Approximate transient analysis of queuing networks by quasi product forms

Alessio Angius[1], András Horváth[1], and Verena Wolf[2]

[1] Dipartimento di Informatica, Università di Torino, Italy
[2] Computer Science Department, Saarland University, Saarbrücken, Germany

Abstract. In this paper we deal with transient analysis of networks of queues. These systems most often have enormous state space and the exact computation of their transient behavior is not possible. We propose to apply an approximate technique based on assumptions on the structure of the transient probabilities. In particular, we assume that the transient probabilities of the model can be decomposed into a *quasi product form*. This assumption simplifies the dependency structure of the model and leads to a relatively small set of ordinary differential equations (ODE) that can be used to compute an approximation of the transient probabilities. We provide the derivation of this set of ODEs and illustrate the accuracy of the approach on numerical examples.

1 Introduction

Queuing networks are heavily used in performance evaluation of distributed systems, such as computer networks, telecommunication systems, and manufacturing or logistics infrastructures. Markovian models of such systems can, in principle, be treated by standard transient and steady state analysis techniques developed for Markov chains [22] but this is almost always precluded because of the huge state space. A family of algorithms that can be applied to models with large state space are based on the fact that the steady state probabilities, under particular assumptions, are in product form. The most classical works in this direction include: Jackson networks [16] which are open networks with Poisson arrivals and infinite capacity, exponential servers; Gordon-Newell networks [11] which are closed networks with exponential servers; and BCMP networks [4] which can be either open, closed or mixed and can contain multiple classes of customers. When the probabilities are not in product form and the state space is large, approximate techniques can be used. One category of approximate approaches is decomposition. By this approach, nodes are analyzed in isolation and their output traffic is characterized and forwarded to the subsequent nodes in order to evaluate the whole network. Methods based on decomposition ranges from simple nodes with simple inter-node traffic description [17, 23] to more general nodes with correlated inter-node traffic [14, 15]. Approximate techniques can also be based on hierarchical analysis where a subnetwork is aggregated into a single node. Algorithms that use this approach are presented, for example, in [3, 21] where the concept of flow equivalent server is employed [9].

The vast majority of the above and related methods are developed to compute steady state measures, such as average response time, long-run throughput or long-run system

utilization. It is for a long time known, however, that steady state measures can be insufficient to describe the behavior of even a single queue. In [24] the GI/M/1 queue is analyzed for what concerns the range of possible fluctuations in the queue-length process for a fixed steady state queue-length distribution. It is shown that queues with equal equilibrium queue-length distribution can have very different second order performance indices, such as, variance of the busy period. For what concerns networks of queues, [20] reveals that equilibrium traffic streams on non-exit arcs in Jackson networks with loops are not Poisson which means that the product form steady state distribution is not sufficient to characterize the traffic in the network. Moreover, modern distributed systems are such complex and dynamic that they often never reach steady state.

For the above mentioned reasons transient analysis of queuing networks is an important topic. Exact computations are possible only for small models or for very particular situations, like networks of infinite server queues [6, 7, 12, 18]. In these networks clients are independent of each other and this leads to the fact that the number of clients at a station follows a Poisson distribution whose mean can be calculated by a set of ODEs. System of ODEs can be used also to develop approximate techniques. Among these we have moment closure techniques [19] which provide approximate moments of the system and fluid approximations that can be used to find the bottlenecks of the network [10]. Methods based on aggregation can also be developed, see, for example, [5]. There are fewer techniques that maintain the original state space of the model and, as a consequence, allow to calculate distributions and not only moments. In [13] an iterative method is suggested to solve the time-dependent Kolmogorov equations of the model but this approach suffers from the state space explosion problem. A memory efficient approach is proposed instead in [25] where the number of ODEs that describe the transient behavior is decreased by assuming a limited dependency structure among the queues of the network.

In this paper we apply an approximate transient analysis technique that maintains the state space of the model. The technique is based on assuming that the transient probabilities can be written in a *quasi product form* (QPF) and it was proposed in [2] to analyze reaction networks. These networks are characterized by the presence of infinite server-like mechanisms combined with switches. Instead in this work, we apply and experiment the approach in the context of queuing networks with finite number of servers.

The QPF assumption leads to a memory efficient description of the transient probabilities and determines a relatively small set of ODEs that provides an approximation of the transient probabilities. A similar approach was adopted in [8] in the context of closed networks to compute passage time distributions. The method that comes closest to ours is the one introduced in [25] and called *partial product form* (PPF) decomposition. Our exposition differs from [25] in the following points. **i)** We apply a more relaxed way of decomposing the transient probabilities into a product form and therefore allow for a more general dependency structure. In particular, the dependency structure considered in [25] is a special case of the dependency structure applied here. **ii)** Our approach is such that the network topology can be used to provide candidates for the way of decomposing the transient probabilities. **iii)** We illustrate the technique on numerical examples and show that it can outperform the one proposed in [25].

The paper is organized as follows. In Section 2 we describe the considered class of queuing networks. Section 3 provides the approximate transient analysis technique based on QPF. In Section 4, characteristics of the proposed technique are discussed. Numerical examples are provided in Section 5. Conclusions are drawn in Section 6.

2 Considered class of queueing networks

We consider an open network of M queues. The maximum number of jobs at queue i is B_i including the job under service with $B_i \in \mathbb{N} \cup \{\infty\}$, i.e., each buffer is either finite or infinite. Clients that arrive to a full buffer are lost. The ith queue of the network receives jobs (clients) entering from outside according to a Poisson-process with intensity λ_i. The routing probabilities are given by r_{ij} with $i, j \in \{1, 2, ..., M\}$ and the probability that a client leaving queue i exits the system is denoted by $r_{i0} = 1 - \sum_{j=1}^{M} r_{ij}$. Service times are exponential and can depend on the length of the local queue. The service intensity of queue i in the presence of j jobs at the station is denoted by $\mu_{i,j}$ (in the following equations we assume that $\mu_{i,0} = 0$).

A given state of the model is an M-dimensional vector $x = |x_1, ..., x_M|$ where x_i denotes the number of jobs at station i. We assume that the arrival intensities, λ_i, and the routing probabilities, r_{ij}, are such that every queue can be reached by the clients. Accordingly, the state space is $\mathcal{S} = \left\{x = |x_1, ..., x_M| \,\middle|\, 0 \leq x_i \leq B_i\right\}$. We use the following notation

$$f_{x,i} = \begin{cases} 1 \text{ if } x_i = B_i \\ 0 \text{ otherwise} \end{cases}$$

to indicate that the ith queue in state x is full and apply also its complement as $\bar{f}_{x,i} = 1 - f_{x,i}$. The number of jobs at queue i at time t is denoted by $Q_i(t)$ and the full system state by $Q(t) = |Q_1(t), ..., Q_M(t)|$. The probability that the system is in state x at time t is denoted by $p(t, x)$, i.e.,

$$p(t, x) = P\{Q(t) = x\} = P\{\wedge_{i \in \{1,2,...,M\}} Q_i(t) = x_i\}$$

As throughout the paper we consider transient analysis, in the following we omit the dependency on time and write Q instead of $Q(t)$ and $p(x)$ instead of $p(t, x)$.

As the network of queues form a continuous time Markov chain, $p(x)$ satisfies the following ODE

$$\begin{aligned} \frac{dp(x)}{dt} = &- p(x) \left(\sum_{i=1}^{M} \bar{f}_{x,i} \lambda_i + \sum_{i=1}^{M} \mu_{i,x_i} \right) + \sum_{i=1}^{M} p(x - h_i) \lambda_i + \\ &\sum_{i=1}^{M} \sum_{j=0}^{M} p(x + h_i - h_j) \mu_{i,x_i} r_{ij} + \sum_{i=1}^{M} \sum_{j=1}^{M} p(x + h_i) \mu_{i,x_i} \bar{f}_{x,j} r_{ij} \end{aligned} \tag{1}$$

where h_0 denotes the vector of zeros, h_i with $1 \leq i \leq M$ the vector with a 1 in the ith place and zeros elsewhere and we assumed that $p(x) = 0$ if $x \notin \mathcal{S}$. In (1) the first term of the right hand side collects the outgoing probabilities of state x; the second the incoming probabilities due to arrivals from the outside; the third the incoming probabilities due to a job leaving station i and joining station j; and the fourth is similar to the third but when station j is full and the job gets lost.

3 Quasi product form approximation

In [1] we applied an approximate transient analysis method for stochastic reaction networks arising in systems biology. This method is based on the assumption that the transient probabilities are in product form, i.e., we can write

$$p(x) = P\{Q = x\} = \prod_{i=1}^{M} P\{Q_i = x_i\}. \tag{2}$$

This assumption leads to a set of ODEs that can be used to compute an approximation of the transient probabilities. The number of ODEs in this set is much lower than the number of states of the system. Roughly speaking, the number of ODEs grows linearly with the number of queues while the number of states grows in an exponential manner. This means that the space complexity of the resulting algorithm is much lower than that of computing the transient probabilities by the classical and widely used uniformisation approach (see, for example, [22]).

The product form assumption given in (2) leads to exact results if the model corresponds to a network of M/M/∞ queues. In [1] we showed that the approximation is satisfactory if the model resembles a network of M/M/∞ queues but can give imprecise results in other cases. In this paper we apply a more relaxed assumption that leads to a good approximation for a wider range of models. In particular, we will assume that there exist sets of queues whose conditional probabilities depend only on a set of other queues and not on all the rest of the queues. For example, if we assume that the conditional probabilities of the number of clients in queue 1 and 2 depend only on the number of clients in queue 3, 4 and 5 then we can write

$$P\{Q_1 = x_1, Q_2 = x_2 | Q_3 = x_3, Q_4 = x_4, ..., Q_M = x_M\} =$$
$$P\{Q_1 = x_1, Q_2 = x_2 | Q_3 = x_3, Q_4 = x_4, Q_5 = x_5\}.$$

A set of assumptions like the one above allows us to decompose the probability $P\{\wedge_{i\in\{1,2,...,M\}} Q_i = x_i\}$ into a product. As this product is not in the classical product form given in (2), we will refer to it as *quasi product form* and in the following we provide its formal description.

The QPF decomposition of the transient probabilities is conveniently described by a directed acyclic graph (DAG), denoted by $\mathcal{F}$. The set of nodes is denoted by $\mathcal{V}$ and a node $v \in \mathcal{V}$ represents a subset of the queues. The index set of the queues represented by node v is denoted by $I(v)$. The set $\mathcal{V}$ must be such that it provides a partitioning of the set of queues, i.e., $\bigcup_{v\in\mathcal{V}} I(v) = \{1, 2, ..., M\}$ and $\forall v_1, v_2 \in \mathcal{V}, v_1 \neq v_2 : I(v_1) \bigcap I(v_2) = \emptyset$. The edge set of the DAG, denoted by $\mathcal{E}$, provides the assumed dependency structure of the transient probabilities. Specifically, if $e = (u, v) \in \mathcal{E}$ (denoted as $u \longrightarrow v$) then the conditional probability of the queues in v *depends* on those queues that are present in u. The set of queues present in the predecessors of v will be denoted by $P(v)$, i.e., $P(v) = \bigcup_{u:u\longrightarrow v} I(u)$. The conditional probability of the queues in $I(v)$ is independent of those queues that are not present in $P(v)$, i.e.,

$$P\{\wedge_{i\in I(v)}(Q_i = x_i) | \wedge_{j\in\{1,2,...,M\}/I(v)} (Q_j = x_j)\} =$$
$$P\{\wedge_{i\in I(v)}(Q_i = x_i) | \wedge_{j\in P(v)} (Q_j = x_j)\}$$

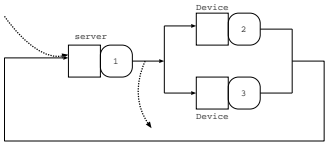

Fig. 1. Open central server network

By considering every node of the tree, the probability of a given state of the system, $|x_1, ..., x_M|$, can be written as

$$P\{\wedge_{i\in\{1,2,...,M\}}(Q_i = x_i)\} = \prod_{v\in\mathcal{V}} P\{\wedge_{i\in I(v)}(Q_i = x_i)|\ \wedge_{j\in P(v)}\ (Q_j = x_j)\} = \prod_{v\in\mathcal{V}} \frac{P\{\wedge_{i\in D(v)}(Q_i = x_i)\}}{P\{\wedge_{j\in P(v)}(Q_j = x_j)\}} \tag{3}$$

where we applied the notation $D(v) = I(v) \bigcup P(v)$. In the following we give three examples for the DAG $\mathcal{F}$.

Example 1. The assumption of complete product form would be expressed by the DAG with M nodes, $v_1, ..., v_M$, such that $I(v_i) = \{i\}$, and an empty set of arcs, $\mathcal{E} = \emptyset$. With this DAG the probabilities are in the form given in (2).

Example 2. The decomposition called *partial product form* in [25] would be expressed by a DAG in which the nodes provide a partitioning of the queues and the set of arcs, $\mathcal{E}$, is empty.

Example 3. Let us consider the simple network depicted in Fig. 1 which will also serve as a numerical example in Section 5. The topology suggests to use a DAG with three nodes, v_1, v_2 and v_3, corresponding to the three stations, respectively, and two arcs as $\mathcal{E} = \{(v_1, v_2), (v_1, v_3)\}$. With this DAG the transient probabilities are approximated in the form

$$\begin{aligned} &P\{\wedge_{j\in\{1,2,3\}}(Q_j = x_j)\} = \\ &P\{Q_1 = x_1\}P\{Q_2 = x_2|Q_1 = x_1\}P\{Q_3 = x_3|Q_1 = x_1\} = \\ &P\{Q_1 = x_1\}\frac{P\{Q_1 = x_1, Q_2 = x_2\}}{P\{Q_1 = x_1\}}\frac{P\{Q_1 = x_1, Q_3 = x_3\}}{P\{Q_1 = x_1\}} \end{aligned} \tag{4}$$

In order to compute the transient probabilities based on the QPF assumption expressed by the DAG $\mathcal{F}$, we need the quantities appearing in (3). Since $P(v) \subseteq D(v)$, the quantities in the denominator can be computed simply by appropriate summation of the quantities in the numerator. The quantities in the numerator can instead be computed by the ODEs provided by the following theorem.

Theorem 1. *If the transient probabilities satisfy the QPF decomposition expressed by the DAG $\mathcal{F}$, then the following ODE holds for all nodes $v \in \mathcal{V}$ and every possible*

values of $x_i, i \in D(v)$*:*

$$\frac{dP\{\wedge_{i\in D(v)}(Q_i=x_i)\}}{dt} = \sum_{\substack{|y_1,\ldots,y_M|:\\ \forall k\in D(v),\, y_k=x_k}} \Bigg(\tag{5}$$
$$-\prod_{v\in\mathcal{V}} \frac{P\{\wedge_{l\in D(v)}(Q_l=y_l)\}}{P\{\wedge_{m\in P(v)}(Q_m=y_m)\}} \left(\sum_{i=1}^M \bar{f}_{y,i}\lambda_i + \sum_{i=1}^M \mu_{i,y_i}\right) +$$
$$\sum_{i=1}^M \prod_{v\in\mathcal{V}} \frac{P\{\wedge_{l\in D(v)}(Q_l=y_l-h_{il})\}}{P\{\wedge_{m\in P(v)}(Q_m=y_m-h_{im})\}}\lambda_i +$$
$$\sum_{i=1}^M \sum_{j=0}^M \prod_{v\in\mathcal{V}} \frac{P\{\wedge_{l\in D(v)}(Q_l=y_l+h_{il}-h_{jl})\}}{P\{\wedge_{m\in P(v)}(Q_m=y_m+h_{im}-h_{jm})\}}\mu_{i,y_i}r_{ij} +$$
$$\sum_{i=1}^M \sum_{j=1}^M \prod_{v\in\mathcal{V}} \frac{P\{\wedge_{l\in D(v)}(Q_l=y_l+h_{il})\}}{P\{\wedge_{m\in P(v)}(Q_m=y_m+h_{im})\}}\mu_{i,y_i}f_{y,j}r_{ij}\Bigg)$$

where h_{ij} *denotes the* j*th entry of* h_i*.*

Proof. It is easy to see that we have

$$\frac{dP\{\wedge_{i\in D(v)}(Q_i=x_i)\}}{dt} = \frac{d}{dt} \sum_{\substack{|y_1,\ldots,y_M|:\\ \forall k\in D(v),\, y_k=x_k}} P\{\wedge_{i\in\{1,\ldots,M\}}(Q_i=y_i)\} \tag{6}$$

where the order of the derivative and the summation can be exchanged. By applying (1) and the QPF assumption given in (3) the theorem follows.

Given a DAG, the set of equations provided by Theorem 1 can be redundant. This happens when we have two nodes, v and u, such that $D(v) \subset D(u)$ because in this case the probabilities $P\{\wedge_{i\in D(v)}(Q_i=x_i)\}$ can be computed by appropriate summation of the quantities $P\{\wedge_{i\in D(u)}(Q_i=x_i)\}$.

In the following example we apply Theorem 1 to the network shown in Fig. 1 with the DAG described in Example 3.

Example 4. Since we have $D(v_1) \subset D(v_2)$ the equations associated with node v_1 can be discarded. For what concerns v_2, assuming that the queues of the network are with infinite buffer, we have for all $x_1 \geq 0, x_2 \geq 0$

$$\frac{dP\{Q_1=x_1, Q_2=x_2\}}{dt} = \sum_{x_3\geq 0} \Bigg($$
$$-P\{|Q_1,Q_2,Q_3| = |x_1,x_2,x_3|\}\left(\lambda_1 + \sum_{i=1}^3 \mu_{i,x_i}\right)+$$
$$P\{|Q_1,Q_2,Q_3| = |x_1-1,x_2,x_3|\}\,\lambda_1+$$
$$P\{|Q_1,Q_2,Q_3| = |x_1+1,x_2,x_3|\}\,\mu_{1,x_1+1}r_{10}+$$
$$P\{|Q_1,Q_2,Q_3| = |x_1+1,x_2-1,x_3|\}\,\mu_{1,x_1+1}r_{12}+$$
$$P\{|Q_1,Q_2,Q_3| = |x_1+1,x_2,x_3-1|\}\,\mu_{1,x_1+1}r_{13}+$$
$$P\{|Q_1,Q_2,Q_3| = |x_1-1,x_2+1,x_3|\}\,\mu_{2,x_2+1}+$$
$$P\{|Q_1,Q_2,Q_3| = |x_1-1,x_2,x_3+1|\}\,\mu_{3,x_3+1}\Bigg)$$

which, by applying the assumed QPF decomposition given in (4) and using

$$P\{|Q_1, Q_2| = |x_1, x_2|\} = \sum_{x_3 \geq 0} P\{|Q_1, Q_2, Q_3| = |x_1, x_2, x_3|\}$$

can be written as

$$\begin{aligned}
&\frac{dP\{Q_1 = x_1, Q_2 = x_2\}}{dt} = \\
&- P\{|Q_1, Q_2| = |x_1, x_2|\} \left(\lambda_1 + \sum_{i=1}^{2} \mu_{i,x_i}\right) - \\
&P\{|Q_1, Q_2| = |x_1, x_2|\} \sum_{x_3 \geq 0} \frac{P\{Q_1 = x_1, Q_3 = x_3\}}{P\{Q_1 = x_1\}} \mu_{3,x_3} + \\
&P\{|Q_1, Q_2| = |x_1 - 1, x_2|\} \lambda_1 + \\
&P\{|Q_1, Q_2| = |x_1 + 1, x_2|\} \mu_{1,x_1+1} r_{10} + \\
&P\{|Q_1, Q_2| = |x_1 + 1, x_2 - 1|\} \mu_{1,x_1+1} r_{12} + \\
&P\{|Q_1, Q_2| = |x_1 + 1, x_2|\} \mu_{1,x_1+1} r_{13} + \\
&P\{|Q_1, Q_2| = |x_1 - 1, x_2 + 1|\} \mu_{2,x_2+1} + \\
&P\{|Q_1, Q_2| = |x_1 - 1, x_2|\} \times \\
&\qquad \sum_{x_3 \geq 0} \frac{P\{Q_1 = x_1 - 1, Q_3 = x_3 + 1\}}{P\{Q_1 = x_1 - 1\}} \mu_{3,x_3+1}
\end{aligned} \tag{7}$$

The above ODE provides a clear interpretation of the impact of applying the QPF decomposition. The events whose intensity does not depend on the number of jobs at the third queue (i.e., arrival from outside, service of clients at station 1 and 2) are considered in an exact manner. The event of serving a job at station 3, whose intensity certainly depends on the number of clients at station 3, is seen from the point of view of the 1st and the 2nd queue as it was independent of the number of clients at station 2. Indeed, the summations in the 2nd and the 8th term of the right hand side of (7) correspond to the intensity of the event of serving a client at station 3 provided that the number of clients at station 1 is x_1 and independent of the number of clients at station 2.

In order to have a complete set of ODEs for the network in Fig. 1 with the DAG described in Example 3, Theorem 1 must be applied to node v_3 as well. The resulting equations, which provide the ODEs for the quantities $P\{Q_1 = x_1, Q_3 = x_3\}$ with $x_1 \geq 0, x_3 \geq 0$, are symmetric to the those presented for v_2 in (7).

4 Characteristics of the approximation

4.1 Number of equations

Assuming that the highest considered buffer level for queue i is $\tilde{B}_i$, the number of equations in the original set of ODEs given in (1) is $\prod_{i=1}^{M}(\tilde{B}_i + 1)$. The number of equations describing the QPF in (5) can be determined as follows. Let Y denote that subset of the nodes of the DAG that provides a complete and not redundant set of ODEs,

i.e., $Y = \{v \mid v \in \mathcal{V}, \nexists u \in \mathcal{V} \text{ such that } D(v) \subset D(u)\}$. Having defined Y, the necessary number of ODEs for the QPF is given by $\sum_{v \in Y} \prod_{i \in D(v)} (\tilde{B}_i + 1)$. This means that the original M-dimensional state space is reduced to $\max_{v \in Y} |D(v)|$ dimensions.

4.2 Choice of DAG

There are two simple factors that can be used to construct the DAG that describes the QPF decomposition. The first is the topology together with the routing probabilities. As considering only this first factor can lead to large system of ODEs, a second factor can also be considered, namely, the dimensionality reduction we want to obtain.

More sophisticated arguments to construct the DAG can be based on the theoretical results presented in the papers dealing with networks of infinite servers [6, 7, 12, 18] that enjoy product form in transient. According to these results, a crucial aspect for the presence of transient product form is that traffic flows are inhomogeneous Poisson processes. This suggests that if the output of a queue is close to an inhomogeneous Poisson process then the dependency between this queue and those that receive its outgoing jobs can be neglected. A necessary condition to have Poisson output in transient is that jobs are not queuing at the station. This condition can be fulfilled by a station with finite number of servers only if its load is low. This implies that if some dependencies of connected queues must be neglected in order to keep low the number of ODEs, then it is convenient to neglect dependencies on the queues with lower loads.

In Section 5 we show examples for which we considered only the topology to guide the choice of the DAG and others for which more factors were taken into account.

4.3 Coherence of the set of ODEs

By looking at (5), one can check that the ODEs provided by Theorem 1 maintain unity of the total probability, i.e., for every node $v \in \mathcal{V}$ summing $P\{\wedge_{i \in D(v)}(Q_i = x_i)\}$ for every possible values of $x_i, i \in D(v)$, gives one.

If there exists a queue whose index, i, is present in both $D(u)$ and $D(v)$ with $u, v \in \mathcal{V}, u \neq v$, then the marginal probabilities of queue i, i.e., $P\{Q_i = x_i\}$, can be derived using the quantities associated with u, i.e., $P\{\wedge_{j \in D(u)}(Q_j = y_j)\}$, or using the quantities associated with v, i.e., $P\{\wedge_{j \in D(v)}(Q_j = y_j)\}$. By considering the derivative $dP\{Q_i = x_i\}/dt$ which can be computed based on both u and v by summation of their associated ODEs given in (5), it is easy to show that the different ways of calculating $P\{Q_i = x_i\}$ lead to the same result. The above reasoning can be generalized to any marginal distribution of the model.

4.4 Limiting behavior

In principle, the steady state determined by the QPF assumption can be calculated by setting the left hand side of (5) to zero and solving the resulting set of equations. In practice, this is not feasible because the set of equations is not linear and the number of equations can be large. Exact steady state is determined by the equations given in (1) by setting the left hand side to zero. One can observe that the right hand side of (5) contains summations of the right hand side of (1) for given set of states. This implies that the limiting behavior of the QPF approximation is such that it satisfies sums of

those equations that determine the exact steady state. Moreover, if the exact steady state is uniquely determined by these sums of equations then the limiting behavior of the approximation is exact.

5 Numerical examples

In this section we apply the QPF approximation to three models with various settings of the parameters. The first example has a small state space and thus allows us to compare the results obtained by the approximation to the numerical solution of the underlying CTMC. We use this small example to show that the QPF decomposition can outperform the PPF decomposition [25]. In case of the last two models with large state spaces we compare the QPF approximation to results obtained using Monte Carlo simulation.

The algorithm based on the QPF assumption has been implemented in JAVA using the odeToJava package[1] for the solution of the system of ODEs. In particular, we applied the "explicit Runge-Kutta triple" solver of the package. The accuracy of this method is determined by two parameters, called relative and absolute tolerance, and we set these values to 10^{-10} and 10^{-12}, respectively. The reported run times refer to these settings and by choosing less restrictive values the computation times can be lowered significantly, by about one order of magnitude. All the experiments have been performed on an Intel Centrino Dual Core with 4Gb of RAM.

5.1 Open central server network

As first, we consider a network representing a server connected to two devices (Fig. 1). Jobs arrive to the server with a constant rate λ_1 and compete for the resources of the system. After each service at the first station, the job can leave the system with probability $r_{1,0}$ or use one of the two devices with probabilities $r_{1,2}$ and $r_{1,3}$, respectively. Jobs leaving the two devices go back to the server. We tested the approximation with the parameters $r_{1,2} = 0.2$, $r_{1,3} = 0.3$, $\lambda_1 = 0.6$, $\mu_1 = 1$, $\mu_2 = 5$, $\mu_3 = 1$. We assume that the server is triplicated, i.e., three jobs can be under service at a time at the server. Note that with single server policy the system would not be stable due to the presence of the loop. The number of jobs at a queue is at most 15, further arrivals are lost.

We analyzed the model by using its original CTMC, by all possible partial product form decompositions, and by the QPF decomposition given in Example 3. With this QPF decomposition the set of required marginal distributions are the probabilities $P\{Q_1 = x_1, Q_2 = x_2\}$ and $P\{Q_1 = x_1, Q_3 = x_3\}$. We considered two situations: starting with empty queues and starting with five jobs at the server. In Fig. 2 and 3 we depicted the mean and the variance of the number of jobs at the server as function of time. Starting with empty queues, both the QPF and PPF approximations lead to good results with the exceptions of the third PPF decomposition that fails to provide an accurate estimate of the variance. Starting with five jobs at the server, the approximations are worse and only the QPF decomposition captures the mean correctly and approximate the variance well.

The original CTMC is composed of $16^3 = 4096$ states. The PPF approximations lead to $16^2 + 16 = 272$ ODEs while the QPF uses $2 \times 16^2 = 512$ ODEs. The number

[1] Available at *http://www.netlib.org/ode/* and developed by M. Patterson and R. J. Spiteri.

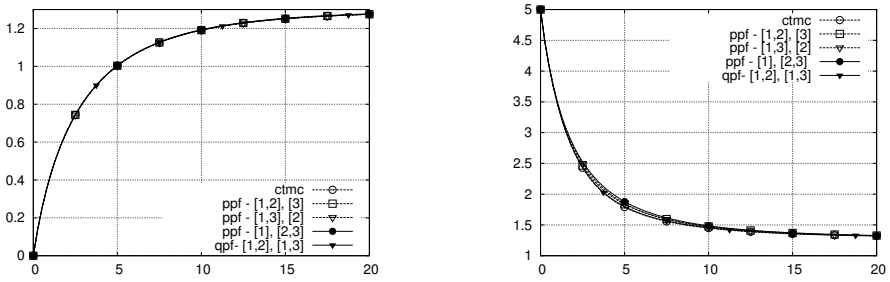

Fig. 2. Open server network: mean number of jobs at the server vs. time; starting with empty queues (left), with five jobs at the server (right).

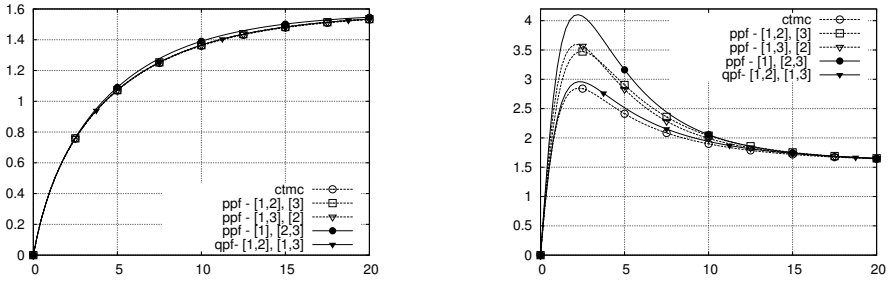

Fig. 3. Open server network: variance of number of jobs at the server vs. time; starting with empty queues (left), with five jobs at the server (right).

of equation of the QPF decomposition is about two times more but the dimensionality of the the approaches are the same as both consider the dependencies of at most two stations. The calculations with QPF decomposition required about 1 second of CPU time.

5.2 Multi-path network

As second example, we consider a more complex pipeline where each request can be satisfied by following four different production paths. The network is depicted in Fig. 4. Requests arrive with a fixed rate $\lambda_1 = 0.6$ to the first station that constitutes a pre-processing step. After that, the possible paths for a request are stations 2-4, 2-5, 3-4 and 3-5 followed by a post-processing phase that takes place at station 6. The parameters that we consider are such that the routing probabilities are symmetric with $r_{1,2} = r_{2,4} = r_{3,5} = 0.5$ but the processing intensities are asymmetric with $\mu_2 = \mu_4 = 0.4$ and $\mu_3 = \mu_5 = 2$, i.e., the branch composed by stations 2 and 4 is slow while the one containing stations 3 and 5 is fast. The pre-processing and the post-processing stations are serving the jobs with intensity $\mu_1 = \mu_6 = 1$. The maximum number of clients for each queue is 50.

The QPF decomposition suggested by the topology is given by the DAG in which each station defines a node and the routing of the network coincides with the set of edges, i.e., $\mathcal{E} = \{(1,2),(1,3),(2,4),(2,5),(3,4),(3,5),(4,6),(5,6)\}$. It is easy to

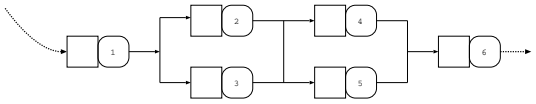

Fig. 4. Multi-path network

see that the corresponding set of necessary marginal distributions are composed of the probabilities $P\{Q_1 = x_1, Q_2 = x_2\}$, $P\{Q_1 = x_1, Q_3 = x_3\}$, $P\{Q_2 = x_2, Q_3 = x_3, Q_4 = x_4\}$, $P\{Q_2 = x_2, Q_3 = x_3, Q_5 = x_5\}$, $P\{Q_4 = x_4, Q_6 = x_6\}$ and $P\{Q_5 = x_5, Q_6 = x_6\}$. Since this way the calculations would require three dimensional marginals we apply a DAG with less edges. According to the arguments provided in Sec. 4.2, we neglect the dependencies on the less loaded queues which are in station 3 and 5. This way the edges of the DAG are $\mathcal{E} = \{(1,2),(1,3),(2,4),(2,5),(4,6)\}$. The necessary marginals with this DAG are all two dimensional: $P\{Q_1 = x_1, Q_2 = x_2\}$, $P\{Q_1 = x_1, Q_3 = x_3\}$, $P\{Q_2 = x_2, Q_4 = x_4\}$, $P\{Q_2 = x_2, Q_5 = x_5\}$ and $P\{Q_4 = x_4, Q_6 = x_6\}$.

We tested the model starting from two different initial states. We first consider the case in which initially all queues are empty. In the second case the system starts with 10 requests in stations 2, 3, 4, and 5. Expectations and variances of the number of jobs at stations 4, 5 and 6 for the two cases are depicted in Fig.s 5 and 6. By comparing the two figures, one can observe to what extent starting from the second initial state penalizes station 4. After about 10 time units the average number of clients at station 4 is about 14 while the same average never exceeds 3 starting from an empty system. The longer run effect can be seen instead looking at the variances: for station 4 this quantity is increasing up to about 100 time units and for station 5 as well it reaches much higher values than in case of starting from an empty network. All these behaviours are captured well by the QPF approximation. In Fig. 7 we depicted the probabilities of having no clients at station 4 and 6. On these curves as well one can observe the effect of the choice of the initial state.

The network is composed of $51^6 = 1.76 \times 10^{10}$ states. The number of ODEs for the QPF approximation is $5 \times 51^2 = 13005$. The presented results were calculated in about one minute.

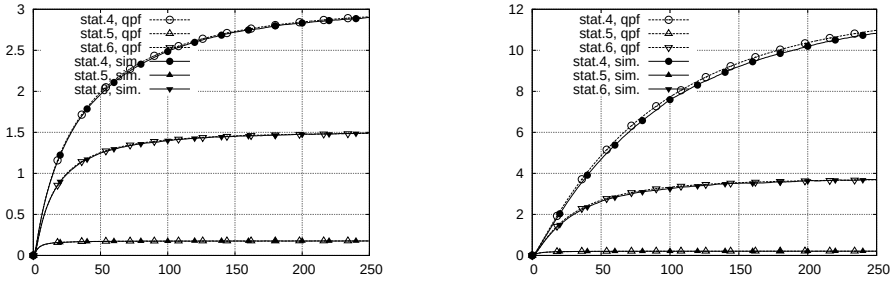

Fig. 5. Multi-path: expectations (left) and variances (right) of the stations 4, 5 and 6 vs. time starting from state $\{0,0,0,0,0,0\}$

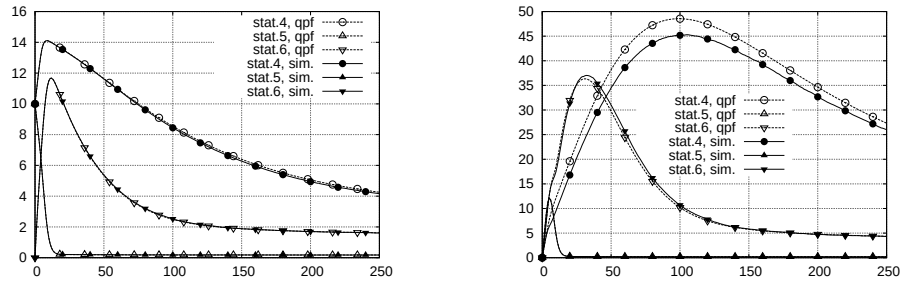

Fig. 6. Multi-path: expectations (left) and variances (right) of the stations 4, 5, and 6 vs. time starting from state $\{0, 10, 10, 10, 10, 0\}$

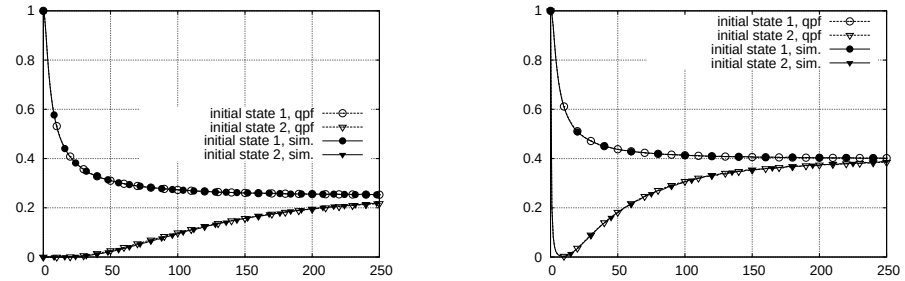

Fig. 7. Multi-path: Probability of empty queue in station 4 and 6 vs. time starting from the two different initial states

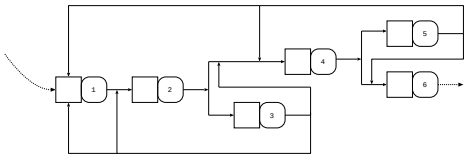

Fig. 8. On-demand production system

5.3 On-demand production system

As last example, we propose a network representing a production pipeline. The network is depicted in Fig. 8. Customer requests arrive at station 1 where they are processed and sent to the production stations. In the optimal case, each customer request requires only two production steps that are performed at station 2 and 4. The product undergoes also a quality check at station 2 and 4. When this check at station 2 (4) is not successful the product is sent to station 3 (5) where it is monitored. If the problem is a false positive or fixable, then the product is reintegrated at the next station of the production line. Otherwise the production request is sent back to station 1 or 2 $(1$ or $4)$. At station 6, the final product is delivered to the customer.

The QPF decomposition we apply is described by the DAG composed of a set of nodes corresponding to the set of stations and the set of edges is $\mathcal{E} = \{(1,2),(2,3),(2,4),(4,5),(4,6)\}$. The differences between the set of edges of the DAG and the

routing of the network are the arcs $(3,1)$, $(3,2)$, $(3,4)$, $(5,1)$, $(5,4)$, and $(5,6)$. Four of these arcs are not present in the DAG because they form cycles. The other two, $(3,4)$ and $(5,6)$, are excluded instead to keep low the number of dimensions of the marginal distributions that are necessary to compute the approximation. With the above described DAG the necessary marginal distributions are $P\{Q_1 = x_1, Q_2 = x_2\}$, $P\{Q_2 = x_2, Q_3 = x_3\}$, $P\{Q_2 = x_2, Q_4 = x_4\}$, $P\{Q_4 = x_4, Q_5 = x_5\}$ and $P\{Q_4 = x_4, Q_6 = x_6\}$.

The parameters that we use are $\lambda_1 = 0.4$, $r_{2,3} = r_{4,5} = 0.5$, $r_{3,1} = r_{5,1} = 0.2$, $r_{3,2} = r_{5,4} = 0.7$, $\mu_1 = \mu_2 = \mu_4 = \mu_6 = 1$, and $\mu_3 = \mu_5 = 1.5$. Note that with these parameters clients enter the cycles with high probability. The maximum number of clients for each queue is 50. We assume that the queues are empty initially. The number of states, the number of ODEs representing the QPF and the computation times are the same as in case of the example in Sec. 5.2.

In Fig. 9 and 10 we show the mean and the variance of the number of clients at the stations as function of time. Fig. 11 depicts instead the probability of the empty queue. It can be seen that the QPF approximation provides quite a precise view of these quantities.

6 Conclusions

In this paper we applied an approximate transient analysis technique for networks of queues. The technique is based on the assumption that the transient probabilities can be

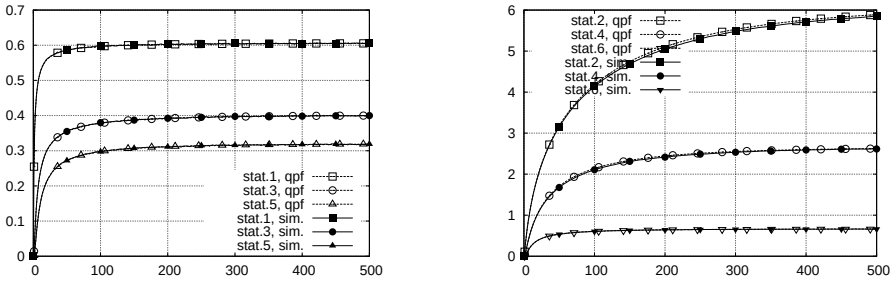

Fig. 9. On-demand production system: expectation of the number of jobs at the stations vs. time

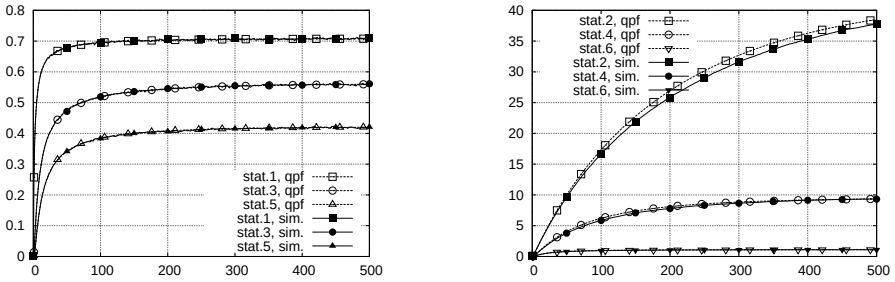

Fig. 10. On-demand production system: variance of the number of jobs at the stations vs. time

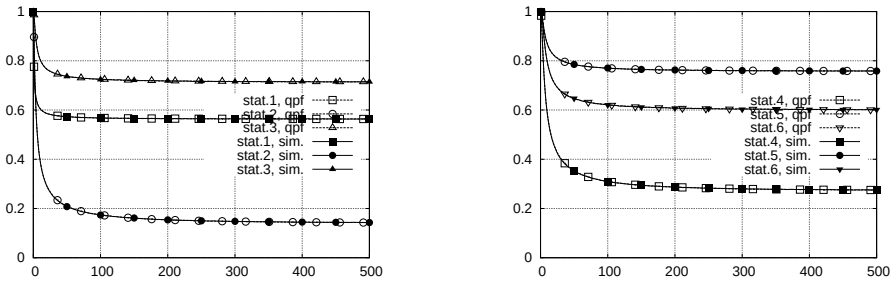

Fig. 11. On-demand production system: probability of empty queue at the stations vs. time

expressed approximately in QPF. This assumption leads to a memory efficient algorithm to analyze networks even with huge state space. We discussed the properties of the algorithm and provided several numerical examples to illustrate its accuracy.

In the future we plan to develop solvers that are specific to the ODEs arising from the QPF approximation. In this paper we intentionally concentrated on a relatively simple class of networks. The technique can be applied to networks in which arrival streams and service processes are correlated in a Markovian manner. Moreover, time dependent arrival rates and service processes can also be incorporated into the ODEs that describe the approximation. We plan to implement the resulting method and apply it to a wide range of cases.

Acknowledgments. This work has been supported in part by project grant Nr. 10-15-1432/HICI from the King Abdulaziz University of Kingdom of Saudi Arabia and by project grant AMALFI (Advanced Methodologies for the AnaLysis and management of the Future Internet) financed by the Intesa Sanpaolo banking group.

Bibliography

[1] A. Angius and A. Horváth. Product form approximation of transient probabilities in stochastic reaction networks. *Electronic Notes on Theoretical Computer Science*, 277:3–14, 2011.

[2] A. Angius, A. Horváth, and V. Wolf. Quasi product form approximation for Markov models of reaction networks. *Transactions on Computational Systems Biology*, XIV, 2012.

[3] J. Anselmi, G. Casale, and P. Cremonesi. Approximate solution of multiclass queuing networks with region constraints. In *Proc. of 15th Int. Symp. on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS '07)*, pages 225–230, 2007.

[4] F. Baskett, K.M. Chandy, R.R. Muntz, and G. Palacios. Open, closed, and mixed networks of queues with different classes of customers. *Journal of the ACM*, 22(2):248–260, 1975.

[5] P. Bazan and R. German. Approximate transient analysis of large stochastic models with WinPEPSY-QNS. *Computer Networks*, 53:1289–1301, 2009.

[6] R. J. Boucherie. *Product-form in queueing networks*. PhD thesis, Vrije Universiteit, Amsterdam, 1992.

[7] R. J. Boucherie and P.G. Taylor. Transient product form distributions in queueing networks. *Discrete Event Dynamic Systems: Theory and Applications*, 3:375–396, 1993.

[8] G. Casale. Approximating passage time distributions in queueing models by Bayesian expansion. *Perform. Eval.*, 67(11):1076–1091, 2010.

[9] K. M. Chandy, U. Herzog, and L. Woo. Parametric analysis of queueing networks. *IBM Journal of Research and Development*, 19(1):36–42, 1975.

[10] H. Chen and A. Mandelbaum. Discrete flow networks: Bottleneck analysis and fluid approximations. *Mathematics of Operations Research*, 16(2):408–446, 1991.

[11] W.J. Gordon and G.F. Newell. Cyclic queueing networks with exponential servers. *Operations Research*, 15(2):254–265, 1967.

[12] J. M. Harrison and A. J. Lemoine. A note on networks of infinite-server queues. *J. Appl. Probab.*, 18(2):561–567, 1981.

[13] P. G. Harrison. Transient behaviour of queueing networks. *Journal of Applied Probability*, 18(2):482–490, 1981.

[14] A. Horváth, G. Horváth, and M. Telek. A traffic based decomposition of two-class queueing networks with priority service. *Computer Networks*, 53:1235–1248, 2009.

[15] A. Horváth, G. Horváth, and M. Telek. A joint moments based analysis of networks of MAP/MAP/1 queues. *Performance Evaluation*, 67:759–778, 2010.

[16] J.R. Jackson. Jobshop-like queueing systems. *Management Science*, 10(1):131–142, 1963.

[17] P. Kuehn. Approximate analysis of general queuing networks by decomposition. *IEEE Transactions on Communications*, 27(1):113 – 126, 1979.

[18] W. A. Massey and W. Whitt. Networks of infinite-server queues with nonstationary Poisson input. *Queueing Systems*, 13:183–250, 1993.

[19] T. I. Matis and R. M. Feldman. Transient analysis of state-dependent queueing networks via cumulant functions. *Journal of Applied Probability*, 38(4):841–859, 2001.

[20] B. Melamed. Characterizations of Poisson traffic streams in jackson queueing networks. *Advances in Applied Probability*, 11(2):422–438, 1979.

[21] C. H. Sauer. Approximate solution of queueing networks with simultaneous resource possession. *IBM Journal of Research and Development*, 25(6):894–903, 1981.

[22] W. J. Stewart. *Introduction to the Numerical Solution of Markov Chains.* Princeton University Press, 1995.

[23] W. Whitt. The queueing network analyzer. *Bell System Technical Journal*, 62(9):2779–2815, 1983.

[24] W. Whitt. Untold horrors of the waiting room. what the equilibrium distribution will never tell about the queue-length process. *Management Science*, 29(4):395–408, 1983.

[25] W. Whitt. Decomposition approximations for time-dependent Markovian queueing networks. *Operations Research Letters*, 24:97–103, 1999.