Performance Evaluation 91 (2015) 80–98

Contents lists available at ScienceDirect

Performance Evaluation

journal homepage: www.elsevier.com/locate/peva

QD-AMVA: Evaluating systems with queue-dependent service requirements

Giuliano Casale *, Juan F. Pérez, Weikun Wang

Department of Computing, Imperial College London, SW7 2AZ, London, UK

ARTICLE INFO

Article history:
Available online 2 July 2015

keywords:
Closed queueing network
Product-form
Approximate mean value analysis
State-dependent service

ABSTRACT

Workload measurements in enterprise systems often lead to observe a dependence between the number of requests running at a resource and their *mean* service requirements. However, multiclass performance models that feature these dependences are challenging to analyze, a fact that discourages practitioners from characterizing workload dependences. We here focus on closed multiclass queueing networks and introduce QD-AMVA, the first approximate mean-value analysis (AMVA) algorithm that can efficiently and robustly analyze queue-dependent service times in a multiclass setting. A key feature of QD-AMVA is that it operates on mean values, avoiding the computation of state probabilities. This property is an innovative result for state-dependent models, which increases the computational efficiency and numerical robustness of their evaluation. Extensive validation on random examples, a cloud load-balancing case study and comparison with a fluid method and an existing AMVA approximation prove that QD-AMVA is efficient, robust and easy to apply, thus enhancing the tractability of queue-dependent models.

1. Introduction

Performance analysis of real-world systems often relies on analytical performance models such as queueing networks to capture limits on the maximum concurrency levels and pooling at software and hardware servers. In particular, when the model focuses on describing contention at CPUs, the scheduling assumed at the resources is processor-sharing and queueing network models become tractable thanks to the product-form result by Baskett et al. [1] and the availability of several approximate mean-value analysis (AMVA) algorithms for load-independent models [2]. Load-independent AMVA algorithms have been very successful since they are both efficient and accurate. These methods have found application, for instance, in the analysis of batch jobs in large databases [3], and in the prediction of memory contention on multicore servers [4]. Load-independent AMVA methods are also extensively used in the analysis of Layered Queueing Networks (LQNs) [5–7], a class of queueing networks well-suited for the analysis of software systems.

Although the evaluation of load-independent models is a mature area, a common problem in real-world studies is the fact that the service requirements of an application are seldom independent of the mix and number of requests in execution, even if one considers the average service time only. A typical example occurs in servers with multiple CPUs. Parallel jobs may utilize multiple cores to process sub-tasks concurrently, leading to a variable service rate [8], i.e., the effective service rate depends on the load, specifically on the number of sub-tasks in execution. Databases deployed on multiple servers have also been modeled as load-dependent servers [9]. Similarly, in disks it has been shown [10] that the mean seek time depends on the number of requests queueing at the disk, resulting in a variable disk service rate that depends on

* Corresponding author.
E-mail addresses: g.casale@imperial.ac.uk (G. Casale), j.perez-bernal@imperial.ac.uk (J.F. Pérez), weikun.wang11@imperial.ac.uk (W. Wang).

http://dx.doi.org/10.1016/j.peva.2015.06.006

the load (number of requests) being processed. Furthermore, the congestion control mechanisms common in broadband networks [11] are also a source of variable service rates. Another example of load-dependent behavior arises in enterprise web applications [12], which are complex transaction-based systems that are known to display different behaviors in low and high loads due to caching and shared data structures. Even if one characterizes these dependences from empirical data, the limiting factor for the subsequent analyses is the lack of methods to efficiently evaluate the resulting queueing network models, unless the model is small. On the other hand, relying on load-independent AMVA methods limits the expressiveness of the performance models. This has become apparent also in recent systems research, which is increasingly advocating the use of mix-dependent methods for performance prediction and control [13,14]. Another area where load-dependent behaviors have been consistently observed are call centers [15]. Here the dependence arises from the agent heterogeneity due to their different skill sets, training, and fatigue levels, among others.

To support the raising interest for characterizing load dependence in queueing networks, this paper introduces QD-AMVA, a novel AMVA algorithm for queue-dependent models, i.e., models of systems where the service requirements depend on the number and mix of requests in execution at the resource where the request executes. Although AMVA methods for load-dependent models exist [1], these methods have severe limitations in the multiclass setting, including excessive computational requirements, numerical instabilities, and being restricted to the simplest types of queue-dependence [16,17]. This has led to multiclass state-dependent networks being seldom used in real-world studies. Moreover, existing methods are perceived as complex since they focus on evaluating state probabilities even when mean values are sufficient for basic performance assessment.

QD-AMVA is a queue-dependent AMVA algorithm that operates on mean values only, avoiding the computation of state probabilities. This feature implies that the algorithm is efficient, since its computational requirements become independent of the population size, and it is also numerically stable. Our algorithm is applicable to a large family of dependence functions satisfying mild differentiability assumptions. Furthermore, we prove that the AMVA equations always admit a solution and this solution can be guaranteed to be unique under monotonicity assumptions for the dependence functions. A noticeable case that falls under these monotonicity conditions is the multi-server queue, which can be readily evaluated by the QD-AMVA technique. In this setting, we show that our approximation of the multi-server queue improves over existing AMVA approaches for multi-server queues and it is also better than a well-known fluid approximation.

We then investigate theoretical properties of the method, including existence and uniqueness of the QD-AMVA solutions, and develop exact formulas for sensitivity analysis of queue-dependent performance measures. A by-product of this investigation is that we find it possible to evaluate the normalizing constant of the state probabilities for a queue-dependent model by combining QD-AMVA sensitivity formulas with an existing approximation scheme for load-independent models.

We illustrate the efficiency and accuracy of QD-AMVA using random test instances, a running case, and a comparison with existing approximations. We also showcase the applicability of our methodology to real-world systems in a load-balancing use case for a cloud application. Using real-world measurements from a multi-tier enterprise application deployed on the Amazon EC2 cloud, which shows queue-dependent behavior, we show that a model solved with QD-AMVA provides a better configuration of the load-balancing weights compared to those recommended by an ordinary load-independent AMVA.

Summarizing, the present paper considers performance evaluation of systems under queue-dependent workloads and makes the following main contributions:

- QD-AMVA, a novel approximate algorithm for evaluating closed queue-dependent queueing networks in a numerically stable and efficient manner, without computing state probabilities;
- A characterization of the properties of QD-AMVA, including existence and uniqueness of the solutions;
- Sensitivity analysis results to compute gradients of performance measures in queue-dependent models;
- Extension of an existing approximation for the normalizing constant of queue-dependent models, which enables assessment of probabilistic measures;
- Validation against random models, a case study, and two approximations for multi-server queues.

The rest of the paper is organized as follows. After providing background in Section 2, we further discuss motivation in Section 3. The AMVA method for queue-dependent models is defined in Section 4 and characterized in Section 5. Extensions to compute normalizing constants and sensitivity measures are given in Section 6. Evaluation of the methods is given in Sections 7 and 8. Finally, Section 9 gives final remarks.

2. Background

2.1. Queue-dependent product-form solution

We begin by reviewing exact results for queue-dependent closed networks considered by the BCMP theorem [1]. The reference model is composed of M stations and R job classes; indexes $k, i \in \{1, \ldots, M\}$ are used throughout to reference stations, while indexes $r, s \in \{1, \ldots, R\}$ reference classes. Each class r has a constant population of $K_r \geq 1$ jobs. The total job population in the model is denoted by $K = \sum_{r=1}^{R} K_r$. Scheduling at queueing stations is assumed to be processor sharing (PS) or any other scheduling discipline satisfying the BCMP theorem assumptions [1]. The state of the system is described by an $M \times R$ matrix $\boldsymbol{n}$ with generic element $n_{k,r}$ being a random variable counting the number of jobs of class r at station k. We define $n_k = \sum_{r=1}^{R} n_{k,r}$ to be the total number of jobs at station k.

The equilibrium probability $\pi(\boldsymbol{n})$ of state $\boldsymbol{n}$ is given by the BCMP product-form solution for a queue-dependent network, which may be written as

$$\pi(\boldsymbol{n}|\boldsymbol{K}) = G^{-1} \prod_{k=1}^{M} C_k(\boldsymbol{n}_k) F_k(\boldsymbol{n}_k), \quad \boldsymbol{n} \in \mathcal{S}, \tag{1}$$

where $\boldsymbol{n}_k$ is the kth row of $\boldsymbol{n}$ describing the state of station k, $C_k(\boldsymbol{n}_k) = n_k!(\prod_{r=1}^{R} n_{k,r}!)^{-1}$, G is a normalizing constant relatively to the state-space

$$\mathcal{S} = \left\{ \boldsymbol{n} | n_{k,r} \geq 0, \sum_{k=1}^{M} n_{k,r} = K_r, \text{for all } r = 1, \ldots, R \right\},$$

and the product-form factors $F_k(\cdot)$ are recursively defined by

$$F_k(\boldsymbol{n}_k) = D_{k,r}(\boldsymbol{n}_k) F_k(\boldsymbol{n}_k - \mathbf{1}_r), \tag{2}$$

for any station k and class r such that $n_{k,r} \geq 1$, where $\mathbf{1}_r$ is a vector with a 1 in position r and 0 elsewhere. This recurrence relation has termination condition $F_k(\mathbf{0}) = 1$ where $\mathbf{0} = (0, \ldots, 0)$.

We assume that jobs are routed across the network according to an irreducible discrete-time Markov chain. Under this assumption, each state-dependent function $D_{k,r}(\boldsymbol{n}_k)$ is given by the mean number of visits of class-r jobs to station k, with respect to an arbitrary reference station, divided by the class-r service rate at station k when this station is in state $\boldsymbol{n}_k$. We assume $D_{k,r}(\boldsymbol{n}_k)$ to be either a positive function or, if class r does not visit station k, to be identically zero for all values of its argument. In the case where all stations are load-independent, $D_{k,r}(\boldsymbol{n}_k) = \theta_{k,r}$, where $\theta_{k,r}$ is the mean service demand of class r at station k, the formula becomes the well-known expression of the BCMP theorem for load-independent stations [1].

Throughout the paper, we focus on networks where one or more classes have queue-dependent service requirements specified through the functions $D_{k,r}(\boldsymbol{n}_k)$. Following [18], it is known that a function $D_{k,r}(\boldsymbol{n}_k)$ preserves the product-form property (1) if and only if, in addition to (2), it also satisfies the constraint

$$D_{k,s}(\boldsymbol{n}_k - \mathbf{1}_r) D_{k,r}(\boldsymbol{n}_k) = D_{k,r}(\boldsymbol{n}_k - \mathbf{1}_s) D_{k,s}(\boldsymbol{n}_k), \tag{3}$$

for all population vectors $\boldsymbol{n}_k$ and classes r and s such that $n_{k,s} \geq 1$ and $n_{k,r} \geq 1$. This requirement stems from the observation that, if $n_{k,r} \geq 1$ for multiple values of r, then (2) can be expanded recursively in different ways according to the choice of r. The constraint (3) simply ensures that all the possible recursion branches terminate providing an identical value for $F_k(\boldsymbol{n}_k)$.

2.1.1. Performance measures

To simplify notation, unless needed we omit the dependence of performance measures on $\boldsymbol{K}$, but we keep explicit the dependence on related population vectors such as $\boldsymbol{K} - \mathbf{1}_r$. Common performance measures for state-dependent models include the class-r throughput [19], $T_r = G(\boldsymbol{K} - \mathbf{1}_r)/G$, and the *mean* queue-length of class-r jobs at station k [17],

$$x_{k,r} = \sum_{\boldsymbol{n}_k \in \mathcal{N}} n_k T_r D_{k,r}(\boldsymbol{n}_k) \pi(\boldsymbol{n}_k - \mathbf{1}_r | \boldsymbol{K} - \mathbf{1}_r), \tag{4}$$

where $\mathcal{N} \equiv \mathcal{N}(k, r) = \{\boldsymbol{n} \in \mathcal{S} | 1 \leq n_{k,r} \leq K_r\}$, $\mathcal{N} \subset \mathcal{S}$, and $\pi(\boldsymbol{n}_k | \boldsymbol{K} - \mathbf{1}_r)$ is the marginal probability of station k being in state $\boldsymbol{n}_k$ in a model with $\boldsymbol{K} - \mathbf{1}_r$ jobs. Note that due to the closed nature of the model, the sum of mean queue-lengths is constant for each class, i.e., $\sum_{k=1}^{M} x_{k,r} = K_r$. Finally, we will group mean queue-lengths into a vector $\boldsymbol{x}_k = (x_{k,1}, \ldots, x_{k,R})$, and use the shorthand notation $x_k = \sum_{r=1}^{R} x_{k,r}$.

2.2. Product-form queue-dependence

In general, it is difficult to find a class of queue-dependent functions $D_{k,r}(\boldsymbol{n}_k)$ that satisfies (3) and at the same time allows for fairly general behaviors as a function of $\boldsymbol{n}_k$. However, this flexibility is needed, since real-world workload measurements can show complex trends. To allow for more flexibility, we restrict our attention to functions of the type $D_{k,r}(\boldsymbol{n}_k) \equiv D_{k,r}(n_k, n_{k,r}) = \theta_{k,r} \beta_{k,r}(n_{k,r}) \gamma_k(n_k)$, where we assume $\beta_{k,r}(n_{k,r})$ and $\gamma_k(n_k)$ are bounded strictly-positive functions in the range $1 \leq n_{k,r} \leq K_r$ and $1 \leq n_k \leq K$, respectively. This form encompasses the most popular types of queue-dependent functions used in the literature,[1] for example, in the so-called load-dependent models, these functions reduce to $D_{k,r}(n_k, n_{k,r}) = \theta_{k,r} \gamma_k(n_k)$ [17].

[1] Forms that do not satisfy this form are seldom used in the literature. One such example is the multiclass flow-equivalent server [20]. However, in a multiclass flow-equivalent server the form of the service demands is dictated by the throughput of the subnetwork that it replaces. Thus, in these models, it is not possible to use arbitrary mathematical expressions for the queue-dependence. A few other examples can be found in [17], however none of these other forms can be efficiently analyzed in multiclass models of practical size.

We refer to the queue-dependent service requirements in the form $D_{k,r}(\boldsymbol{n}_k) = \theta_{k,r}\beta_{k,r}(n_{k,r})\gamma_k(n_k)$ as *product-form demands*. Their key property is that they always satisfy condition (3) since

$$\frac{D_{k,r}(\boldsymbol{n}_k)}{D_{k,s}(\boldsymbol{n}_k)} = \frac{\beta_{k,r}(n_{k,r})}{\beta_{k,s}(n_{k,s})} = \frac{D_{k,r}(\boldsymbol{n}_k - \mathbf{1}_s)}{D_{k,s}(\boldsymbol{n}_k - \mathbf{1}_r)},$$

for all $n_{k,r} \geq 1$ and $n_{k,s} \geq 1$, regardless of the expressions of $\beta_{k,r}(n_{k,r})$ and $\gamma_k(n_k)$. Thus, arbitrary mathematical expressions can be used to specify the queue-dependent functions $\beta_{k,r}(n_{k,r})$ and $\gamma_k(n_k)$. This dependence is therefore rather flexible, and its main limitation is that it allows the service time of a job of class r to depend on the number of jobs of classes $s \neq r$ only through the total queue-length value n_k. Assumptions of this kind are rather common for tractability, for example they arise in mean-field analysis techniques [21] and in the analysis of state-dependent behaviors that preserve the $M \Rightarrow M$ property [22].

Without loss of generality, we define $\theta_{k,r} = D_{k,r}(\mathbf{1}_r)$, such that $\theta_{k,r}$ represents the service demand of a single job of class r when this is served alone at resource k, from service start to completion. With this definition, we can assume from now on that $\beta_{k,r}(1) = 1$ and $\gamma_k(1) = 1$.

3. Motivation

In this section, we provide evidence that the class of models described in the previous section is intractable with state-of-the-art methods. We first review existing methods, and then introduce a running case for the paper that cannot be analyzed efficiently with existing techniques as the number of classes grows.

3.1. Related work

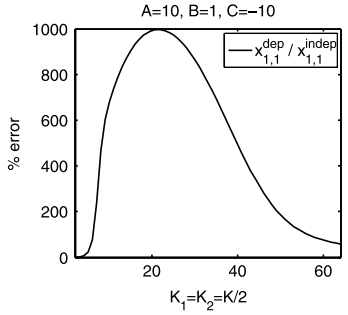
Load-dependent queueing systems have been recognized as an important modeling tool. In open systems, particular attention has been paid to the single-station case. For instance, [23,24] study a queueing system where the service times change according to the queue length, while [25] considers the case where the service time depends on the waiting time experienced by the job in process.

In closed systems, the analysis of closed networks with queue-dependent stations is an unsolved problem from a computational standpoint, since multiclass methods require exponentially large time and space as the number of classes and jobs in the system grows [18,17]. Computational limitations have led previous work to focus on simpler load-dependent models, which are a subset of the class of models considered in this paper. Exact analysis of load-dependent models has been derived in [26], which introduces the multiclass convolution method, and in [19], which defines exact mean-value analysis (MVA) for multiclass models, including load-dependent models. However, load-dependent algorithms are notoriously unstable, and scalable and stable computational algorithms exist only for single-class models with a single load-dependent queue [27]. Also, [16] reports numerical difficulties for the queue-dependent MVA and proposes a correction for the single-class case. A systematic review of the state-of-the-art for single-class and multiclass models is provided in [17,18]. The main limitation of these algorithms is that their computational requirements grow exponentially with the model size, becoming soon prohibitive on networks with more than a few jobs. The community has later tried to address this issue by extending AMVA methods to the queue-dependent case [28,29], but the resulting methods are brittle and difficult to implement, not really addressing the problem. For this reason, several other works have proposed approximations based on non-AMVA approaches, such as asymptotic expansions [30,31], throughput bounds [32], and probabilistic methods [33].

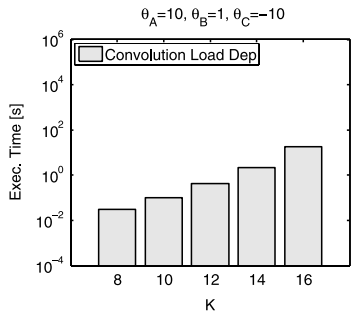
The QD-AMVA method we propose differs from existing techniques as it encompasses a much wider class of queue-dependence, where service requirements of class r at station k can depend on both $n_{k,r}$ and n_k, not just on n_k as in the above works. Existing approaches for open load-dependent queueing systems focus on the case of a single station with a single server and single-class customers, which is analyzed by means of Markovian processes for either the queue lengths or the waiting times. Instead we focus on the case of a closed queueing network made of many stations, which we analyze iteratively by means of an AMVA algorithm especially designed for queue-dependent services. Furthermore, our method is straightforward to implement, provides guarantees on existence and uniqueness of the solutions, and delivers accurate results. Finally, since our technique operates directly on mean values, its computational costs are $O(MR)$, for a model with M stations and R classes, and thus independent of the population size K. Conversely, all existing methods for multiclass models need to recursively or iteratively evaluate at least $O(MRK)$ state probabilities. This makes the overall cost of exact analysis of queue-dependent models $O(MRK^{R+1})$ in the total population size depending on the solution approach [16].

3.2. Running case

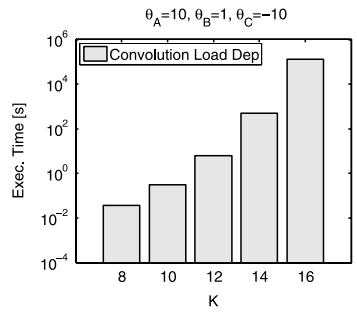
To illustrate our results throughout the paper, we introduce a running case. We consider a small closed network composed of $M = 2$ tandem queues; station 1 is an infinite server (i.e., a $-/GI/\infty$ station), station 2 is a processor-sharing queue. There are $R = 2$ workload classes, with identical population sizes $K_1 = K_2 = K/2$. The mean processing time of a class-1 request is $\theta_{1,1} = 90.0$ at station 1, and $\theta_{2,1} = 1.0$ at station 2; similarly, for class-2 requests we set $\theta_{1,2} = 90.0$ and $\theta_{2,2} = 1.0$.



(a) Approximation error for $x_{1,1}$ if queue-dependence at the processor sharing station 2 is ignored.



(b) Execution times (s) — 2 classes.



(c) Execution times (s) — 4 classes.

Fig. 1. Running case — $x^{\text{dep}}_{1,1}$ and $x^{\text{indep}}_{1,1}$ refer respectively to the mean queue-length $x_{1,1}$ in queue-dependent and load-independent models.

As the stations are assumed to be processor-sharing, we can model the infinite-server station by defining $\gamma_1(n_1) = n_1^{-1}$. Thus, we use this representation in the running case. For the processor-sharing queue, we assume that jobs of class 1 execute sequentially and place a queue-dependent demand that is captured by a polynomial function $\beta_{2,1}(n_{2,1}) = \theta_A n_{2,1}^2 + \theta_B n_{2,1} + \theta_C$, where θ_A, θ_B, θ_C are scalar parameters that we vary to explore the sensitivity of the results. The requirement $\beta_{2,1}(1) = 1$ implies $\theta_C = 1 - \theta_A - \theta_B$. All queue-dependent functions other than $\gamma_1(n_1)$ and $\beta_{2,1}(n_{2,1})$ are set equal to 1 for all inputs.

3.2.1. Impact of queue-dependence

To illustrate the large impact that queue-dependence can have in performance predictions, we illustrate the approximation error incurred by ignoring queue-dependence at station 2. Fig. 1(a) shows numerical results obtained by evaluating exactly with the convolution algorithm in [26] the running case (*dep*) and its load-independent variant where we ignore the queue-dependence at station 2 (*indep*). In all experiments, we assume $\theta_A = 10$ and $\theta_B = 1$. The results indicate that ignoring queue-dependence yields a very large error, above 1000% in some cases. Conversely, in low and heavy load, this network is well-approximated by a load-independent model. This overall indicates that, even for a small model, the practical implication of ignoring queue-dependence can be significant.

Furthermore, while in this example it is possible to span a range of populations from low to heavy load, in more realistic models with several queues and classes, the scalability of the convolution algorithm is limited. This is already apparent in Fig. 1(b). For $K = 64$, the convolution algorithm takes 0.4 s to complete on a commodity desktop,[2] but for $K = 256$ the algorithm takes 17.6 s to complete and returns an incorrect answer due to the normalizing constant exceeding the floating-point range. In general, with R classes, the convolution algorithm takes $O(MRK^{R+1})$ to complete; for instance, adding to the running case two new classes identical to class 2, so that the model has $R = 4$ classes, the completion time for $K = 64$ jobs of the convolution algorithm grows from 0.2 s to 490 s, as depicted in Fig. 1(c). Further increasing the total population to $K = 128$ requires an execution time of over 36 h, making the analysis impractical, especially for optimization studies that may require the evaluation of thousands of models.

4. Approximate analysis

4.1. Approximating mean performance

We begin by deriving some foundational results for the QD-AMVA algorithm. Our main finding is that, using first-order Taylor approximations, it is possible to obtain a novel recurrence equation operating only on the mean values of the performance metrics. To apply this method, we assume that the scaling factors $\beta_{k,r}(n_{k,r})$ and $\gamma_k(n_k)$, in addition to being bounded and strictly positive, are differentiable functions in the range $1 \leq n_{k,r} \leq K_r$ and $1 \leq n_k \leq K$, respectively. Notice that the following results involve the mean queue-lengths x_k and $x_{k,r}$, instead of the discrete variables n_k and $n_{k,r}$, which are real quantities in the range $1 \leq x_{x,r} \leq K_r$ and $1 \leq x_k \leq K$, respectively. The statement is given for the general case of queue-dependent scaling factors $D_{k,r}(\boldsymbol{n}_k)$, from which the specialization to the scalings $D_{k,r}(n_k, n_{k,r})$ is immediate.

Theorem 1. *Consider a queue-dependent closed queueing network where the real-valued functions $D_{k,r}(\cdot)$ are differentiable on $\{\boldsymbol{x} \in \mathbb{R}^{M\times R} | x_{k,r} \geq 0, \sum_{k=1}^{M} x_{k,r} = K_r, \sum_{r=1}^{R}\sum_{k=1}^{M} x_{k,r} = K\}$. Then, the first-order Taylor approximation of the mean queue-lengths at population $\boldsymbol{K}$ is*

$$x_{k,r} = T_r D_{k,r}(\mathbf{1}_r + \boldsymbol{x}_k(\boldsymbol{K} - \mathbf{1}_r))(1 + x_k(\boldsymbol{K} - \mathbf{1}_r)), \tag{5}$$

for all stations $1 \leq k \leq M$ and classes $1 \leq r \leq R$.

[2] The running case experiments have been performed on a desktop with 16GB RAM and an Intel Core i7-2600 CPU at 3.40 GHz.

The proofs of this theorem and the other results in the paper are provided in the Appendix. From this result, we readily have that under product-form scaling factors the first-order expansion of the mean queue-lengths becomes

$$x_{k,r} = T_r\theta_{k,r}\beta_{k,r}(1 + x_{k,r}(\boldsymbol{K} - \boldsymbol{1}_r))\gamma_k(1 + x_k(\boldsymbol{K} - \boldsymbol{1}_r))(1 + x_k(\boldsymbol{K} - \boldsymbol{1}_r)), \tag{6}$$

for all stations $1 \leq k \leq M$ and classes $1 \leq r \leq R$. By Little's law, the mean response time of class-r requests at station k is then $W_{k,r} = \frac{x_{k,r}}{T_r}$. Exploiting the population constraints $\sum_{k=1}^{M} x_{k,r} = K_r$, $1 \leq r \leq R$, it is useful to note that Little's law also implies $T_r = K_r(\sum_{i=1}^{M} W_{i,r})^{-1}$. Similarly, the utilization will be $U_{k,r} = T_r\mathbb{E}_{n_{k,r}\geq 1}[D_{k,r}(\boldsymbol{n})]$, where the expectation $\mathbb{E}_{n_{k,r}\geq 1}[\cdot]$ is defined for states $\boldsymbol{n}$ where $n_{k,r} \geq 1$. As we assume that $D_{k,r}(\boldsymbol{n})$ are differentiable functions, then the utilization may be similarly approximated by a first-order Taylor approximation as $U_{k,r} \approx T_r D_{k,r}(\boldsymbol{x}) = T_r\theta_{k,r}\beta_{k,r}(1 + \delta_r x_{k,r})\gamma_k(1 + \delta x_k)$.

4.1.1. Running case

Using Theorem 1, we consider the running case and compare against the convolution method, to determine the errors introduced by the approximate analysis. For $K = 64, R = 2, \theta_A = 10$ and $\theta_B = 1$, we find that the approximate mean queue-lengths obtained by the above first-order approximations are $x_{1,1} = 0.2011, x_{1,2} = 21.7008, x_{2,1} = 31.7989, x_{2,2} = 10.2992$. The exact values for these mean queue-lengths obtained by the convolution algorithm are $x_{1,1} = 0.2010, x_{1,2} = 21.7008, x_{2,1} = 31.7990, x_{2,2} = 10.2992$, thus the first-order approximation has negligible error. Varying the population from $K = 2$ to $K = 64$, on average the worst-case error on $x_{k,r}$ is just 1.45% of the total class-r population.

4.2. AMVA for queue-dependent models

Using Theorem 1 it is simple to develop an AMVA algorithm. We use the following AMVA approximations for the per-class and aggregate mean queue-lengths

$$x_{k,r}(\boldsymbol{K} - \boldsymbol{1}_s) = \begin{cases} \delta_r x_{k,r}, & \forall k, r, s : s = r, \\ x_{k,r}, & \forall k, r, s : s \neq r, \end{cases} \tag{7}$$

$$x_k(\boldsymbol{K} - \boldsymbol{1}_s) = \delta x_k, \quad \forall k, s, \tag{8}$$

where $x_{k,r} \equiv x_{k,r}(\boldsymbol{K})$, $\delta_r = (K_r - 1)K_r^{-1}$, and $\delta = (K-1)K^{-1}$. It should be noted that while the approximation for $x_{k,r}(\boldsymbol{K} - \boldsymbol{1}_s)$ is standard and corresponds to the one used by the Bard–Schweitzer AMVA algorithm [34,2], the approximation used for $x_k(\boldsymbol{K} - \boldsymbol{1}_s)$ is non-standard and we adopt it to interpolate the mean queue-length in a class-independent manner to ensure the uniqueness guarantees for the resulting AMVA solution, as we discuss later in Section 5.

Using (7)–(8), we can rewrite (6) as

$$x_{k,r} = T_r\theta_{k,r}\beta_{k,r}(1 + \delta_r x_{k,r})\gamma_k(1 + \delta x_k)(1 + \delta x_k), \tag{9}$$

for $1 \leq k \leq M$, $1 \leq r \leq R$. Exploiting that the population is constant, i.e., $\sum_k x_{k,r} = K_r$, we can obtain T_r and write the *MR* equations (9) as a non-linear system

$$x_{k,r} = \frac{K_r\theta_{k,r}\beta_{k,r}(1 + \delta_r x_{k,r})\gamma_k(1 + \delta x_k)(1 + \delta x_k)}{\sum_{i=1}^{M} \theta_{i,r}\beta_{i,r}(1 + \delta_r x_{i,r})\gamma_i(1 + \delta x_i)(1 + \delta x_i)}, \tag{10}$$

$1 \leq k \leq M$, $1 \leq r \leq R$, where $x_i = \sum_{s=1}^{R} x_{i,s}$. This is a set of *MR* non-linear equations in the *MR* unknowns $x_{k,r}$. Existence and uniqueness of the solutions to this system of equations are discussed in Section 5.

Starting from a random, but feasible, initial guess, the non-linear system may be solved for the mean queue-lengths $x_{k,r}$ using the successive substitutions method, as in established AMVA algorithms. This involves seeking for a fixed point of a continuous mapping

$$x_{k,r}^{n+1} = \frac{K_r\theta_{k,r}\beta_{k,r}(1 + \delta_r x_{k,r}^n)\gamma_k(1 + \delta x_k^n)(1 + \delta x_k^n)}{\sum_{i=1}^{M} \theta_{i,r}\beta_{i,r}(1 + \delta_r x_{i,r}^n)\gamma_i(1 + \delta x_i^n)(1 + \delta x_i^n)}, \tag{11}$$

where $x_i^n = \sum_{s=1}^{R} x_{i,s}^n$ and $n \geq 1$ is the iteration index. A pseudo-code summarizing the procedure is shown in Algorithm 1 in the Appendix. The algorithm uses an initial guess for the queue-length $x_{k,r}$ proportional to $\theta_{k,r}$, but other initializations may be considered.

It should be mentioned that known cases exist, in the mathematical literature, where the successive substitutions method does not converge on well-behaved mappings. However, we are not aware of this behavior having been observed before in existing AMVA methods [2]. Using QD-AMVA, we have evaluated thousands of models, as documented in Section 7, and we have never experienced a failure of the successive substitution method to converge. Therefore, while it remains possible that the method may not converge in some instances, the impact of this seems likely to affect only contrived examples. For such instances, one may consider adopting a different approach to solve the AMVA system (10), such as Newton-type methods that can provide guaranteed convergence under some assumptions [35].

4.3. Running case

For the running case introduced in Section 3.2, we consider an instance with $K = 64$ jobs, $R = 2$ classes, $\theta_A = 10$, and $\theta_B = 1$. We find the approximate queue lengths with the method introduced in this section, and compare against the exact results obtained with the convolution algorithm. The maximal relative error obtained with QD-AMVA is just 0.17% of the per-class populations. The AMVA algorithm returns this solution after 12 iterations in just 2 *ms*, which in this small example is already about 35 times faster than convolution. AMVA memory requirements are negligible, below 1MB. Considering again the scenario with $K = 64$ jobs and $R = 4$ classes discussed in Section 3.2, the AMVA algorithm requires 13 iterations, with a completion time of 3 ms instead of the 1341 s required by the convolution algorithm.

5. Characterization

We now focus on characterizing existence and uniqueness of the solutions for the AMVA system (10). This is a relevant problem, since the generality of the scaling factors makes it possible for the domain of the solution of (10) to be non-convex and therefore multiple feasible solutions may be possible for the AMVA equations. In the following we focus on the $K > 1$ case, as in the single-job case ($K = 1$) the AMVA equations clearly have a single solution. To see this, recall that $\delta = (K-1)K^{-1}$ and $\delta_r = (K_r - 1)K_r^{-1}$, thus $K = 1$ implies $K_r = 1$ and $\delta = \delta_r = 0$. Further, from the definition of $\beta_{k,r}(\cdot)$ and $\gamma_k(\cdot)$ we know that $\beta_{k,r}(1) = \gamma_k(1) = 1$. Replacing in (10), we obtain $x_{k,r} = \theta_{k,r} / \sum_{i=1}^{M} \theta_{i,r}$, which is the only solution to this system.

In addition, to avoid unnecessary notation complexity, in the following we assume that $\theta_{k,r} > 0$ for every $1 \leq k \leq M$, $1 \leq r \leq R$. All the results can be generalized to the case where some $\theta_{k,r} = 0$ as in this case the vector of mean queue-lengths $\boldsymbol{x}$ only needs to consider (k, r) tuples such that $\theta_{k,r} > 0$, as all others are simply zero.

5.1. Existence of solutions

In order to prove existence, we first prove that the mean queue-lengths $x_{k,r}$ obtained from the AMVA system are strictly positive, whenever the demands $\theta_{k,r}$ are positive.

Proposition 1. *Let $\boldsymbol{x}$ be a solution of the AMVA equations (10) such that $0 \leq x_{k,r} \leq K_r$, $0 \leq x_k \leq K$. Then the solution $\boldsymbol{x}$ must be strictly positive, with $x_{k,r} \geq x^{-}_{k,r} > 0$, where*

$$x^{-}_{k,r} = \frac{K_r \theta_{k,r} \min\limits_{1 \leq u_r \leq K_r} \beta_{k,r}(u_r) \min\limits_{1 \leq u \leq K} \gamma_k(u)}{K \sum\limits_{j=1}^{M} \theta_{j,r} \max\limits_{1 \leq u_r \leq K_r} \beta_{j,r}(u_r) \max\limits_{1 \leq u \leq K} \gamma_j(u)},$$

$1 \leq k \leq M$, $1 \leq r \leq R$. Moreover, if $K > 1$, $x_{k,r} > x^{-}_{k,r}$.

We are now ready to prove existence of solutions using the Brouwer fixed point theorem.

Theorem 2. *Under the assumptions of Theorem 1, and given the definitions in Proposition 1, the continuous mapping (10) maps each point in the compact set*

$$\mathcal{B} = \left\{ \boldsymbol{x} \in \mathbb{R}^{M \times R} : \sum_{k=1}^{M} x_{k,r} = K_r; x_{k,r} \geq x^{-}_{k,r}, \forall k, r \right\}, \tag{12}$$

to $\mathcal{B}$ itself. Thus, (10) has at least one fixed point $\boldsymbol{x}^* \in \mathcal{B}$.

It is possible to show examples where, even in the basic load-independent case, the non-linear system (10) admits in general multiple solutions if initialized at arbitrary real points $\boldsymbol{x} \notin \mathcal{B}$. However, this does not happen if the initial point is in $\mathcal{B}$. From now on, we therefore always assume that the initial point used in the AMVA system (10) belongs to $\mathcal{B}$, which is the case for the initial point used in Algorithm 1.

5.2. Optimization-based reformulation

With the goal of characterizing the number of solutions of the AMVA equations, we now introduce a reformulation of the nonlinear system (10) as an optimization program. Recall that $x_k = \sum_{s=1}^{R} x_{k,s}$, and introduce the following function

$$f(\boldsymbol{x}) = \sum_{k=1}^{M} \sum_{r=1}^{R} \int_0^{x_{k,r}} \log\left(\frac{u_{k,r}}{\theta_{k,r} \beta_{k,r}(1 + \delta_{k,r} u_{k,r})} \right) du_{k,r} - \sum_{k=1}^{M} \int_0^{x_k} \log\left((1 + \delta u_k) \gamma_k (1 + \delta u_k) \right) du_k, \tag{13}$$

for $\boldsymbol{x} \in \mathcal{B}$ and $\mathcal{B}$ defined as in (12). Assuming the scaling factors are twice differentiable, then $f(\boldsymbol{x})$ is also twice differentiable in $\mathcal{B}$, since the first summation in (13) is on strictly positive values of $x_{k,r} \geq \bar{x_{k,r}} > 0$ and thus the first derivative does not require to evaluate logarithms at zero. The key property of $f(\boldsymbol{x})$ is that

$$\frac{\partial f(\boldsymbol{x})}{\partial x_{k,r}} = \log\left(\frac{x_{k,r}}{\theta_{k,r}\beta_{k,r}(1+\delta_r x_{k,r})\gamma_k(1+\delta x_k)(1+\delta x_k)}\right),$$

which by (9) may be seen as the logarithm of the AMVA throughput T_r. It should be noted that this relationship exists just because we interpolate the total queue-length in a class-independent manner, i.e., $x_k(\boldsymbol{K}-\mathbf{1}_r) \approx 1+\delta x_k$, where δ does not depend on r. This, together with the dependence of the scaling factors $D_{k,r}(n_{k,r}, n_k)$ only on the random variables $n_{k,r}$ and n_k, simplifies the expression of the partial derivative and allows us to draw a connection between $f(\boldsymbol{\theta})$ and the AMVA throughput T_r.

Stemming from this connection, we can now prove that the AMVA solutions satisfy the necessary conditions to be stationary points for a minimization of $f(\boldsymbol{x})$ in $\mathcal{B}$, which are provided by the first-order Karuhn–Kuhn–Tucker (KKT) conditions [35].

Lemma 1. *If all the queue-dependent functions $\beta_{k,r}(1+\delta_r x_{k,r})$ and $\gamma_k(1+\delta x_k)$ are twice differentiable in $\mathcal{B}$, then the solutions of the AMVA equations (10) in $\mathcal{B}$ satisfy the first-order KKT conditions for the constrained nonlinear program*

$$\min_{\boldsymbol{x}\in\mathcal{B}} \quad f(\boldsymbol{x}). \tag{14}$$

5.2.1. Specialization to load-independent models

An important property of (14) for load-independent models is that the class-independent interpolation of $x_{k,r}(\boldsymbol{K}-\mathbf{1}_r)$ allows us to prove strict convexity of the minimization program in this case. This readily implies uniqueness of the solution.

Theorem 3. *For a load-independent model, the minimization program (14) has a unique solution, and therefore the QD-AMVA equations (10) have a unique solution in $\mathcal{B}$.*

Theorem 3 implies that for load-independent models with finite populations, a solution to the AMVA equations exists and is unique in the feasible set $\mathcal{B}$. To the best of our knowledge, uniqueness of AMVA results in the sub-asymptotic case is established for throughputs [36,37], but has not been proved before for queue-lengths ([37], Table 1, Fixed Point, case $J > 1$).

5.2.2. Uniqueness of solutions for queue-dependent models

Stemming from these results, we are now able to prove the main characterization result for the solutions of the AMVA system (10).

Theorem 4. *If all queue-dependent functions $\beta_{k,r}(1+\delta_r x_{k,r})$ and $\gamma_k(1+\delta x_k)$ are twice differentiable in $\mathcal{B}$, and respectively non-increasing in $0 \leq x_{k,r} \leq K_r$ and $0 \leq x_k \leq K$, then the QD-AMVA equations (10) have a unique solution in $\mathcal{B}$.*

The significance of Theorem 4 is that, when combined with the results in the previous sections, it sheds light on the guarantees that QD-AMVA offers. For queue-dependent models that satisfy the assumptions of Theorem 4, the fixed point exists and it is unique. This class of models is broad. For example, non-increasing queue-dependent functions are a natural way to express parallelism in the service at a queue, see Section 7.2.

It is also important to note that for queue-dependent models that do *not* satisfy the assumptions of the theorem, Theorem 4 does *not* exclude the existence of a unique solution in $\mathcal{B}$. For example, despite using an exhaustive search based on homotopy continuation [38], we have been unable to find for the running case multiple solutions in $\mathcal{B}$, even though it does not satisfy the monotonicity assumptions of Theorem 4. More generally, despite having evaluated by homotopy continuation thousands of queue-dependent models, we have never observed the case where AMVA has multiple solutions in $\mathcal{B}$ for a given model. Our investigation has been primarily focused on models with polynomial and rational queue-dependent functions, thus not excluding in principle that multiple solutions may arise in presence of other types of dependence functions. However, our experience provides circumstantial evidence that the issue may not arise frequently enough to represent a problem for applications of the QD-AMVA method.[3]

[3] In the hypothetical situation where one has to discriminate between multiple feasible solutions for QD-AMVA, the following heuristic might help in choosing a solution. Lemma 1 ensures that all such solutions are also feasible solutions for (14). This implies that we can score all such solutions $\boldsymbol{x}^*$ using the objective function $f(\boldsymbol{x}^*)$ and select the one that minimizes the objective function as the preferred solution. Ties may be resolved differently depending on the application area, for example in capacity planning one may select the most conservative solution according to some criteria other than the objective function (e.g., cost).

6. Further applications

6.1. Normalizing constant

We now consider the normalizing constant for a queue-dependent model, given by $G = \sum_{\boldsymbol{n}\in\mathcal{S}} \prod_{k=1}^{M} C_k(\boldsymbol{n}_k)F_k(\boldsymbol{n}_k)$. Recently, the authors of [39] have shown that the normalizing constant for a load-independent model can be computed recursively through a Taylor approximation and AMVA. The proposed method is based on the first-order Taylor approximation of $G(\boldsymbol{\theta} + \Delta\theta_{k,r}) \approx G(\boldsymbol{\theta}) + \frac{\partial G(\boldsymbol{\theta})}{\partial\theta_{k,r}}\Delta\theta_{k,r}$, exploiting the property that the derivative of $\log G$ with respect to $\theta_{k,r}$ depends only on the mean queue length $x_{k,r}$, which can be computed with load-independent AMVA methods. Then G can be iteratively evaluated along each dimension $\theta_{k,r}$ by gradually increasing it with a step size $\Delta\theta_{k,r}$. In addition, the initial value of G comes from a degenerate model for which G is trivial to compute. The approach has a computational complexity of $O(M^2R^2)$ per iteration using the Bard–Schweitzer algorithm to estimate $x_{k,r}$. Compared to the convolution algorithm, which has $O(M\prod_{r=1}^{R} K_r)$ complexity, this approach is significantly more efficient since normally K is much larger than M and R. It is known that the above-mentioned property of the derivative of $\log G$ is true for both load-independent [40] and load-dependent models [33], but we are not aware of this problem having been considered before for queue-dependent models, which also include dependence on the number of jobs for each class. We now show that this property holds also in this setting.

Theorem 5. *For a closed queueing network model with product-form demands*

$$\frac{\partial \log G}{\partial\theta_{k,r}} = \frac{x_{k,r}}{\theta_{k,r}}, \tag{15}$$

for all stations $1 \le k \le M$ *and classes* $1 \le r \le R$.

Therefore, we can combine QD-AMVA with the method in [39] to approximate G. A pseudo-code summarizing the required approximation steps is given in the Appendix, together with an assessment on our running case.

6.2. Sensitivity analysis

The definitions given in Section 2.1.1 suggest that the derivatives of the performance measures with respect to an arbitrary parameter of the model require to differentiate G. This need commonly arises in optimization-driven search, which finds application in load balancing problems and maximum-likelihood estimation. In this section, we prove that these gradients can be accurately approximated in terms of mean queue-lengths. This means that the QD-AMVA method can be directly applied for computing the expression of the gradients, thus avoiding the need of probabilistic analysis. This is especially important for the integration of QD-AMVA in numerical optimization programs, where accurate computation of the Jacobian matrix through explicit expressions is preferable to finite-difference schemes.

Let us assume that the queue-dependent functions may be expressed in terms of a parameter set $\boldsymbol{\theta}_{k,r} = (\theta_{k,r,1}, \ldots, \theta_{k,r,p}, \ldots, \theta_{k,r,P})$, such that $D_{k,r}(\boldsymbol{n}_k) \equiv D_{k,r}(n_{k,r}, n_k) \equiv D_{k,r}(n_{k,r}, n_k, \boldsymbol{\theta}_{k,r})$ is determined by an arbitrary set of P real-valued parameters. For example, these may be coefficients of a polynomial defining a queue-dependent function. We assume $F_i(\boldsymbol{n}_i)$ to be independent of $\theta_{k,r,p}$ for $i \neq k$, and otherwise we assume $F_k(\boldsymbol{n}_k)$ to be dependent on $\theta_{k,r,p}$ unless station k has no jobs of class r (i.e., $n_{k,r} = 0$).

We now extend (15) to cover sensitivity with respect to an arbitrary parameter $\theta_{k,r,p} \in \boldsymbol{\theta}_{k,r}$. Notice that, thanks to the identity $\partial \log G/\partial\theta_{k,r,p} = 1/G\ \partial G/\partial\theta_{k,r,p}$, computing the logarithmic derivative $\partial \log G/\partial\theta_{k,r,p}$ is enough to obtain the derivative $\partial G/\partial\theta_{k,r,p}$.

Theorem 6. *For a function* $g(\boldsymbol{x})$*, let* $S_{x_i}[g(\boldsymbol{x})] = \frac{x_i}{g(\boldsymbol{x})}\frac{\partial g(\boldsymbol{x})}{\partial x_i}$ *be the elasticity of* $g(\boldsymbol{x})$ *with respect to variable* x_i*. Then the logarithmic derivative of the normalizing constant with respect to a parameter* $\theta_{k,r,p}$ *is*

$$\frac{\partial \log G}{\partial\theta_{k,r,p}} = \theta_{k,r,p}^{-1}\mathbb{E}_{n_{k,r}\ge 1}\left(\sum_{u=0}^{n_{k,r}-1} S_{\theta_{k,r,p}}\left[D_{k,r}(\boldsymbol{n}_k - u\mathbf{1}_r)\right]\right),$$

where $\mathbb{E}_{n_{k,r}\ge 1}(\cdot)$ *denotes expectation over states* $\boldsymbol{n}$ *with* $n_{k,r} \ge 1$.

The above expression can be approximated in terms of $x_{k,r}$ in two ways. If the product-form queue-dependent functions allow to explicitly compute the summation over the elasticities, a first-order Taylor approximation may be applied to the result if it is a continuous function. We show one such instance below in Section 9. Conversely, define $g(u) = S_{\theta_{k,r,p}}[D_{k,r}(\boldsymbol{n}_k - u\mathbf{1}_r)]$, then by the asymptotic Euler–Cauchy formula

$$\sum_{u=0}^{n_{k,r}-1} g(u) \approx \frac{g(0) + g(n_{k,r}-1)}{2} + \int_{u=0}^{n_{k,r}-1} g(u)du,$$

Table 1
Parameters.

$\theta_{k,r} = \text{Unif.Integer}(1, 100)$
$M \in \{2, 3, 4\}$
$R \in \{1, 2, 4\}$
$K \in \{R, 2R, 4R\}$
$K_r/K = 1/R$
$\beta_{k,r}(\cdot) \equiv 1, \forall (k, r) \neq (1, 1)$
$\gamma_k(\cdot) \equiv 1, \forall k$

and applying a first-order Taylor approximation to the expectation $\mathbb{E}_{n_{k,r}\geq 1}(\cdot)$ we find with the AMVA queue-length approximation

$$\frac{\partial \log G}{\partial \theta_{k,r,p}} \approx \theta_{k,r,p}^{-1} \left(\frac{g(0) + g(\delta_r x_{k,r})}{2} + \int_{u=0}^{\delta_r x_{k,r}} g(u)du \right). \tag{16}$$

Based on this first-order approximation, we can readily approximate the sensitivity of performance measures using the QD-AMVA results. For example, differentiating $\log T_r$, we find

$$\frac{\partial \log T_r}{\partial \theta_{k,r,p}} = T_r \left(\frac{\partial \log G(\boldsymbol{K} - \mathbf{1}_r)}{\partial \theta_{k,r,p}} - \frac{\partial \log G}{\partial \theta_{k,r,p}} \right).$$

Using the definitions in Section 2.1.1, similar expressions can be generated for other performance measures. An example of this analysis is provided in the Appendix for the running case.

7. Validation

We now describe the validation methodology used to evaluate the QD-AMVA algorithm.

7.1. Random models

We have first evaluated the QD-AMVA algorithm under a range of randomly generated model instances. Our evaluation covers about 11,000 models, for different choices of the number of queues, classes, job populations, and queue-dependent functions. Table 1 summarizes the choices of parameters used in the random model generation. A few remarks are needed:

- For each choice of M, R, K, and queue-dependent functions, we re-run the experiment 50 times with different sets of random demands $\theta_{k,r}$. The same assignment of the demands is considered for increasing values of the population K.
- For ease of interpretation, we consider a single service demand to be queue-dependent, i.e., for station $k = 1$ and class $r = 1$. In particular, we assign $\beta_{1,1}(\cdot)$ to be queue-dependent, since this is generally more challenging than assigning $\gamma_1(\cdot)$, which would not depend on the per-class populations. All other queues and classes are load-independent.
- For each choice of queue-dependent function $\beta_{1,1}(\cdot)$, we also run experiments for its reciprocal $\beta_{1,1}^{-1}(\cdot)$. This means that our experiments consider both increasing and decreasing functions in the same proportions.
- Models are either small-sized or medium-sized, since we need to consider a number of jobs and classes that allow us to solve the model *exactly* using the convolution algorithm. Larger models can only be evaluated approximately, due to the excessively high cost of the convolution algorithm, thus complicating the error assessment for the AMVA method. We evaluate some large models in Section 7.2.

The QD-AMVA algorithm is implemented with tolerance $\tau = 10^{-6}$ and no bounds on the maximum number of iterations. Let $x_{k,r}^{exact}$ be the mean number of jobs for class r at station k in the exact solution and let $x_{k,r}^{approx}$ be the corresponding approximation computed by QD-AMVA. To assess the performance of the AMVA method, we consider these metrics:

$$\epsilon_{avg} = \frac{1}{MR} \sum_{k,r} \frac{|x_{k,r}^{exact} - x_{k,r}^{approx}|}{K_r}, \qquad \epsilon_{max} = \max_{k,r} \frac{|x_{k,r}^{exact} - x_{k,r}^{approx}|}{K_r}. \tag{17}$$

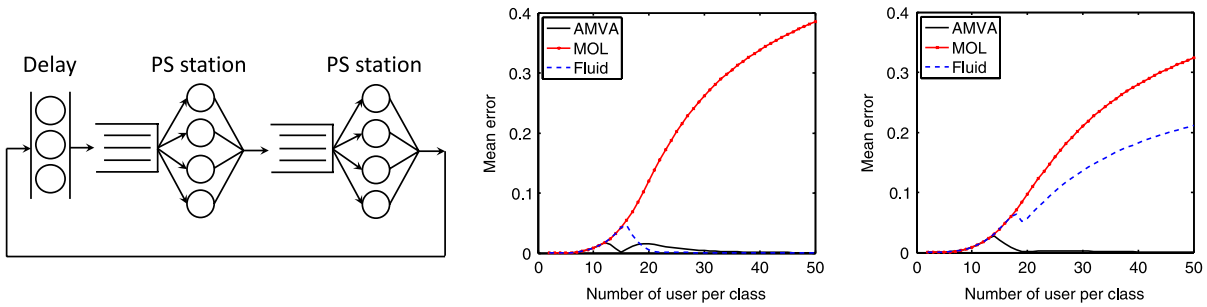
Therefore ϵ_{avg} is the average fraction of jobs, relatively to the class population, that is misplaced in the AMVA solution compared to the exact one determined by the convolution algorithm; ϵ_{max} is similarly defined as the maximum misplaced fraction of jobs.

Validation results

Table 2 summarizes the results, which are aggregated based on the choice of the queue-dependent function $\beta_{1,1}(\cdot)$. The results indicate that the QD-AMVA has low errors in all cases, always between 1.5% and 6%. An interesting pattern that emerges from the experimental results is that the number of iterations and the errors are generally lower for functions that are non-increasing. These correspond to functions that satisfy the uniqueness conditions of Theorem 4. We also see that the non-increasing functions have a number of iterations that is generally in the range 8–10, whereas for increasing functions this is in the range of 10–12 iterations. Coupled with Theorem 4, this finding strengthens the observation that non-increasing queue-dependent functions are better suited to use in queue-dependent models. In these experiments the instance with the longest execution time required 200 ms, while the average execution time was under 10 ms.

Table 2
Results for random models.

	$\beta_{1,1}(x)$			$\beta_{1,1}^{-1}(x)$		
	ϵ_{avg}	ϵ_{max}	iterations	ϵ_{avg}	ϵ_{max}	iterations
x	0.020	0.040	12.7	0.015	0.031	8.3
$x(x-\frac{1}{2})$	0.029	0.059	12.6	0.019	0.037	9.8
$x(x+\frac{1}{2})$	0.024	0.049	12.9	0.016	0.032	9.9
$x(x^2+x+\frac{5}{4})$	0.023	0.048	12.0	0.016	0.034	8.8
$\min(x, 2)$	0.020	0.039	13.4	0.016	0.032	8.8
$\log(e+x-1)$	0.017	0.034	12.1	0.015	0.030	9.5
e^x	0.023	0.046	12.1	0.016	0.033	8.1



(a) Test model – $m_1 = m_2 = 4$. (b) $\theta_{2,1} = 0.1$, $\theta_{2,2} = 0.2$. (c) $\theta_{2,1} = 1$, $\theta_{2,2} = 2$.

Fig. 2. Comparing AMVA to a fluid limit and MOL.

7.2. Comparison with other methods

We now compare the performance of QD-AMVA with fluid approximation methods [41]. We focus on a specific type of queue-dependence, which describes well processor-sharing multi-server stations, and fits within the density-dependent models that enable a fluid approximation in the sense of [41]. Assuming m_k servers in station k, the mean demand of class-r jobs at station k is given by $D_{k,r}(n_k, n_{k,r}) = \theta_{k,r} n_k (\min\{n_k, m_k\})^{-1}$, such that $\beta_{k,r}(n_{k,r}) = 1$ and $\gamma_k(n_k) = \frac{n_k}{\min\{n_k, m_k\}}$. With this definition, when $n_k \leq m_k$, the job mean demand is $\theta_{k,r}$, while when $n_k > m_k$, this becomes $\theta_{k,r} n_k / m_k$. Notice that $\gamma_k(n_k)$ does not have a continuous derivative at $n_k = m_k$, but this function can be approximated arbitrarily close using the smooth *softmin* function

$$\hat{\gamma}_k(n_k) = \frac{n_k e^{\alpha n_k} + m_k e^{\alpha m_k}}{e^{\alpha n_k} + e^{\alpha m_k}} n_k, \tag{18}$$

which converges to $\gamma_k(n_k)$ as $\alpha \to -\infty$. We use the associated rate function to define a fluid limit [41], which is a system of non-linear differential equations, and use the fixed-point of this system, obtained numerically, to approximate the mean queue-lengths. We compare the mean queue-length obtained with the fluid limit with the QD-AMVA solution, using the tandem network in Fig. 2(a), which is made of one infinite server and two processor sharing stations, each with 4 servers, serving 2 job classes. We also set the parameter α in (18) to be equal to 20. The demand at the infinite server is set to $\theta_{3,1} = \theta_{3,2} = 10$ for both job classes, while the mean demands in the processor sharing station 1 are set to $\theta_{1,1} = 1$, and $\theta_{1,2} = 2$, and those in the processor sharing station 2 are $\theta_{2,1} = 0.1$, and $\theta_{2,2} = 0.2$. Fig. 2(b) depicts the mean error ϵ_{avg}, as defined in (17), obtained with the fluid and the QD-AMVA method for an increasing number of jobs of each class. Clearly, the QD-AMVA method offers smaller errors than the fluid for small populations, but it shows a slower convergence to the asymptotic solution than the fluid method. In Fig. 2(c) we modify the demands in the processor sharing station 2 to be equal to those of station 1, that is, $\theta_{2,1} = 1$, and $\theta_{2,2} = 2$. In this case, the fluid method has much larger errors than QD-AMVA, even for large populations. In fact, we have observed that in cases like this one, where the demands at the processing stations are balanced, the fluid shows very different errors depending on its initial conditions. The QD-AMVA method, instead, offers a similar performance than in the previous case, not being affected by the change in the service demands. Fig. 2 also compares QD-AMVA with the results obtained with the approximation used in the Method of Layers (MOL) [5] to model multi-server stations. Although the error is similar to the fluid and the AMVA for small populations, it becomes very large when the population increases. Similar trends to those in Fig. 2 can be observed for different choices of the number of servers at the two nodes.

Summarizing, this comparison highlights that QD-AMVA is generally more robust than two existing approximations. It consistently delivers low approximation errors, in particular on models with non-asymptotic populations that are difficult to analyze with fluid methods.

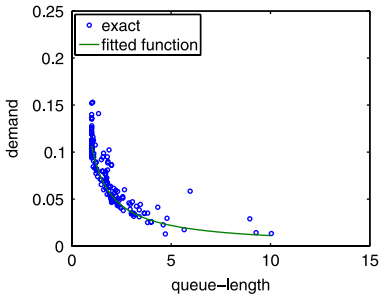


Fig. 3. Measured queue-dependent service.

Table 3
Demand θ_{kr} for the load balancing model in seconds.

VM	Class 1	Class 2	Class 3	Class 4
Inf. Server	0.10	0.20	0.40	0.80
VM1	0.50	2.28	1.29	2.66
VM2	0.26	1.23	0.69	1.44
VM3	0.16	0.77	0.44	0.91
VM4	0.11	0.52	0.30	0.61

8. Case study

To illustrate the benefits of considering queue-dependence when modeling real systems, we consider a load balancing problem for an e-commerce application deployed on the cloud. The deployment consists of a set of heterogeneous front servers, each of which contains an instance of both the application and database servers. These servers are deployed on Amazon EC2, using virtual machines (VMs) of different characteristics, e.g., CPU speed, memory. The application requests, submitted by an application-specific workload generator, are handled by a load-balancer, which forwards them according to a weighted round-robin policy. The problem under study is to define the routing weights that maximize the application throughput. A common approximation for this consists in formulating a probabilistic routing problem and using the resulting probabilities to configure the load-balancer weights. We thus formulate an optimization problem to determine the routing probabilities that maximize the application throughput, as in, for instance, [42]. The application servers are modeled as processor-sharing stations, considering both queue-dependent and load-independent descriptions.

To parameterize the queueing-network model, we have run a large number of experiments using a real-world application, Apache OFBiz,[4] observing that some of the application requests clearly show queue-dependent service times. This is exemplified in Fig. 3, where we depict the observed request service demands as a function of the queue-lengths. We fit the observed measurements using a power function, and obtain the monotone decreasing $\gamma_k(n_k) = n_k^{-0.9805}$ function leading to the trend in the figure.

Specifically, we consider an EC2 deployment with $M = 4$ frontend servers (VMs) and $R = 4$ session types (classes). These user classes are emulated with the workload generator, which submits a different sequence of application requests for each user class. The demand of each user class is obtained by summing up the demand for each request as profiled for each VM. In addition to the four processor-sharing stations that model the frontend servers, we add an infinite-server station to model the user think times. In the queueing model that we use to find the optimal load-balancing probabilities, we assume all the VMs are load-dependent, following the real scaling function depicted in Fig. 3. The mean demands for the load balancing model are shown in Table 3. The number of jobs of each class K_r is chosen in $\{5, 25, 50\}$ to consider different load levels.

Using the queueing-network model to evaluate a set of routing probabilities, we use MATLAB's fmincon solver to obtain the optimal routing probabilities of each class to each VM such that the overall throughput is maximized. We consider two runs of the solver and for each of them we adopt a multi-start approach to cope with the presence of multiple local optima. In the first run, we use QD-AMVA, as defined in Algorithm 1, to evaluate the throughput of the model for each feasible allocation of the routing probabilities. In the second run, we use a load-independent AMVA, the Bard–Schweitzer algorithm, parameterized just with the mean demands in Table 3, but without the scaling function $\gamma_k(n_k)$ that cannot be expressed in a load-independent model. After solving the optimization program, we use the obtained optimal routing probabilities within a discrete-event simulation tool and obtain the simulated throughput under the queue-dependent behavior. Simulation

[4] Apache OFBiz: http://ofbiz.apache.org/.

is required since the models are too large for exact analysis. The result is shown in Fig. 4, where QD stands for queue-dependent and LI stands for load-independent. By relying on QD-AMVA, the throughput-optimization problem achieves a marked improvement in overall throughput compared to the load-independent AMVA. This illustrates how explicitly considering queue-dependent service times can be exploited to improve the application performance.

9. Conclusion

This paper has proposed QD-AMVA, the first approximate mean value analysis (AMVA) algorithm that can evaluate closed queueing networks with queue-dependent nodes operating only on mean values. The algorithm does not need to compute state probabilities and therefore is numerically stable and its computational requirements are independent of the number of jobs in the system. Future work may explore the integration of QD-AMVA method in layered queueing network solvers, which are AMVA-based but are currently unable to deal with load-dependence, and applications to resource management at run-time.

Acknowledgments

The research presented in this paper has received funding from the Engineering and Physical Sciences Research Council (EPSRC) under grant agreement No. EP/M009211/1 (OptiMAM), from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 644869 (DICE), and from an Amazon Web Services (AWS) in Education Research Grant. This publication reflects only the authors' view and the European Commission is not responsible for any use that may be made of the information it contains. The data generated in this paper is available at http://wp.doc.ic.ac.uk/optimam/datasets.

Appendix

Pseudocode of the QD-AMVA algorithm

Algorithm 1 QD-AMVA Algorithm

Require: $M, K, R, K_r, \theta_{k,r}, \beta_{k,r}(\cdot), \gamma_k(\cdot)$, for $1 \leq r \leq R$, $1 \leq k \leq M$, $\tau =$ tolerance
  $\delta = K^{-1}(K-1)$, $\delta_r = K_r^{-1}(K_r - 1)$
  $n = 0$
  for $r = 1, \ldots, R$ **do**
    for $k = 1, \ldots, M$ **do**
      $x_{k,r}^0 = K_r \theta_{k,r} (\sum_{i=1}^{M} \theta_{i,r})^{-1}$,
    end for
  end for
  while $n < 1$ **or** $\max_{k,r}\{x_{k,r}^n - x_{k,r}^{n-1}\} > \tau$ **do**
    for $k = 1, \ldots, M$ **do**
      $x_k^n = \sum_{s=1}^{R} x_{k,s}^n$
    end for
    for $r = 1, \ldots, R$ **do**
      for $k = 1, \ldots, M$ **do**

$$x_{k,r}^{n+1} = \frac{K_r \theta_{k,r} \beta_{k,r}(1 + \delta_r x_{k,r}^n)\gamma_k(1 + \delta x_k^n)(1 + \delta x_k^n)}{\sum_{i=1}^{M} \theta_{i,r} \beta_{i,r}(1 + \delta_r x_{i,r}^n)\gamma_i(1 + \delta x_i^n)(1 + \delta x_i^n)}$$

      end for
    end for
    $n = n + 1$
  end while
  return Mean queue lengths $x_{k,r}^n$, for $1 \leq r \leq R$, $1 \leq k \leq M$

Normalizing constant approximation

The method, which has the pseudocode shown in Algorithm 2, initially evaluates the normalizing constant on a degenerate model with a single station. Then, it progressively increases the service demand values for the other stations, updating at each iteration the mean queue-length estimates. This can be done efficiently if one uses as initial point in Algorithm 1 the mean queue-length values obtained in the previous inner iteration of the algorithm, since small increases of service demand typically imply negligible changes in mean queue-lengths.

Algorithm 2 Normalizing constant approximation

Require: σ = step size
  Compute explicitly G for $M = 1$, i.e.,
  $\log G = \log K! + \sum_{r=1}^{R}\left(K_r \log \theta_{1,r} + \sum_{u=1}^{K_r} \log \beta_{1,r}(u)\right) + \sum_{u=1}^{K} \log \gamma_1(u) - \sum_{s=1}^{R} \log K_s!$
  $\theta'_{1,r} = \theta_{k,r}$, for $1 \leq r \leq R$
  $\theta'_{k,r} = 0$, for $2 \leq k \leq M$, $1 \leq r \leq R$
  for $k = 2 : M$ **do**
    for $r = 1 : R$ **do**
      while $\theta'_{k,r} < \theta_{k,r}$ **do**
        $\theta'_{k,r} = \theta'_{k,r} + \sigma$
        Compute $x_{i,c}$ by QD-AMVA using demands $\theta'_{k,r}$, $\forall k, r$
        $\log G = \log G + \log(1 + x_{i,c})\sigma\,(\theta'_{k,r} + \sigma)^{-1}$
      end while
    end for
  end for
  return $\log G$

Running case

We apply the approximation described above to the evaluation of the running case. We consider again the model for parameters $K = 64$, $\theta_A = 10$, $\theta_B = 1$. We apply the algorithm by computing first the value of $\log G$ on the processor sharing node, and then iterating on the demands of the infinite servers. The step size is set to $\sigma = 10^{-2}$. We obtain an approximate normalizing constant value of $1.9873 \cdot 10^{47}$, against an exact value of $2.0064 \cdot 10^{47}$, with a 0.95% error. The iterations require 25 s to complete and negligible memory usage. Importantly, if one neglects the queue-dependence at the processor sharing node, the normalizing constant becomes $G = 1.2062 \cdot 10^{36}$. This proves that combining QD-AMVA with the scheme in [39] effectively paves the way to computing normalizing constants in queue-dependent models. We are not aware of any other existing scheme that can deliver this result.

Sensitivity analysis

For illustration, we first consider the running case for $\theta_A = 0$, $K = 64$ and $\theta_B \in [2, 10]$, such that the queue-dependent function at the processor sharing node is now $\beta_{2,1}(n_{2,1}) = \theta_B n_{2,1} + 1 - \theta_B$. The elasticity with respect to θ_B is therefore

$$S_{\theta_B}(D_{2,1}(\boldsymbol{n}_2 - u\mathbf{1}_1)) = S_{\theta_B}(\beta_{2,1}(n_{2,1} - u)) = \frac{\theta_B(n_{2,1} - u - 1)}{\theta_B(n_{2,1} - u - 1) + 1}.$$

In the range of definition of θ_B, we can explicitly close the summation in Theorem 6 as

$$\frac{\partial \log G}{\partial \theta_B} = \theta_B^{-1}\mathbb{E}_{n_{2,1}\geq 1}\left(\sum_{u=0}^{n_{2,1}-1} S_{\theta_B}(\beta_{2,1}(n_{2,1} - u))\right) = \theta_B^{-1}\mathbb{E}_{n_{2,1}\geq 1}\left(n_{2,1} + \theta_B^{-1}\psi(1 - \theta_B^{-1}) - \theta_B^{-1}\psi(1 - n_{2,1} - \theta_B^{-1})\right),$$

where $\psi(\cdot)$ is the digamma function. Recall that we can write $\mathbb{E}_{n_{k,r}\geq 1}(n_{k,r}) = \mathbb{E}_r(1 + n_{k,r}) \approx 1 + \delta_r x_{k,r}$, where $\mathbb{E}_r$ is the expectation with respect to the probabilities $\boldsymbol{\pi}(\cdot|\boldsymbol{K} - \mathbf{1}_r)$. Since the summation can be expressed directly as a continuous function, we can immediately apply a first-order approximation without resorting to the Euler–Cauchy formula and write $\partial \log G/\partial\theta_B \approx \theta_B^{-1}\left(1 + \delta_1 x_{2,1} + \theta_B^{-1}\psi(1 - \theta_B^{-1}) - \theta_B^{-1}\psi(-\delta_1 x_{2,1} - \theta_B^{-1})\right)$. Fig. 5 compares the approximation against the exact values obtained by the convolution algorithm, showing a very good approximation accuracy. Here we focus on values above 1 since $\partial \log G/\partial\theta_B$ has a discontinuity at $\theta_B = 1$.

We now illustrate the approach with the Euler–Cauchy summation. For this we revert to the queue-dependence used in the previous examples, i.e., $\beta_{2,1}(n_{2,1}) = \theta_A n_{2,1}^2 + \theta_B n_{2,1} + 1 - \theta_A - \theta_B$ and set $\theta_A = 10$ and $K = 64$. We consider again the sensitivity with respect to θ_B. Following the definitions, we have that the integrand in the Euler–Cauchy formula is $g(u) = \theta_B(u - K_2 + 1)/(\theta_B - \theta_B(K_2 - u) - 10(K_2 - u)^2 + 9)$. Using (16), we find that the Euler–Cauchy approximation is $\partial \log G/\partial\theta_B \approx 0.27334$. Using convolution, we find an exact value of $\partial \log G/\partial\theta_B = 0.25295$, with a 8.1% error.

Proof of Theorem 1. We begin by observing that we can rewrite the definition of the mean queue-lengths as an expectation $\mathbb{E}_r[\cdot]$ defined over the probabilities $\pi(\cdot|\boldsymbol{K} - \mathbf{1}_r)$. Let $\mathcal{N} \equiv \mathcal{N}(k, r) = \{\boldsymbol{n} \in \mathcal{S}|1 \leq n_{k,r} \leq K_r\}$ and $\mathcal{N}' \equiv \mathcal{N}'(k, r) = \{\boldsymbol{n}' \in \mathcal{S}|0 \leq n'_{k,r} \leq K_r - 1\}$. Recalling that $\boldsymbol{n}_k$ is the kth row of $\boldsymbol{n}$, and using expression (4) for the mean queue-lengths, then

$$\begin{aligned} x_{k,r} &= \sum_{\boldsymbol{n}_k \in \mathcal{N}} n_k T_r D_{k,r}(\boldsymbol{n}_k)\pi(\boldsymbol{n}_k - \mathbf{1}_r|\boldsymbol{K} - \mathbf{1}_r) \\ &= T_r\left(\sum_{\boldsymbol{n}'_k \in \mathcal{N}'} D_{k,r}(\boldsymbol{n}'_k + \mathbf{1}_r)\pi(\boldsymbol{n}'_k|\boldsymbol{K} - \mathbf{1}_r) + \sum_{\boldsymbol{n}'_k \in \mathcal{N}'} n'_k D_{k,r}(\boldsymbol{n}'_k + \mathbf{1}_r)\pi(\boldsymbol{n}'_k|\boldsymbol{K} - \mathbf{1}_r)\right) \\ &= T_r\left(\mathbb{E}_r[D_{k,r}(\boldsymbol{n}'_k + \mathbf{1}_r)] + \mathbb{E}_r[n'_k D_{k,r}(\boldsymbol{n}'_k + \mathbf{1}_r)]\right). \end{aligned}$$

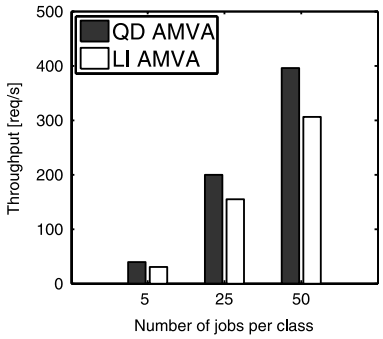


Fig. 4. Comparison between queue-dependent and load-independent AMVA results.

Using standard results for the expectation of a function of random variables, we can consider a first-order Taylor approximation of the last expression and write

$$\begin{aligned} x_{k,r} &= T_r\Big(\mathbb{E}_r[D_{k,r}(\boldsymbol{n}'_k+\mathbf{1}_r)]+\mathbb{E}_r[n'_k D_{k,r}(\boldsymbol{n}'_k+\mathbf{1}_r)]\Big)\\ &\approx T_r\Big(D_{k,r}(\mathbb{E}_r[\boldsymbol{n}'_k+\mathbf{1}_r])+\mathbb{E}_r[n'_k]D_{k,r}(\mathbb{E}_r[\boldsymbol{n}'_k+\mathbf{1}_r])\Big)\\ &= T_r D_{k,r}\Big(\mathbb{E}_r[\boldsymbol{n}'_k+\mathbf{1}_r]\Big)\Big(1+\mathbb{E}_r[n'_k]\Big) = T_r D_{k,r}\Big(\mathbf{1}_r+\boldsymbol{x}_k(\boldsymbol{K}-\mathbf{1}_r)\Big)\Big(1+x_k(\boldsymbol{K}-\mathbf{1}_r)\Big). \quad \square \end{aligned}$$

The last step follows since $\mathbb{E}_r$ is precisely the expectation based on the probabilities $\pi(\cdot|\boldsymbol{K}-\mathbf{1}_r)$.

Proof of Proposition 1. Since $0 \le x_{k,r} \le K_r$, $0 \le x_k \le K$, and $\beta_{k,r}(\cdot)$ and $\gamma_k(\cdot)$ are bounded and strictly positive in the range $1 \le u_{k,r} \le K_r$ and $1 \le u_k \le K$, respectively, we get

$$\begin{aligned} x_{k,r} &= \frac{K_r\theta_{k,r}\beta_{k,r}(1+\delta_r x_{k,r})\gamma_k(1+\delta x_k)(1+\delta x_k)}{\sum_{j=1}^{M}\theta_{j,r}\beta_{j,r}(1+\delta_r x_{j,r})\gamma_j(1+\delta x_j)(1+\delta x_j)} \ge \frac{K_r\theta_{k,r}\min\limits_{1\le u_r\le K_r}\beta_{k,r}(u_r)\min\limits_{1\le u\le K}\gamma_k(u)}{\sum_{j=1}^{M}\theta_{j,r}\max\limits_{1\le u_r\le K_r}\beta_{j,r}(u_r)\max\limits_{1\le u\le K}\gamma_j(u)(1+\delta x_j)}\\ &\ge \frac{K_r\theta_{k,r}\min\limits_{1\le u_r\le K_r}\beta_{k,r}(u_r)\min\limits_{1\le u\le K}\gamma_k(u)}{K\sum_{j=1}^{M}\theta_{j,r}\max\limits_{1\le u_r\le K_r}\beta_{j,r}(u_r)\max\limits_{1\le u\le K}\gamma_j(u)} = x^-_{k,r} > 0. \end{aligned}$$

The last inequality follows from the strict positivity assumed for $\beta_{k,r}(\cdot)$ and $\gamma_k(\cdot)$. The strict inequality $x_{k,r} > x^-_{k,r}$ follows now by revisiting the steps above to notice that

$$\begin{aligned} x_{k,r} &= \frac{K_r\theta_{k,r}\beta_{k,r}(1+\delta_r x_{k,r})\gamma_k(1+\delta x_k)}{\sum_{j=1}^{M}\theta_{j,r}\beta_{j,r}(1+\delta_r x_{j,r})\gamma_j(1+\delta x_j)(1+\delta x_j)}(1+\delta x_k)\\ &\ge \frac{K_r\theta_{k,r}\min\limits_{1\le u_r\le K_r}\beta_{k,r}(u_r)\min\limits_{1\le u\le K}\gamma_k(u)}{\sum_{j=1}^{M}\theta_{j,r}\max\limits_{1\le u_r\le K_r}\beta_{j,r}(u_r)\max\limits_{1\le u\le K}\gamma_j(u)(1+\delta x_j)}(1+\delta x_k)\\ &\ge \frac{K_r\theta_{k,r}\min\limits_{1\le u_r\le K_r}\beta_{k,r}(u_r)\min\limits_{1\le u\le K}\gamma_k(u)}{K\sum_{j=1}^{M}\theta_{j,r}\max\limits_{1\le u_r\le K_r}\beta_{j,r}(u_r)\max\limits_{1\le u\le K}\gamma_j(u)}(1+\delta x_k) = x^-_{k,r}(1+\delta x_k). \end{aligned}$$

Thus $x_{k,r} > x^-_{k,r}$ whenever $K > 1$, as in this case $\delta = (K-1)K^{-1} > 0$.

Proof of Theorem 2. Since the scaling factors are continuous and positive functions, (10) defines a continuous mapping that we denote by Y. Let $\boldsymbol{x} \in \mathcal{B}$ and consider $\boldsymbol{y} = Y\boldsymbol{x}$. The point $\boldsymbol{y}$ also belongs to $\mathcal{B}$ since $\sum_k y_{k,r} = K_r$ by the structure of Y, the right-hand side of (10) sums to K_r, and $y_{k,r} \ge x^-_{k,r}$, since $x^-_{k,r}$ is defined as a lower bound on the mapping; thus, $Y : \mathcal{B} \to \mathcal{B}$. Since $\mathcal{B}$ is a compact set, the continuous mapping Y therefore admits at least a fixed point $\boldsymbol{x}^*$ by the Brouwer fixed point theorem.

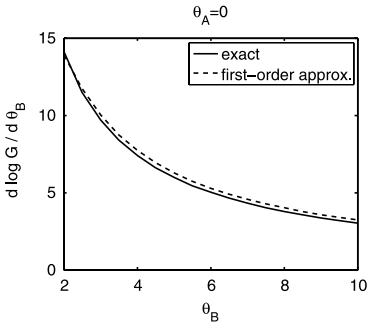


Fig. 5. Running case. Sensitivity of log G to a queue-dependent function parameter.

Proof of Lemma 1. Let $\mu_{k,r}$ be the Lagrange multiplier of the constraint $x_{k,r} \geq x^{-}_{k,r}$ and let μ_r be the multiplier of the constraint $\sum_{k=1}^{M} x_{k,r} = K_r$. The KKT conditions for (14) are

$$\begin{aligned} -\frac{\partial f(\boldsymbol{x})}{\partial x_{k,r}} &= \mu_r - \mu_{k,r}, \quad \forall k, r : \theta_{k,r} > 0, \qquad x_{k,r} \geq x^{-}_{k,r}, \quad \forall k, r \\ (x_{k,r} - x^{-}_{k,r})\mu_{k,r} &= 0, \qquad \forall k, r, \qquad \mu_{k,r} \geq 0, \quad \forall k, r, \\ \sum_{k=1}^{M} x_{k,r} &= K_r, \qquad \forall r. \end{aligned} \tag{19}$$

We immediately verify that every AMVA solution satisfies the conditions associated to primal feasibility as every solution $\boldsymbol{x}$ is in $\mathcal{B}$. Further, from Proposition 1 we know that, for $K > 1$, $x_{k,r} > x^{-}_{k,r}$, thus we set the associated multipliers to $\mu_{k,r} = 0$, verifying dual feasibility and complementary slackness. Also, the first condition becomes

$$\log\left(\frac{x_{k,r}}{\theta_{k,r}\beta_{k,r}(1+\delta_r x_{k,r})\gamma_k(1+\delta x_k)(1+\delta x_k)}\right) = -\mu_r, \quad \theta_{k,r} > 0.$$

Letting T_r be the mean throughput in an AMVA solution, and setting $\mu_r = -\log(T_r)$, we immediately see from (9) that every AMVA solution complies with the KKT conditions.

Proof of Theorem 3. In the load-independent case, $\beta_{k,r}(\cdot) \equiv \gamma_k(\cdot) \equiv 1, \forall k, r$. Note that $f(\boldsymbol{x})$ is twice differentiable in $\mathcal{B}$, since $x^{-}_{k,r} > 0$. Let $H(\boldsymbol{x})$ be the Hessian matrix for $f(\boldsymbol{x})$. It is sufficient to show that, for any $\boldsymbol{x} \in \mathcal{B}$ and direction $\boldsymbol{w} \in \mathbb{R}^{MR}/\{\boldsymbol{0}\}$, $V = \boldsymbol{w}^T H(\boldsymbol{x})\boldsymbol{w} > 0$. We denote by $w_{k,r}$ the element of $\boldsymbol{w}$ corresponding to variable $x_{k,r}$. Also, for a continuous positive function $g(\boldsymbol{x})$, define $\Lambda_x(g(\boldsymbol{x})) = \partial \log g(\boldsymbol{x})/\partial x$ to be its logarithmic derivative. Simple calculations yield

$$\begin{aligned} V &= \sum_{k=1}^{M}\left(\sum_{r=1}^{R} w_{k,r}^2 \Lambda_{x_{k,r}}(x_{k,r}) - \sum_{r=1}^{R}\sum_{\substack{s=1 \\ s\neq r}}^{R} w_{k,r}w_{k,s}\Lambda_{x_k}(1+\delta x_k)\right) \\ &= \sum_{k=1}^{M}\left(\sum_{r=1}^{R}\frac{w_{k,r}^2}{x_{k,r}} - \sum_{r=1}^{R}\sum_{s=1}^{R}\frac{\delta w_{k,r}w_{k,s}}{(1+\delta x_k)}\right). \end{aligned} \tag{20}$$

To show that $V > 0$ we start by noting that this is true if $\delta = 0$, which occurs when $K = 1$, as in this case the second set of sums in (20) is zero, and $\sum_{k=1}^{M}\sum_{r=1}^{R}\frac{w_{k,r}^2}{x_{k,r}} > 0$ because $x_{k,r} > 0$ and $\boldsymbol{w}$ is a nonzero vector. Now we focus on the case $\delta > 0$. Let us write $V = \sum_{k=1}^{M} V_k$, where

$$V_k = \sum_{r=1}^{R}\frac{w_{k,r}^2}{x_{k,r}} - \sum_{r=1}^{R}\sum_{s=1}^{R}\frac{\delta w_{k,r}w_{k,s}}{(1+\delta x_k)}.$$

It is proved in [43, Lemma 6.3] that, for a superset of $\mathcal{B}$,

$$\sum_{r=1}^{R}\frac{w_{k,r}^2}{x_{k,r}} - \sum_{r=1}^{R}\sum_{s=1}^{R}\frac{w_{k,r}w_{k,s}}{x_k} \geq 0, \quad k = 1, \ldots, M. \tag{21}$$

Multiplying V_k by $(1+\delta x_k)/(\delta x_k) > 0$, we obtain

$$V_k\frac{1+\delta x_k}{\delta x_k} = \frac{1}{\delta x_k}\sum_{r=1}^{R}\frac{w_{k,r}^2}{x_{k,r}} + \sum_{r=1}^{R}\frac{w_{k,r}^2}{x_{k,r}} - \sum_{r=1}^{R}\sum_{s=1}^{R}\frac{w_{k,r}w_{k,s}}{x_k} \geq \frac{1}{\delta x_k}\sum_{r=1}^{R}\frac{w_{k,r}^2}{x_{k,r}} \geq 0,$$

where the first inequality follows from (21), and the second from noting that $\delta x_k > 0$ and $x_{k,r} > 0$. We therefore conclude that $V_k \geq 0$, thus $V \geq 0$. To prove $V > 0$ we reconsider the expression above, summing over k, thus

$$\sum_{k=1}^{M} V_k \frac{1+\delta x_k}{\delta x_k} = \sum_{k=1}^{M} \left(\frac{1}{\delta x_k} \sum_{r=1}^{R} \frac{w_{k,r}^2}{x_{k,r}} + \sum_{r=1}^{R} \frac{w_{k,r}^2}{x_{k,r}} - \sum_{r=1}^{R} \sum_{s=1}^{R} \frac{w_{k,r} w_{k,s}}{x_k} \right) \geq \sum_{k=1}^{M} \frac{1}{\delta x_k} \sum_{r=1}^{R} \frac{w_{k,r}^2}{x_{k,r}} > 0.$$

The last inequality follows since δ, x_k, and $x_{k,r}$ are strictly positive, and $\boldsymbol{w}$ is a nonzero vector. As a result, we have that $\sum_{k=1}^{M} V_k \frac{1+\delta x_k}{\delta x_k} > 0$, and since we know that each $V_k \geq 0$ and $\frac{1+\delta x_k}{\delta x_k} > 0$, at least one V_k must be strictly positive, thus $V > 0$. This implies that $f(\mathbf{x})$ is strictly convex in $\mathcal{B}$, and therefore (14) has a unique global minimum in $\mathcal{B}$. Since $f(\mathbf{x})$ is convex and the constraints in (14) are affine, the KKT conditions are both necessary and sufficient for optimality, thus, by Lemma 1, every solution of the AMVA equations is an optimal solution of (14). As $f(\mathbf{x})$ is strictly convex, this solution must be unique.

Proof of Theorem 4. We use a similar notation as in the proof of Theorem 3. In the general case, the Hessian can be written as

$$V = \boldsymbol{w}^T H(\boldsymbol{x}) \boldsymbol{w} = \boldsymbol{w}^T (H_{LI}(\boldsymbol{x}) + H_\gamma(\boldsymbol{x}) + H_\beta(\boldsymbol{x})) \boldsymbol{w},$$

where $H_{LI}(\boldsymbol{x})$ is the Hessian as in Theorem 3, while $H_\gamma(\boldsymbol{x})$ and $H_\beta(\boldsymbol{x})$ group the contributions of the $\gamma_k(\cdot)$ and $\beta_{k,r}(\cdot)$ factors, for all r, k. Since, from Theorem 3, we know that $\boldsymbol{w}^T H_{LI}(\boldsymbol{x}) \boldsymbol{w} > 0$, we only need to prove that $V_\gamma = \boldsymbol{w}^T H_\gamma(\boldsymbol{x}) \boldsymbol{w} \geq 0$ and $V_\beta = \boldsymbol{w}^T H_\beta(\boldsymbol{x}) \boldsymbol{w} \geq 0$. Recall that $\Lambda_x(g(\boldsymbol{x})) = \partial \log g(\boldsymbol{x}) / \partial x$ for a positive function $g(\boldsymbol{x})$. After simple passages we find

$$V_\beta = \sum_{r=1}^{R} -w_{k,r}^2 \Lambda_{x_{k,r}}(\beta_{k,r}(1+\delta_r x_{k,r})).$$

Thus, $V_\beta \geq 0$ if the $\beta_{k,r}(\cdot)$'s are monotone non-increasing.

For V_γ, $H_\gamma(\boldsymbol{x})$ has the property of being a block-diagonal matrix with M blocks of size R. Letting block k, corresponding to station k, be $H_\gamma^{(k)}(\boldsymbol{x})$, and partitioning in a similar way $\boldsymbol{w} = [\boldsymbol{w}_1, \ldots, \boldsymbol{w}_M]$, we can write

$$V_\gamma = \boldsymbol{w}^T H_\gamma(\boldsymbol{x}) \boldsymbol{w} = \sum_{i=1}^{M} \boldsymbol{w}_k^T H_\gamma^{(k)}(\boldsymbol{x}) \boldsymbol{w}_k.$$

For each of the terms in the sum we observe that

$$H_\gamma^{(k)}(\boldsymbol{x}) = -\Lambda_{x_k}(\gamma_k(1+\delta x_k)) \mathbf{1}_{R \times R},$$

where $\mathbf{1}_{R \times R}$ is an $R \times R$ matrix of ones, which is positive semi-definite. It therefore follows that $\boldsymbol{w}_k^T H_\gamma^{(k)}(\boldsymbol{x}) \boldsymbol{w}_k \geq 0$ since $\Lambda_{x_k}(\gamma_k(1+\delta x_k)) \leq 0$ as $\gamma_k(\cdot)$ is monotone non-increasing. As a result, $V_\gamma \geq 0$, and thus $V > 0$, which ensures that $f(\boldsymbol{x})$ is strictly convex in $\mathcal{B}$, so it has a unique minimizer that corresponds to the AMVA solution by Lemma 1.

Proof of Theorem 5. Observe that

$$\frac{\partial \log G}{\partial \theta_{k,r}} = G^{-1}(\boldsymbol{K}) \sum_{\boldsymbol{n}: n_{k,r} \geq 1} C_k(\boldsymbol{n}_k) \frac{\partial F_k(\boldsymbol{n}_k)}{\partial \theta_{k,r}} \prod_{i \neq k} F_i(\boldsymbol{n}_i), \tag{22}$$

thus differentiating (2), we find

$$\frac{\partial F_k(\boldsymbol{n}_k)}{\partial \theta_{k,r}} = \beta_{k,r}(n_{k,r}) \gamma_k(n_k) F_k(\boldsymbol{n}_k - \mathbf{1}_r) + D_{k,r}(\boldsymbol{n}_k) \frac{\partial F_k(\boldsymbol{n}_k - \mathbf{1}_r)}{\partial \theta_{k,r}}.$$

Letting $S_{\theta_{k,r}}(F_k(\boldsymbol{n}_k)) = (\theta_{k,r} / F_k(\boldsymbol{n}_k)) \partial F_k(\boldsymbol{n}_k) / \partial \theta_{k,r}$ and using (2), the last expression can be rewritten as

$$S_{\theta_{k,r}}(F_k(\boldsymbol{n}_k)) = 1 + S_{\theta_{k,r}}(F_k(\boldsymbol{n}_k - \mathbf{1}_r)) = n_{k,r},$$

where the last passage is obtained after telescoping the recurrence relation on $n_{k,r}$, with terminating condition $S_{\theta_{k,r}}(F_k(\boldsymbol{n}_k - n_{k,r}\mathbf{1}_r)) = 0$. Plugging this result in (22), we obtain

$$\frac{\partial \log G}{\partial \theta_{k,r}} = G^{-1}(\boldsymbol{K}) \sum_{\boldsymbol{n}: n_{k,r} \geq 1} C_k(\boldsymbol{n}_k) \frac{n_{k,r}}{\theta_{k,r}} \prod_{i=1}^{M} F_i(\boldsymbol{n}_i) = \frac{x_{k,r}}{\theta_{k,r}}.$$

Proof of Theorem 6. Generalizing the derivation of (15), we find

$$\frac{\partial F_k(\boldsymbol{n}_k)}{\partial \theta_{k,r,p}} = \frac{\partial D_{k,r}(\boldsymbol{n}_k)}{\partial \theta_{k,r,p}} F_k(\boldsymbol{n}_k - \mathbf{1}_r) + D_{k,r}(\boldsymbol{n}_k) \frac{\partial F_k(\boldsymbol{n}_k - \mathbf{1}_r)}{\partial \theta_{k,r,p}} = \frac{\partial D_{k,r}(\boldsymbol{n}_k)}{\partial \theta_{k,r,p}} \frac{F_k(\boldsymbol{n}_k)}{D_{k,r}(\boldsymbol{n}_k)} + D_{k,r}(\boldsymbol{n}_k) \frac{\partial F_k(\boldsymbol{n}_k - \mathbf{1}_r)}{\partial \theta_{k,r,p}}$$
$$= \frac{F_k(\boldsymbol{n}_k)}{D_{k,r}(\boldsymbol{n}_k)} \left(\frac{\partial D_{k,r}(\boldsymbol{n}_k)}{\partial \theta_{k,r,p}} + \frac{D_{k,r}(\boldsymbol{n}_k)}{F_k(\boldsymbol{n}_k - \mathbf{1}_r)} \frac{\partial F_k(\boldsymbol{n}_k - \mathbf{1}_r)}{\partial \theta_{k,r,p}} \right).$$

Therefore

$$\frac{D_{k,r}(\boldsymbol{n}_k)}{F_k(\boldsymbol{n}_k)} \frac{\partial F_k(\boldsymbol{n}_k)}{\partial \theta_{k,r,p}} = \frac{\partial D_{k,r}(\boldsymbol{n}_k)}{\partial \theta_{k,r,p}} + \frac{D_{k,r}(\boldsymbol{n}_k)}{D_{k,r}(\boldsymbol{n}_k - \mathbf{1}_r)} \frac{D_{k,r}(\boldsymbol{n}_k - \mathbf{1}_r)}{F_k(\boldsymbol{n}_k - \mathbf{1}_r)} \frac{\partial F_k(\boldsymbol{n}_k - \mathbf{1}_r)}{\partial \theta_{k,r,p}}$$
$$= D_{k,r}(\boldsymbol{n}_k) \sum_{u=0}^{n_{k,r}-1} \frac{1}{D_{k,r}(\boldsymbol{n}_k - u\mathbf{1}_r)} \frac{\partial D_{k,r}(\boldsymbol{n}_k - u\mathbf{1}_r)}{\partial \theta_{k,r,p}},$$

which finally yields

$$\frac{\partial F(n_{k,r})}{\partial \theta_{k,r,p}} = F_k(\boldsymbol{n}_k) \left(\sum_{u=0}^{n_{k,r}-1} \frac{1}{D_{k,r}(\boldsymbol{n}_k - u\mathbf{1}_r)} \frac{\partial D_{k,r}(\boldsymbol{n}_k - u\mathbf{1}_r)}{\partial \theta_{k,r,p}} \right) = \theta_{k,r,p}^{-1} F_k(\boldsymbol{n}_k) \left(\sum_{u=0}^{n_{k,r}-1} S_{\theta_{k,r,p}} \left[D_{k,r}(\boldsymbol{n}_k - u\mathbf{1}_r) \right] \right).$$

The final result is obtained by using the last expression in the derivative of $\log G$ with respect to $\theta_{k,r,p}$, see (22) in the proof of Theorem 5.

References

[1] F. Baskett, K.M. Chandy, R.R. Muntz, F.G. Palacios, Open, closed, and mixed networks of queues with different classes of customers, J. ACM 22 (2) (1975) 248–260.
[2] P. Cremonesi, P.J. Schweitzer, G. Serazzi, A unifying framework for the approximate solution of closed multiclass queuing networks, IEEE Trans. Comput. 51 (2002) 1423–1434.
[3] M.N. Bennani, D. Menasce, Resource allocation for autonomic data centers using analytic performance models, in: ACM ICAC, 2005, pp. 229–240.
[4] S. Bardhan, D. Menasce, Predicting the effect of memory contention in multi-core computers using analytic performance models, IEEE Trans. Comput. (2014).
[5] J. Rolia, K. Sevcik, The method of layers, IEEE Trans. Softw. Eng. 21 (1995) 689–700.
[6] G. Franks, T. Al-Omari, M. Woodside, O. Das, S. Derisavi, Enhanced modeling and solution of layered queueing networks, IEEE Trans. Softw. Eng. 35 (2009) 148–161.
[7] G. Franks, L. Li, Efficiency improvements for solving layered queueing networks, in: ACM/SPEC ICPE, 2012.
[8] A. Kattepur, M. Nambiar, Performance modeling of multi-tiered web applications with varying service demands, in: IPDPS Workshops, 2015.
[9] D.A. Menasce, V.A. Almeida, L.W. Dowdy, Performance by Design: Computer Capacity Planning by Example, Prentice Hall Professional, 2004.
[10] J. Padhye, A.D. Rahatekar, L.W. Dowdy, A simple LAN file placement strategy, in: Int. CMG Conference, 1995, pp. 396–406.
[11] K.K. Leung, Load-dependent service queues with application to congestion control in broadband networks, Perform. Eval. 50 (1) (2002) 27–40.
[12] D. Kumar, L. Zhang, A. Tantawi, Enhanced inferencing: Estimation of a workload dependent performance model, in: Valuetools, 2009.
[13] C. Stewart, T. Kelly, A. Zhang, Exploiting nonstationarity for performance prediction, in: Proc. EuroSys, 2007.
[14] Z. Wang, X. Liu, A. Zhang, C. Stewart, X. Zhu, T. Kelly, S. Singhal, Autoparam: Automated control of application-level performance in virtualized server environments, in: FeBid, 2007.
[15] N. Gans, N. Liu, A. Mandelbaum, H. Shen, H. Ye, Service times in call centers: Agent heterogeneity and learning with some operational consequences, IMS Collect. 6 (2010) 99–123.
[16] M. Reiser, Mean-value analysis and convolution method for queue-dependent servers in closed queueing networks, Perform. Eval. 1 (1981) 7–18.
[17] S.C. Bruell, G. Balbo, P.V. Afshari, Mean value analysis of mixed, multiple class BCMP networks with load dependent service stations, Perform. Eval. 4 (1984) 241–260.
[18] C.H. Sauer, Computational algorithms for state-dependent queueing networks, ACM Trans. Comput. Syst. 1 (1) (1983) 67–92.
[19] M. Reiser, S.S. Lavenberg, Mean-value analysis of closed multichain queueing networks, J. ACM 27 (2) (1980) 312–322.
[20] P.S. Kritzinger, S. van Wyk, A.E. Krzesinski, A generalisation of Norton's theorem for multiclass queueing networks, Perform. Eval. 2 (2) (1982) 98–107.
[21] J.L. Boudec, D. McDonald, J. Mundinger, A generic mean field convergence result for systems of interacting objects, in: QEST, 2007.
[22] R.R. Muntz, Poisson Departure Processes and Queueing Networks, IBM Research Report RC-4145, 1972.
[23] B.I. Choi, C. Knessl, C. Tier, A queueing system with queue length dependent service times, with applications to cell discarding in ATM networks, J. Appl. Math. Stoch. Anal. 12 (1999) 35–62.
[24] B.D. Choi, Y.C. Kim, Y.W. Shin, C.E.M. Pearce, The M^X/G/1 queue with queue length dependent service times, J. Appl. Math. Stoch. Anal. 14 (2001) 399–419.
[25] R. Bekker, G.M. Koole, B.F. Nielsen, T.B. Nielsen, Queues with waiting time dependent service, Queueing Syst. 68 (2011) 61–78.
[26] M. Reiser, H. Kobayashi, Queueing networks with multiple closed chains: Theory and computational algorithms, IBM J. Res. Dev. 19 (3) (1975) 283–294.
[27] G. Casale, A note on stable flow-equivalent aggregation in closed networks, Queueing Syst. 60 (2008) 193–202.
[28] A.E. Krzesinski, J. Greyling, Improved linearizer methods for queueing networks with queue dependent centres, in: ACM SIGMETRICS, 1984, pp. 41–51.
[29] J. Zahorjan, E.D. Lazowska, Incorporating load dependent servers in approximate mean value analysis, in: ACM SIGMETRICS, 1984, pp. 52–62.
[30] D. Mitra, J. McKenna, Asymptotic expansions for closed markovian networks with state-dependent service rates, J. ACM 33 (3) (1985) 568–592.
[31] A.I. Lyakhov, Closed queue networks with load-dependent servers: an asymptotic analysis, Automa. Remote Control 58 (3) (1997) 435–444.
[32] J. Shanthikumar, D.D. Yao, Throughput bounds for closed queueing networks with queue-dependent service rates, Perform. Eval. 9 (1) (1988) 69–78.
[33] I.F. Akyildiz, A. Sieber, Approximate analysis of load dependent general queueing networks, IEEE Trans. Softw. Eng. 14 (11) (1988) 1537–1545.
[34] G. Schwarz, Estimating the dimension of a model, Ann. Statist. 6 (1978) 461–464.
[35] J. Nocedal, S.J. Wright, Numerical Optimization, Springer-Verlag, New York, 1999.
[36] E. de Souza e Silva, S.S. Lavenberg, R.R. Muntz, A perspective on iterative methods for the approximate analysis of closed queueing networks, in: The Intl. Workshop on Computer Performance and Reliability, 1984.
[37] K.R. Pattipati, M.M. Kostreva, J.L. Teele, Approximate mean value analysis algorithms for queueing networks: Existence, uniqueness, and convergence results, J. ACM 37 (1990) 643–673.

[38] J. Verschelde, Algorithm 795: PHCpack: a general-purpose solver for polynomial systems by homotopy continuation, ACM Trans. Math. Software 25 (2) (1999) 251–276.
[39] W. Wang, G. Casale, Bayesian service demand estimation using Gibbs sampling, in: IEEE MASCOTS, 2013, pp. 567–576.
[40] E. de Sousa e Silva, R.R. Muntz, Simple relationships among moments of queue lengths in product form queueing networks, IEEE Trans. Comput. 37 (9) (1988) 1125–1129.
[41] S.N. Ethier, T.G. Kurtz, Markov Processes: Characterization and Convergence, Wiley, 1986.
[42] S.K. Tripathi, C.M. Woodside, A vertex-allocation theorem for resources in queuing networks, J. ACM 35 (1) (1988) 221–230.
[43] N.S. Walton, Proportional fairness and its relationship with multi-class queueing networks, Ann. Appl. Probab. 19 (2009) 2301–2333.

Giuliano Casale is a Lecturer in performance analysis and operations research at the Department of Computing of Imperial College London. Prior to this, he was a full-time researcher at SAP Research UK. His research interests include cloud computing, modeling, and resource management, topics on which he has published more than 70 refereed papers in international conferences and journals. He has served as co-chair for several conferences in the area of performance engineering, such as ACM SIGMETRICS/Performance 2012, IEEE MASCOTS, USENIX ICAC and ACM/SPEC ICPE. He is member of the IFIP WG 7.3 group on Computer Performance Analysis and the ACM SIGMETRICS Board of Directors.

Juan F. Pérez is a Research Associate in performance analysis at Imperial College London, Department of Computing. He obtained a Ph.D. in Computer Science from the University of Antwerp, Belgium, in 2010. His research interest centers around the performance analysis of computer systems, especially on cloud computing and optical networking.

Weikun Wang is a Ph.D. student and research assistant in performance analysis at Imperial College London, Department of Computing, since January 2013. He obtained a Master in Computing from Imperial College London in 2012. His research interest focuses on performance modeling of computer systems deployed in cloud.