–241.

[9] Kemeney, J. G. and Snell, J. L. (1960). *Finite Markov Chains*. D. Van Nostrand Inc.

[10] Lui, J. C. S. and Muntz, R. R. (1994). Computing bounds on steady state availability of repairable computer systems. *Journal of the ACM* **41,** 676–707.

[11] Muller, A. and Stoyan, D. (2002). *Comparison Methods for Stochastic Models and Risks*. Wiley Series in Probability and Mathematical Statistics. John Wiley & Sons.

[12] Pekergin, N. (1999). Stochastic performance bounds by state space reduction. *Performance Evaluation* **36-37,** 1–17.

[13] Semal, P. (1995). Refinable bounds for large Markov chains. *IEEE Transactions on Computers* **44,** 1216–1222.

[14] Stewart, W. J. (1994). *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press.

[15] Trivedi, K. S., Ciardo, G., Mahorta, M. and Sahner, R. A. (1993). Dependability and performability analysis. In *Performance evaluation of computer and communication systems: Joint tutorial papers of Performance '93 and Sigmetrics '93*. ed. L. Donatelli and R. Nelson. Springer, LNCS 729. pp. 587–612.