Moving Horizon Estimation of Service Demands in Queuing Networks

Emilio Incerto
IMT School for Advanced Studies
Piazza San Francesco 19
Lucca, Italy
emilio.incerto@imtlucca.it

Annalisa Napolitano
IMT School for Advanced Studies
Piazza San Francesco 19
Lucca, Italy
annalisa.napolitano@imtlucca.it

Mirco Tribastone
IMT School for Advanced Studies
Piazza San Francesco 19
Lucca, Italy
mirco.tribastone@imtlucca.it

***Abstract*—Accurate estimation of resource demands is one of the key challenges to be able to use queuing networks (QNs) for performance prediction, especially in cases where the profiling is to be performed through a non-intrusive system instrumentation. This problem is worsened when one needs to obtain a continuously updated model (e.g., for control and adaptation purposes) because it becomes crucial to use fast estimation methods that do not interfere with the behavior of the running system. A crucial limitation in the state of the art is the assumption that the measurement are taken from a system in the steady state regime. To the best of our knowledge, this paper presents the first approach—here developed for single-class QNs—that does not make such assumption. Our service-demand estimation technique relies on a deterministic approximation of the QN where the transient evolution of the queue lengths is modeled by means of a compact analytical representation based on a system of coupled nonlinear ordinary differential equations. We set up a *moving-horizon estimation* problem whereby the governing equations of the model, appropriately unfolded over a given time horizon, represent the constraints of a quadratic program that seeks to find the optimal choice of service demands that minimize the error between the measured queue lengths and the predicted ones. An extensive numerical evaluation demonstrates the efficiency and the effectiveness of our approach against the state-of-the-art techniques for service demands estimation.**

I. Introduction

Accurately predicting the performance of a computing system hinges on the availability of a model with well calibrated parameters. When the system under consideration can be satisfactorily modeled as a queuing network (QN), key quantities to estimate are the service demands, i.e., the parameters that uniquely characterize the distributions of the service times that govern each node of the network. Recent trends in software engineering advocate the use of quantitative models at runtime to perform (self-)adaptation to continuously meet given performance-related quality-of-service goals [1]–[7]. Such a context calls for efficient techniques that can provide frequent up-to-date estimates of service demands in a minimally intrusive manner so as not to alter the running system significantly. The new perspective of online estimation brings about two main challenges:

i) It may not be appropriate to assume that the system is in the steady-state regime, in particular when it is subjected to control actions that update its parameters frequently enough. The steady-state assumption is important because it allows the use of many well-known relationships and/or analytical results for QNs (see e.g., [8], [9]). Importantly, none of techniques reported in the recent survey on service demand estimation [10] is applicable to systems in a transient regime.

ii) It is not possible to intrude into the running system by, for instance, injecting measurement traffic to exercise the system under different utilization levels (e.g., [11]). Regression techniques that involve different quantities (e.g., throughput and utilization) may introduce instrumentation overheads and may cause inaccuracies and bias in the estimation due to multicollinearity [12]. This has motivated the development of techniques that make use of queue-length information only [13], [14], which in some cases may be accessible externally (i.e., from the operating system) without the need for direct instrumentation of the application.

In this paper we present the first estimation method that, to the best of our knowledge, does not make the steady-state assumption, using low-overhead measurements of queue lengths only for systems that can be modeled as single-class QNs with exponentially distributed service times and load-dependent (i.e., multiple-server) service rates. The key intuition behind our approach is to consider a dynamical model of the QN that provides an estimate of its transient evolution *approximately*, thus avoiding the well-known state space explosion problem of the exact description based on the forward equations of a continuous-time Markov chain (CTMC, see e.g., [8]). In particular, we use a *mean-field* (or *fluid*) approximation of the QN (see [15] for an exhaustive review), where the time course of the queue length at each station is described by a single (non-linear) ordinary differential equation (ODE), which can be used as an estimate of the true mean.

By appropriately discretizing time, the mean-field ODEs of the QN are turned into difference equations; starting from some known initial condition that gives the queue length at each station in the QN, it is then possible to unfold the discrete dynamics over H steps, obtaining explicit equations for the queue lengths at such subsequent steps. This unfolding can be interpreted as a set of constraints for an optimization problem where the decision variables are the service demands to be estimated, with the objective of minimizing the error between the predicted queue lengths and the measured ones across the whole observation horizon H. Naturally, this set-up is agnostic

to the knowledge of the measured system being in steady state since it involves dynamical equations that can be used over any given fixed time horizon. In addition, it allows for an on-line estimation of the service demands through a *moving horizon* strategy, by shifting forward the time window at each step in order to continuously obtain updated estimates. This represents a novel approach to service demands estimation which is fundamentally due to the availability of a simple (approximating) ODE system, unleashing the adaptation of ideas on system identification based on moving-horizon estimation that are originally rooted in control theory (e.g., [16]–[18]).

The ODEs have piecewise-linear derivatives due to the presence of minimum functions that encode the load-dependent rates in a multi-server queue [19]–[21]. However, we avoid to explicitly model this nonlinearity in the optimization problem since it is naturally encoded in the measured queue-length dynamics. Indeed, in this paper we show how to formulate the moving horizon estimation (MHE) into a quadratic program (QP) which can be efficiently used for the estimation. Then, only after the solution of each optimization problem the explicit throughput non linear relation is considered for computing the actual system service demands. Our approach is somewhat dual to that in [7], where a mixed integer linear optimization (for the same class of QNs) was obtained for the purposes of model predictive control [22]. In [7] similar constraints as in this paper are used, but the objective function represents a quality-of-service requirement (i.e., throughput, response time, or queue length) that can be achieved using decision variables related to the routing probabilities and number of servers at each station; here, as discussed, the objective function is the prediction error, while routing probabilities and server multiplicities are assumed to be known.

The accuracy of the service-demand estimates ultimately depends on the accuracy of the approximation of the mean-field model. In general, it affords two related interpretations. The first is that the ODE solutions coincide with a sample path of the underlying CTMC under appropriate *limiting and scaling conditions* [23]; these, in our case, correspond to having QNs of increasing size where the server multiplicities at each queue grow proportionally with the workload [20]. The second interpretation is a first-order moment-closure approximation of a given fixed CTMC representing a population process [15]; in this paper's setting, this corresponds to replacing the expectation of a nonlinear function of random variables with the function of the expectation of the random variables, which introduces an error in general [24], [25].

The main implication is that the ODEs incur approximation errors which depend on the parameters of the QN, but which tend to become more negligible as the number of jobs and server multiplicities increase. With a substantial numerical assessment using both controlled and randomized experiments, we show that our service demand technique is efficient and yield accurate estimates, with an average error of 3.82% and a maximum average runtime of 0.52s on QNs with over 20 stations, thus promoting our approach as effective technique for on-line estimation of service demands.

Related work: When applied to a system in the steady state, our technique can be compared with many approaches that rely on fluid model [26] or on a corollary of Little's Law, the Utilization Law [10], using different statistical inference approaches such as linear regression [27], non-linear optimization [28], clustering regression [29], independent component analysis [30], pattern matching [31] and Gibbs sampling [13], [32] based on measured values of utilization and/or throughput. A main drawback of these techniques is that they require observations of quantities that are difficult to obtain: indeed, utilization is not available when there is no complete control of the underlying physical layer (e.g., when using a Platform-as-a-Service environment).

Most approaches require *active probing*, i.e., observing the system in different configurations (e.g., at different utilization levels). For example, the technique in [33], which is related to ours since it is based on a QP optimization, measures the utilization of every station under different system configurations (i.e., different load combinations, load intensity, and so on), and it estimates demands only relying on a steady-state closed equation for the QN. However, active probing at runtime induces extra interfering traffic. This is necessarily intrusive and the observed metrics will differ from those that would have been generated by the regular traffic only (see [34]). This difference makes the identification problem more difficult since reconstructing the original metric (i.e. the *ground-truth*) from the measured one is not straightforward.

The approach presented in [11] is non-intrusive since it measures only the end-to-end response time and throughput of the transactions submitted to the system, modeled as a QN. The service demands of all network stations are estimated by means of a nonlinear optimization problem that fits the computed performance metrics with the measured ones. The technique requires active probing for the collection of several steady-state observations at different utilization levels of the network. In addition, because it relies only on end-to-end measurements taken where the workload is generated, the method can only yield a feasible assignment for the service demands; for example it cannot distinguish service demands that differ up to a permutation in a series of queuing stations.

Most closely related to ours are two recent techniques that consider queue-length measurements only [13], [14]. In [13], the authors develop an estimation method based on Gibbs sampling. However, the computational cost of the algorithm is high already for networks of small/moderate size, making it not applicable online. Instead, in [14] a closed-form expression to evaluate all the network's stations service demands for a multiclass application is presented for the load-independent case. The load-dependent case is approximated in closed form by an appropriate scaling of the estimated service demands. We postpone a more detailed comparison to Section IV.

Paper structure: The remainder of this paper is organized as follows. Section II provides an overview of single-class QNs and their approximation by means of mean-field ODEs. Section III presents our QP-based estimation method. Section IV discusses the results of our numerical evaluation,

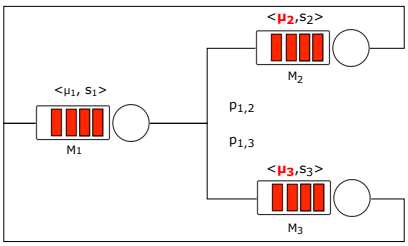


Fig. 1: Queuing network for the load balancer running example

while Section V concludes with perspectives on future work.

II. Background

In this section all the steps briefly discussed in Section I are formally defined by means of a simple running example.

A. Running Example

Load balancing is an established design technique in performance engineering [35] and it is considered a building block for scalable and reliable distributed systems. Figure 1 shows a closed single class QN model for a load balancing system with two replicas M_2, M_3 and a workload generator M_1 (i.e., users terminals). Each station in the network is characterized by an exponentially distributed service demand $1/\mu_i$ (i.e., time required by station i for completing one single job) and a parallelism level s_i (i.e., maximum number of requests processable by parallel equivalent servers at station i). User requests are distributed across the computational nodes according to the probabilities $p_{1,2}$ and $p_{1,3}$.

Typically, designers employ such model for studying the performance of systems in a "what-if" analysis fashion (i.e., combining different workload and queuing parameters) so as to identify the most suitable hardware/software configuration satisfying requirements under worst-case or average execution conditions. Clearly, in order to conduct meaningful design-time or runtime analysis, model parameters need to be accurately discovered otherwise the computed predictions will significantly deviate from the real system behavior.

In this setting, our goal is to identify the unknown service demands $1/\mu_2,\ 1/\mu_3$ at runtime, relying only on queue-length traces collected without altering the normal operational behavior of the system. For doing so we assume that the remaining system parameters, i.e., $\mu_1, s_1, s_2, s_3, p_{1,2}, p_{1,3}$ are known; indeed, they can be easily derived from system logs [36].

In the following the definition of a QN model, its ODEs based representation and the QP encoding are formalized.

B. Single Class Queuing Networks

We consider closed QNs where we assume that a fixed population of jobs circulate in the system; the extension to an open QN, admitting exogenous arrivals, is straightforward.

A single-class QN is described by the following quantities:

- M is the number of queuing stations;
- $s = (s_1, \ldots, s_M)$ is the vector of server multiplicities, where s_i denotes the number of servers at station i, with $1 \leq i \leq M$;
- $\mu = (\mu_1, \ldots, \mu_M)$ is the exponentially distributed service rate at station i, with $1 \leq i \leq M$, hence $1/\mu_i$ is the corresponding service demand;
- $P = (p_{i,j})_{1 \leq i,j \leq M}$ is the routing probability matrix, i.e., a stochastic matrix where each element $p_{i,j}$ gives the probability that a job goes to station j upon completion at station i;
- $x(0) = (x_1(0), \ldots, x_M(0))$ is the *initial condition*, i.e., the number of jobs assigned to each station at time 0.

C. ODE Model

The mean-field ODEs for the QN model can be derived from a stochastic description of the system in terms of a Markov Population Process which tracks the queue-length processes at each station. Similarly to what presented in [7], the ODE system is as follows:

$$\dot{x}_i(t) = -\mu_i \min\{x_i(t), s_i\} + \sum_{j=1}^{M} p_{j,i}\mu_j \min\{x_j(t), s_j\} \quad (1)$$

for $1 \leq i \leq M$. Considering the initial value problem where each variable x_i is given initial condition $x_i(0)$, the solution $x(t) = (x_1(t), \ldots, x_M(t))$ gives an estimate of the average queue length at each station.

A key aspect of the ODE formulation is the *nonlinear* instantaneous average throughput of station i, given by $\mu_i \min\{x_i(t), s_i\}$: when the queue length $x_i(t)$ in station i is less than the available number of servers s_i, then the $x_i(t)$ jobs are served in parallel; otherwise some of the jobs are enqueued and only s_i of them are processed simultaneously. Network topologies are represented in the model by weighting throughputs with the routing probabilities $p_{j,i}$.

In our running example, the following ODE system gives the mean-field approximation for the load balancer depicted in Figure 1:

$$\begin{aligned} \dot{x}_1(t) &= -\mu_1 \min\{x_1(t), s_1\} + \sum_{i=1}^{2} \mu_i \min\{x_i(t), s_i\} \\ \dot{x}_2(t) &= -\mu_2 \min\{x_2(t), s_2\} + p_{1,2}\mu_1 \min\{x_1(t), s_1\} \\ \dot{x}_3(t) &= -\mu_3 \min\{x_3(t), s_3\} + p_{1,3}\mu_1 \min\{x_1(t), s_1\} \end{aligned} \quad (2)$$

where we use the dot notation in the left-hand sides to denote derivative with respect to time. In [7] we validated the ODE model in (2) by comparing prediction results against real measurements taken from a running load balanced system, assuming that all model parameters (including the service demands) were known. In this paper, from the dynamics of (1) we extract a set of constraints which will be used in an optimization problem where decision variables are given by the vector of service rates μ.

III. Moving Horizon Estimation of Service Demands for Queuing Networks

A. Discrete-time model

Our estimation procedure is based on discretization of time for (1), by considering the usual approximation of the time derivative as $\dot{x}_i(t) \approx (x_i(t + \Delta t) - x_i(t))/\Delta t$. Then, assuming

a fixed time step, we denote by $\overline{x}_i(k)$ the approximation at the k-th step, i.e., $\overline{x}_i(k) \approx x_i(k\Delta t)$, for $k \geq 0$. The approximation can be computed by solving the following system of equations:

$$\overline{x}_i(k+1) = \overline{x}_i(k) - \Delta t \mu_i \min\{\overline{x}_i(k), s_i\} + \\ + \Delta t \sum_{j=1}^{M} p_{j,i} \mu_j \min\{\overline{x}_j(k), s_j\} \quad (3)$$

with $k \geq 0$ and $1 \leq i \leq M$, starting fro $\overline{x}_i(0) = x_i(0)$.

B. Nonlinear service demand estimator

We use (3) as constraints in an optimization problem which seeks to minimize the error between the predicted queue lengths $\overline{x}_i(k)$ and the measured ones over a given time horizon H. Denoting by $\tilde{x}_i(k)$ the measured queue length at time step k, we can write the optimization problem as follows:

$$\begin{aligned} &\text{minimize} \sum_{k=1}^{H} \sum_{i=1}^{M} (\overline{x}_i(k) - \tilde{x}_i(k))^2 \\ &\text{subject to:} \\ &\quad \text{Eq. } (3), \overline{x}_i(0) = \tilde{x}_i(0) \quad \text{for } 0 \leq k \leq H-1,\ 1 \leq i \leq M \end{aligned} \quad (4)$$

In other words, we search for the optimal vector of service rates μ that minimizes the difference between the measurements and the model predictions across all stations and all discrete time points over the horizon H, when the model dynamics is initialized with the measured queue lengths. In this general formulation we assume that all service rates are unknown since known service rates can be encoded by simply adding further equalities to the optimization problem.

C. Quadratic programming formulation

The main drawback of the optimization problem (4) is the presence of the nonlinear terms appearing in (3). We now consider a formulation in terms of a quadratic programming problem. The key point is to replace each nonlinear term with an auxiliary variable which linearizes the constraints. We do this by setting

$$T_i(k) := \Delta t \mu_i \min\{\overline{x}_i(k), s_i\}, \quad k \geq 0, 1 \leq i \leq M. \quad (5)$$

Essentially, $T_i(k)$ represents the instantaneous discretized throughput at station i. Then, the constraints (3) become:

$$\overline{x}_i(k+1) = \overline{x}_i(k) - T_i(k) + \sum_{j=1}^{M} p_{j,i} T_j(k) \quad (6)$$

Overall, the optimization problem (4) is rewritten thus:

$$\begin{aligned} &\text{minimize} \sum_{k=1}^{H} \sum_{i=1}^{M} (\overline{x}_i(k) - \tilde{x}_i(k))^2, \\ &\text{subject to:} \\ &\quad \text{Eq. } (6), \overline{x}_i(0) = \tilde{x}_i(0) \quad \text{for } 0 \leq k \leq H-1,\ 1 \leq i \leq M \end{aligned} \quad (7)$$

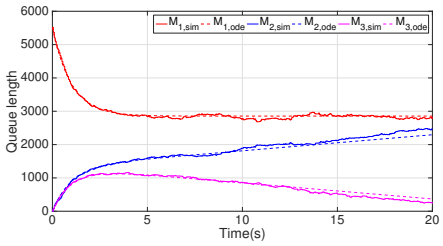


Fig. 2: Simulation and mean-field solution for the queue lengths of the running example.

N	μ_1	μ_2	μ_3	s_1	s_2, s_3	$p_{1,2}$, $p_{1,3}$	$p_{2,1}$, $p_{3,1}$
5500	1.0	68.97	73.80	∞	20	0.5	1.0

TABLE I: Model parameters for the load balancer of Fig. 1

We denote by $\overline{x}_i^*(k)$ and $T_i^*(k)$, with $0 \leq k \leq H-1$ and $1 \leq i \leq M$, the solution found by the above optimization problem. Using (5), we can define our service demand estimator as follows:

$$\mu_i^* := \frac{\sum_{k=0}^{H-1} T_i^*(k)}{\sum_{k=0}^{H-1} \min\{\overline{x}_i^*(k), s_i\}}, \qquad 1 \leq i \leq M. \quad (8)$$

This essentially estimates the average service demand at station i, i.e., $1/\mu_i^*$, across the entire observation window.

D. Application to the load balancer case study

Let us now apply the moving-horizon estimation introduced in the previous paragraphs to our running example. We consider a parametrization as in Table I, where we assume that the service demands at stations 2-3 (here chosen at random) must be estimated, while the demand at station 1 is known and fixed to 1.0. In order to model station 1 as a delay we encoded the infinite-server semantics by simply choosing a value $s_1 \geq N$.

Queue-length traces for each station were generated from the simulation of one sample path of the CTMC underlying the QN model for 20 time units, starting from an initial condition where all jobs are located in station 1. Then, we resampled the obtained queue-length traces with a time step $\Delta t = 0.01$, hence for a total of 2000 time steps. Figure 2 depicts the simulated trace against the numerical solution of the ODE (2).

We then applied the moving horizon estimator iteratively over fixed observation windows with $H = 100$, obtaining average percentage errors of the service demand estimates equal to 1.4% and 2.6% for station 2 and station 3 respectively. Moreover, we measured an average solution time of 0.01 s on a ordinary laptop.

IV. Numerical Evaluation

In this section we evaluate the effectiveness and the scalability of our MHE approach, on simulated queuing networks of different sizes and topologies. The replication package is publicly available at https://goo.gl/zNdr5f.

A. Methodology

In Section IV-B we assess the effectiveness of our MHE approach against the recent Queue Length Maximum Likelihood Estimation (QMLE) method proposed in [14]. As

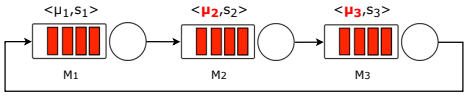
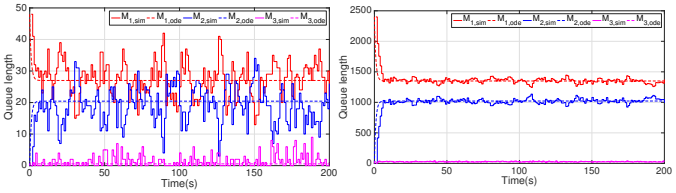


Fig. 3: Queuing network topology used in the comparison experiments against QMLE.



(a) $K = 1$ (b) $K = 50$

Fig. 4: Comparison between a sample path of the CTMC and the discretized ODE solution for the QN described in Fig. 3 with parameters: $x(0) = K(48, 0, 0), \mu = (1, 27, 48), s = K(\infty, 1, 1)$.

discussed in Section I, QMLE estimates service demands from queue-length data only. In Section IV-C we carry out an analysis of the scalability of MHE, by measuring runtime statistics on networks with an increasing number of stations. In both cases the queue length traces, i.e., the inputs of the estimation experiments, have been collected by simulating the underlying CTMC of the QN using stochastic simulation based on Gillespie's algorithm [37]. Moreover, we ran the QP optimization problems underlying MHE by using the Julia [38] interface of the CPLEX optimization tool [39].We executed all the experiments on a laptop equipped with an Intel dual-core i5 processor operating at a 2.6 GHz with 8 GB of memory.

B. Comparison with QMLE

In [14] QMLE is presented in two variants. The first uses nonlinear optimization on an exact closed-form expression based on the BCMP theorem. The second relies on a formula derived from the Bard-Schweitzer approximate mean value analysis (BS-AMVA) [40], [41], for both load-independent (i.e., single-server) and load-dependent (i.e., multi-server) queuing centers. In this paper we propose a comparison with the latter approach because the exact one is not suitable for online estimation purposes in real-world systems due to its computational complexity: indeed, in [14] are reported execution times of the order of 10^4 s for tandem networks.

In order to conduct as fair a comparison as possible, we considered synthetic networks, with topology given in Fig. 3, that stress known sources of approximation errors by both methods. MHE is essentially derived from a deterministic dynamical system, i.e., equations (1), that approximates the true stochastic behavior of the QN. Thus, we expect the estimation error to depend on that approximation.

The precision of the ODE approximation ultimately depends on the *system size*, intended as the number of jobs and the server multiplicities of the QN. More precisely, for a given network topology one can fix a sequence of QNs where both the initial job population $x(0)$ and the vector of server multiplicities K are scaled by the same factor $K \in \mathbb{N}$. The limit result of Kurtz [23], applied to the QNs of this paper, states that, in the limit when K goes to infinity, a sample path of the Markov chain becomes indistinguishable from the ODE solution (under an appropriate normalization that tracks the *density* process, i.e., the total number of jobs in each queue divided by the scaling factor K). Thus, for larger values of K one may expect increasingly high accuracy of the ODE solution with respect to a sample queue-length path. Figure 4 provides an graphical illustration of this effect.

The precision of QMLE depends on the accuracy of BS-AMVA and the normalization factor used in [14] to treat load-dependent service rates. The accuracy of the former tends to increase with the number of users circulating in the network [40]. The latter approximation uses the closed-form expression for a load-independent queue and scales it by a factor equals to the minimum between the number of servers in the queue and the average queue length, similarly to the denominator in (8). This may introduce an error at low-utilization regimes (i.e., when the queue length tends to be less than the number of servers with higher probability).

In order to take into account all these potential sources of approximation error, we generated our synthetic benchmarks by controlling the system size as well as the utilization levels of the QN. In order to do so, we considered 5 experiments which spanned the steady-state utilization levels between 0.1 and 0.8. For each experiment we considered varying system sizes $K \in \{1, 2, 5, 10, 20, 50\}$; the case $K = 1$ corresponds to a QN with a delay station at node M_1 and two single-server queues at nodes M_2 and M_3, i.e., $s = (\infty, 1, 1)$; we fixed the service demands arbitrarily with $\mu_1 = 1.0$, $\mu_2 = 27.0$ and $\mu_3 = 48.0$. The experiments differed in the choice of the initial condition $x(0)$, which was fixed in order to achieve varying steady-state utilizations of the queues.

In accordance with the experimental settings proposed in [14], we executed QMLE with 10^5 steady-state queue-length samples. MHE depends on the observation window H and the discretization parameter Δt. Here we used a different value of H for each experiment (i.e., for each utilization level) such that in the time window $H\Delta t$ we observed a roughly constant number of service events (i.e., approximately 1500) while choosing a discretization step $\Delta t = 0.1$ to ensure an accurate enough ODE solution. For MHE we considered 100 non-overlapping intervals of length $H\Delta t$ which were used as input to the optimization problem (7), in order to compute statistics about the estimation error.

Table II shows the results of the comparison. For each experiment we report the value of H that was used, as well as the utilizations at the station M_2 (i.e., detailed results reporting statistics about station M_3 can be found in the replication package of this paper), denoted by U_2 (this utilization was roughly constant at every system size K), and the initial condition $x(0)$. For each experiment, each station, and each value of K we measured the accuracy of QMLE and MHE, computed as the mean absolute percentage errors between the estimate and the true service demands; for MHE, we report the

	$x(0)=(3,0,0)$ $H=2347, U_2\approx 0.10$		$x(0)=(9,0,0)$ $H=688, U_2\approx 0.30$		$x(0)=(12,0,0)$ $H=521, U_2\approx 0.40$		$x(0)=(19,0,0)$ $H=353, U_2\approx 0.60$		$x(0)=(26,0,0)$ $H=262, U_2\approx 0.80$	
K	QMLE	MHE	QMLE	MHE	QMLE	MHE	QMLE	MHE	QMLE	MHE
1	0.52	9.25 ± 1.03	1.37	9.63 ± 1.06	2.07	7.90 ± 1.01	3.40	6.58 ± 0.81	5.15	4.89 ± 0.69
2	448.30	4.13 ± 0.62	126.54	3.93 ± 0.58	67.18	4.20 ± 0.63	5.46	3.90 ± 0.56	2.33	3.59 ± 0.54
5	184.02	2.26 ± 0.33	60.41	3.02 ± 0.43	42.09	2.76 ± 0.38	8.78	2.07 ± 0.33	1.65	2.06 ± 0.34
10	92.29	1.65 ± 0.27	30.53	1.99 ± 0.31	23.18	1.82 ± 0.31	9.50	2.09 ± 0.30	3.89	1.50 ± 0.24
20	45.18	1.37 ± 0.21	15.01	1.13 ± 0.19	11.32	1.36 ± 0.18	6.41	1.36 ± 0.19	5.81	1.17 ± 0.18
50	18.67	0.74 ± 0.10	6.08	0.81 ± 0.14	4.57	0.78 ± 0.11	2.72	0.81 ± 0.12	5.17	0.73 ± 0.10

TABLE II: Comparison between QMLE and MHE.

	Errors				*Runtimes* (s)			
M	min	avg	95-th	max	min	avg	95-th	max
5	1.60	2.53	4.12	4.50	0.03	0.03	0.03	0.04
10	1.63	2.46	3.28	3.56	0.08	0.08	0.09	0.09
15	1.59	2.63	3.48	4.56	0.18	0.18	0.19	0.19
20	1.62	2.52	3.19	3.82	0.34	0.38	0.48	0.52

TABLE III: Scalability analysis.

95% confidence intervals computed over the 100 independent samples, and the average execution runtime. Relying on the obtained results we make three main observations:

i) In the load-independent (i.e., $K = 1$) case, QMLE tends to outperform MHE at low utilizations, while at higher utilizations ($U_2 \geq 0.60$) the two techniques provide comparable accuracy. Larger MHE errors at $K = 1$ can be explained by the fact that this system size is considerably away from a deterministic regime, as previously discussed. Conversely, here QMLE does not feature the multi-server correction to the BS-AMVA solution. Overall, the largest extremum of the confidence interval for MHE is ca. 16%, still indicating an acceptable performance even for small QNs.

ii) In the load-dependent scenarios, i.e., for all $K > 1$, MHE consistently outperforms QMLE. We attribute this to the aforementioned multi-server correction to BS-AMVA approach. Indeed, for a fixed K the error tends to decrease for larger utilizations, coherently with the fact that when the queue is highly utilized the difference between the dynamics of a single server (i.e., under BS-AMVA) and that of multiple servers with the same overall maximum service rate (i.e., the multi-service correction) become negligible.

iii) As expected for MHE, larger values of the system size K tend to yield more accurate estimates. The results show that this also holds for QMLE.

C. Scalability analysis

Here we further study the effectiveness and scalability of MHE on QNs with randomly generated topologies and parameters. We fixed total number of stations M equal to 5, 10, 15, 20. For each case we generated 20 QNs with random parameters. In particular, the routing probability matrices were randomly generated stochastic matrices (i.e., ensuring a closed QN workload); the number of servers at each station was picked uniformly at random in $\{20, \ldots, 50\}$ while the service rates were chosen randomly from the interval $[10, 50]$. For each such random QN, the initial number of jobs was chosen such that the bottleneck queue attained a utilization of about 0.8, using the approximate formulas presented in [19]. For each random QN we computed the average service demand estimate across all stations and with 100 non-overlapping observation windows with $H = 200$ taken from a sample path.

Table III shows the collected results. In particular for each value of M we report the minimum, average, the 95-th quantile and the maximum error across the 20 random QNs. We observe that the number of stations M does not significantly affect the accuracy of the estimation. As expected, the execution time of MHE grows almost linearly with increasing number of stations in the network. Moreover, for large systems (i.e., $M = 20$) the average time needed for obtaining new estimations by solving the QP optimization, does not exceed the fraction of a second in the worst case (i.e., 0.52s).

V. Conclusion

We have presented a technique to service demand estimation in queueing networks (QNs) using a moving horizon approach which can be used both in the transient and in the steady-state regime. Our method requires the solution of a quadratic programming problem that fits the instantaneous throughputs of each station to minimize the error between the measured queue lengths and the estimated ones. The numerical results demonstrate high accuracy across a wide range of operating regimes and network sizes at a low computational cost, which make it appealing for online use.

In this paper we have considered closed QNs supporting both load independent and load independent stations with exponentially distributed service times and with a single class of jobs. Future work will be concerned with the development of extensions for multi-class QNs with non-exponentially distributed service times, e.g., by fitting service demands against phase-type distributions.

VI. Acknowledgement

This work is partially supported by a DFG Mercator Fellowship (SPP 1593, DAPS2 Project).

REFERENCES

[1] N. Huber, F. Brosig, and S. Kounev, "Model-based self-adaptive resource allocation in virtualized environments," in *Proceedings of the 6th International Symposium on Software Engineering for Adaptive and Self-Managing Systems*, ser. SEAMS '11, 2011, pp. 90–99.

[2] R. Calinescu, C. Ghezzi, M. Kwiatkowska, and R. Mirandola, "Self-adaptive software needs quantitative verification at runtime," *Commun. ACM*, vol. 55, no. 9, pp. 69–77, 2012.

[3] R. de Lemos and et al., *Software Engineering for Self-Adaptive Systems: A Second Research Roadmap*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 1–32.

[4] M. Kowal, I. Schaefer, and M. Tribastone, "Family-based performance analysis of variant-rich software systems," in *Fundamental Approaches to Software Engineering (FASE)*, 2014, pp. 94–108.

[5] D. Arcelli, V. Cortellessa, A. Filieri, and A. Leva, "Control theory for model-based performance-driven software adaptation," in *International Conference on Quality of Software Architectures (QoSA)*, 2015, pp. 11–20.

[6] E. Incerto, M. Tribastone, and C. Trubiani, "Symbolic performance adaptation," in *International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*, 2016, pp. 140–150.

[7] ——, "Software performance self-adaptation through efficient model predictive control," in *Proceedings of the 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE)*, 2017, pp. 485–496.

[8] G. Bolch, S. Greiner, H. de Meer, and K. Trivedi, *Queueing networks and Markov chains: modeling and performance evaluation with computer science applications*. Wiley, 2005.

[9] F. Brosig, S. Kounev, and K. Krogmann, "Automated extraction of palladio component models from running enterprise java applications," in *Proceedings of the Fourth International ICST Conference on Performance Evaluation Methodologies and Tools*, ser. VALUETOOLS '09, 2009.

[10] S. Spinner, G. Casale, F. Brosig, and S. Kounev, "Evaluating approaches to resource demand estimation," *Performance Evaluation*, vol. 92, pp. 51–71, 2015.

[11] M. Awad and D. A. Menasce, "Deriving parameters for open and closed qn models of operational systems through black box optimization," in *Proceedings of the International Conference on Performance Engineering (ICPE)*, 2017.

[12] A. Kalbasi, D. Krishnamurthy, J. Rolia, and M. Richter, "MODE: mix driven on-line resource demand estimation," in *7th International Conference on Network and Service Management*, 2011.

[13] W. Wang and G. Casale, "Bayesian service demand estimation using Gibbs sampling," in *21st International Symposium on Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS)*, 2013.

[14] W. Wang, G. Casale, A. Kattepur, and M. Nambiar, "Maximum likelihood estimation of closed queueing network demands from queue length data," in *Proceedings of the 7th ACM/SPEC on International Conference on Performance Engineering*. ACM, 2016, pp. 3–14.

[15] L. Bortolussi, J. Hillston, D. Latella, and M. Massink, "Continuous approximation of collective system behaviour: A tutorial," *Performance Evaluation*, vol. 70, no. 5, pp. 317–349, 2013.

[16] A. Bemporad, D. Mignone, and M. Morari, "Moving horizon estimation for hybrid systems and fault detection," in *Proceedings of the American Control Conference*, vol. 4, 1999, pp. 2471–2475.

[17] F. Allgöwer, T. A. Badgwell, J. S. Qin, J. B. Rawlings, and S. J. Wright, "Nonlinear predictive control and moving horizon estimation—an introductory overview," in *Advances in control*. Springer, 1999, pp. 391–449.

[18] J. B. Rawlings and B. R. Bakshi, "Particle filtering and moving horizon estimation," *Computers & Chemical Engineering*, vol. 30, no. 10, pp. 1529–1541, 2006.

[19] M. Kowal, M. Tschaikowski, M. Tribastone, and I. Schaefer, "Scaling size and parameter spaces in variability-aware software performance models," in *International Conference on Automated Software Engineering (ASE)*, 2015, pp. 407–417.

[20] M. Tribastone, "A fluid model for layered queueing networks," *IEEE Transactions on Software Engineering*, vol. 39, no. 6, pp. 744–756, 2013.

[21] M. Tribastone, S. Gilmore, and J. Hillston, "Scalable differential analysis of process algebra models," *IEEE Transactions on Software Engineering*, vol. 38, no. 1, pp. 205–219, 2012.

[22] C. E. García, D. M. Prett, and M. Morari, "Model predictive control: Theory and practice—a survey," *Automatica*, vol. 25, no. 3, pp. 335–348, 1989.

[23] T. G. Kurtz, "Solutions of ordinary differential equations as limits of pure Markov processes," in *J. Appl. Prob.*, vol. 7, no. 1, 1970, pp. 49–58.

[24] A. Stefanek, M. C. Guenther, and J. T. Bradley, "Normal and inhomogeneous moment closures for stochastic process algebras," in *10th Workshop on Process Algebra and Stochastically Timed Activities (PASTA)*, 2011.

[25] M. C. Guenther, A. Stefanek, and J. T. Bradley, "Moment closures for performance models with highly non-linear rates," in *Computer Performance Engineering — 9th European Workshop (EPEW) and 28th UK Workshop (UKPEW), Revised Selected Papers*, 2012, pp. 32–47.

[26] J. F. Pérez, G. Casale, and S. Pacheco-Sanchez, "Estimating computational requirements in multi-threaded applications," *IEEE Transactions on Software Engineering*, vol. 41, no. 3, pp. 264–278, 2015.

[27] G. Pacifici, W. Segmuller, M. Spreitzer, and A. Tantawi, "Cpu demand for web serving: Measurement analysis and dynamic estimation," *Performance Evaluation*, vol. 65, no. 6-7, pp. 531–553, 2008.

[28] D. A. Menasce, "Computing missing service demand parameters for performance models." in *Int. CMG Conference*, 2008, pp. 241–248.

[29] P. Cremonesi, K. Dhyani, and A. Sansottera, "Service time estimation with a refinement enhanced hybrid clustering algorithm," in *International Conference on Analytical and Stochastic Modeling Techniques and Applications*. Springer, 2010, pp. 291–305.

[30] A. B. Sharma, R. Bhagwan, M. Choudhury, L. Golubchik, R. Govindan, and G. M. Voelker, "Automatic request categorization in internet services," *ACM SIGMETRICS Performance Evaluation Review*, vol. 36, no. 2, pp. 16–25, 2008.

[31] P. Cremonesi and A. Sansottera, "Indirect estimation of service demands in the presence of structural changes," *Performance Evaluation*, vol. 73, pp. 18–40, 2014.

[32] C. Sutton and M. I. Jordan, "Bayesian inference for queueing networks and modeling of Internet services," *The Annals of Applied Statistics*, pp. 254–282, 2011.

[33] Z. Liu, L. Wynter, C. H. Xia, and F. Zhang, "Parameter inference of queueing models for IT systems using end-to-end measurements," *Performance Evaluation*, vol. 63, no. 1, pp. 36–60, 2006.

[34] F. Baccelli, B. Kauffmann, and D. Veitch, "Inverse problems in queueing theory and Internet probing," *Queueing Systems*, vol. 63, no. 1-4, p. 59, 2009.

[35] A. N. Tantawi and D. Towsley, "Optimal static load balancing in distributed computer systems," *J. ACM*, vol. 32, no. 2, pp. 445–465, 1985.

[36] M. Awad and D. A. Menascé, "Performance model derivation of operational systems through log analysis," in *Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS), 2016 IEEE 24th International Symposium on*. IEEE, 2016, pp. 159–168.

[37] D. Gillespie, "Exact stochastic simulation of coupled chemical reactions," *Journal of Physical Chemistry*, vol. 81, no. 25, pp. 2340–2361, December 1977.

[38] I. Dunning, J. Huchette, and M. Lubin, "Jump: A modeling language for mathematical optimization," *SIAM Review*, vol. 59, no. 2, pp. 295–320, 2017.

[39] I. ILOG, "Cplex optimizer," 2012. [Online]. Available: http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer

[40] Y. Bard, "Some extensions to multiclass queueing network analysis," in *Proceedings of the Third International Symposium on Modelling and Performance Evaluation of Computer Systems: Performance of Computer Systems*. North-Holland Publishing Co., 1979, pp. 51–62.

[41] P. Schweitzer, "Approximate analysis of multiclass closed networks of queues," *J. ACM*, vol. 29, no. 2, 1981.