Loss Queueing Networks with Bursty Arrival Processes and Phase Type Service Times: Approximate Analysis

Johann Christoph Strelen
Rheinische Friedrich–Wilhelms–Universität Bonn

Abstract

Open queueing networks with bursty arrival processes, phase type service times, and losses are considered. A technique is proposed for the specification of the models. Using such a specification, a Markov chain can be generated automatically. Furthermore, this description can be used immediately for simulation. An approximate disaggregation–aggregation technique for finite state Markov chains is applied. Thus large models with many states can be solved, so large that the state probabilities could neither be stored nor calculated in any computer. Accurate approximations are obtained with different degrees of accuracy, depending on the investigated computational effort.

Introduction

Queueing networks are widely used in the stochastic modeling of computer, communication, and manufacturing systems. For the restricted class of separable queueing networks, efficient evaluation algorithms are known but if more general arrival processes and service times, or losses are considered, exact analysis is expensive or even impossible due to computational complexity. In order to obtain performance measures like throughput of jobs, utilization and availability of system components, and response times, such queueing networks are transformed into Markov chains which have typically a very large state space and a sparse transition probability matrix. The measures are calculated from the state probabilities of the Markov chain.

Unfortunately, the number of states grows quickly as the complexity of the model increases. This is why much research was done about the solution of large Markov chains [2, 7, 3]. But the state space is subject to a combinatorial explosion law. Therefore exact methods will never suffice for all models, approximation techniques are needed. We propose the *disaggregation-aggregation (DA) iteration* [8]. This is a general approximate technique for finite state Markov chains with discrete time which is based on aggregation. The calculated state probabilities are in product form; we call them *disaggregation-aggregation product-form (DAΠ)* solutions. Here, neither the complete state space, nor all state probabilities, nor the transition probability matrix are handled explicitly, marginal probabilities are considered instead. Therefore Markov chains with really large state spaces can be analyzed, for example billions of states.

Time and space demands are by orders of magnitude lower than for the exact solution of large Markov chains. The results of DA iteration are usually accurate, and the more aggregates are considered the more accuracy can be obtained, if the aggregation is chosen suitably.

It is interesting to note that the steady state probabilities of separable queueing networks are equal to DAΠ solutions calculated according to suitably chosen aggregates. Here our DA method is exact.

In [8] we propose also a description technique which we call *transition classes*. It is defined in the terms system components, component

Correspondence to: J. Ch. Strelen, Roemerstr. 164, 53117 Bonn, Germany, E-mail: strelen@cs.uni-bonn.de

June 25, 1997

states, state transitions, transition probabilities. This technique helps to structure the design of the Markov models and renders possible to generate the Markov chain automatically. With the transition classes a model can be immediately simulated. Furthermore this idea turns out to be usefull for the DA technique.

In [10] transition classes and the DA technique are applied to queueing networks with different service disciplines like priorities and polling, Coxian service and interarrival time distributions, blocking, or fork and join. Queueing networks with repetitive service blocking are discussed more thoroughly in [9].

Here we use transition classes and the DA technique for approximate solutions of loss queueing networks with phase type (PH) service time distributions and bursty arrival processes which can be modeled using hyperexponential interarrival times.

In this paper we consider steady state solutions. In section 1 we describe phase type distributions and the considered queueing networks, in section 2 we speak about the transformation of queueing network models into Markov chain models using transition classes, in section 3 DAΠ solutions are discussed, and some numerical examples are given in section 4.

1 Loss Queueing Networks with Bursty Arrival Processes and Phase Type Service Times

In this paper disributions of *phase typ* (PH-distributions) are used for interarrival times and service times. According to [6, page 231] a probability distribution $F(\cdot)$ on $[0,\infty)$ is of phase type if it is the distribution of the time until absorption in a finite-state Markov chain with a single absorbing state, that is, there exists a probability vektor $(\boldsymbol{\beta}, \beta_\kappa)$ and a generator matrix

$$\begin{bmatrix} \boldsymbol{\tau} & \boldsymbol{\tau}^0 \\ \mathbf{0}^\top & 0 \end{bmatrix}$$

such that

$$F(x) = 1 - \boldsymbol{\beta}^\top e^{\boldsymbol{\tau} x}\mathbf{1}, \quad x \geq 0,$$

where $\boldsymbol{\beta}^\top = (\beta_0, \ldots, \beta_{\kappa-1})$, $\mathbf{1} = (1, \ldots, 1)^\top$, $\mathbf{0} = (0, \ldots, 0)^\top$. The $\kappa \times \kappa$ matrix $\boldsymbol{\tau}$ is nonsingular, has negative diagonal elements and nonnegative off-diagonal elements. The vector $\boldsymbol{\tau}^0$ is nonnegative and satisfies $\boldsymbol{\tau}\mathbf{1} + \boldsymbol{\tau}^0 = \mathbf{0}^\top$.

The PH-distribution $F(\cdot)$ has a point mass β_κ at 0 and a density

$$f(x) = \boldsymbol{\beta}^\top e^{\boldsymbol{\tau} x}\boldsymbol{\tau}^0$$

on $(0,\infty)$.

We will use the triplet $(\kappa, \boldsymbol{\tau}, \boldsymbol{\beta})$ to denote the phase type distribution and the according renewal process which is defined as follows.

Upon absorption in the Markov process we instantaneously restart the process by selecting a new initial state independently of the past, using the probability vector $(\boldsymbol{\beta}, \beta_\kappa)$. This resetting is repeated indefinitely. It is obvious that the epochs of successive visits to the "instantaneous" state κ form a renewal process with the underlying life time distribution $F(\cdot)$.

If the PH-distribution is an interarrival time distribution this is exactly the process of interarrival times. If it is a service time distribution the service process is slightly different: If upon completion of a service the waiting room is not empty, that means if there is work to do, the Markov process is instantaneously restarted. If there is no job the restart is delayed until the next job arrives.

In the sequel we will use $\beta_\kappa = 0$. That means that the distribution is continuous, there is no point mass at 0.

Let us consider examples. A mixed Erlang distribution

$$F(x) = \sum_{k=1}^{\kappa} \alpha_k E_k(\lambda; x), \quad \kappa \geq 1,$$

is of phase type ($E_k(\lambda; x)$ denotes an Erlang distribution with parameters k and λ). Here $\boldsymbol{\beta}^\top = (\alpha_\kappa, \ldots, \alpha_1)$, $\beta_\kappa = 0$, and

$$\boldsymbol{\tau} = \begin{bmatrix} -\lambda & \lambda & & \\ & \ddots & & \\ & & -\lambda & \lambda \\ & & & -\lambda \end{bmatrix}.$$

The same but $\boldsymbol{\beta}^\top = (1, 0, \ldots, 0)$ holds for a pure Erlang-κ distribution. Clearly this is a exponential distribution if $\kappa = 1$ holds.

A hyperexponential distribution

$$F(x) = \sum_{k=0}^{\kappa-1} \beta_k (1 - e^{-\lambda_k x}), \quad \kappa \geq 2,$$

is of phase type and may be represented by $\boldsymbol{\beta}^\top = (\beta_0, \ldots, \beta_{\kappa-1})$, $\beta_\kappa = 0$, and $\boldsymbol{\tau} = -\operatorname{diag}(\lambda_0, \ldots, \lambda_{\kappa-1})$.

If $\kappa = 2$, $C^2 \geq 1$, and $\mathrm{E}[X] > 0$ is given, and

$$\begin{aligned} \beta_0 &= \frac{1}{2} - \frac{1}{2}\sqrt{\frac{C^2 - 1}{C^2 + 1}}, \quad \beta_1 = 1 - \beta_0 \\ \lambda_0 &= 2\beta_0/\mathrm{E}[X], \quad \lambda_1 = 2\beta_1/\mathrm{E}[X] \end{aligned}$$

hold, the hyperexponential distribution has the expectation $\mathrm{E}[X]$ and the squared coefficient of varation C^2.

Here we did not use the third degree of freedom: The third moment can also be taken into account [1, page 151].

Now we consider bursty arrivals using a hyperexponential distribution. To this end we predefine $\bar{B}$, the mean number of arrivals within a burst, λ, the overall rate of arrivals, and $V = t_\mathrm{P}/t_\mathrm{B}$, the ratio of the mean epoch of no arrivals, t_P, and the mean epoch with burst arrivals, t_B, restricted to $\bar{B} - 1 > 1/V$.

If the coefficients of a hyperexponential distribution with $\kappa = 2$ are according to

$$\begin{aligned} \beta_0 &= 1/\bar{B}, \quad \beta_1 = 1 - \beta_0, \\ \lambda_1 &= (1 + V)\beta_1 \lambda, \\ \lambda_0 &= \frac{\lambda_1}{V\beta_1/\beta_0 - 1}, \end{aligned}$$

and $\tau_{0,1} = \lambda_0$, the measures $\bar{B}$, λ, and V are satisfied by the distribution. The number of jobs in a burst has a geometric distribution.

Such a process may suitably model arrivals which originate from splitting up messages into packets or cells. Another application could be as follows. A lot of articles is to be treated on a machine, but the machine must be set before. Here the service time is modeled suitably by such a bursty type distribution.

We consider open queueing networks with phase type interarrival times and service times. $\mathcal{K}' \in I\!N^+$ denotes the set of nodes. Each node $k \in \mathcal{K}'$ consists of some system components.

Component k is the waiting room. It has the component state $z_k \in [0 : \nu_k]$ which counts the number of jobs in the node. The buffer capacity of the node is ν_k. If a job tries to enter the node and there is no buffer space available, the job is lost.

Component $k+1$ models the service process which is according to a phase type distribution. The state z_{k+1} of this component indicates the phase of this distribution.

The components $k-1$, $k-2$, ... are optional. They model arrival processes. z_{k-i} indicates the phase which the i-th arrival process is in. Obviously a node may have multiple arrival processes.

To each arrival process belongs a phase type distribution which models the interarrival times. At the end of an interarrival time an arrival occurs.

As usual, a job whose service is finished in node k, leaves the network (destination 0), or tries to go to node i. The selection of the destination i is done according to transition probabilities $t_{k,i}$, $k \in \mathcal{K}'$, $i \in \mathcal{K}' \cup \{0\}$. If the waiting room in the target node is full, the job is lost.

2 Markov Chains for Queueing Networks

Irreducible aperiodic Markov chains $(\mathbf{Z}(t),\ t \in I\!N)$, discrete and homogeneous in time, with finite state space $\mathcal{Z}^T$, $|\mathcal{Z}^T| = n \in I\!N$, are considered. They stem from stochastic models of computer, communication, or manufacturing systems. These systems consist of interacting subsystems, their *(system) components.* At every moment, each component is in one of its possible *component states.* This is why we denote the whole system states by *state tuples* $\mathbf{z} = (z_1, \ldots, z_K)$, where z_k indicates the state of component k. The states z_k of component k belong to the component state space $\mathcal{S}_k$ for all components $k = 1, \ldots, K$. Quite often we use $\mathcal{S}_k = [0 : N_k]$.

Sometimes virtual system components are also usefull, for example a "component" which indicates the number of jobs in an open queueing network.

Aggregates are sets of system states. An aggregate can be defined by the state of a certain system component, for example the aggregate $\mathcal{Z}_{k,j}$ is the set of all system states where the component k is in state j, irrespective of the states of the other components:

$$\mathcal{Z}_{k,j} = \{\mathbf{z} \in \mathcal{Z}^T \mid z_k = j\},\ j \in \mathcal{S}_k,\ k = 1, \ldots, K.$$

We call these aggregates *canonical*. For k fixed, all canonical aggregates $\mathcal{Z}_{k,0}, \ldots, \mathcal{Z}_{k,N_k}$ form a *canonical* partition $\mathcal{Z}_k$ of the state space. All the partitions $\mathcal{Z}_k$, $k = 1, \ldots, K$, constitute the *canonical aggregation*.

In general, a partition $\mathcal{Z}_c$ consists in aggregates $\mathcal{Z}_{c,j}$, $j \in \mathcal{N}_c$. A class $\{\mathcal{Z}_1, \ldots, \mathcal{Z}_C\}$, $C \in I\!N$, of partitions is an *aggregation*. For canonical aggregation, there are $C = K$ partitions, and $\mathcal{N}_c = [0 : N_c]$, $c = 1, \ldots, C$, holds.

A partition can be defined by two or more system components $k, k', \ldots, \hat{k}$, for example the partition with the aggregates

$$\mathcal{Z}_{(k,k'),(j,j')} = \{\mathbf{z} \in \mathcal{Z}^T \mid z_k = j,\ z_{k'} = j'\}, \\ j \in \mathcal{S}_k,\ j' \in \mathcal{S}_{k'},\ k \neq k',\ k, k' \in [1 : K], \quad (1)$$

combines the components k and k'. If more than one component define aggregates, the partition and the aggregation are termed *fine*. The sets of components which are used to define a partition $\mathcal{Z}_c$ are denoted $\mathcal{H}_c = \{k, k', \ldots, \hat{k}\}$, for example $\mathcal{H}_c = \{c\}$, $c = 1, \ldots, K$, for the canonical aggregation.

$p^{(t)}(\mathbf{z}) \stackrel{\triangle}{=} \mathrm{P}\{\mathbf{Z}(t) = \mathbf{z}\}$, $\mathbf{z} \in \mathcal{Z}^T$, denotes the state probabilities of the Markov chain at time t. These probabilities are collected in the n-vector $\mathbf{p}^{(t)}$. $p(\mathbf{x}, \mathbf{y})$, $\mathbf{x}, \mathbf{y} \in \mathcal{Z}^T$, are the state transition probabilities. They are the elements of the transition probability matrix $\mathbf{P}$.

Sometimes, if we are not interested in a particular time, or if steady state probabilities are considered, we omit the specification (t).

Each aggregate $\mathcal{Z}_{c,j}$ defines a *macro probability* or *aggregate probability* $\mathrm{P}\{\mathbf{Z} \in \mathcal{Z}_{c,j}\}$:

$$r_c(j) = \sum_{\mathbf{z} \in \mathcal{Z}_{c,j}} p(\mathbf{z}),\ j \in \mathcal{N}_c,\ c = 1, \ldots, C. \quad (2)$$

We collect the aggregate probabilities of a partition $\mathcal{Z}_c$ into tuples $\mathbf{r}_c$ and, in turn, these into $\mathbf{R} \stackrel{\triangle}{=} (\mathbf{r}_1, \ldots, \mathbf{r}_C)$.

By (2), the *aggregation function*

$$\mathcal{A} : [0,1]^n \to [0,1]^{|\mathcal{N}_1|} \times \ldots \times [0,1]^{|\mathcal{N}_C|},\ \mathbf{p} \mapsto \mathbf{R},$$

is defined. It calculates the macro probabilities for given state probabilities.

The states of the system components are described by the random variables $Z_k : \mathcal{Z}^T \to \mathcal{S}_k$, $\mathbf{z} \mapsto z_k$. The marginal distribution $\mathbf{r}_k$, calculated according to the canonical aggregation, is the probability distribution of Z_k. The state of the whole system is given by the random vector $\mathbf{Z} = (Z_1, \ldots, Z_K)$ with the probability distribution $\mathrm{P}\{\mathbf{Z} = \mathbf{z}\} = p(\mathbf{z})$, $\mathbf{z} \in \mathcal{Z}^T$.

Example. In a loss queueing network with Poisson arrivals and $\cdot/\mathrm{M}/1/\nu_k/\mathrm{FCFS}$ nodes, the system components are the nodes. Their states are the numbers z_k of jobs in them, $z_k \leq \nu_k$, $k = 1, \ldots, K$. Therefore, a system state is given by the K-tuple $\mathbf{z} = (z_1, \ldots, z_K)$. The state space $\mathcal{Z}^T$ equals $\mathcal{S}_1 \times \ldots \times \mathcal{S}_K$ where $\mathcal{S}_k = [0 : \nu_k]$ is the state space of node k. □

We assume that the Markov chain transitions from state $\mathbf{y}$ into state $\mathbf{z}$, $\mathbf{y} \neq \mathbf{z}$, $\mathbf{y}, \mathbf{z} \in \mathcal{Z}^T$, can be classified as follows: Each state transition belongs to one and only one of the *transition classes*, which are triplets $(\mathcal{U}_\nu, u_\nu, \alpha_\nu)$, $\nu = 1, \ldots, \bar{\nu}$. The introduction of these transition classes helps to structure the design of the Markov models, renders possible to generate the Markov chain automatically, and turns out to be very usefull for the DA technique.

A transition $\mathbf{y}$ into $\mathbf{z}$ of transition class ν may occur only if $\mathbf{y} \in \mathcal{U}_\nu$, where $\mathcal{U}_\nu \subset \mathcal{Z}^T$ denotes the *source states set* of this class. The *destination state function* $u_\nu : \mathcal{Z}^T \to \mathcal{Z}^T$ defines the new states, i.e. a transition of class ν leads from state $\mathbf{y} \in \mathcal{U}_\nu$ into state $u_\nu(\mathbf{y})$. The tuple u_ν consists of functions $u_{\nu,k}$, $u_\nu(\mathbf{y}) = \big(u_{\nu,1}(\mathbf{y}), \ldots, u_{\nu,K}(\mathbf{y})\big)$. Transitions of class ν occur with probability $\alpha_\nu(\mathbf{y})$. For DA iteration some restrictions on the transition classes will be required later on.

Example. We consider an open queueing network with losses. It consists of $K \in I\!N$ nodes. In node k, $k = 1, \ldots, K$, there is a single exponential server, service rate μ_k. Buffer space is available for no more than ν_k jobs. Jobs arrive according to a Poisson process with parameter λ_k but only if buffer space is available. When the service of a job in node k is finished the job tries to go to node l with probability $t_{k,l}$, $l \in [1:K]$. If there is no buffer space available, the job is lost. A job which is completely served in node k, leaves the network with probability $t_{k,0}$.

The system may change its state due to an arrival at node k from outside. The source state set of this transition class is $\mathcal{U}_\nu = \{\mathbf{z} \in \mathcal{Z}^T \mid z_k < \nu_k\}$. That means, there is space left in node k for an arriving job. Only the state of node k changes, the other remain unchanged. Hence the destination state function is given by $u_{\nu,k}(\mathbf{z}) = z_k + 1$, identity otherwise: $u_{\nu,i}(\mathbf{z}) = z_i$ if $i \neq k$. The transition probability is $\alpha_\nu = \lambda_k \Delta$.

Δ is the randomization constant: Initially, the Markov chain is continuous in time. Let $\mathbf{Q}$ denote its generator matrix. We consider the discrete time Markov chain with the transition probability matrix $\mathbf{P} = \mathbf{Q}\Delta + \mathbf{I}$ instead, where $\Delta = 0.99/\max_x |q_{x,x}|$. It has the same steady state probabilities as the continuous time Markov chain. By the factor 0.99, numerical problems are avoided, which may occur due to rounding errors.

Other state transitions take place when a job of node k leaves the network. The according transition class ν is defined as follows. The source states set is

$$\mathcal{U}_\nu = \{\mathbf{z} \in \mathcal{Z}^T \mid z_k > 0\}, \tag{3}$$

the destination state function $u_{\nu,k}(\mathbf{z}) = z_k - 1$, identity otherwise, and the transition probability is $\alpha_\nu = \mu_k t_{k,0} \Delta$.

The third kind of state transitions occur when a job leaves node k for node i, $i \neq k$. If there no buffer space is available, the job is lost. In node k, there must be at least one job. Therefore, the source states set is

$$\mathcal{U}_\nu = \{\mathbf{z} \in \mathcal{Z}^T \mid z_k > 0\}. \tag{4}$$

Only the states of the nodes k and i change, $u_{\nu,k}(\mathbf{z}) = z_k - 1$, $u_{\nu,i}(\mathbf{z}) = \min(\nu_i, z_i + 1)$, and the transition probability is $\alpha_\nu = \mu_k t_{k,i} \Delta$. $\square$

Applying transition classes, the state transition probabilities of a Markov chain can be expressed as follows:

$$\begin{aligned} p(\mathbf{y},\mathbf{z}) &= \sum_{\nu=1}^{\bar{\nu}} I(\mathbf{y} \in \mathcal{U}_\nu) I\Big(u_\nu(\mathbf{y}) = \mathbf{z}\Big) \alpha_\nu(\mathbf{y}), \\ & \qquad \mathbf{y} \neq \mathbf{z},\ \ \mathbf{y},\mathbf{z} \in \mathcal{Z}^T, \\ p(\mathbf{y},\mathbf{y}) &= 1 - \sum_{\nu=1}^{\bar{\nu}} I(\mathbf{y} \in \mathcal{U}_\nu) \alpha_\nu(\mathbf{y}),\ \mathbf{y} \in \mathcal{Z}^T \end{aligned} \tag{5}$$

where $I : \{\text{true}, \text{false}\} \to \{0,1\}$ denotes the indicator function.

It is not difficult to design an algorithm which enumerates the system states by application of the source states sets and the destination state functions. Using such an algorithm and the transition probabilities (5), the Markov chain can be generated automatically. We used such generated Markov chains for many examples.

Similarly a model which is described by means of transition classes can be immediately simulated.

Now we present the transition classes for the loss networks with phase type interarrival and service times which we described in section 1.

The service process $(\kappa, \boldsymbol{\tau}, \boldsymbol{\beta})$ of a node k may change from phase $e \in [0:\kappa-1]$, its component state, into the new phase $j \in [0:\kappa-1]$ if $\tau_{e,j} > 0$. According to the definition of phase type distributions the service is not finished by this transition. There is such a transition class ν for each service process, each old phase e, and each new phase j if $\tau_{e,j} > 0$.

These transitions occur with rate $\alpha_\nu = \tau_{e,j}$ but only if actually a job is served, i.e. if the system is in one of the states

$$\mathcal{U}_\nu = \{\mathbf{z} \in \mathcal{Z}^T \mid z_k > 0\}.$$

The new phase is

$$u_{\nu,k+1}(\mathbf{z}) = j,$$

the other system components remain unchanged,

$$u_{\nu,i}(\mathbf{z}) = z_i \text{ if } i \neq k.$$

A service in node k may be finished if the service process $(\kappa, \boldsymbol{\tau}, \boldsymbol{\beta})$ is in phase $e \in [0:\kappa-1]$ where $\tau_{e,\kappa} > 0$. The new phase may be $j \in [0:\kappa-1]$ if $\beta_j > 0$. The finished job may go to node $i \in \mathcal{K}'$ if $t_{k,i} > 0$. There is such a transition class ν for each pair (k,i) of nodes if $t_{k,i} > 0$, each old phase e if $\tau_{e,\kappa} > 0$, and each new phase $j \in [0:\kappa-1]$ if $\beta_j > 0$.

These transitions occur with rate

$$\alpha_\nu = \tau_{e,\kappa} \beta_j t_{k,i},$$

but only if there is actually a job in node k, if the service process is in phase e, and if buffer space is left in node i:

$$\mathcal{U}_\nu = \{\mathbf{z} \in \mathcal{Z}^T \mid z_k > 0, z_{k+1} = e, z_i < \nu_i\}.$$

The job leaves node k,

$$u_{\nu,k}(\mathbf{z}) = z_k - 1,$$

goes into node i,

$$u_{\nu,i}(\mathbf{z}) = z_i + 1,$$

and the new phase is j,

$$u_{\nu,k+1}(\mathbf{z}) = j.$$

Similar transition classes account for the case where in the destination node i no buffer space is available. The job is lossed, node i remains unchanged. Again there is such a transition class ν for each pair (k,i) of nodes if $t_{k,i} > 0$, each old phase e if $\tau_{e,\kappa} > 0$, and each new phase $j \in [0:\kappa-1]$ if $\beta_j > 0$.

These transitions occur with rate

$$\alpha_\nu = \tau_{e,\kappa}\beta_j t_{k,i},$$

but only if there is actually a job in node k, if the service process is in phase e, and if no buffer space is left in node i:

$$\mathcal{U}_\nu = \{\mathbf{z} \in \mathcal{Z}^T \mid z_k > 0, z_{k+1} = e, z_i = \nu_i\}.$$

The job leaves node k,

$$u_{\nu,k}(\mathbf{z}) = z_k - 1,$$

and the new phase is j,

$$u_{\nu,k+1}(\mathbf{z}) = j.$$

A service in node k may be finished if the service process $(\kappa, \boldsymbol{\tau}, \boldsymbol{\beta})$ is in phase $e \in [0:\kappa-1]$ where $\tau_{e,\kappa} > 0$. The new phase may be $j \in [0:\kappa-1]$ if $\beta_j > 0$. The finished job may leave the network if $t_{k,0} > 0$. There is such a transition class ν for each node k if $t_{k,0} > 0$, each old phase e if $\tau_{e,\kappa} > 0$, and each new phase $j \in [0:\kappa-1]$ if $\beta_j > 0$.

These transitions occur with rate

$$\alpha_\nu = \tau_{e,\kappa}\beta_j t_{k,0},$$

but only if there is actually a job in node k and if the service process is in phase e:

$$\mathcal{U}_\nu = \{\mathbf{z} \in \mathcal{Z}^T \mid z_k > 0, z_{k+1} = e\}.$$

The job leaves node k,

$$u_{\nu,k}(\mathbf{z}) = z_k - 1,$$

and the new phase is j,

$$u_{\nu,k+1}(\mathbf{z}) = j.$$

An arrival process $(\kappa, \boldsymbol{\tau}, \boldsymbol{\beta})$ may change from phase $e \in [0:\kappa-1]$, which is indicated by its component state, $z_k = e$, into the new phase $j \in [0:\kappa-1]$ if $\tau_{e,j} > 0$. According to the definition of phase type distributions no arrival occurs. There is such a transition class ν for each arrival process and each new phase j if there exists an arrival process state $z_k = e$ with $\tau_{e,j} > 0$.

These transitions occur with rate $\alpha_\nu = \tau_{z_k,j}$ but only if $\tau_{z_k,j} > 0$:

$$\mathcal{U}_\nu = \{\mathbf{z} \in \mathcal{Z}^T \mid \tau_{z_k,j} > 0\}.$$

The new phase is

$$u_{\nu,k}(\mathbf{z}) = j,$$

the other system components remain unchanged,

$$u_{\nu,i}(\mathbf{z}) = z_i \text{ if } i \neq k.$$

An arrival may take place at node k' due to one of its arrival processes $(\kappa, \boldsymbol{\tau}, \boldsymbol{\beta})$ where this arrival process changes its state z_k from phase $e \in [0:\kappa-1]$ into the new phase $j \in [0:\kappa-1]$ via the absorbing state κ if $\tau_{e,\kappa} > 0$ and $\beta_j > 0$.

There is such a transition class ν for each arrival process, each old phase e if $\tau_{e,\kappa} > 0$, and each new phase j if $\beta_j > 0$.

These transitions occur with rate $\alpha_\nu = \tau_{e,j}\beta_j$ but only if the arrival processes is in state e,

$$\mathcal{U}_\nu = \{\mathbf{z} \in \mathcal{Z}^T \mid z_k = e\}.$$

The arriving job enters waiting room k' if buffer space is left, or is lost otherwise,

$$u_{\nu,k}(\mathbf{z}) = \begin{cases} z_{k'} + 1 & \text{if } z_{k'} < \nu_{k'}, \\ \nu_{k'} & \text{otherwise.} \end{cases}$$

The new phase is

$$u_{\nu,k}(\mathbf{z}) = j,$$

the other system components remain unchanged,

$$u_{\nu,i}(\mathbf{z}) = z_i \text{ if } i \neq k.$$

3 Disaggregation–Aggregation Iteration

In this section we survey the DA method, details are given in [8].

Our proposed solution technique needs *disaggregation*: Given positive macro probabilities $\mathbf{R}$, a distribution $\boldsymbol{\pi}$ over $\mathcal{Z}^T$ is defined as follows where $\boldsymbol{\pi}$ is a n-vector with the probabilities $\pi(\mathbf{z})$, $\mathbf{z} \in \mathcal{Z}^T$. First, the distribution satisfies the macro probabilities, $\mathbf{R} = \mathcal{A}(\boldsymbol{\pi})$, and secondly, $\boldsymbol{\pi}$ maximizes the entropy. According to the *product-form theorem*, see [5, theorem 13.2.1, page 409], this maximum entropy distribution (MED) $\boldsymbol{\pi}$ is uniquely determined and has a product form,

$$\pi(\mathbf{z}) = \rho_{1,z_1} \cdot \ldots \cdot \rho_{C,z_C}/G, \tag{6}$$

where the $\rho_{c,z_c} > 0$ are some real numbers, and G is a normalizing constant. In fact, the MED is completely determined by all *MED parameters* $\rho_{c,j}$, $\boldsymbol{\rho}$ for short.

By (6) and (2), the *disaggregation function* is defined:

$$\mathcal{D} : [0,1]^{|\mathcal{N}_1|} \times \ldots \times [0,1]^{|\mathcal{N}_C|} \to [0,1]^n, \ \mathbf{R} \mapsto \boldsymbol{\pi}.$$

It calculates the MED over the state space for given aggregate probabilities.

Instead of calculating the steady state probabilities $\mathbf{q}$ of a Markov chain we determine an approximation $\mathbf{p} = \mathcal{D}(\mathbf{R})$ where $\mathbf{R} = \mathcal{T}(\mathbf{R})$ holds. Here the function $\mathcal{T}$ is defined as follows:

$$\begin{aligned} \mathcal{T} \quad : \quad & [0,1]^{|\mathcal{N}_1|} \times \ldots \times [0,1]^{|\mathcal{N}_C|} \to \\ & \quad [0,1]^{|\mathcal{N}_1|} \times \ldots \times [0,1]^{|\mathcal{N}_C|}, \\ \mathbf{R} \quad \mapsto \quad & \mathcal{A}\Big(\mathcal{D}(\mathbf{R})\mathbf{P}\Big). \end{aligned}$$

This approximation $\mathbf{p}$ and its marginal probabilities $\mathbf{R}$ are referred to as *steady state disaggregation-aggregation product-form (DAΠ) solution*. In fact, only the marginal probabilities $\mathbf{R}$ and the MED parameters $\boldsymbol{\rho}$ and G must be calculated. This is practicable even if the state space is large, so large that neither all state probabilities nor the transition probabilities could be handled.

We calculate the DAΠ solutions using *disaggregation-aggregation (DA) iteration*,

$$\mathbf{R}^{(t+1)} = \mathcal{T}(\mathbf{R}^{(t)}), \ t = 1, 2, \ldots \tag{7}$$

until $\max_{c,j} |r_c^{(t+1)}(j) - r_c^{(t)}(j)|$ is sufficiently small, beginning with feasible marginal probabilities $\mathbf{R}^{(1)}$.

This DA iteration resembles vector iteration, $\mathbf{q}^{(t+1)} = \mathbf{q}^{(t)}\mathbf{P}$, $t = 1, 2, \ldots$, beginning with any distribution $\mathbf{q}^{(1)}$. $\mathbf{q}^{(t)}$ are the time-dependent state probabilities and $\mathcal{A}(\mathbf{q}^{(t)})$ the time-dependent aggregate probabilities. $\mathbf{q}^{(t)}$ converges to the steady state probability vector and $\mathcal{A}(\mathbf{q}^{(t)})$ to the steady state aggregate probabilities. Accordingly, when DA iteration $\mathbf{R}^{(t+1)} = \mathcal{T}(\mathbf{R}^{(t)})$, $t = 1, 2, \ldots$, is performed, $\mathbf{R}^{(t)}$ are approximations of the time-dependent aggregate probabilities and $\mathcal{D}(\mathbf{R}^{(t)})$ approximations of the time-dependent state probabilities. If DA iteration is done for a sufficient number of iterations, $\mathbf{R}^{(t)}$ is an approximation of the steady state aggregate probabilities, and $\mathcal{D}(\mathbf{R}^{(t)})$ an approximation of the steady state probabilities.

The function $\mathcal{T}$ is evaluated with the *aggregate transition equations*

$$\begin{aligned} r_c^{(t+1)}(l) \quad = \quad & \sum_{j \in \mathcal{N}_c} \pi_c(j,l) r_c^{(t)}(j), \\ & l \in \mathcal{N}_c, \ c = 1, \ldots, C, \end{aligned} \tag{8}$$

using the *aggregate transition probabilities*

$$\begin{aligned} \pi_c(j,l) \quad \triangleq \quad & \mathrm{P}\{\mathbf{Z}_c^{(t+1)} \in \mathcal{Z}_{c,l} \mid \mathbf{Z}_c^{(t)} \in \mathcal{Z}_{c,j}\}, \\ & j, l \in \mathcal{N}_c, \ c = 1, \ldots, C. \end{aligned} \tag{9}$$

For canonical aggregation these are efficiently calculated with

$$\pi_k(j,l) = \frac{1}{r_k(j)} \sum_{\nu=1}^{\bar{\nu}} \begin{cases} \alpha_\nu \mathrm{P}\{\mathbf{Z} \in \mathcal{U}_\nu \cap \mathcal{Z}^T \cap \mathcal{X}\} \\ \quad \text{if } j \in \mathcal{U}_{\nu,k}, \ u_{\nu,k} = l, \\ 0 \text{ otherwise}, \end{cases} \tag{10}$$

where $\mathcal{X} = \mathcal{S}_1 \times \ldots \times \mathcal{S}_{k-1} \times \{j\} \times \mathcal{S}_{k+1} \times \ldots \times \mathcal{S}_K$, and similarly if a partition $\mathcal{Z}_c$ is defined by the states of more system components.

Here we must observe some restrictions on the transition classes. Each source states set must be a cartesian product, $\mathcal{U}_\nu = \mathcal{U}_{\nu,1} \times \ldots \times \mathcal{U}_{\nu,K}$ or the union of not too many such products. Under these circumstances the probabilities in (10) can be evaluated efficiently. Furthermore, broadly speaking, if there is a partition $\mathcal{Z}_c$ which is defined by the system components $h, h', \ldots, \hat{h} \in \mathcal{H}_c$, and if the transition class ν changes the state z_k of the component k where $k \in \mathcal{H}_c$, then the transition probability $\alpha(\mathbf{z})$ and the destination state function $u_\nu(\mathbf{z})$ must not depend on component states $z_i,\ i \notin \mathcal{H}_c$.

More precisely, let $\mathcal{M}_\nu \in [1 : K]$ denote the set of system components whose states are modified by transitions of class ν, $\mathcal{V}_{\nu,k} \subseteq [1 : K]$ the set of system components whose states are used for the calculation of the destination state function $u_{\nu,k}(\mathbf{z})$, except k, i.e. $k \notin \mathcal{V}_{\nu,k}$, $\mathcal{V}_\nu = \bigcup_{i \in \mathcal{M}_\nu} \mathcal{V}_{\nu,k}$, and $\mathcal{W}_\nu \subseteq [1 : K]$ the set of system components whose states are used for the calculation of the transition probability $\alpha_\nu(\mathbf{z})$.

The aggregate transition probabilities (10) can be calculated if the following *independence conditions on transition classes*

$$i \in \mathcal{H}_c \cap \mathcal{M}_\nu \Rightarrow \mathcal{V}_{\nu,i} \subseteq \mathcal{H}_c, \qquad (11)$$

and

$$\mathcal{H}_c \cap \mathcal{M}_\nu \neq \emptyset \Rightarrow \mathcal{W}_\nu \subseteq \mathcal{H}_c \qquad (12)$$

are fulfilled for all partitions $c \in \mathcal{C}$ and all transition classes $\nu = 1, \ldots, \bar{\nu}$.

The problem is as follows. If the aggregate transitions (8) are considered for $c \in \mathcal{C}$, only the states $z_i,\ i \in \mathcal{H}_c$, are available. Only if (11) is fulfilled, the new states can be calculated, and only if (12) is fulfilled, the transition probabilities $\alpha_\nu(\mathbf{z})$ can be determined.

If (11) is violated for i, or if (12) is violated with $i \in \mathcal{W}_\nu,\ i \notin \mathcal{H}_c$, this difficulty can be eliminated as follows. Split up the transition class ν into new classes, one for each $z_i \in \mathcal{S}_i$. In each of these new transition classes z_i is fixed, the destination state functions and the transition probabilities do not depend on z_i. Or alternatively, if only (11) is violated for i, split up the transition class ν into new classes, one for each new value $u_{\nu,i}(\mathbf{z})$.

For the calculation of the probabilities in (10), the MED parameters $\boldsymbol{\rho}$ are needed. In general, the nonlinear equations (2) must be solved in order to calculate all MED parameters $\rho_{k,j}$ of the MED $\boldsymbol{\pi}$, given the marginal probabilities $\mathbf{R}$. This is easily accomplished in some special cases. For example, if the aggregation is canonical and if the state space is *rectangular*, i.e. if $\mathcal{Z}^T$ is a cartesian product, $\mathcal{Z}^T = \mathcal{S}_1 \times \ldots \times \mathcal{S}_K$, the MED is given by $\pi(\mathbf{z}) = r_1(z_1) \cdot \ldots \cdot r_K(z_K),\ \mathbf{z} \in \mathcal{Z}^T$. If such a simple disaggregation is not available, an approximation is sufficient which becomes more and more accurate during the DA iteration:

$$\rho_{c,j}^{(t+1)} = 0.5 \left(1 + r_c^{(t+1)}(j)/r_c^{(t)}(j)\right) \rho_{c,j}^{(t)}.$$

Unfortunately, we have not yet found error estimates or a proof of convergence of the iteration. Also other authors who developed fixed point techniques observed that proofs of existence of a solution, of the uniqueness, and of convergence are difficult to state, as well as bounds for the accuracy, for example Haverkort [4]. But in [8] the existence of DAII solutions is proven, and it is proven that there is always an aggregation with only a small number of aggregates for which an approximating product form (6) exists which is as accurate as desired. Accuracy is verfied empirically: We solved many models using DA iteration, among others polling, fork join, stochastic Petri net models, queueing networks with blocking, losses, different service disciplines, phase type service times, and MAP arrivals, and we found very accurate results; the iteration did never diverge. Moreover, the accuracy could be substantially ameliorated in all examples if more and suitably chosen aggregates are considered. These matters are discussed in [8].

4 Numerical Examples

Here we consider the steady state probabilities of the component states. Other performance measures are calculated from them. Therefore their accuracy is of particular interest. In the examples with different aggregations the reader may note the gain in accuracy when finer aggregation with more aggregates is used.

Example: A loss network with Erlang-2 distributed service times. An open tandem network with 5 nodes is considered, the capacity is $\nu = 2$ in all nodes, Poisson arrivals occur at node 1 with rate 1, and all service times are Erlang-2 distributed with mean service time 0.5. The jobs go from node 1 to node 2, then to node 3, node 4, and finally to node 5, if they are not lossed before due to full storage in a destination node.

First the states of each node with the waiting room and the system component which describes its service process are used to define a partition. The maximum error of the steady component state probabilities, i.e. the job numbers in waiting rooms and the phases of the service processes, is 0.06... .

The DA approximations are compared with exact Markov chain state probabilities. These were obtained from a Markov chain which was generated with the transition class model description. This is also true for the following example.

If the joint states of all pairs of connected nodes are used for the definition of the aggregation, the maximum error is 0.007... .

If the joint states of all triplets of connected nodes are used for the definition of the aggregation, the maximum error is 0.00008... . □

Example: A loss network with bursty arrivals. An open tandem network with 8 nodes is considered. The capacity is $\nu = 2$ in all nodes. All service times are exponentially distributed with rate 2. The arrival process is defined by a hypergeometric distribution with mean arrival rate 1 and squared coefficient of variation 80.8... . The arrivals occur in bursts, the mean number of jobs per burst is 10, and an epoch without arrivals is ten times as long as an epoch with arrivals, in mean.

First the states of each node with the waiting room are used to define a partition, in node 1 also the system component which describes the arrival process. The maximum error of the steady component state probabilities, i.e. the job numbers in waiting rooms and the phases of the arrival process, is 0.048... .

If the joint states of all pairs of connected nodes are taken for the definition of the aggregation, this maximum error is 0.019... .

If the joint states of all triplets of connected nodes are taken for the definition of the aggregation, this maximum error is 0.009... . □

Conclusion

Steady state probabilities of open queueing networks with losses, bursty arrival processes and phase type service times are investigated. Product form approximations of them are calculated with the disaggregation-aggregation (DA) method which is suitable for very large state spaces. The accuracy of this method was empirically verified for many models, among others priority, polling, fork join, blocking, loss, and stochastic Petri net models. By choosing aggregations with more aggregates – fine aggregation – the accuracy can be substantially ameliorated.

The queueing models are described using a specification technique which we call transition classes. Such a specification can be used to generate the Markov chain, its state space and the transition probability matrix, or for simulation. Furthermore these transition classes are applied to calculate efficiently the aggregate transition probabilities which are essential for our disaggregation-aggregation method.

At that time, a tool is developped which exploits transition classes. This tool generates Markov chains, calculatates the state probabilities, and analyzes the models with simulation. Additionally it supports the approximate disaggregation-aggregation method.

Further research concerns applications of the DA method on other Markov models, time dependent state probabilities, and stiff Markov chains.

References

[1] A. O. Allen. *Probability, Statistics and Queueing Theory.* Academic Press, 1990.

[2] W. J. Stewart (ed.). *Numerical Solution of Markov Chains.* Marcel Dekker, New York, Basel, Hong Kong, 1991.

[3] W. J. Stewart (ed.). *Computation with Markov Chains*. Kluwer, 1995.

[4] B.R. Haverkort. Approximate analysis of networks of PH/PH/1/K queues: Test results. *Accepted for publication in Annals of Operations Research*, 1997.

[5] A. M. Kagan, J. V. Linnik, and C. R. Rao. *Characterization Problems in Mathematical Statistics*. Wiley, New York, 1973.

[6] M. F. Neuts. *Structured Stochastic Matrices of M/G/1–Type and their Applications*. Marcel Dekker, New York, 1989.

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

[8] J. Ch. Strelen. Approximate product form solutions for Markov chains. *Accepted for publication in Performance Evaluation*, 1997.

[9] J. Ch. Strelen, B. Bärk, J. Becker, and V. Jonas. Analysis of queueing networks with blocking using a new aggregation technique. *Accepted for publication in Annals of Operations Research*, 1997.

[10] J. Ch. Strelen. Approximate disaggregation aggregation solutions for general queueing networks. In *Proceedings of the ESM 97 Conference*. To appear, 1997.