Approximating Closed Queueing Networks in Semi-Markov Random Environments

Yaqi Zhou
Department of Computing
Imperial College London, UK
yaqi.zhou22@imperial.ac.uk

Matthew Sheldon
Department of Computing
Imperial College London, UK
matthew.sheldon20@imperial.ac.uk

Giuliano Casale
Department of Computing
Imperial College London, UK
g.casale@imperial.ac.uk

***Abstract*—Modeling queueing networks in random environments is a challenging problem that finds application in routing, reliability analysis, and evaluation of systems processing bursty workloads. In this paper, we introduce a method to evaluate closed queueing networks in random environments that can deal with non-Markovian transitions among environment stages and accurately approximate time-averaged performance metrics. Our technique supports both state-independent and state-dependent random environments, the latter allowing us to analyze challenging queueing network models with bursty service processes. Using simulation on representative case studies, we show that our technique provides low approximation errors. We further demonstrate the applicability of our result to a mobility case study demonstrating the applicability of the proposed approach.**

***Index Terms*—Semi-Markov random environments, mean field approximation, performance analysis**

I. Introduction

Queueing network models have been widely used for performance analysis in cloud computing, software architecture, telecommunication systems, and several other areas [1], [2]. A closed queueing network model typically consists of a set of jobs circulating among delay and queueing stations. The service processes at the stations are modeled by probability distributions, often exponential distributions. However, real-world workloads can show heterogeneous features and be affected by external events, such that service rates and the underpinning distributions may vary over time, requiring specialized performance evaluation methods.

Time-varying characteristics of a queueing network can be modeled in terms of a random environment (RE) [3], [4], which stochastically assigns the service characteristics of the stations. An RE is itself a stochastic model where states, called *stages*, determine the service characteristics of all the stations in the closed queueing network. In a *state-independent* RE, the active stage determines the current service distribution irrespective of the state of the queueing network; numerical approximations capable of addressing this case when the RE model is Markovian have been available for some time [5], [6]. Conversely, in a *state-dependent* RE, the active stage can itself be affected by the state of the queueing network, for example disabling further RE transitions whenever a given station is idle. The latter feature enables us in particular to model Markov-modulated service processes by means of an RE that modulates the service rate of the station only in non-idling periods. Approximating non-renewal processes of this kind is generally difficult, yet it remains critical to the accuracy of performance predictions [7]–[10].

The blending method proposed in [5], [11] defines iterative approximation methods for queueing models in state-independent REs by means of fluid ordinary differential equations (ODEs). However, the method has not been developed in full generality for state-dependent REs; only a specific proof-of-concept for a two-stage two-station model is illustrated in [11, Fig. 6]. Thus, the first contribution of this work is to extend the blending algorithm to the state-dependent RE case and establish its effectiveness in a much more general case with an arbitrary number of stages and stations. The resulting extension considers each RE stage separately and conducts transient analysis during stage transitions to derive approximate queue lengths in each stage. It then derives mean performance metrics for the model as a whole, such as mean queue lengths and utilization. Experimental results show that our generalized technique can effectively calculate performance metrics, achieving average errors below 4% in most experiments.

A second major contribution of this work is to generalize the blending theory to non-Markovian REs, specifically those described by semi-Markov processes (SMPs). This extension allows for the first time to address closed queueing networks modulated by general non-Markovian distributions, broadening the expressivity of the models and the ability to support real-world application scenarios. In particular, since a state-dependent RE can approximate a switching service process, this extension implies that our generalized blending method can also solve closed queueing networks parameterized by a class of SMPs. We illustrate this by showing that service processes modulated by Erlang, gamma or lognormal distributions are accurately approximated. To the best of our knowledge, no analytical approximation exists that can approximate non-Markovian closed networks with the size of the models we consider, which feature several tens of jobs. We illustrate the applicability of this extension through a queueing case study using a real-world mobility traffic dataset from the city of Oslo that involves hundreds of stations.

The rest of the paper is structured as follows. Section II gives a brief example to introduce closed queueing networks

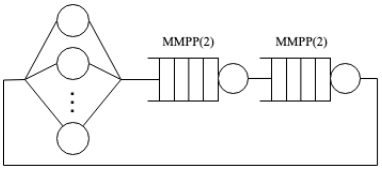


Fig. 1: A closed queueing network model containing 1 delay and 2 switching queues (SQ1 and SQ2).

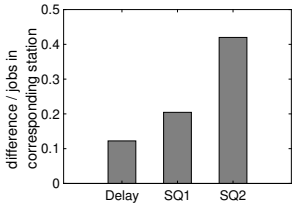


Fig. 2: Difference in performance for queueing network example with exponential and MMPP service.

in random environments. Section III and Section IV present the original blending method and our proposed generalization. Section V provides the numerical results and a case study on mobility. Section VI reviews related work on modeling random environments, followed by conclusions in Section VII.

II. Random Environment Example

To better illustrate the definition of the RE, we first give an example of a Markovian RE. A Markovian RE is a state-dependent RE and the service rate in each station is defined by the corresponding stage. The Markov-modulated Poisson process (MMPP) is a common model to describe this kind of RE. A two-stage MMPP can be defined by the following two matrices [12]

$$\boldsymbol{Q} = \begin{pmatrix} -r_1 & r_1 \\ r_2 & -r_2 \end{pmatrix}, \quad \boldsymbol{\Lambda} = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} \tag{1}$$

The matrix $\boldsymbol{Q}$ is an infinitesimal generator for the continuous-time Markov chain (CTMC) that describes the hidden transitions between the stages. The diagonal elements in matrix Λ are the arrival rates in each stage. $\boldsymbol{Q}$ can be seen to modulate the service rates and λ_i is the service rate in the corresponding MMPP state. The transition times are exponentially distributed. Thus, using the MMPP as the service process for a queue, we can model the service behaviors under an RE.

For multiple MMPP queues, we assume that the stage of one queue will not impact the stage of another queue in the network. We generate all combinations of the stages from different queues and $\boldsymbol{Q}$ can then modulate the service parameters. For example, for a queueing network model consisting of a delay station and two MMPP queues shown in Fig. 1, if the service process of these two queues is both modeled by a two-stage MMPP, the RE of this model has 4 stages in total. We have the stage set $S = \{(1,1,1),(1,1,2),(1,2,1),(1,2,2)\}$; for example, $S = (1,2,2)$ describes a RE stage where the switching queues are all in stage 2 of their corresponding MMPP. The RE has an infinitesimal generator

$$\boldsymbol{E} = \begin{pmatrix} -r_{11} & r_1' & r_1 & 0 \\ r_2' & -r_{12} & 0 & r_1 \\ r_2 & 0 & -r_{21} & r_1' \\ 0 & r_2 & r_2' & -r_{22} \end{pmatrix} \tag{2}$$

where r_1 and r_2 denote the transition rates of the RE modulating the first queue, r_1' and r_2' represent the ones modulating the second queue, and $r_{ij} = r_i + r_j'$. Suppose r_1, r_2, r_1' and r_2' are set to 1, 0.2, 1 and 0.5. $\lambda_1, \lambda_2, \lambda_1'$ and λ_2' are 1, 2, 0.1 and 10 respectively. We compare the results with the case when there is no RE and the service rates for the queues are 7/6 and 4 derived by averaging the service rates in MMPP using the equilibrium probability. Fig. 2 shows the mean queue length difference divided by the jobs in the corresponding station, indicating that non-exponential service can lead to large variations in performance metrics. The method proposed in the next sections tackles, among others, the problem of approximating models of this kind.

III. Background

A. Mean-Field Approximation

An effective methodology to analyze the transient behaviors of population models is mean-field approximation [13]. In this method, the population size and its flows are modeled as continuous and their dynamics are captured by means of a set of ODEs. By solving the ODEs, we can obtain the expected performance metrics over time and also approximate their equilibrium values.

The mean-field approximation method can be used to deal with a wide range of queueing networks. When the servers have exponential service processes, Kurtz's fluid limit has been shown to converge in mean for networks of processor sharing queues, delay nodes, and in single-class first-come first-serve (FCFS) stations [6], [14]. The above results also support service processes modeled by phase-type distributions. This method has also been used in the LINE queueing network solver to handle multi-class FCFS scheduling [15].

Throughout, we consider a single class of jobs moving in a closed queueing network. The queueing stations use the FCFS scheduling policy and the servers have exponential service distributions with a rate controlled by the active RE stage. Let M and $\boldsymbol{s} = (s_1, s_2, \ldots, s_M)^T$ denote the number of stations in the queueing network and the number of servers in each station, respectively. For an infinite server station, we set $s_i = N$, where N is the total number of jobs in the model. Let $\boldsymbol{R}$ be the routing probability matrix so that $\boldsymbol{R}_{ij}$ indicates the probability that a job is sent to station j after completing service at station i. μ_i is the service rate at station i and x_i is the number of jobs at station i. When we only consider the change in the number of jobs at station i, we can formulate the following expression to describe the rate of change at time t [11]:

$$F(x_i(t)) = \sum_{j=1}^{M} -\boldsymbol{R}_{ij}\mu_i \min(s_i, x_i(t)) + \boldsymbol{R}_{ji}\mu_j \min(s_j, x_j(t)) \tag{3}$$

where $x_i(t)$ is the number of jobs in station i at time t. In the above equation, the negative term represents the outgoing job flow rate to other stations and the positive term describes the arriving job flow rate from others. Let $\boldsymbol{x}(t)$ be the state vector $(x_1(t), x_2(t), \ldots, x_M(t))^T$, the change in the jobs at each station can be represented by

$$F(\boldsymbol{x}(t)) = \sum_{i,j=1}^{M} (\boldsymbol{I} - \boldsymbol{R}^T)(\boldsymbol{\mu} \cdot \min(\boldsymbol{x}(t), \boldsymbol{s})) \tag{4}$$

and the ODEs are then defined as

$$\frac{d\boldsymbol{x}(t)}{dt} = F(\boldsymbol{x}) \tag{5}$$

In practice, the initial condition for the ODEs to be solved is set as $x_i(0) = N/M$.

B. Markovian Random Environments

To model a Markovian random environment, when the number of stages is K, we consider the following infinitesimal generator to describe the transition rate between stages

$$\boldsymbol{E} = \begin{pmatrix} -\lambda_1 & \lambda_{12} & \cdots & \lambda_{1K} \\ \lambda_{21} & -\lambda_2 & \cdots & \lambda_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_{K1} & \lambda_{K2} & \cdots & \lambda_{KK} \end{pmatrix} \tag{6}$$

where $\lambda_k = \sum_{e=1}^{K} \lambda_{ke}$. λ_{ke} is the transition rate from stage k to e and λ_k is the rate of leaving stage k. Combining and solving the two equations $\boldsymbol{\pi E} = \boldsymbol{0}$ and $\boldsymbol{\pi e} = 1$ where $\boldsymbol{e}$ is a vector with all elements being 1, one can obtain the equilibrium probability of this random environment. Given the equilibrium probability $\boldsymbol{\pi}$, the embedded probability p_{ke} which represents the probability of jumping from stage k to stage e can be calculated by [5]

$$p_{ke} = \frac{\pi_k \lambda_{ke}}{\sum_{h \neq e} \pi_h \lambda_{he}} \tag{7}$$

where π_h is the equilibrium probability in stage h. In our assumptions, the stage switches in each station are not synchronized since the random environment of one station is independent of the others, i.e., when one station changes the stage, the remaining stations may not change. For this reason, if a queueing network model contains more than one station whose service process will be affected by the random environment, we combine each possible state of each station. The total number of combinations determines the number of stages in this model. $S = \{(c_1, c_2, \ldots, c_M) | 1 \leq c_k \leq K_i, k = 1, \ldots, M\}$ is used to represent combined stages, where K_i is the number of stages at station i.

C. Blending Method

The blending method is an iterative fixed-point method that seeks to approximate steady-state queue length values embedded at stage transition times. Let $f_k(t)$ be the probability density function of the sojourn time in stage k, the blending algorithm derives the time-averaged queue length L_i^k at station i in stage k by Laplace transform and has the following expression [11]

$$L_i^k = \lim_{T \to \infty} \frac{1}{T} \int_0^T x_i(t) f_k(t) dt \tag{8}$$

and the utilization U_i^k of the reference station i is

$$U_i^k = \lim_{T \to \infty} \frac{1}{T} \int_0^T \min(s_i, x_i(t)) f_k(t) dt \tag{9}$$

Thus the mean queue length and the utilization across all the stages at equilibrium can then be approximated by

$$L_i = \sum_k \pi_k L_i^k, \quad U_i = \sum_k \pi_k U_i^k \tag{10}$$

The initial values of the ODEs, which is the queue length at the instant of switching stages, will greatly affect the accuracy of the transient behaviors at the beginning of the system entering the next stage. For a stage i, there might be multiple stages that can switch to this stage. The blending method considers all possible stages and adopts an iterative approach to approximate this value. In each iteration, the initial value $x_i^e(0)$ in stage e at iteration n can be computed by

$$x_i^e(0) = \sum_{k \neq e}^{K} p_{ke} L_i^k(n-1) \tag{11}$$

where $n = 1, 2, \ldots$ is the number of the current iteration. In most cases, as the number of iterations increases, the difference in initial values between consecutive iterations will decrease and the approximated queue length will also converge gradually. However, there exist scenarios when the queue lengths may not converge. Non-convergent cases in heavy-load and corresponding analysis can be found in [5]; even in these scenarios, an average approximation can be returned by the model by averaging across a time-window of iterations.

IV. Generalized Blending Methodology

A. Semi-Markov Random Environments

We now extend the original blending algorithm to include random environments modulated by more general distributions and use a semi-Markov process (SMP) to define the random environments.

Given the initial state probability $\boldsymbol{v}_0$ and a semi-Markov kernel $\boldsymbol{H}(t)$, one can define a specific SMP. Let Z_n denote the current stage when n transitions have occurred and Y_n represent the sojourn time before the nth transition takes place. This semi-Markov kernel gives the probability of transiting to stage e when the current stage is k and is defined by [16]

$$\boldsymbol{H}(t) = (Pr\{Y_n \leq t, Z_n = e | Z_{n-1} = k\})_{ke} \tag{12}$$

In our context, we adopt the simulation mechanism which is to draw samples from every possible sojourn time density and choose the stage with minimum time as the next one according to [17]. Let $F_{ke}(x)$ and $f_{ke}(t)$ be the cumulative distribution function (cdf) and probability density function (pdf) of the sojourn time when transitioning from stage k to

stage e, respectively. The derivative of the semi-Markov kernel entries is given by

$$h_{ke}(t) = \frac{dH_{ke}(t)}{dt} = f_{ke}(t) \prod_{h \neq e} (1 - F_{kh}(t)) \tag{13}$$

where $f_{ke}(t)$ can be regarded as the probability that the SMP will transition to stage e and $\prod_{h \neq e} (1 - F_{kh}(t))$ describes the case when it has not jumped to stage h before time t [17].

The sojourn time distribution in each stage k can then be transformed to calculate the distribution of minimum order statistics in these non-identical distributions, which is given by [18]

$$F_k(t) = 1 - \prod_{e=1}^{|S|} (1 - F_{ke}(t)) \tag{14}$$

By substituting $f_k(t)$ in (8) with the derivative of $F_k(t)$ in (14), we now obtain the time-averaged queue length in stage k of the SMP random environment.

The approach to deriving the infinitesimal generator of the SMP environment is similar to the MMPP case. We need first to derive the rate of the transitions between the two stages. If the transition is from state k to e, the mean transition time is approximated by the expectation of the corresponding distribution. We then use the reciprocal of the time to obtain the rate λ_{ke} when $f_{ke}(t) \neq 0$

$$\lambda_{ke} = \frac{1}{\int_0^\infty t f_{ke}(t) dt} \tag{15}$$

To obtain the equilibrium probability of each stage, we then construct the infinitesimal generator $\boldsymbol{E}$ as shown in (6).

B. State-Dependent Random Environments

For the state-dependent RE, the transition rates are approximated according to the probability that the stage switches are enabled [11], which can be represented by the utilization of an arbitrary reference station. In the exponential transition case, the modification is to multiply the utilization of the reference station with the transition rates λ and use the updated rate as the parameter. Since the transitions are not only exponentially distributed in SMP, a straightforward multiplication is not applicable.

As per (9), the utilization of the reference station i in stage k is U_i^k. The pdf of the sojourn time in stage k at iteration n is approximated as

$$f'_k(x) = U_i^k f_k(U_i^k x) \tag{16}$$

By (16), the mean of the updated distribution is the previous mean divided by U_i^k under the assumption that the support of the pdf is $[0, \infty]$. Since $\int_0^\infty x f_k(x) dx = {\lambda_k}^{-1}$, we can derive the mean of the updated distribution as follows:

$$\int_0^\infty x f'_k(x) dx = \int_0^\infty U_i^k x f_k(U_i^k x) dx$$

Substituting $t = U_i^k x$, we have

$$\int_0^\infty x f'_k(x) dx = \int_0^\infty t f_k(t) \frac{1}{U_i^k} dt = \frac{1}{\lambda_k U_i^k}$$

We can now approximate the time-averaged queue length by

$$L_i^k = \lim_{T \to \infty} \frac{1}{T} \int_0^T U_i^k x_i(t) f_k(U_i^k t) dt \tag{17}$$

After changing the integration formula for the queue lengths in each stage, each row in the infinitesimal generator $\boldsymbol{E}$ also needs modifications. The transition rate λ is now approximated as $U_i \lambda$. Each row of matrix $\boldsymbol{E}$ is then multiplied by the utilization of the corresponding stage. The updated matrix is as follows:

$$\boldsymbol{E}' = \begin{pmatrix} -U_i^1 \lambda_1 & U_i^1 \lambda_{12} & \cdots & U_i^1 \lambda_{1K} \\ U_i^2 \lambda_{21} & -U_i^2 \lambda_2 & \cdots & U_i^2 \lambda_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ U_i^K \lambda_{K1} & U_i^K \lambda_{K2} & \cdots & -U_i^K \lambda_K \end{pmatrix} \tag{18}$$

The processes for the state-dependent blending algorithm are shown in Algorithm 1. Lines 1 to 4 are to calculate the infinitesimal generator and the sojourn time cdf for each combined stage. Lines 5 to 13 illustrate the procedure of each iteration. In the inner loop from lines 7 to 11, each stage is traversed and the time-averaged queue length is approximated. The infinitesimal generator $\boldsymbol{E}'$ and corresponding probability will be updated after the inner loop ends. In practice, we use the utilisation of the queue instead of a single server for the multi-server queue.

V. Numerical Results

A. Methodology

We adopt simulation for validation. The random environments are modelled by queueing Petri nets using Java Modeling Tools (JMT) [19]. Each simulation has 10^6 samples.

Let $\boldsymbol{L}$ and $\boldsymbol{L}'$ be the vector of queue lengths in each station derived by simulation and the blending method, respectively. We use the following δ to measure the error compared to the simulation results

$$\delta = \frac{||\boldsymbol{L} - \boldsymbol{L}'||_1}{2N} \tag{19}$$

The $2N$ factor ensures that, in the worst case where the two solutions place the whole probability mass on entirely different stations, then $\delta = 1$.

B. Running Example

We first illustrate how to process the model discussed in Section II which contains 1 delay and 2 MMPP switching queues in the state-dependent environment. Its infinitesimal generator for the state-independent case is shown in (2). We generalize the transition distribution to SMP. For switching queue i, the matrix describing the stage transition in a single station is

$$\boldsymbol{H}_i(x) = \begin{pmatrix} 0 & F_1^i(x) \\ F_2^i(x) & 0 \end{pmatrix} \tag{20}$$

Algorithm 1 Blending algorithm

Input:

$\boldsymbol{R}$: the routing matrix denoting the topology of the queueing network

$\boldsymbol{F}$: an array containing transition matrices between stages for each station in the network. Each entry in the matrix is the cdf of sojourn time

$iter_{max}$: the maximum number of iterations

ϵ: the error tolerance

Output:

$L_i, i = 1, \ldots, M$

1: Derive the combinations of stages S
2: Construct the infinitesimal generator $\boldsymbol{E}$ by (15) and (6), which describes the transition between stages in S
3: Compute the sojourn time distribution $F_k(x)$ for each stage by $F_k(t) = 1 - \prod_{e=1}^{|S|}(1 - F_{ke}(t))$
4: $iter = 0$
5: **while** $\epsilon > \max|\frac{L_i(iter) - L_i(iter-1)}{L_i(iter-1)}|$ **and** $iter < iter_{max}$ **do**
6: $iter = iter + 1$
7: **for** each stage $k \in S$ **do**
8: Update the pdf of sojourn time distribution by $f'_k(x) = U_j^k f_k(U_j^k x)$
9: Solve the ODEs:
$$\frac{d\boldsymbol{x}^k(t)}{dt} = F(\boldsymbol{x}^k(t))$$
where $\boldsymbol{x}^k(0) = \sum_{e \neq k}^{|S|} p_{ke} L_i^e(iter - 1)$
10: Compute queue lengths and utilization of the reference station j by (8) and (9)
11: **end for**
12: Calculate matrix $\boldsymbol{E}'$ by (18) and the corresponding equilibrium probability and embedded probability
13: **end while**
14: Compute the mean queue lengths and the utilization:
$L_i = \sum_k \pi_k L_i^k, \quad U_j = \sum_k \pi_k U_j^k$
15: **return** $L_i, U_j, i, j = 1, \ldots, M$

TABLE I: The parameters of the transition in the two switching queues. The columns in mean and SCV correspond to the moments of $F_1^i(x)$ and $F_2^i(x)$

	mean		SCV	
Switching queue 1	1	0.2	0.5	2
Switching queue 2	1	0.5	0.5	2

where $F_1^i(x)$ and $F_2^i(x)$ are assumed to be lognormal cdf in this example. The parameters used are listed in Table I.

Combining the stages of stations, we derive the set S and the following matrix describing the transition probability

$$\boldsymbol{H}(x) = \begin{pmatrix} 0 & F_1^2(x) & F_1^1(x) & 0 \\ F_2^2(x) & 0 & 0 & F_1^1(x) \\ F_2^1(x) & 0 & 0 & F_1^2(x) \\ 0 & F_2^1(x) & F_2^2(x) & 0 \end{pmatrix} \tag{21}$$

Each row corresponds to a combined stage in S. Consequently, we can generate the sojourn time cdf and pdf in each stage. For example, in stage $(1, 1, 1)$, the cdf $F_1(x)$ is $F_1(x) = 1 - (1 - F_1^2(x))(1 - F_1^1(x))$. If the utilization of the reference station in stage k is denoted as U^k, the pdf of the sojourn time in stage $(1, 1, 1)$ is then modified to $f'_1 = U^k f_1(U^k x)$ and the infinitesimal generator of the combined stages is

$$\boldsymbol{E}' = \begin{pmatrix} -U^1 r_{11} & U^1 r'_1 & U^1 r_1 & 0 \\ U^2 r'_2 & -U^2 r_{12} & 0 & U^2 r_1 \\ U^3 r_2 & 0 & -U^3 r_{21} & U^3 r'_1 \\ 0 & U^4 r_2 & U^4 r'_2 & -U^4 r_{22} \end{pmatrix}$$

where $r_{ij} = r_i + r'_j$. In practice, we choose the second switching queue as the reference station. The matrix $\boldsymbol{E}'$ can be calculated in our example as

$$\boldsymbol{E}' = \begin{pmatrix} -2 & 1 & 1 & 0 \\ 0.2132 & -0.3198 & 0 & 0.1066 \\ 5 & 0 & -6 & 1 \\ 0 & 1.1655 & 0.4662 & -1.6317 \end{pmatrix}$$

We set the number of jobs in the network as 10 and δ is 3.12%. The time spent on simulation is 229.68 seconds while the blending method takes 25.256 seconds. The blending algorithm takes 23 iterations to converge.

C. Numerical Results

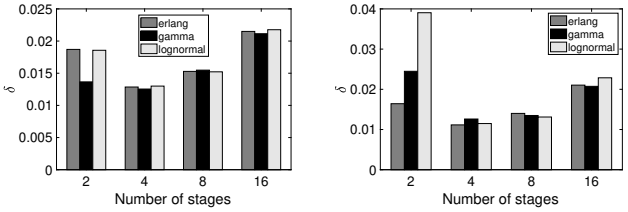
To validate our proposed algorithm, we now use random models with different numbers of stations and stages. If not specified, we set the number of servers in the switching queue to $1, 4, 7$ and 10. The number of jobs in the entire network varies at intervals of 10, ranging from 10 to 100. Erlang, gamma and lognormal distributions are considered to model the transitions for each model. For each model, the mean and SCV of the same transition modeled by different distributions are the same. The values of SCV are set to 0.2, 0.5 and 0.8 randomly.

1) Cyclic models with only delay and switching queues: First, we use a cyclic topology to evaluate Algorithm 1. Table II illustrates the parameters of each model. The number of stations ranges from 2 to 5 and the number of stages in $\boldsymbol{E}$ varies as $2, 4, 8$ and 16. Fig. 3 shows the results of the four models in both state-independent and state-dependent cases. δ is averaged across the same model for different total numbers of jobs and servers. It can be seen that the error is below 4% in all models.

2) Models with servers down in a stage: Now we try to address the case when the servers are idle in a stage. Setting the service rate to 0 might incur a singular infinitesimal generator for the state-dependent case because the jobs may all accumulate in the idle queue and the utilisation of the working queue can be 0. We use 10^{-3} as the service rate in that stage instead. The topology of the models we use is the same as the first and second models in table II. We refer to these two models as D-SQ* and D-SQ-SQ*. Figure 4 shows the error of these two models with Erlang and gamma transitions respectively when the switching queue only has a single server.

TABLE II: Description of the cyclic models. D and SQ denote a delay station and a switching queue, respectively. The models we use are all cyclic queueing networks. The station sequences in the table correspond to the arrangement of stations in the models.

Number of Stages	Model
2	D-SQ
4	D-SQ-SQ
8	D-SQ-SQ-SQ
16	D-SQ-SQ-SQ-SQ



(a) state-independent (b) state-dependent

Fig. 3: Numerical results of cyclic models with only delay and switching queues

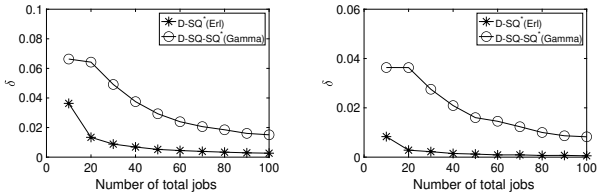
For both models, the error tends to decrease as the number of jobs increases and the error can be generally satisfactory in both the state-independent and the state-dependent cases.

3) Other models: We also experimented on a network which contains a delay, a switching queue and a non-switching queue, and one where there is one delay connected with two switching queues that run in parallel. Each switching queue also contains two stages and the transitions are modeled by the Erlang, gamma or lognormal distributions. For simplicity, we refer to the former one as the D-SQ-Q model and the latter as the parallel model. Table III summarizes the numerical results derived. Similar to the results in subsection V-C1, the average error remains below 4%.

4) Sensitivity analysis on SCV: To better analyze the impact of the distribution SCV values, we now keep the mean of distributions the same and vary the SCV. The model we use has the topology D-SQ-Q and switching queues have a single server. The service rates of the switching queue in each are 10 and 0.1 respectively. We keep the mean of the transition to 1 and 0.1 and the transitions are modeled by the gamma distribution. The SCV values used to fit are 0.5, 2, 10 and 20. Fig. 5 shows the error under different SCV values.

D. A Case Study - Modeling the Oslo Bysykkel System

Lastly, we apply blending to a vehicle-sharing system model with SMP-modulated service rates. The motivation of using REs in this context is to model inhomogeneities in customer arrivals at different stations. Such inhomogeneity is common in vehicle-sharing systems (VSS) [20], [21]. Modeling changes in customer arrivals, and their impact on the system, may be of interest to mobility providers. For example, surge pricing is often used to mitigate and exploit inhomogeneity [22], [23].



(a) state-independent (b) state-dependent

Fig. 4: δ variation with number of jobs for model D-SQ* and D-SQ-SQ* with single server

TABLE III: Average δ of the D-SQ-Q and parallel model

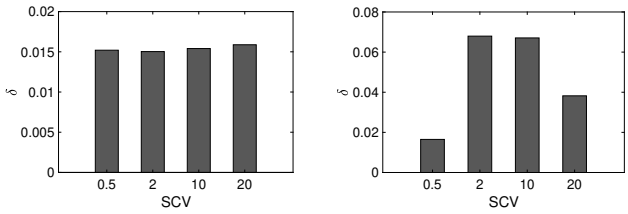
	Dist.	state-independent	state-dependent
Parallel	Erlang	0.0138	0.0169
	Gamma	0.0139	0.0170
	Lognormal	0.0139	0.0157
D-SQ-Q	Erlang	0.0225	0.0206
	Gamma	0.0313	0.0206
	Lognormal	0.0229	0.0180

Changes in customer activity can be divided into local and global activity cycles. In [20], Kaltenbrunner, *et al.*, analyze the Bicing bikeshare system in Barcelona, and detect frequent changes in local customer activity, independent of larger global changes. These changes in demand have been attributed to a large variety of factors, such as multimodal transportation and rush-hour effects [20]. Given the wide variety of causes of customer demand changes, which might not be evident based on the data, SMPs are a flexible tool as they allow for a general distribution of changes between stages.

Queueing models, particularly closed queueing networks, are standard models for vehicle-sharing [24]–[27]. In a typical VSS model, $-/M/1$ queues are used to model points at which customers can access vehicles, such as a bike rack or a general location of a city. $-/G/\infty$ delays are used to model vehicles in transit between stations. In this case, the vehicles in the system are modeled as jobs in the network. The fleet size corresponds to the job population. Customers are assumed to be impatient, they leave if they find the queue empty. Therefore, the service process of the jobs at $-/M/1$ stations represents the process at which customers arrive to access a certain vehicle. It is important to differentiate the process of vehicle arrivals, which is modeled by arrivals into the queue, from the process of customer arrivals, which is modeled by the service rate of vehicles in a queue.

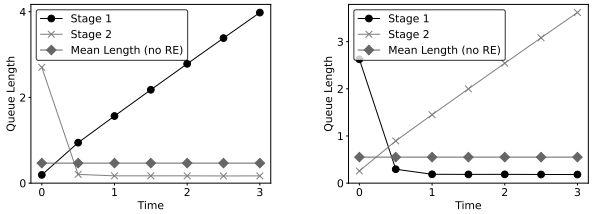
Next, we present a queueing model of the Bysykkel network in Oslo. This model has been parameterized by a sample of $248,041$ Saturday trips [1], taken between 10:00 am and 5:00 pm, from this bikeshare network. The total number of stations is equal to 285, and therefore it is not tractable to model independent changes in demand for each station. Therefore, in each experiment, only one station changes its service rate according to the SMP-modulated random environment, and

[1]https://oslobysykkel.no/en/open-data/historical



(a) state-independent (b) state-dependent

Fig. 5: Numerical results of SCV sensitivity analysis



(a) Olaf Ryes Plass (b) Kirkeristen

Fig. 6: Trajectories for two Oslo stations

the rest use the mean service time based on the trip data, in order to model inhomogeneity of customer arrivals at a given station. The SMP process is fitted based on a variation of the backward-forward algorithm, which bears a resemblance to the algorithm presented in [28]. We also present the baseline case, without any changes in the service rate. Every pair of stations in the model is assumed to be connected by a delay. Two stages are used in the random environment model, which typically corresponds to one stage with a very low customer arrival rate at each station, and another with a very high customer arrival rate. A simplifying assumption is that we do not include rebalancing in our model, which is a standard method for preventing imbalances in a vehicle sharing system [27], [29]. Furthermore, 2 out of the 290 stations, with different IDs for a single station name, were excluded from the analysis.

In Figure 6, we present the trajectories for two Oslo stations, Olaf Ryes Plass and Kirkeristen, within each stage. Both results show significant changes in the queue length between stages, with the system switching between high-load and low-load as the RE transitions. A surprising result is that customer arrival variability can improve access to vehicles at a given station. At Olaf Ryes Plass, the mean queue length is 0.467, without modeling the random environment. With modeling the random environment, the overall mean queue length increases to 0.496, which indicates a 6.07% increase in queue length and utilization. However, this effect is not universal, as placing the RE at Kirkeristen yields a 14.9% decrease in mean queue length.

In Table IV, we show the variations in the trip rate based on the SMP-modulated station. Let e_i be the visit ratio, T_i the throughput at station i, and S be the set of indices of all queueing stations. Then, the trip rate is calculated as,

$$Trips = \frac{\sum_{i \in S} T_i}{\sum_{i \in S} e_i} \tag{22}$$

TABLE IV: RE impact on trips (Oslo)

	Trips (Stage 1)	Trips (Stage 2)	Avg Trips
None	—	—	86.4328
Olaf Ryes Plass	84.8415	89.5909	86.4327
Kirkeristen	89.3235	85.0651	86.4327

When changes in Olaf Ryes Plass are modeled, the trip rate in the high customer arrival stage is 5.6% higher than the low customer arrival stage. Kirkeristen shows similar results with 5.1% higher total trips in the high customer arrival stage vs its low arrival stage. Despite the changes in the queue length at the RE-modulated station and a strongly disproportionate impact on the number of trips at each stage, the average long-term trip rate remains nearly identical in the default and RE-modulated cases. We interpret this to be due to the relatively fast changes in the RE compared to the characteristic timescales of the overall system dynamics; fast RE changes can retain the same mean equilibrium performance albeit the performance conditional on RE stage is different [11]. This indicates that RE modeling on this time scale is more useful for local and conditional analysis in the system, as opposed to it being a necessity for large-scale modeling.

VI. Related Work

Non-renewal processes such as the MMPP or the Markovian arrival process (MAP), are often used to describe temporally-dependent service times [8], [30], [31]. The work in [30] decomposes the analysis of an MMPP/M/1 queueing system into a transient analysis problem for the M/M/1 queue and approximates lower and upper bounds on the response time distribution. For open queueing networks with MAP processes, [8] uses a moment-based decomposition method to analyze performance metrics. Closed queueing networks are instead tackled analytically by the bounding method in [31], which utilizes decomposition and linear programming to obtain bounds on performance metrics.

MMPPs and MAPs may both be seen as special cases of SMP, which relaxes the restriction on transition times among stages being exponentially distributed. SMP transitions can instead be modeled by arbitrary probability distributions. Due to its non-Markovian nature, SMPs can model a wide range of application scenarios, such as real-world failure and repair process in the dependability analysis that is known not to be necessarily Markovian [32]. However, the complexity of analyzing queueing network models in random environments is made visible by the limited literature on this topic, which is particularly scarce in number for closed queueing network models. General methods for modeling queueing networks in the SMP random environment may include non-Markovian analysis methods [33]. However, closed models with tens or hundreds of jobs have intractably large state spaces that hamper the application of direct numerical methods. To address these scenarios, [34] provides an average environment method to analyze queueing networks in a random environment. How-

ever, the accuracy of these approximations is high only when the rate of environment transitions is much faster than the service rate [5]. The work in [16] deals with the SMP random environment using the decomposition method by transforming it to MMPP or a renewal process, which depends on the SCV value of this SMP. Yet, the method is available only for SMPs with 2 stages. More recent work includes [35] which analyzes $M/GI/\infty$ queues in a semi-Markov environment and obtains the probability distribution of the number of jobs in the system.

Compared to the above approaches, the proposed technique does not rely on specific assumptions on the SMP transitions, which can be general. However, the proposed approach requires, as in the original blending method, that a fluid ODE solver is available for the associated queueing network model.

VII. Conclusion

In this paper, a method to analyze the closed queueing networks in non-Markovian random environments is proposed as a generalization of the blending algorithm. This iterative fixed-point method is based on mean-field approximation and can estimate mean performance metrics. We have conducted experiments on models with up to 16 stages, showing the good levels of accuracy of the approach. We also analyze a real-world vehicle-sharing system, which illustrates the effective application of the proposed algorithm.

Future work may generalize the blending method to multi-class models in the semi-Markov RE. Another extension may be to explore a more efficient way to analyze the sojourn time distribution in the SMP than symbolic representations.

References

[1] S. Balsamo, V. D. N. Personè, and P. Inverardi, "A review on queueing network models with finite capacity queues for software architectures performance prediction," *Perf. Eval.*, vol. 51, no. 2-4, pp. 269–288, 2003.

[2] E. Jafarnejad Ghomi, A. M. Rahmani, and N. N. Qader, "Applying queue theory for modeling of cloud computing: A systematic review," *Concurrency and Computation: Practice and Experience*, vol. 31, no. 17, p. e5186, 2019.

[3] C. S. Kim, A. Dudin, V. Klimenok, and V. Khramova, *Performance analysis of multi-server queueing system operating under control of a random environment*. IntechOpen, 2010.

[4] N. Bambos and G. Michailidis, "Queueing and scheduling in random environments," *Advances in Applied Probability*, vol. 36, no. 1, pp. 293–317, 2004.

[5] G. Casale, M. Tribastone, and P. G. Harrison, "Blending randomness in closed queueing network models," *Perf. Eval.*, vol. 82, pp. 15–38, 2014.

[6] J. F. Pérez and G. Casale, "Line: Evaluating software applications in unreliable environments," *IEEE Trans. on Reliability*, vol. 66, no. 3, pp. 837–853, 2017.

[7] S. Gupta and A. D. Dileep, "Long range dependence in cloud servers: a statistical analysis based on google workload trace," *Computing*, vol. 102, no. 4, pp. 1031–1049, 2020.

[8] A. Horváth, G. Horváth, and M. Telek, "A joint moments based analysis of networks of map/map/1 queues," *Perf. Eval.*, vol. 67, no. 9, pp. 759–778, 2010.

[9] G. Casale, N. Mi, and E. Smirni, "Model-driven system capacity planning under workload burstiness," *IEEE Trans. on Computers*, vol. 59, no. 1, pp. 66–80, 2009.

[10] N. Mi, Q. Zhang, A. Riska, E. Smirni, and E. Riedel, "Performance impacts of autocorrelated flows in multi-tiered systems," *Perf. Eval.*, vol. 64, no. 9-12, pp. 1082–1101, 2007.

[11] G. Casale and M. Tribastone, "Fluid analysis of queueing in two-stage random environments," in *Proc. of QEST*. IEEE, 2011, pp. 21–30.

[12] W. Fischer and K. Meier-Hellstern, "The markov-modulated poisson process (MMPP) cookbook," *Perf. Eval.*, vol. 18, no. 2, pp. 149–171, 1993.

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

[14] J. Ruuskanen, T. Berner, K.-E. Arzen, and A. Cervin, "Improving the mean-field fluid model of processor sharing queueing networks for dynamic performance models in cloud computing," *ACM SIGMETRICS Perf. Eval. Review*, vol. 49, no. 3, pp. 69–70, 2022.

[15] G. Casale, "Integrated performance evaluation of extended queueing network models with line," in *2020 Winter Simulation Conference (WSC)*. IEEE, 2020, pp. 2377–2388.

[16] A. Heindl, "Traffic-based decomposition of general queueing networks with correlated input processes," Ph.D. dissertation, Technische Universität Berlin, 2001.

[17] W. R. Nunn and A. M. Desiderio, "Semi-markov processes: an introduction," *Cent. Nav. Anal*, pp. 1–30, 1977.

[18] I. Tsimashenka, W. J. Knottenbelt, and P. G. Harrison, "Controlling variability in split–merge systems and its impact on performance," *Annals of Operations Research*, vol. 239, no. 2, pp. 569–588, 2016.

[19] G. Casale, G. Serazzi, and L. Zhu, "Performance evaluation with java modelling tools: a hands-on introduction," *ACM SIGMETRICS Perf. Eval. Review*, vol. 45, no. 3, pp. 246–247, 2018.

[20] A. Kaltenbrunner, R. Meza, J. Grivolla, J. Codina, and R. Banchs, "Urban cycles and mobility patterns: Exploring and predicting trends in a bicycle-based public transport system," *Pervasive and Mobile Computing*, vol. 6, pp. 455–466, 2010.

[21] H. Sayarshad and J. Chow, "Survey and empirical evaluation of nonhomogeneous arrival process models with taxi data," *Journal of Advanced Transportation*, vol. 50, pp. 1275–1294, 2016.

[22] J. Gao, X. Li, C. Wang, and X. Huang, "Learning-based open driver guidance and rebalancing for reducing riders' wait time in ride-hailing platforms." IEEE, 2020.

[23] H. Wang and H. Yang, "Ridesourcing systems: A framework and review," *Transportation Research Part B*, vol. 129, pp. 122–155, 2019.

[24] G. Zardini, N. Lanzetti, M. Pavone, and E. Frazzoli, "Analysis and control of autonomous mobility-on-demand systems," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 5, pp. 633–658, 2022.

[25] R. Iglesias, F. Rossi, R. Zhang, and M. Pavone, "A BCMP network approach to modeling and controlling autonomous mobility-on-demand systems," *The International Journal of Robotics Research*, vol. 38, no. 2-3, pp. 357–374, 2019.

[26] A. Waserhole and V. Jost, "Pricing in vehicle sharing systems: optimization in queuing networks with product forms," *Euro journal on transportation and logistics*, vol. 5, pp. 293–320, 2014.

[27] R. Zhang and M. Pavone, "Control of robotic mobility-on-demand systems: a queueing-theoretical perspective," *The International Journal of Robotics Research*, vol. 35, no. 1-3, 2016.

[28] S.-Z. Yu, "Hidden semi-markov models," *Artificial intelligence*, vol. 174, no. 2, pp. 215–243, 2010.

[29] C. Fricker and N. Gast, "Incentives and redistribution in homogeneous bike-sharing systems with stations of finite capacity," *Euro journal on transportation and logistics*, vol. 5, no. 3, pp. 261–291, 2016.

[30] P. Romano, B. Ciciani, A. Santoro, and F. Quaglia, "Fast computation of hyper-exponential approximations of the response time distribution of mmpp/m/1 queues," in *Proc. of ANSS*. IEEE, 2008, pp. 113–120.

[31] G. Casale, N. Mi, and E. Smirni, "Model-driven system capacity planning under workload burstiness," *IEEE Trans. on Computers*, vol. 59, no. 1, pp. 66–80, 2010.

[32] G. Rubino and B. Tuffin, "Markovian models for dependability analysis," *Rare Event Simulation using Monte Carlo Methods*, pp. 125–143, 2009.

[33] L. Carnevali, L. Grassi, and E. Vicario, "State-density functions over dbm domains in the analysis of non-markovian models," *IEEE Transactions on Software Engineering*, vol. 35, no. 2, pp. 178–194, 2008.

[34] V. Anisimov, *Switching processes in queueing models*. John Wiley & Sons, 2013.

[35] A. Nazarov and G. Baymeeva, "The $M/GI/\infty$ system subject to semi-markovian random environment," in *14th International Scientific Conference, ITMM*. Springer, 2015, pp. 128–140.