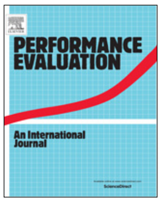
Performance Evaluation 91 (2015) 99–116

Contents lists available at ScienceDirect

Performance Evaluation

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



Beyond the mean in fork-join queues: Efficient approximation for response-time tails

Zhan Qiu [*], Juan F. Pérez, Peter G. Harrison

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

A R T I C L E   I N F O

Article history:
Available online 2 July 2015

Keywords:
Fork-join queues
Response-time distribution
Matrix-analytic methods
Order statistics

A B S T R A C T

Fork-join queues are natural models for various computer and communications systems that involve parallel multitasking and the splitting and resynchronizing of data, such as parallel computing, query processing in distributed databases, and parallel disk access. Job response time in a fork-join queue is a critical performance indicator but its exact analysis is challenging. We introduce a stochastic model for K-node homogeneous fork-join queues ($K \geq 2$) that focuses on the difference in length between any node-queue and the shortest one, truncating the state space such that the maximum difference is at most a constant C. Whilst most previous methods focus on the *mean* response time, our model is also able to evaluate the response time *distribution*, as well as accommodating phase-type processing times and Markovian arrival processes. In order to tackle scenarios with high loads, which require a large value of C to provide sufficient accuracy, we develop an efficient algorithm using matrix-analytic methods. Tests against simulation show that the proposed model yields accurate results for 2-node fork-join queues. As the model becomes numerically intractable for large values of K, we further propose an approximate approach, based on properties of order statistics and extreme values. The approximation gives a high degree of accuracy on response time tails, and has the advantage of being efficient and scalable, requiring only the analytical results for a single-node and 2-node fork-join queues, which we obtain with the aforementioned matrix-analytic model. Comparison with simulation results shows that our approximation yields good fits for the tails, even in very large cases with general processing and inter-arrival times.

1. Introduction

Modern computer and manufacturing systems rely heavily on parallelism to satisfy their performance requirements, exploiting large resource pools to perform tasks that would otherwise be uneconomical or even infeasible. For instance, the Google search engine relies on a high degree of parallelism to return a huge volume of answers in less than a second [1]. Parallelism can be found in many different computer systems, such as query processing in distributed databases, parallel computing environments, distributed stream-processing systems, and parallel disk access to improve I/O throughput in storage systems, amongst others. In manufacturing systems, parallelism can also be found in processes such as product-assembly and logistic operations that involve multiple suppliers [2]. A key ingredient in many of these systems is that, after a step where many subtasks are executed in parallel by a set of resources, the final product can only be delivered once all the subtasks have been completed, thus introducing a synchronization point. Fork-join (FJ) queues are a popular abstraction for

[*] Corresponding author.
E-mail addresses: zhan.qiu11@imperial.ac.uk (Z. Qiu), j.perez-bernal@imperial.ac.uk (J.F. Pérez), pgh@imperial.ac.uk (P.G. Harrison).

http://dx.doi.org/10.1016/j.peva.2015.06.007
0166-5316/© 2015 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

these kinds of operations, as they are able to capture the parallel execution and synchronization aspects precisely. FJ queues have therefore received significant attention, although their analysis has proved challenging.

In a FJ queue, each arriving job is split into K subtasks, each of which is assigned to one of K parallel processors, or nodes. Once assigned to a node, a subtask must wait in a queue until its service starts, according to some scheduling policy. After each subtask is completed, it must wait for all other subtasks in the same job to also complete, at which point the job's service terminates. A key performance metric in a FJ queue is the job response time, although its computation – certainly beyond its mean value – is notoriously difficult due to the synchronization step. In fact, exact solutions for the *mean* response time are only available for FJ queues with $K = 2$ nodes. A large body of work has therefore been devoted to obtaining approximations for the mean, and, in a few cases, the distribution of the response time. Further, most approximations focus on the case with negative exponential execution times and Poisson arrivals. Notice that the FJ queue deals with the *maximum* of node-queue response times, which are not independent due to the synchronized arrivals. This is inherently different from "first to finish" schedules which require a *minimum* value.

In the present work, we introduce a stochastic model for K-node FJ queues to determine the response-time *distribution*. The model focuses on the difference in length between any node-queue and the shortest one, the state space being truncated such that the maximum difference is at most a constant number C. The value needed for this bound is much smaller than the traditional bound imposed on the raw queue length to obtain the same precision, especially at high loads. We propose a numerical approach based on matrix-analytic methods, where the key step lies in the solution of a matrix integral equation. The sizes of the matrices involved in this equation grow rapidly with K and C but our method provides significant improvements over existing methods for moderate values of these parameters and is also applicable in a variety of situations where similar equations arise.

Whilst the matrix-analytic model provides accurate results for small-scale cases, it becomes numerically intractable for large values of K and C. We therefore propose an *Efficient Approximation for Response-Time Tails (EAT)*, which relies on properties of order statistics and extreme values. It stems from an argument that the response-time percentiles of a K-node FJ queue grow at rate $\mathcal{O}(\ln(K))$, based on the observation that both upper and lower bounds also grow at this rate. EAT has the advantage of being efficient and scalable, requiring only the analytical results for a single-node and 2-node FJ queues, which we obtain by the preceding matrix-analytic approach. Experimental results show that EAT yields good fits for the tails, even in very large cases with general processing and inter-arrival times. Its scalability and accuracy make the approximation a good candidate to support scheduling, resource provisioning and optimization decisions.

The main contributions of this paper can be summarized as follows:

- A stochastic model, presented in Section 4, to determine the response-time distribution in homogeneous K-node FJ queues. The model handles phase-type processing times and Markovian arrival processes.
- An efficient method to solve a matrix integral equation, required to obtain the response-time distribution, which is introduced in Section 5. The method is shown to outperform existing approaches and can be used for similar equations that arise in other contexts.
- An approximation based on extreme value theory for FJ queues with a large number of parallel servers, which exploits the stochastic model introduced in Section 4, to provide accurate estimates of the response-time tail. The approximation approach is described in Section 6 and tested in Section 7.

Before introducing the stochastic model, we overview related work and provide background definitions.

2. Related work

There is a considerable body of published research on FJ queues, as recently surveyed in [1]. The complexity of FJ queues is evident in the paucity of exact results available. With Poisson arrivals and exponential service times, the queue-length distribution for the 2-node queue is derived in [3], while [4] provides an expression for the mean response-time, which has also been obtained in [5] for cases with more general inter-arrival times (hyper-exponential) and service times (Erlang). More general cases have been considered only through approximate analysis. In particular, for FJ queues with Poisson arrivals and exponential service, [4] provides an approximation for the mean response-time in FJ queues with K nodes, $2 \leq K \leq 32$. This method is based on the observation that the upper and lower bounds on the mean response time grow at the same rate as a function of the number of nodes K. In this setting, lower and upper bounds on the mean response-time [6] have also been obtained. [7] considered a system with a serial-parallel phase-type arrival process and exponential or Erlang service times, in which a job is split into $1 \leq i \leq K$ subtasks. It derived both upper and lower bounds on the expected response time. For general inter-arrival and service-time distributions, approximations for the mean response-time in heavy-traffic can be found in [8].

Although most previous methods focus on the *mean* response-time, a few have considered its *distribution*. An approximation for the case where each node has multiple exponential servers and a Poisson arrival process is considered in [9]; it relies on estimating two coefficients through extensive simulations and offers good accuracy for the mean response-time when $2 \leq K \leq 50$ as well as for the response-time distribution when $K = 2$. A similar approach is proposed in [10] for the case that inter-arrival times form a general renewal process, with good accuracy for the response-time distribution when $2 \leq K \leq 50$, assuming Erlang inter-arrival times. In the sequel we introduce a model to compute the response-time distribution in a K-node FJ queue where service times are phase-type and the arrival process is Markovian (MAP). The model

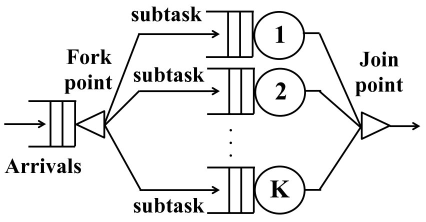


Fig. 1. Fork-join queue.

relies on observing the difference in size between any node-queue and the shortest node-queue, and truncating the state space to limit the maximum difference to a constant C. We demonstrate that the model provides accurate results for the response-time distribution. Since the numerics of the model become intractable at large values of K and C, we introduce a method that exploits the results of the model for $K = 2$ to approximate the response-time tails at larger K. Experimental results show that the approximation is accurate even at very large values of K.

3. Preliminaries

In this section we define phase-type distributions, Markovian arrival processes and a reference model for the FJ queue, which will be considered in subsequent sections.

3.1. Phase-type distributions

Phase-type (PH) distributions have the ability to capture very general behaviors; they are dense in the set of all positive-valued distributions whilst maintaining some of the analytical tractability of the exponential distribution [11]. A PH random variable X represents the absorption time in a Markov Chain (MC) with $n+1$ states, where the states $\{1, \ldots, n\}$ are transient and state 0 is absorbing [11]. Let $\boldsymbol{\tau}$ be the $1 \times n$ vector of the MC initial probability distribution for the transient states, and let S be the $n \times n$ sub-generator matrix holding the transition rates among the transient states. We denote this random variable or its distribution as PH($\boldsymbol{\tau}, S$). The vector $\boldsymbol{S}^* = -S\mathbf{1}$ holds the absorption rates from the transient states, where $\mathbf{1}$ denotes a column vector of ones. Its cumulative distribution function (CDF) is given by $F(x) = 1 - \boldsymbol{\tau} \exp(Sx)\mathbf{1}$, for $x \geq 0$, and its expected value is $E[X] = -\boldsymbol{\tau} S^{-1}\mathbf{1}$.

3.2. Markovian arrival processes

Compared to traditional Poisson arrivals, a Markovian arrival process (MAP) can represent more general inter-arrival times, capturing features such as high variability and auto-correlation. The continuous-time MAP [11] is a marked MC with generator matrix $D = D_0 + D_1$. The transition rates *not* associated with arrivals are held in D_0, while the rates that trigger new arrivals are kept in D_1. The diagonal entries of D_0 hold the total exit rate in each state, such that $(D_0 + D_1)\mathbf{1} = \mathbf{0}$. We denote this process as MAP(m_a, D_0, D_1), where m_a is the number of states, or arrival phases, in the MC. The mean arrival rate is $\lambda = \boldsymbol{d}D_1\mathbf{1}$, where $\boldsymbol{d}$ is the stationary distribution of the underlying MC, i.e. $\boldsymbol{d}D = 0$ and $\boldsymbol{d}\mathbf{1} = 1$.

3.3. Reference model

We consider a fork-join queue made of K parallel processing resources, each with an associated buffer, as shown in Fig. 1. Jobs arrive following a MAP(m_a, D_0, D_1), and fork into K subtasks, which are sent to the K servers for processing. Subtasks in front of a server form a single queue in the order of arrival and receive service when the server becomes available with first-come first-served (FCFS) scheduling. The K servers are homogeneous, thus the processing time of each subtask follows a PH($\boldsymbol{\alpha}_{\text{task}}, S_{\text{task}}$) distribution. For the case of exponential service times, we assume a mean processing time $1/\mu$, such that $\boldsymbol{\alpha}_{\text{task}} = 1$, and $S_{\text{task}} = -\mu$. When a subtask completes service, it waits at the join point until all its partners complete execution. When all the subtasks of a job complete service, the job service is completed and the job leaves the system. The response-time of a job, i.e., the time interval between its arrival and its departure, is thus the maximum of the response-times of its subtasks. Finally, the system utilization U, i.e., the expected fraction of the time that a server is busy, is simply $U = \lambda/\mu$, where λ is the mean arrival rate, as defined in the previous section, and μ is the subtask service rate.

4. The response-time distribution

In this section, we introduce a stochastic model to estimate the response-time distribution of a parallel job in a K-node FJ queue, as in Fig. 1. The distribution is obtained by considering the service-time and waiting-time processes separately, and connecting them via a PH representation. We therefore provide PH representations for the waiting-time, service-time,

Table 1
Transition rates for S and $A^{(\text{jump})}$.

From	To	Rate	Matrix
$(n_0, \dots, n_{i-1}, n_i, \dots, n_C)$	$(n_0, \dots, n_{i-1}+1, n_i-1, \dots, n_C)$	$n_i\mu, i \geq 1$	S
$(n_0, n_1, \dots, n_{C-1}, n_C)$	$(1, n_0-1, n_1, \dots, n_{C-2}, n_{C-1}+n_C)$	$n_0\mu$	$A^{(\text{jump})}$

and response-time distributions. In the following we pay special attention to the shortest of the parallel queues, and refer to a period during which the server of the shortest queue is busy as an *all-busy* period, and a period during which at least one server is idle as a *not-all-busy* period.

4.1. The waiting-time distribution

To determine the waiting-time distribution, we observe the FJ queue only during the *all-busy* periods, and define a bivariate Markov process $\{X(t), J(t)|t \geq 0\}$. The *age* $X(t)$ keeps track of the total time-in-system of the youngest job in service [12], thus taking values in $[0, \infty)$, and increasing linearly with rate 1 if no service completions occur. In case of a service completion in the shortest queue, which triggers the start of a new job service, $X(t)$ has a downward jump, with its new value being equal to the waiting-time of the job starting service. The *phase* $J(t) = (A(t), D(t))$ holds the joint state of the arrival process $A(t)$ and the service process $D(t)$. The arrival process is a MAP with m_a phases, so that, assuming $D(t)$ takes values in a set of size m_s, the phase process takes values in a set of size $m = m_a m_s$. The process $\{X(t), J(t)|t \geq 0\}$ is in fact a Markov process since, at any time t, its evolution depends only on its current state, as the state of the phase $J(t)$ determines not only its own evolution, but also whether $X(t)$ suffers a downward jump or continues increasing at rate 1.

To model the service process of a K-node FJ queue, we first note that the state of the service process is fully described by the vector $(q_1, q_2, \dots, q_K)$, where q_i holds the number of subtasks in queue i, *either waiting or in service*, for $1 \leq i \leq K$. Since all the servers are identical, we order the queue lengths such that $q_i \leq q_j$ for $i < j$ and $1 \leq i, j \leq K$, making q_1 and q_K the length of the shortest and longest queues, respectively. During the all-busy period the shortest queue length is positive ($q_1 > 0$), while during the not-all-busy period, q_1 is 0 since the shortest queue must be empty. Instead of keeping this description, we focus on the *queue-length differences* with respect to the shortest queue. Since the queues are homogeneous, instead of recording this difference for each queue, it suffices to count the number of queues with the same queue length. We thus model the service process $D(t)$ by keeping track of the number of queues with j more subtasks than the shortest queue, $n_j(t)$, for $j \geq 0$. In principle, the difference in length between any queue and the shortest one is unbounded. We thus truncate the state space such that the difference between the longest and the shortest queue is limited to be at most $C < \infty$, such that the $D(t)$ is given by $(n_0(t), n_1(t), \dots, n_C(t))$. Thus $D(t)$ takes values in the set $S_D = \{(n_0, \dots, n_C) | \sum_{i=0}^{C} n_i = K, n_i \in \{0, \dots, K\}\}$. The cardinality of S_D is

$$m_s = \binom{K+C-1}{C}. \tag{1}$$

The accuracy of this approximation depends on the system parameters. For instance, when the utilization is high, we may expect a higher probability of large differences than under low utilizations, thus requiring a larger value of C to achieve the same accuracy. Section 4.5 evaluates the accuracy of this approximation and explores the selection of the limit C.

To determine the PH representation $(\boldsymbol{\alpha}_{\text{wait}}, S_{\text{wait}})$ of the waiting-time distribution, we rely on the stationary distribution $\boldsymbol{\pi}(x)$ of the $(X(t), J(t))$ process, which has a matrix-exponential representation $\boldsymbol{\pi}(x) = \boldsymbol{\pi}(0)\exp(Tx)$, for $x > 0$, as shown in [13]. The $m \times m$ matrix T satisfies the non-linear integral equation

$$T = S^{(\text{MAP})} + \int_0^\infty \exp(Tu) A^{(\text{MAP})}(u) du, \tag{2}$$

where $S^{(\text{MAP})} = S \otimes I_{m_a}$, $A^{(\text{MAP})}(u) = A^{(\text{jump})} \otimes \exp(D_0 u) D_1$, I_n is the identity matrix of size n, and $\otimes$ denotes the Kronecker product. S and $A^{(\text{jump})}$ are $m_s \times m_s$ matrices that hold the transition rates of the service process associated to transitions without and with a new job starting service, respectively [12,13]. The generator of the marginal service phase process during an all-busy period is thus $S + A^{(\text{jump})}$. Notice that the service completion of the subtask currently in service in any of the n_0 shortest queues triggers a new job to start service, thus corresponding to rates in matrix $A^{(\text{jump})}$; transitions in other queues correspond to rates in matrix S. Table 1 summarizes the service transition rates that correspond to matrices S and $A^{(\text{jump})}$. Notice that transitions that lead any of the differences to exceed the limit C are re-assigned to C. A small example of these matrices is provided in Appendix C.

4.1.1. Computing $\pi(0)$

To determine the steady state distribution $\boldsymbol{\pi}(0)$ of the phase at the beginning of an all-busy period, it is essential to connect the not-all-busy and the all-busy periods [12]. During the not-all-busy period we keep the same description of the phase $J(t) = (A(t), D(t))$, with the service phase $D(t)$ taking values in the set S_D. In this case the shortest queue is empty, and therefore n_0 holds the number of idle servers. Let the $m_s \times m_s$ matrix $S_{\text{not-all-busy}}$ hold the transition rates across service phases, *between arrivals*, during a not-all-busy period, as summarized in Table 2. An example is provided in Appendix C.

Table 2
Transition rates for $S_{\text{not-all-busy}}$.

From	To	Rate
$(n_0, \dots, n_{i-1}, n_i, \dots, n_C)$	$(n_0, \dots, n_{i-1}+1, n_i-1, \dots, n_C)$	$n_i\mu,\ i \geq 1$

Since $\boldsymbol{\pi}(0)$ is the distribution of the phase $J(t)$ at the beginning of an all-busy period, it solves

$$\boldsymbol{\pi}(0) = \boldsymbol{\pi}(0)\int_0^\infty \exp(Tu)(A^{(\text{jump})} \otimes \exp(D_0u))du(S_{\text{not-all-busy}} \oplus D_0)^{-1}(I_{m_s} \otimes D_1). \tag{3}$$

Here we consider that, during an all-busy period, the age reaches a value u with density $\boldsymbol{\pi}(0)\exp(Tu)$. At this point a downward jump of size larger than u occurs with density $A^{(\text{jump})} \otimes \exp(D_0u)$. This results in a transition to a not-all-busy period, where the phase process evolves according to the sub-generator $S_{\text{not-all-busy}} \oplus D_0$, until an arrival occurs and triggers the process into a new all-busy period. Notice that when a job arrives during a not-all-busy period, the service phase $D(t)$ remains unchanged since every queue receives a new subtask, keeping their differences unchanged. The distribution of the phase just after the arrival that initiates the all-busy period, given the phase just before this arrival, is thus $(I_{m_s} \otimes D_1)$. To compute the integral in Eq. (3), we define $P = \int_0^\infty \exp(Tu)(I_{m_s} \otimes \exp(D_0u))du$, and integrate P by parts to obtain $TP + P(I_{m_s} \otimes D_0) = -I_m$, which, since T is known, is a Sylvester matrix equation that can be solved in $O(m^3)$ time to find P. Once P has been obtained, $\pi(0)$ can be found by solving the linear system (3) in $O(m^3)$ time. Recall that $m = m_a\binom{K+C-1}{C}$.

Notice that the stationary distribution $\boldsymbol{\pi}(x)$ exists if $\{X(t), J(t)|t \geq 0\}$ is positive recurrent, which occurs under the condition [13]

$$\varrho = \boldsymbol{\rho}\int_0^\infty uA^{(\text{MAP})}(u)du\mathbf{1} > 1, \tag{4}$$

where $\boldsymbol{\rho}$ is the invariant probability vector of the matrix $S^{(\text{MAP})} + (A^{(\text{jump})} \otimes D_0^{-1}D_1)$. Letting the steady state distribution of the phase $J(t)$ during the all-busy period be $\boldsymbol{\alpha}_{\text{busy}} = -\boldsymbol{\pi}(0)T^{-1}$, and defining $\boldsymbol{\varphi} = (T - S^{(\text{MAP})})\mathbf{1}$, the PH representation of the waiting-time is given by [12]

$$\boldsymbol{\alpha}_{\text{wait}} = \gamma\boldsymbol{\alpha}_{\text{busy}} \circ \boldsymbol{\varphi}/((\boldsymbol{\alpha}_{\text{busy}} \circ \boldsymbol{\varphi})\mathbf{1}), \qquad S_{\text{wait}} = \Lambda^{-1}T'\Lambda, \tag{5}$$

where $\circ$ stands for the Hadamard product [14], $'$ denotes the matrix transpose, Λ is a diagonal matrix such that $\Lambda\mathbf{1} = \boldsymbol{\alpha}'_{\text{busy}}$, and γ is the probability that a job has to wait. For the sake of completeness, we recall the computation of γ, by first defining η_0 to be the number of service completions in an all-busy period, and η_1 the number of arrivals in a not-all-busy period, respectively. Their expected values can be obtained as [12, Section 7.2] $E[\eta_0] = -(\boldsymbol{\pi}(0)T^{-1}(A^{(\text{jump})} \otimes I_{m_a})\mathbf{1})/(\boldsymbol{\pi}(0)\mathbf{1})$, and $E[\eta_1] = 1$, since an arrival during the not-all-busy period adds one subtask to every queue, initiating the all-busy period. Thus the probability that a job has to wait is $\gamma = (E[\eta_0] - 1)/(E[\eta_0] - 1 + E[\eta_1]) = 1 - 1/E[\eta_0]$ since, in a cycle made of one all-busy and one not-all-busy period, $E[\eta_0] - 1$ is the expected number of jobs that have to wait, and $E[\eta_0] - 1 + E[\eta_1]$ is the expected number of job arrivals.

4.2. The service-time distribution

Although the subtask service-time follows an exponential distribution, the job service-time does not, as it is composed of multiple subtasks, and completes service only when all subtasks terminate. We now show that the job processing times follow a PH distribution, with parameters PH($\boldsymbol{\alpha}_{\text{service}}, S_{\text{service}}$). Let $Y(t)$ be the service state of a tagged job in service at time t, which we define as $Y(t) = (R(t), N(t))$. Here $R(t)$ records the number of subtasks in the tagged job that *have not finished service* at time t, thus $R(t) \in \{1, \dots, K\}$. Similarly to the service phase defined for the waiting-time distribution, the variable $N(t) = (n_0, \dots, n_C)$ holds the state of the queues, with n_i being the number of queues with i more subtasks in queue than the shortest queue. However, here we focus on the queue-lengths in front of the tagged job only, ignoring any job behind the tagged one. This allows us to keep track only of the state of the r queues where subtasks in the tagged job have not completed, as the state of the other $K - r$ queues is set to 0 after the tagged subtasks complete. $(R(t), N(t))$ thus takes values in the set $S_{RN} = \{(r, (n_0, \dots, n_C))|r \in \{1, \dots, K\}, n_i \in \{0, \dots, r\}, \sum_{i=0}^{C} n_i = r\}$.

Therefore, the cardinality of S_{RN} is

$$|S_{RN}| = \binom{K+C-1}{C} + \sum_{r=1}^{K-1}\binom{r+C-1}{C-1}. \tag{6}$$

For example, for the case $K = 3$ and $C = 2$, if only one subtask has not finished service, thus $R(t) = 1$, and $N(t)$ takes values in $\{(2, 0, 1), (2, 1, 0)\}$, as the 2 queues where the tagged subtasks finished service have queue-length 0, and the other queue can have a queue-length of either 1 or 2.

To obtain a PH representation of the service-time distribution we describe the transitions among the states in S_{RN} with a sub-generator S_{service}. To this end, we partition S_{RN} into two sets: in the *full* phases none of the tagged subtasks have completed service, thus $R(t) = K$; and in the *not-full* phases, at least one tagged subtask already finished service, thus $1 \leq$

Table 3
Transition rates for $S_{\text{(full)-(not-full)}}$ and $S_{\text{not-full}}$.

From	To	Rate	Matrix
$(K, (n_0, \ldots, n_i, \ldots, n_C))$	$(K-1, (1, n_0-1, \ldots, n_{C-2}, n_{C-1}+n_C))$	$n_0\mu$	$S_{\text{(full)-(not-full)}}$
$(r, (n_0, \ldots, n_{i-1}, n_i, \ldots, n_C))$	$(r, (n_0, \ldots, n_{i-1}+1, n_i-1, \ldots, n_C))$	$n_i\mu\ (i \geq 2)$	$S_{\text{not-full}}$
$(r, (n_0, n_1, \ldots, n_C))$	$(r-1, (n_0+1, n_1-1, \ldots, n_C))$	$n_1\mu$	$S_{\text{not-full}}$

$R(t) < K$. Notice that the first set has m_s phases, as defined in Eq. (1), and letting the number of *not-full* phases be m_{nf}, we see from Eq. (6) that $|S_{RN}| = m_s + m_{nf}$. Partitioning the sub-generator S_{service} according to these subsets, we have

$$S_{\text{service}} = \begin{bmatrix} S_{\text{full}} & S_{\text{(full)-(not-full)}} \\ 0 & S_{\text{not-full}} \end{bmatrix} \otimes I_{m_a},$$

thus the S_{service} matrix is of size $m_{\text{ser}} = (m_s + m_{nf})m_a$. Notice that here we have included the arrival phase, such that during service we keep track of the arrival phase at the time when the service started. Although this is not necessary for the service-time distribution on its own, it will become necessary when building the response-time distribution in the next section. Since S_{full} holds the service transition rates without a service completion in the shortest queue, we have $S_{\text{full}} = S$. The transition rates in $S_{\text{(full)-(not-full)}}$ and $S_{\text{not-full}}$ are summarized in Table 3. Notice that the absorbing vector $\boldsymbol{S}^*_{\text{service}} = -S_{\text{service}}\mathbf{1}$ has a single nonzero element equal to μ, which corresponds to the phase where the last subtask completes service. Having obtained S_{service}, it remains to determine the initial probability vector $\boldsymbol{\alpha}_{\text{service}}$, which is the stationary probability with which a job starts service in each phase. The proof of the following result can be found in Appendix A.

Proposition 1. *The job service-time follows a PH distribution with parameters* $(\boldsymbol{\alpha}_{\text{service}}, S_{\text{service}})$, *where*

$$\boldsymbol{\alpha}_{\text{service}} = [(1-\gamma)\boldsymbol{\pi}(0) + c\gamma\boldsymbol{\alpha}_{\text{busy}}(T - S^{(\text{MAP})}) \quad \mathbf{0}_{1\times m_{nf}m_a}], \tag{7}$$

where $c = \boldsymbol{\alpha}_{\text{busy}}(T - S^{(\text{MAP})})\mathbf{1}$.

4.3. The response-time distribution

We now derive the PH representation of the response-time distribution, the proof of which can be found in Appendix B.

Theorem 1. *The job response-time follows a PH distribution with parameters* $(\boldsymbol{\alpha}_{\text{res}}, S_{\text{res}})$, *where*

$$\boldsymbol{\alpha}_{\text{res}} = \left[(1-\gamma)\boldsymbol{\pi}(0) \quad \tilde{\boldsymbol{\alpha}}_{\text{busy}} \quad \mathbf{0}_{1\times m}\right], \qquad S_{\text{res}} = \begin{bmatrix} S_{\text{service}} & 0_{m_{\text{ser}}\times m_{\text{ser}}} & 0_{m_{\text{ser}}\times m} \\ 0_{m_{\text{ser}}\times m_{\text{ser}}} & \tilde{S}_{\text{service}} & (-\tilde{S}_{\text{service}}\mathbf{1})P_{s,w} \\ 0_{m\times m_{\text{ser}}} & 0_{m\times m_{\text{ser}}} & S_{\text{wait}} \end{bmatrix}, \tag{8}$$

where $\tilde{S}_{\text{service}} = \Delta^{-1}S'_{\text{service}}\Delta$, Δ *is a diagonal matrix such that* $\Delta\mathbf{1} = \boldsymbol{\eta}'$, *and* $\boldsymbol{\eta} = -\boldsymbol{\alpha}_{\text{service}}S^{-1}_{\text{service}}$ *is the stationary distribution of the phase during service. Also,* $\tilde{\boldsymbol{\alpha}}_{\text{busy}} = (-S_{\text{service}}\mathbf{1})'\Delta$. *Finally,* $P_{s,w}$ *is an* $m_{\text{ser}} \times m$ *matrix given by*

$$P_{s,w} = \begin{bmatrix} \tilde{P}_{s,w} \\ 0_{m_a m_{nf}\times m} \end{bmatrix},$$

where $\tilde{P}_{s,w} = \Gamma^{-1}(T - S^{(\text{MAP})})'\Lambda$, *and* Γ *and* Λ *are diagonal matrices such that* $\Gamma\mathbf{1} = (T - S^{(\text{MAP})})'\Lambda\mathbf{1}$ *and* $\Lambda\mathbf{1} = \boldsymbol{\alpha}'_{\text{busy}}$.

Notice that this result is exact as long as the PH representations of the service and waiting-time distributions are exact. Since we obtain approximate representations for these distributions, the response-time distribution is also approximate.

4.4. Extension to PH services

The model presented so far has focused on exponential service-times, but it can be extended to the more general case of PH distributions, as we now show. The main difference lies in the service process $D(t)$ as we now need to keep track of the phase of the subtasks in service. Thus, assuming the subtask service-time distribution is PH$(\boldsymbol{\alpha}_{\text{task}}, S_{\text{task}})$ with m_{task} phases, we define $D^{\text{PH}}(t) = (n_{i,j}(t), 0 \leq i \leq C, 1 \leq j \leq m_{\text{task}})$, where $n_{i,j}(t)$ is the number of queues with queue-length difference i with respect to the shortest queue, and such that the subtask in service is in phase j, at time t. Thus $D^{\text{PH}}(t)$ takes values in the set $S_D^{\text{PH}} = \{\boldsymbol{n} = (\boldsymbol{n}_1, \ldots, \boldsymbol{n}_C) | \boldsymbol{n}_i = (n_{i,1}, \ldots, n_{i,m_{\text{task}}}), n_{i,j} \in \{0, \ldots, K\}, \sum_{i=0}^{C}\sum_{j=1}^{m_{\text{task}}} n_{i,j} = K\}$.

Similarly, the matrices S and $A^{(\text{jump})}$ need to be re-defined, considering that $A^{(\text{jump})}$ only keeps transition rates associated to the start of a new job service, i.e., service completions in any of the shortest queues. Table 4 summarizes the service transition rates that correspond to matrices S and $A^{(\text{jump})}$. The first row in this table considers the case where the subtask in service goes through a service-phase transition (from phase j to phase k, with $1 \leq j, k \leq m_{\text{task}}$), without completing service, in any of the queues. The second row covers the case of service completions at any queue but the shortest one, which are also kept in S. In this case, the subtask in service phase j completes, letting a new subtask start service in phase k, with $1 \leq j, k \leq m_{\text{task}}$. Finally, the third row considers the case where the service completion occurs in any of the shortest queues,

Table 4
Transition rates for S and $A^{(\text{jump})}$ with PH services.

From	To	Rate	Matrix
$(\boldsymbol{n}_0, \ldots, \boldsymbol{n}_i, \ldots, \boldsymbol{n}_C)$	$(\boldsymbol{n}_0, \ldots, \boldsymbol{n}_i - \boldsymbol{e}_j + \boldsymbol{e}_k, \ldots, \boldsymbol{n}_C)$	$n_{i,j} S_{\text{task}}(j,k), i \geq 0$	S
$(\boldsymbol{n}_0, \ldots, \boldsymbol{n}_{i-1}, \boldsymbol{n}_i, \ldots, \boldsymbol{n}_C)$	$(\boldsymbol{n}_0, \ldots, \boldsymbol{n}_{i-1} + \boldsymbol{e}_k, \boldsymbol{n}_i - \boldsymbol{e}_j, \ldots, \boldsymbol{n}_C)$	$n_{i,j} S^*_{\text{task}}(j) \boldsymbol{\alpha}_{\text{task}}(k), i \geq 1$	S
$(\boldsymbol{n}_0, \boldsymbol{n}_1, \ldots, \boldsymbol{n}_{C-1}, \boldsymbol{n}_C)$	$(\boldsymbol{e}_k, \boldsymbol{n}_0 - \boldsymbol{e}_j, \boldsymbol{n}_1, \ldots, \boldsymbol{n}_{C-2}, \boldsymbol{n}_{C-1} + \boldsymbol{n}_C)$	$n_{0,j} S^*_{\text{task}}(j) \boldsymbol{\alpha}_{\text{task}}(k)$	$A^{(\text{jump})}$

Table 5
Transition rates for $S_{\text{not-all-busy}}$ with PH services.

From	To	Rate
$(n_0, \ldots, \boldsymbol{n}_i, \ldots, \boldsymbol{n}_C)$	$(n_0, \ldots, \boldsymbol{n}_i - \boldsymbol{e}_j + \boldsymbol{e}_k, \ldots, \boldsymbol{n}_C)$	$n_{i,j} S_{\text{task}}(j,k), i \geq 1$
$(n_0, \ldots, \boldsymbol{n}_{i-1}, \boldsymbol{n}_i, \ldots, \boldsymbol{n}_C)$	$(n_0, \ldots, \boldsymbol{n}_{i-1} + \boldsymbol{e}_k, \boldsymbol{n}_i - \boldsymbol{e}_j, \ldots, \boldsymbol{n}_C)$	$n_{i,j} S^*_{\text{task}}(j) \boldsymbol{\alpha}_{\text{task}}(k), i \geq 2$
$(n_0, \boldsymbol{n}_1, \ldots, \boldsymbol{n}_C)$	$(n_0 + 1, \boldsymbol{n}_1 - \boldsymbol{e}_j, \ldots, \boldsymbol{n}_C)$	$n_{1,j} S^*_{\text{task}}(j)$

which is recorded by matrix $A^{(\text{jump})}$. Here again the subtask completes service in phase j, and the new subtask starts in phase k, such that now there is a single shortest queue, which is in phase k.

During the not-all-busy period, a similar modification is necessary, although in this case the service process $D^{\text{PH}}(t)$ needs to reflect that empty queues do not have an associated subtask in service that requires keeping track of its phase. To differentiate it from the $D^{\text{PH}}(t)$ process during the all-busy period, we label this process $\bar{D}^{\text{PH}}(t)$. The $\bar{D}^{\text{PH}}(t)$ process takes values in the set $\bar{S}_D^{\text{PH}} = \{(n_0, \boldsymbol{n}_1, \ldots \boldsymbol{n}_C) | \boldsymbol{n}_i = (n_{i,1}, \ldots, n_{i,m_{\text{task}}}), 1 \leq i \leq C, n_0 \in \{0, \ldots, K\}, n_{i,j} \in \{0, \ldots, K\}, n_0 + \sum_{i=1}^{C} \sum_{j=1}^{m_{\text{task}}} n_{i,j} = K\}$. In this case, the first entry of the vector $(n_0, \boldsymbol{n}_1, \ldots \boldsymbol{n}_C)$ holds the number of idle servers, while the remaining entries are the same as for $D^{\text{PH}}(t)$. Table 5 summarizes the entries of the matrix $S_{\text{not-all-busy}}$, which are the transition rates across service phases during a not-all-busy period. Here the first row corresponds to phase transitions without a service completion, while the second and third rows correspond to service completions. The third row shows how a subtask in a queue with queue-length one completes service in phase j, thus incrementing the number of idle queues n_0. A final adaptation is needed to compute $\boldsymbol{\pi}(0)$, the distribution of the phase $J(t)$ at the beginning of an all-busy period, which now solves

$$\boldsymbol{\pi}(0) = \boldsymbol{\pi}(0) \int_0^{\infty} \exp(Tu) (A^{(\text{jump})}_{\text{not-all-busy}} \otimes \exp(D_0 u)) du (S_{\text{not-all-busy}} \oplus D_0)^{-1} (K^{(\text{start})} \otimes D_1).$$

The difference with respect to Eq. (3) lies in the matrices $A^{(\text{jump})}_{\text{not-all-busy}}$ and $K^{(\text{start})}$, which connect the state space when going from an all-busy period to a not-all-busy period, and vice versa, respectively. The matrix $A^{(\text{jump})}_{\text{not-all-busy}}$ is similar to $A^{(\text{jump})}$ as defined in Table 4, but from a state $(\boldsymbol{n}_0, \boldsymbol{n}_1, \ldots, \boldsymbol{n}_C) \in S_D^{\text{PH}}$, the transition is to a state $(1, \boldsymbol{n}_1, \ldots, \boldsymbol{n}_C) \in \bar{S}_D^{\text{PH}}$ with rate $n_{0,j} S^*_{\text{task}}(j)$. Thus this transition corresponds to a service completion in the shortest queue, which starts a not-all-busy period with one empty queue. On the other hand, the matrix $K^{(\text{start})}$ maps a state $(n_0, \boldsymbol{n}_1, \ldots, \boldsymbol{n}_C) \in \bar{S}_D^{\text{PH}}$ into a state $(\boldsymbol{n}_0, \boldsymbol{n}_1, \ldots, \boldsymbol{n}_C) \in S_D^{\text{PH}}$ with probability $p_{n_0, \boldsymbol{n}_0}$. Again, the only change is related to the shortest queues, which in this case go from the idle state to an initial phase of service. As each of the n_0 idle servers selects initial service phase according to $\boldsymbol{\alpha}_{\text{task}}$, the probability that the shortest queues start the all-busy period in state $\boldsymbol{n}_0 = (n_{0,1}, \ldots, n_{0,m_{\text{task}}})$ follows a multinomial distribution, thus

$$p_{n_0, \boldsymbol{n}_0} = \frac{n_0!}{n_{0,1}! \cdots n_{0,m_{\text{task}}}!} \boldsymbol{\alpha}_{\text{task}}(1)^{n_{0,1}} \cdots \boldsymbol{\alpha}_{\text{task}}(m_{\text{task}})^{n_{0,m_{\text{task}}}},$$

where $\boldsymbol{\alpha}_{\text{task}}(i)$ denotes the ith entry of vector $\boldsymbol{\alpha}_{\text{task}}$. Once $\boldsymbol{\pi}_0$ is found, the waiting-time distribution is obtained as in the case with exponential services. A similar extension is then needed to obtain the service-time and response-time distributions, which we omit for brevity.

4.5. Experimental validation

In this section we show that the proposed model is able to provide accurate results as long as the limit C is large enough. Although the model is defined for any number of parallel queues K, it becomes time consuming and computationally expensive with increasing K, especially in cases with high loads, as these cases require a large value of the limit C to provide sufficient accuracy. To handle cases with large K, we propose an approximation approach in Section 6, which provides a high degree of accuracy for the response-time tails in FJ queues with $K > 2$, exploiting the results of the model proposed for the $K = 2$ case. We thus focus on the 2-node FJ queue, considering different distributions for the arrival and service processes, and different utilization levels. For the service times, we consider exponential (Exp), Erlang-2 (ER$_2$), and 2-phase hyper-exponential (HE$_2$) distributions. For the arrival process, we consider Poisson (Exp), and general order-2 MAPs (MAP). We thus cover a broad range of behaviors in terms of variability, measured by the squared coefficient of variation (SCV), defined as $C_X^2 = \text{Var}[X]/E^2[X]$, for a random variable X. The Exp and ER$_2$ distributions have C_X^2 equal to 1 and 0.5, respectively, while for the HE$_2$ distribution we set it to 10. For MAP arrivals we also set $C_X^2 = 10$, and the decay rate of the auto-correlation function is set to 0.5. The parameters of the HE$_2$ distribution are computed with the moment-matching method in [15], while the matrices D_0 and D_1 of the MAP are obtained with the method in [16].

Table 6
Approximation error compared with exact results and simulations.

Arr	Ser	U	Measure	Err(%)					
				$C = 5$	$C = 50$	$C = 100$	$C = 500$	$C = 750$	$C = 1000$
Exp	Exp	0.10	mean	<0.01	<0.01	<0.01	<0.01	<0.01	<0.01
			$R_2(99)$	0.06	0.06	0.06	0.06	0.06	0.06
		0.50	mean	2.14	<0.01	<0.01	<0.01	<0.01	<0.01
			$R_2(99)$	5.85	0.04	0.04	0.04	0.04	0.04
		0.90	mean	49.30	0.84	0.01	<0.01	<0.01	<0.01
			$R_2(99)$	55.21	3.46	0.05	0.04	0.04	0.04
		0.95	mean	68.19	9.47	0.98	0.03	0.03	0.03
			$R_2(99)$	70.76	20.98	4.17	0.01	0.01	0.01
Exp	HE_2	0.50	mean	33.23	0.42	0.01	0.01	0.01	0.02
			$R_2(99)$	26.81	0.88	0.01	<0.01	<0.01	<0.01
		0.90	mean	83.57	47.28	25.28	0.03	0.03	0.03
			$R_2(99)$	79.44	54.46	38.43	0.11	0.07	0.07
		0.95	mean	91.31	67.96	49.98	3.35	0.38	0.07
			$R_2(99)$	94.26	83.16	59.31	12.60	1.89	0.10
Exp	ER_2	0.5	mean	0.51	<0.01	<0.01	<0.01	<0.01	<0.01
			$R_2(99)$	2.01	0.02	0.02	0.02	0.02	0.02
		0.90	mean	34.11	0.07	<0.01	<0.01	<0.01	<0.01
			$R_2(99)$	40.69	0.23	<0.01	<0.01	<0.01	<0.01
		0.95	mean	54.01	2.60	0.07	0.01	0.01	0.01
			$R_2(99)$	76.50	47.37	45.90	<0.01	<0.01	<0.01
MAP	Exp	0.50	mean	25.43	<0.01	<0.01	<0.01	<0.01	<0.01
			$R_2(99)$	35.23	0.02	<0.01	<0.01	<0.01	<0.01
		0.90	mean	57.32	11.06	3.37	0.07	0.07	0.07
			$R_2(99)$	56.48	13.87	6.24	<0.01	<0.01	<0.01
		0.95	mean	71.05	21.78	11.02	0.23	0.09	0.08
			$R_2(99)$	70.16	22.26	13.26	0.45	0.04	<0.01
MAP	HE_2	0.50	mean	52.35	5.49	0.40	<0.01	<0.01	<0.01
			$R_2(99)$	47.61	11.00	1.24	<0.01	<0.01	<0.01
		0.90	mean	86.40	54.38	36.53	2.10	0.19	0.14
			$R_2(99)$	85.77	57.24	42.63	7.59	1.37	0.08

For each setting, the simulations were run for 5000 times with 200,000 samples each time, from which we obtain the response-time mean and percentiles, and their 95% confidence intervals. Let the pth percentile of the response-time of a K-node FJ queue be $R_K(p)$. We show in Table 6 the errors of the mean response-time, and $R_2(99)$ as an example. We also consider different load levels, 0.1, 0.5, 0.9, and 0.95. The case with MAP arrivals, HE_2 services, and load 0.95 is not shown here, as we could not obtain a stable result for the tails via simulation due to the high variability. For increasing values of C, Table 6 depicts the relative error obtained with the approximation. For the mean response-time we compare against exact methods for 2-node FJ queues under exponential arrivals and services [4,8]. For queues with MAP arrivals and PH services, and for the percentiles, we compare against simulation results. With $C = 100$, the relative error is below 1% for most of the test cases, with the exception of HE_2 services, and the cases with 0.95 load, which require a larger C to reach this accuracy. We expect this to be the most challenging case, as the large SCV implies service-times with a long tail, potentially causing large differences among the parallel queues. For instance, with Exp arrivals and HE_2 services, under 0.9 load, the error with $C = 100$ and 500 is 38.34% and 0.04%, respectively. In addition, the statistical characteristics of the inter-arrival times have a clear effect on the approximation accuracy, as the error achieved with Exp arrivals is always smaller than the one achieved with MAP arrivals. For instance, for the case with Exp arrivals and services with a 0.95 load, it requires $C = 500$ to reach an error lower than 1%, while for the same case but under MAP arrivals, it requires $C = 1000$. This is caused by the high variability and auto-correlation of the arrival process, as this leads to a high probability that short inter-arrival times come in bursts, creating longer queues. To deal with these large cases we rely on a novel method to compute the matrix T, which is the topic of the next section.

5. The computation of the T matrix

As described in Section 4, finding the PH representation of the waiting-time distribution involves solving Eq. (2) to find T. In this section we propose a new method for solving Eq. (2), and illustrate how it compares with existing methods.

5.1. The Sylvester-equation approach (Sylv)

The standard method [13] to solve Eq. (2) is to start with $T^{(0)} = S^{(\mathrm{MAP})}$, and iteratively compute

$$T^{(n)} = S^{(\mathrm{MAP})} + \int_0^\infty \exp(T^{(n-1)}u)A^{(\mathrm{MAP})}(u)du,$$

until convergence is reached. Although each iteration can be performed using numerical integration, a more efficient and numerically stable approach [17] relies on defining a matrix L as

$$L = \int_0^\infty \exp(Tu)(I_{m_s}\otimes)\exp(D_0u)du,$$

such that Eq. (2) becomes $T = S^{(\mathrm{MAP})} + L(A^{(\mathrm{jump})} \otimes D_1)$. Integrating L by parts, we find that L solves the Sylvester equation $TL + L(I_{m_s} \otimes D_0) = -I$. The method then starts with $T^{(0)} = S^{(\mathrm{MAP})}$, and solves $T^{(n)}L^{(n)} + L^{(n)}(I_{m_s} \otimes D_0) = -I$ in each iteration to find $L^{(n)}$. This is a Sylvester matrix equation that can be solved with the Hessenberg–Schur algorithm [18]. The next iteration continues with $T^{(n+1)} = S^{(\mathrm{MAP})} + L^{(n)}(A^{(\mathrm{jump})} \otimes D_1)$ until convergence is reached. Each step of this iteration therefore takes $O(m^3)$ time.

5.2. A Riccati-equation approach (NARE)

We now propose a different approach for finding T, which consists in defining a non-symmetric algebraic Riccati equation (NARE) [19] associated to an M-matrix. An M-matrix can be written as $\sigma I - B$, with $B \geq 0$ and $\sigma \geq \rho(B)$, where $\rho(B)$ is the spectral radius of B [19]. While NAREs have been used in the past to solve fluid queues, this is first time a NARE is proposed to find the T matrix associated to the age process. Let $\bar{L}$ be

$$\bar{L} = \int_0^\infty \exp(Tu)(A^{(\mathrm{jump})} \otimes \exp(D_0u))du$$

which we integrate by parts to obtain

$$T\bar{L} + \bar{L}(I_{m_s} \otimes D_0) = -A^{(\mathrm{jump})} \otimes I_{m_a}. \tag{9}$$

Since Eq. (2) can be written as $T = S^{(\mathrm{MAP})} + \bar{L}(I \otimes D_1)$, we replace T in (9) to obtain

$$S^{(\mathrm{MAP})}\bar{L} + \bar{L}(I_{m_s} \otimes D_1)\bar{L} + \bar{L}(I_{m_s} \otimes D_0) + A^{(\mathrm{jump})} \otimes I_{m_a} = 0, \tag{10}$$

which is a NARE with $\bar{L}$ unknown. Let H and M be defined as

$$H = -\begin{bmatrix} I_{m_s} \otimes D_0 & I \otimes D_1 \\ -A^{(\mathrm{jump})} \otimes I_{m_a} & -S^{(\mathrm{MAP})} \end{bmatrix}, \qquad M = -\begin{bmatrix} I_{m_s} \otimes D_0 & I \otimes D_1 \\ A^{(\mathrm{jump})} \otimes I_{m_a} & S^{(\mathrm{MAP})} \end{bmatrix}.$$

Then every solution to (10) corresponds to an invariant subspace of H, and M is in fact an M-matrix. Therefore, we can use the Schur decomposition method [19] to determine $\bar{L}$, from which we obtain T. As a result, we can find T in a single iteration that requires $O(8m^3)$ time. Notice that M being an M-matrix is closely related to the definition of $\bar{L}$. Other choices of this matrix do not lead to a NARE with an associated M-matrix.

5.3. Comparison of approaches for calculating T

To illustrate the proposed approach, we compare **NARE** against the **Sylv** method. Table 7 summarizes the computation times that each method requires to find T for the 2-node FJ queue, under different settings. The experiments were performed in MATLAB, using an Intel Core i7-3770 machine, with 16 GB of memory, running at 3.4 GHz. The rows in Table 7 consider different arrival and service processes, and different load levels, as shown in the first two columns. The missing results correspond to cases where the computation time exceeds 2 h. We consider different values of the limit C: 100, 500, 1000. The column labeled m shows the corresponding size of the T matrix. Columns labeled Sylv and NARE hold the computation times for each of these two methods, while column $R_{S/N}$ depicts the ratio between them. In all the test cases, the Sylv and NARE approaches show good numerical stability, which we measure by means of the residual error. From Eq. (9), we define the residual error as $\|T\bar{L} + \bar{L}(I_{m_s} \otimes D_0) + A^{(\mathrm{jump})} \otimes I_{m_a}\|_\infty$, where both T and $\bar{L}$ are obtained with the NARE method, and we use the infinity norm. NARE offers a good numerical behavior under all the experiments conducted, providing a residual error below 10^{-13}. The residual error of Sylv is decided by the stopping criterion of the iteration, which we set to 10^{-10}.

It can be seen that NARE offers much shorter times than Sylv. For instance, with Exp arrivals and services, NARE is one order of magnitude faster than the Sylv method. With MAP arrivals and/or PH services, we observe an increase in the computation times for both methods, since the size of T increases. In this case, we observe a significant gain with NARE over Sylv in all test cases. In particular, with 2-phase PH services, and $C = 1000$, the size of the matrix T is 4004 with Exp arrivals and 8008 with MAP arrivals, and the Sylv approach fails to finish within 2 h. Furthermore, the time that Sylv takes to compute T increases with increasing utilization, while the time for NARE is stable under all load levels, leading to more significant gains under high loads. Overall, the computation times with the NARE method are up to 200 times shorter than with the Sylv method. Clearly, the proposed NARE approach provides a numerically stable and efficient way to calculate the T matrix, which enables the model of the FJ queue to provide higher accuracy by using a large limit C.

Table 7
Computation times (sec) for calculating T.

Arr/Ser	U	$C = 100$				$C = 500$				$C = 1000$			
		m	Sylv	NARE	$R_{S/N}$	m	Sylv	NARE	$R_{S/N}$	m	Sylv	NARE	$R_{S/N}$
Exp/Exp	0.1	101	0.08	0.04	**2.04**	501	8.05	2.03	**3.96**	1001	93.35	17.70	**5.27**
	0.5		0.25	0.04	**6.55**		22.59	1.98	**11.42**		220.08	17.74	**12.40**
	0.9		1.20	0.04	**30.31**		107.16	1.97	**54.38**		1072.61	17.75	**60.43**
MAP/Exp	0.1	202	0.63	0.14	**4.54**	1002	118.18	14.05	**6.93**	2002	1057.43	131.4545	**6.98**
	0.5		3.60	0.12	**29.15**		669.78	13.89	**48.22**		5926.53	127.7037	**46.41**
	0.9		9.45	0.12	**77.75**		1719.85	12.94	**132.93**		/	107.5752	/
Exp/ER$_2$	0.1	404	3.18	0.87	**3.65**	2004	/	150.18	/	4004	/	892.47	/
	0.5		9.77	0.88	**11.07**		/	149.41	/		/	914.74	/
	0.9		43.03	0.84	**51.37**		/	147.74	/		/	909.53	/
Exp/HE$_2$	0.1	404	8.58	0.80	**10.74**	2004	/	130.89	/	4004	/	1243.26	/
	0.5		38.06	0.73	**52.42**		/	130.23	/		/	1238.76	/
	0.9		147.62	0.74	**200.05**		/	128.03	/		/	1256.80	/
MAP/HE$_2$	0.1	808	50.50	8.73	**5.79**	4008	/	1238.74	/	8008	/	6824.39	/
	0.5		218.98	8.65	**25.31**		/	1224.05	/		/	7017.51	/
	0.9		686.77	8.17	**84.08**		/	1218.42	/		/	6614.62	/

6. Approximate analysis

The analytical model introduced in Section 4 produces accurate results for 2-node FJ queues but it becomes time consuming and computationally expensive with increasing K, especially in cases with high loads, which require a large value of the constant C to provide sufficient accuracy. Furthermore, as the value of K increases, simulations must be run for very long times in order to achieve sufficiently accurate values for the response time percentiles, especially in the tail; again, this is more pronounced at high load. We therefore seek an approximate approach based on the maximum of the response times of the spawned subtasks, referred to as the *efficient approximation for tails* (EAT). We start with an approximation for FJ queues with Poisson arrivals and exponential service times – parallel M/M/1 queues – and then extend to queues with MAP arrivals and PH service. We denote the response time random variable for a K-node FJ queue by R_K, with pth percentile $R_K(p)$.

Nelson and Tantawi derived the mean response-time for a 2-node FJ queue by decomposing the response-time of each subtask into the sum of the time R_1 that it spends in its M/M/1 queue and the synchronization delay D spent waiting for its sibling after completion of service, which is non-zero for the subtask that finishes first and zero for its sibling [4]. Then the total response-time for a job in the 2-node queue is shown to be

$$E[R_2] = E[R_1] + E[D] = (12 - \rho)E[R_1]/8,$$

where $E[R_1] = 1/(\mu - \lambda)$ is the mean of the response-time of a single M/M/1 queue, $\rho = \lambda/\mu$ is its utilization, and D is obtained using Little's law, utilizing the average number of subtasks that have completed service and are waiting for their siblings to complete. An approximation is then formulated for a K-node FJ queue, with $2 \le K \le 32$, by observing that the upper and lower bounds of the mean response-time increase at the same rate. The bounds are derived using properties of the random variables concerned: in particular, if the random variables $\{X_1, X_2, \dots, X_n\}$ are *associated*, i.e., the covariance of $f(\dots X_i \dots)$ and $g(\dots X_i \dots)$ is positive for any pair of increasing functions f, g,

$$P(\max_{1\le i\le n} X_i > x) = 1 - P\left(\bigwedge_{i=1}^{n} X_i \le x\right) \le 1 - \prod_{i=1}^{n} P(X_i \le x).$$

The upper bound is attained when the random variables $\{X_1, X_2, \dots, X_n\}$ are independent. It can be shown that the response-times R_i are associated and so

$$P(R_K > t) = P(\max_{1\le i\le K} R_i > t) \le 1 - \prod_{i=1}^{K} P(R_i \le t) = 1 - (1 - \exp^{-(\mu-\lambda)t})^K,$$

the expected value of which is

$$E[R_K] = \int_0^\infty (1 - P(R_K \le t))\mathrm{d}t \le \int_0^\infty (1 - (1 - \exp^{-(\mu-\lambda)t})^K)\mathrm{d}t = \int_0^1 \frac{1 - u^K}{(\mu - \lambda)(1 - u)}\mathrm{d}u = \frac{H_K}{\mu - \lambda}$$

where H_K is the harmonic series $1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{K}$. Similarly, the lower bound H_K/μ is obtained by neglecting queueing effects, so that

$$H_K/\mu \le E[R_K] \le H_K/(\mu - \lambda).$$

The tightness of the bounds is not the main concern, but the authors conclude that $E[R_K]$ must grow at rate H_K since both bounds grow at that same rate. The mean response-time of a $K > 2$-node FJ queue can then be expressed as $E[R_K] = S_K(\rho)E[R_2]$, for $K \geq 2$, where the scaling factor $S_K(\rho)$ is approximated by $S_K(\rho) = \alpha(\rho) + (1-\alpha(\rho))H_K/H_2$ and $\alpha(\rho)$ is to be determined. It follows from simulation results for $K = 4, 8, 16$ and 32 that $\alpha(\rho) \approx 4\rho/11$ provides a good approximation. Thus

$$E[R_K] \approx \left[\frac{H_K}{H_2} + \frac{4\rho}{11}\left(1 - \frac{H_K}{H_2}\right)\right], \quad \text{for } K \geq 2. \tag{11}$$

Using Eq. (11), relative errors of less than 5% were obtained for $K = 4, 8, 16$ and 32.

EAT relies on a similar observation. However, whilst [4] approximates the mean response time in homogeneous K-node FJ queues with exponential inter-arrival and service times for $2 \leq K \leq 32$, we derive the corresponding approximation for response time *percentiles* in more general K-node FJ queues with MAP arrivals and PH service times, for unlimited $K > 2$. We start with an approximation for FJ queues with Poisson arrivals and exponential service times, then extend to queues with MAP arrivals and PH service. We first recap on the necessary background in order statistics and extreme-value theory.

The maximum random variable is a special case of an *order statistic*, the last one, and it is convenient to express our analysis more generally in terms of order statistics. Let $X_{(1)}, X_{(2)}, \ldots, X_{(K)}$ be the ascending ordered sequence of random variables $X_1, X_2, \ldots, X_K$ (with respect to the normal arithmetic ordering), i.e. $X_{(1)} \leq X_{(2)} \leq \cdots \leq X_{(K)}$. Then $X_{(i)}$ is the ith order statistic of $X_1, X_2, \ldots, X_K$ $(1 \leq i \leq K)$ [20]. In particular, $X_{(K)}$ is the maximum, one of the two *extreme* order statistics (1st and Kth). The cumulative distribution function (CDF) of the maximum of K *independent* random variables is

$$F_{X_{(K)}}(x) = \prod_{i=1}^{K} F_i(x),$$

where $F_i(x) = P(X_i \leq x)$. When the X_i are also identically distributed with common CDF $F(\cdot)$, this simplifies to $F_{X_{(K)}}(x) = F^K(x)$.

For i.i.d. random variables with known distribution functions, so-called extreme theory can be applied to approximate the distribution of extreme values, with increasing accuracy as the number of random variables increases. There are three types of distributions for extreme values, among which Type I relates to sets of random variables with exponential tails [21] and includes the *Gumbel distribution* (also called the log-Weibull distribution), with CDF $\exp(-\exp^{-(x-\mu)/\beta})$, for $x \in \mathbb{R}$, which has mode μ (location of the peak of the density function), mean $\mu + \gamma\beta$, and variance $\beta^2\pi^2/6$, where γ is Euler's constant. The *standard Gumbel distribution* has $\mu = 0$ and $\beta = 1$, giving

$$G(x) = \exp(-\exp^{-x}).$$

As $K \to \infty$, the maximum order statistic $X_{(K)}$ for type I distributions is asymptotically distributed as G with normalizing constants a_K and b_K, i.e.,

$$P((X_{(K)} - b_K)/a_K \leq x) = F^K(a_K x + b_K) \to G(x) \quad \text{as } K \to \infty. \tag{12}$$

Here b_K and a_K are the mode and the measure of dispersion of the Gumbel distribution. In the following, we exploit this approximation of extreme values to derive an efficient and accurate approximation for the tail of the response-time distribution in K-node FJ queues. The approximation requires values for the response-time tail when $K = 2$, for which it uses the model detailed in Section 4.

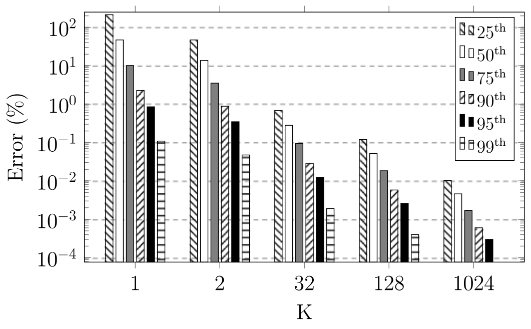
6.1. Asymptotic approximation for Poisson arrivals and exponential services

We first consider a K-node FJ queue where the constituent nodes are M/M/1 queues. We assume the system is homogeneous, i.e. that the M/M/1 queueing nodes are stochastically identical, having the same defining parameters. By using Eq. (13), which we derive below, we can replace the time consuming analytical method for FJ queues with a large number of nodes by a fast alternative that gives a high degree of accuracy, requiring only the analytical results for a single-node and 2-node FJ queues. The usual trade-offs arise amongst accuracy, speed, reliability and even tractability at large K, but we believe our approximation is viable.

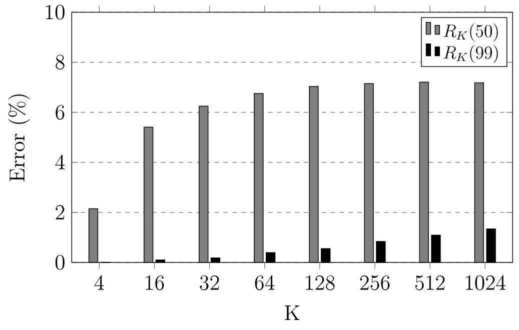
Proposition 2. *Following the conjecture of* [4] *regarding growth rates discussed above, the pth response-time percentile $R_K(p)$ of a K-node homogeneous FJ queue with Poisson arrivals and exponential service times, is given by*

$$R_K(p) \approx R_1(p) + (R_2(p) - R_1(p)) \ln(K)/\ln(2), \tag{13}$$

where $R_1(p)$ and $R_2(p)$ are the pth response-time percentiles of single-node and 2-node FJ queues with the same arrival and service rates, respectively, and ln *is the natural logarithm.*



(a) MAP/HE$_2$.



(b) $R(50)$, $R(99)$.

Fig. 2. Errors (%) of approximation.

Proof. As explained above, the maximum value in a sample of i.i.d. negative exponential random variables approaches the Gumbel distribution as the sample size increases, i.e., when $X_1, X_2, \dots, X_K \overset{\text{i.i.d.}}{\sim} \exp(\alpha)$,

$$F_{X_{(K)}}(x) \approx \exp(-K \exp^{-\alpha x})$$

as $K \to \infty$. It follows that the pth percentile of R_K, which satisfies $F_{X_{(K)}}(x) = p$, can be approximated as

$$R_K(p) \approx L(K,p)/\alpha, \tag{14}$$

where $L(K,p) = \ln(K) - \ln\ln(1/p)$.

An upper bound $R_K^u(p)$ on the response-time percentiles is obtained by assuming that the K parallel queues are independent, as in [4]. Since the response-time distribution of an M/M/1 queue is exponentially distributed with rate $\mu - \lambda$, where λ and μ are respectively the arrival and service rates of the M/M/1 queue (see [22] for example), we have $\alpha = \mu - \lambda$ and so

$$R_K^u(p) \approx \frac{L(K,p)}{\mu-\lambda} = \frac{\ln(K)}{\mu-\lambda} - \frac{\ln\ln(1/p)}{\mu-\lambda}.$$

Hence, the upper bound of $R_K(p)$ grows at rate $\mathcal{O}(\ln(K))$. On the other hand, a lower bound $R_K^l(p)$ is obtained by neglecting queueing effects: in this case, the response-time is the maximum of K *i.i.d.* exponentially distributed service times, each with mean $1/\mu$, i.e., we have

$$R^l(K,p) \approx \frac{L(K,p)}{\mu} = \frac{\ln(K)}{\mu} - \frac{\ln\ln(1/p)}{\mu}.$$

Therefore, this lower bound grows at the same rate $\mathcal{O}(\ln(K))$.

We thus observe that both bounds grow at the same asymptotic rate $\mathcal{O}(\ln(K))$ at large K, from which we conclude that $R_K(p)$ itself must be growing at the same rate, as in [4]. Clearly we cannot assert that this conjecture holds at every value of $K \geq 1$. However, the average growth rate of $R_K(p)$ over an interval $K \in [1,k]$ must be $L(k,p)$ *asymptotically* as $k \to \infty$; if it were greater, the upper bound would eventually be violated, if it were smaller, the lower bound would be violated. Consequently, knowing the value of $R_1(p)$, we may write

$$R_K(p) \approx \alpha(p)\ln(K)R_1(p) + \beta(p), \quad \text{for } K \geq 1, \tag{15}$$

where $\alpha(p)$ and $\beta(p)$ are both functions of p, and are to be determined. Substituting K by 1 in Eq. (15), we find $\beta(p) = R_1(p)$. To obtain $\alpha(p)$, we utilize the matrix analytic model, from which we obtain $R_1(p)$ for the single-node, and $R_2(p)$ for the 2-node FJ queues with the same arrival and service processes. Thus

$$R_2(p) = \alpha(p)\ln(2)R_1(p) + R_1(p),$$

from which we obtain

$$\alpha(p) = \frac{R_2(p) - R_1(p)}{\ln(2)R_1(p)}. \tag{16}$$

Combining Eqs. (16), (15) and $\beta(p)$, we obtain Eq. (13). $\square$

We now provide an example to illustrate this approximation.

Example 1. We use approximation (13) to compute the response-time's 99th percentile in a 1024-node FJ queue ($R_{1024}(99)$), where the arrival and service rates are $\lambda = 0.5$ and $\mu = 1$, respectively. To approximate $R_{1024}(99)$, we first need the percentiles $R_1(99)$ and $R_2(99)$ for the same setting. For a single-node queue with $\lambda = 0.5$ and $\mu = 1$, we obtain $R_1(99) = 9.215$. For the 2-node case, we rely on the analytical model presented in Section 4 to find $R_2(99) = 10.555$, by setting $C = 100$. Thus, following (13), $R_{1024}(99)$ is given by

$$R_{1024}(99) \approx R_1(99) + (R_2(99) - R_1(99)) \ln(K)/\ln(2) = 22.615,$$

while the result from simulation is 22.287, giving a relative error of 1.47%.

To further illustrate the approximation, we look at the behavior of the Gumbel distribution as an approximation of the upper bound for this example. Fig. 2(a) shows the error in the response-time percentiles achieved using the Gumbel distribution, against the actual maximum of the independent exponential random variables that define the upper bound. We depict results for $K = 1, 2, 32, 128$ and 1024, for the 25th, 50th, and 75th percentiles, also known as the first three quartiles, and the 90th, 95th and 99th percentiles, representative of the tail. As expected, the response-time distribution of the upper bound converges to the Gumbel distribution with increasing sample size, but the approximation for the tail is significantly more accurate than for the body. For instance, for $K = 1$, the errors of the first three quartiles are 213.54%, 47.12%, and 10.13%, while for the 90th, 95th and 99th percentiles they are 2.27%, 0.85%, and 0.11%, respectively. Since the approximation (14) depends on the values of $R_1(p)$ and $R_2(p)$, we can expect EAT to provide accurate results for the tail of the distribution, even if its accuracy is limited for the first three quartiles. This is also illustrated in Fig. 2(b), where we show the errors of EAT in approximating the response-time percentiles $R_K(50)$, $R_K(99)$ for values of K between 4 and 1024. Clearly, the error in the 50th percentile is larger than in the 99th percentile, even though both are below 10%, and the latter surpasses 1% error for the largest values of K only. Section 7 provides a thorough evaluation of the proposed approximation.

6.2. Asymptotic approximation for MAP arrivals and PH services

We now extend the preceding approximation for exponential service and inter arrival times to queues with MAP arrivals and PH services. The proof in this case requires a more intricate analysis since the extreme value approximations for the upper and lower bounds may have different normalizing constants.

Proposition 3. *The pth response-time percentile $R_K(p)$ of a K-node homogeneous FJ queue with MAP arrivals and PH services is given by*

$$R_K(p) \approx R_1(p) + (R_2(p) - R_1(p)) \ln(K)/\ln(2), \tag{17}$$

given the same growth rate conjecture as in Proposition 2.

Proof. The distribution of the maximum of i.i.d. random variables whose distributions are PH($\boldsymbol{\tau}, S$) converges to the Gumbel extreme-value distribution since they have exponential tails and so are Class I [23, Theorem 9]. However, different PH representations converge to Gumbel distributions with different normalizing constants, where the two main cases considered in [23] are:

- Case 1: if S is irreducible, the distribution of the maximum converges to a Gumbel distribution with normalizing constants

$$a_K = 1/\eta, \qquad b_K = \eta^{-1} \ln(Kc), \tag{18}$$

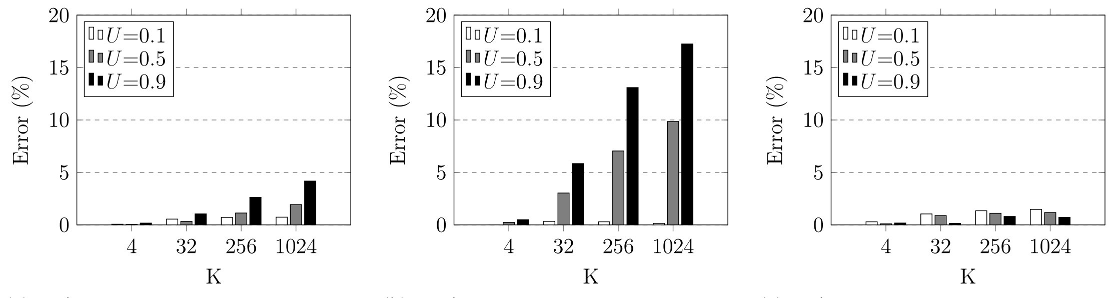
  where $-\eta$ is the eigenvalue of S with largest real part, and $c = \boldsymbol{\tau}\boldsymbol{v}$, with $\boldsymbol{v}$ its right eigenvector.
- Case 2: if no two states in S communicate with each other, i.e., if S can be expressed as an upper-diagonal matrix, then the distribution of the maximum converges to the Gumbel distribution with normalizing constants

$$a_K = 1/\eta, \qquad b_K = \eta^{-1} \left[\ln(K) + (\upsilon - 1) \ln\ln(K) - \ln(\upsilon - 1)! + \ln(\gamma)\right], \tag{19}$$

  where the parameters η, υ, γ are related to the set of paths from any initial state to the absorbing state, as described in [23, Remark 10].

In the following we show that both upper and lower bounds of the response-time percentile $R_K(p)$ grow at a rate $\mathcal{O}(\ln(K))$ if the PH service-time distribution falls into either of these two cases.

As above, the response-time percentiles in the case that the parallel queues are independent provide upper bounds for the K-node FJ queue. Each of these queues is then a MAP/PH/1 queue, the response-time of which follows a PH distribution with parameters $(\boldsymbol{\alpha}^1_{\text{res}}, S^1_{\text{res}})$, from which the eigenvalue and eigenvector pair $(\eta^u, \boldsymbol{v}^u)$ is obtained. To compute PH$(\boldsymbol{\alpha}^1_{\text{res}}, S^1_{\text{res}})$, we rely on the Q-MAM tool [24], which utilizes the age-process approach originally proposed in [13]. Moreover, from [13, Lemma 2.5], we know that the matrix S^1_{res} is irreducible as long as the queue is stable. Hence we can use Eq. (12) and the normalizing constants in Eq. (18) to obtain an approximation for the maximum $X_{(K)}$ of a set of K independent PH variables. It follows that



(a) Exp/Exp. (b) MAP/Exp. (c) Exp/HE$_2$.

Fig. 3. Errors (%) of approximation for $R_K(95)$.

$$F_{X_{(K)}}(x) \approx \exp\left(-\exp^{-(x-b_K)/a_K}\right) = \exp\left(-\exp^{-(\eta^u x - \ln(Kc^u))}\right) = p,$$

where $c^u = \boldsymbol{\alpha}^1_{\text{res}}\boldsymbol{v}^u$, so that the upper bound of $R^u_K(p)$ has the form

$$R^u_K(p) \approx \frac{\ln(K)}{\eta^u} + \frac{\ln(c^u) - \ln\ln(1/p)}{\eta^u}.$$

Therefore the upper bound of $R_K(p)$ grows at asymptotic rate $\mathcal{O}(\ln(K))$.

For the lower bound, we again ignore queueing delays, thus taking the maximum of the service time in the K queues, each of which follows a PH distribution with parameters $(\boldsymbol{\alpha}_{\text{subtask}}, S_{\text{subtask}})$. Different from the upper bound, where the matrix S^1_{res} is always irreducible as long as the queue is stable, the matrix S_{subtask} may fall into either of the 2 cases mentioned. Below we show that, in either of these two cases, the lower bound of $R_K(p)$ grows at a rate $\mathcal{O}(\ln(K))$. In the case that S_{subtask} is irreducible, let $-\eta^l$ be the eigenvalue with largest real part of S_{subtask}, and $\boldsymbol{v}^l$ its associated right eigenvector. Then the lower bound of $R^u_K(p)$ has the form

$$R^l_K(p) \approx \frac{\ln(K)}{\eta^l} + \frac{\ln(c^l) - \ln\ln(1/p)}{\eta^l},$$

which grows at the same rate $\mathcal{O}(\ln(K))$. If S_{subtask} falls into case 2, e.g., the S_{subtask} corresponds to an Erlang or hyper-exponential distribution, the normalizing constants are given by Eq. (19), and it follows that the lower bound $R^l_K(p)$ has the form

$$R^l_K(p) \approx \frac{[\ln(K) + (\upsilon-1)\ln\ln(K)]}{\eta^l} + \frac{[-\ln(\upsilon-1)! + \ln(\gamma) - \ln\ln(1/p)]}{\eta^l}.$$

Thus the lower bound of $R_K(p)$ grows asymptotically at rate $\mathcal{O}(\ln(K))$, as the term $\ln\ln(K)$ is dominated by $\ln(K)$ [25]. Consequently, we conclude that both bounds grow at the same rate $\mathcal{O}(\ln(K))$. Following a similar procedure to that for queues with Poisson arrivals and exponential services, we obtain Eq. (17). □

7. Evaluating the EAT approximation

To evaluate the quality of the EAT approximation, we compare its results against simulation for a broad range of system setups. For each setting, the response time percentiles and their 95% confidence intervals were obtained after 5000 simulations with 200,000 samples each time. We considered different utilization levels, service time distributions, and arrival processes, following the same parametrization as in Section 4.5. To obtain the values for the $K = 2$ case, on which EAT relies, we used $C = 500$ and 1000 to ensure accurate results.

Fig. 3 shows the relative error of $R_K(95)$ in cases with number of nodes $K = 4, 32, 256, 1024$ and at load levels of 0.1, 0.5 and 0.9. Fig. 3(a) considers exponential service times and Poisson arrivals, where we observe that EAT produces accurate results, the error being as low as 4.17% for the most challenging case with $K = 1024$ at load 0.9. We also observe that the error increases with the load, although the errors remain below 5%. Replacing the arrival process by a MAP, as depicted in Fig. 3(b), we observe an increase in EAT's errors, particularly at high utilization. Instead, replacing exponential service times by hyper-exponential, Fig. 3(c) shows how the errors are actually smaller, staying under 2% even at large K. This result suggests that EAT performs better with more variable service times, but worse with more variable inter-arrival times.

Since MAP arrivals are the most challenging, Fig. 4 focuses on this case, fixing the utilization at 0.5 and considering different response-time percentiles and service time distributions. Fig. 4(a) shows how the error increases with K, but decreases with the percentile. In fact, the lowest errors are achieved for the 99th percentile, a result that is related to the faster convergence of the Gumbel distribution for higher percentiles, as discussed in Section 6. If we replace the service-time

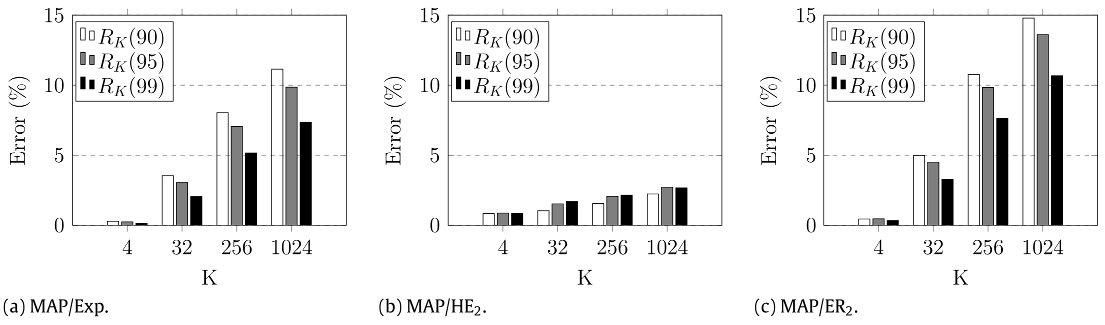


Fig. 4. Errors (%) of approximation when $U = 0.5$.

distribution by a hyper-exponential, we observe a large reduction in the errors, which now remain under 3%. The opposite effect occurs if Erlang service times are considered, as the errors increase to as much as 15%. We therefore observe that EAT offers much better results under highly-variable service times, even under MAP arrivals. Low variability in the service times appears more challenging for EAT, although the errors are remarkably small considering that we approximate cases as large as $K = 1024$ with data from the cases $K = 1$ and $K = 2$ only.

8. Conclusion

We have proposed a stochastic model for K-node homogeneous FJ queues with phase-type processing times and MAP arrivals. Instead of focusing only on the mean response-time, this model facilitates the estimation of the response-time distribution and offers accurate results for small-scale FJ queues. For large-scale FJ queues, where the numerics of the stochastic model become unviable, we proposed an efficient and scalable approximate approach, which only requires the analytical results for a single-node and 2-node FJ queues. Tests against simulation show that the approximation gives a high degree of accuracy for the response-time tails, so that, since mean values can be approximated by known methods, these models cover the majority of response-time requirements in such queueing systems. Future work may consider the extension of the model to consider heterogeneous servers. Also, the efficiency of the approximation makes it a good candidate to support resource provisioning decisions in parallel computing frameworks.

Acknowledgments

The research presented in this paper has received funding from the Engineering and Physical Sciences Research Council (EPSRC) under grant agreement No. EP/L00738X/1. The scripts implementing the methods in this paper are available at http://www.mathworks.com/matlabcentral/fileexchange/51273-efficient-approximation-for-response-time-tails-in-fork-join-queues.

Appendix A. Proof of Proposition 1

First notice that a job can only start service in the m_s service phases with $R(t) = K$, since in other phases one or more subtasks have already completed service, thus the $\mathbf{0}_{1\times m_{nf} m_a}$ vector in $\boldsymbol{\alpha}_{\text{service}}$ corresponds to phases with $R(t) < K$. Now, if a job finds at least one idle server, which occurs with probability $1 - \gamma$, it initiates an all-busy period, and starts service with phase according to $\boldsymbol{\pi}(0)$. On the other hand, a job finds all the servers busy with probability γ, and its starting phase is given by the distribution of the phase in the (X_t, N_t) process after a downward jump, as these are the times when new jobs start service. We know that the joint distribution of the age and the phase is $\boldsymbol{\pi}(x)$. Further, thanks to the level-homogeneity of the process, given that the age is x and the phase is i, the probability that the age reaches $[x + u, x + u + du)$ is independent of x. Thus, the probability that level x is visited after a downward jump, and that this occurs by visiting phase j is given by the jth entry of

$$c \int_0^\infty \int_0^\infty \boldsymbol{\pi}(x) \exp(Tu)(A^{(\text{jump})} \otimes \exp(D_0 u) D_1) du dx = c\boldsymbol{\pi}(0)(-T)^{-1}(T - S^{(\text{MAP})}) = c\boldsymbol{\alpha}_{\text{busy}}(T - S^{(\text{MAP})}),$$

where the inner integral is equal to $T - S^{(\text{MAP})}$ from (2), and c is a normalizing constant that is therefore given by $c^{-1} = \boldsymbol{\alpha}_{\text{busy}}(T - S^{(\text{MAP})})\mathbf{1}$. Thus, a job that waits starts service according to $\boldsymbol{\alpha}_{\text{busy}}(T - S^{(\text{MAP})}) / \left(\boldsymbol{\alpha}_{\text{busy}}(T - S^{(\text{MAP})})\mathbf{1}\right)$. Multiplying this vector by the probability that a job has to wait γ, and adding it to the $(1 - \gamma)\boldsymbol{\pi}(0)$ term completes the result. $\square$

Appendix B. Proof of Theorem 1

To build the PH representation of the response-time distribution we consider separately the paths of jobs that start service immediately after arrival from those that must wait. As stated before, a job starts service without waiting with probability $1-\gamma$, its initial service phase is given by $\boldsymbol{\pi}(0)$, and the sub-generator associated to its service-time is S_{service}. This is reflected in the first set of phases in (8).

If a job has to wait, its response-time is composed of a first stage of waiting followed by a second stage of service. Further, the phase in which the waiting stage ends determines the stage in which the service stage begins. Thus we need to keep track of the phase in which the waiting-time concludes, and find the transition probabilities with which the service stage starts given the phase in which the waiting stage ended. However, the PH representation of the waiting-time distribution cannot be used directly for this purpose as it results from a time-reversal argument [13]. To overcome this, we find the time-reversed version of the service-time distribution, and determine the probability that the time-reversed waiting-time starts in each phase, given the final phase of the time-reversed service-time stage.

The PH representation of the time-reversed service-time distribution $(\boldsymbol{\alpha}_{\text{service}}, S_{\text{service}})$ is given by [26]

$$\tilde{S}_{\text{service}} = \Delta^{-1}S'_{\text{service}}\Delta, \qquad \tilde{\boldsymbol{\alpha}}_{\text{busy}} = (-S_{\text{service}}\mathbf{1})'\Delta,$$

where Δ is a diagonal matrix such that $\Delta\mathbf{1} = \boldsymbol{\eta}'$, and $\boldsymbol{\eta} = -\boldsymbol{\alpha}_{\text{service}}S^{-1}_{\text{service}}$ is the stationary distribution of the phase during service. This distribution is such that its initial phase is actually given by the final phase of the original distribution, and vice versa.

To find how the phase transitions from the end of the service stage to the beginning of the waiting stage, in the time-reversed process, we observe that the joint probability that after a downward jump the level is at least x and the phase is j is given by the jth entry of

$$W(x) = \int_x^\infty \boldsymbol{\pi}(y)\int_0^\infty \exp(Tu)A^{(\text{MAP})}(u)du dy,$$

as this expression considers the probability that any level greater than x is visited by means of a downward jump of any size u. From (2) we know that the inner integral is equal to $T - S^{(\text{MAP})}$, obtaining

$$W(x) = \int_x^\infty \boldsymbol{\pi}(y)(T - S^{(\text{MAP})})dy = -\boldsymbol{\pi}(0)T^{-1}\exp(Tx)(T - S^{(\text{MAP})}) = \boldsymbol{\alpha}_{\text{busy}}\exp(Tx)(T - S^{(\text{MAP})}).$$

Now we perform the same similarity transform as in [13] to obtain the PH representation of the waiting-time process. We thus define a diagonal matrix Λ such that $\Lambda\mathbf{1} = \boldsymbol{\alpha}'_{\text{busy}}$ to obtain

$$W(x) = \boldsymbol{\alpha}_{\text{busy}}\Lambda^{-1}\Lambda\exp(Tx)\Lambda^{-1}\Lambda(T - S^{(\text{MAP})}) = \mathbf{1}'\exp(S'_{\text{wait}}x)\Lambda(T - S^{(\text{MAP})}) = (T - S^{(\text{MAP})})'\Lambda\exp(S_{\text{wait}}x)\mathbf{1}.$$

Notice that we use $S_{\text{wait}} = \Lambda^{-1}T'\Lambda$, as in (5). We therefore have the standard representation for the sub-generator of the waiting-time distribution, but the term $(T - S^{(\text{MAP})})'\Lambda$ captures the transition from the final phase in the time-reversed service process to the initial phase in the time-reversed waiting-time process. We now normalize this expression by evaluating $W(x)$ at $x = 0$, as in this case every entry of $W(0)$ must be 1 since it considers every possible length of the waiting-time process for each final phase of the time-reversed service process. We therefore find the diagonal matrix Γ such that

$$\Gamma W(0) = \Gamma(T - S^{(\text{MAP})})'\Lambda\mathbf{1} = \mathbf{1}.$$

Notice that, from (2), $T - S^{(\text{MAP})}$ is a non-negative matrix since T has non-zero diagonal elements [13]. Thus, from the definition of Γ, the matrix $\tilde{P}_{s,w} = \Gamma^{-1}(T-S)'\Lambda$ is a stochastic matrix. This leads to the second and third blocks in (8), where the response-time process starts with the time-reversed service process, with parameters $(\tilde{\boldsymbol{\alpha}}_{\text{busy}}, \tilde{S}_{\text{service}})$. After the service completes, the waiting-time process starts, selecting the initial phase of the time-reversed waiting process according to the matrix $P_{s,w}$. Notice that absorption from the service-time process can only occur in its first m phases, which correspond to the *full* set of phases defined in Section 4.2. The matrix $\tilde{P}_{s,w}$ corresponds to this m phases. The process then continues according to the sub-generator S_{wait} until absorption. $\square$

Appendix C. Example

This appendix illustrates the different matrices required in the proposed model with a small example. We consider a FJ queue with $K = 3$ and $C = 2$. In this case, the matrices S and $A^{(\text{jump})}$ are given by

$$\begin{array}{c|cccccc} & (1,0,2) & (1,1,1) & (1,2,0) & (2,0,1) & (2,1,0) & (3,0,0) \\ \hline (1,0,2) & -3\mu & 2\mu & 0 & 0 & 0 & 0 \\ (1,1,1) & 0 & -3\mu & \mu & \mu & 0 & 0 \\ (1,2,0) & 0 & 0 & -3\mu & 0 & 2\mu & 0 \\ (2,0,1) & 0 & 0 & 0 & -3\mu & \mu & 0 \\ (2,1,0) & 0 & 0 & 0 & 0 & -3\mu & \mu \\ (3,0,0) & 0 & 0 & 0 & 0 & 0 & -3\mu \end{array},$$

$$\begin{array}{c|cccccc} & (1,0,2) & (1,1,1) & (1,2,0) & (2,0,1) & (2,1,0) & (3,0,0) \\ \hline (1,0,2) & \mu & 0 & 0 & 0 & 0 & 0 \\ (1,1,1) & \mu & 0 & 0 & 0 & 0 & 0 \\ (1,2,0) & \mu & 0 & 0 & 0 & 0 & 0 \\ (2,0,1) & 0 & 2\mu & 0 & 0 & 0 & 0 \\ (2,1,0) & 0 & 2\mu & 0 & 0 & 0 & 0 \\ (3,0,0) & 0 & 0 & 3\mu & 0 & 0 & 0 \end{array}.$$

In this example, the $S_{\text{not-all-busy}}$ matrix is given by

$$S_{\text{not-all-busy}} = \begin{array}{c|cccccc} & (1,0,2) & (1,1,1) & (1,2,0) & (2,0,1) & (2,1,0) & (3,0,0) \\ \hline (1,0,2) & -2\mu & 2\mu & 0 & 0 & 0 & 0 \\ (1,1,1) & 0 & -2\mu & \mu & \mu & 0 & 0 \\ (1,2,0) & 0 & 0 & -2\mu & 0 & 2\mu & 0 \\ (2,0,1) & 0 & 0 & 0 & -\mu & \mu & 0 \\ (2,1,0) & 0 & 0 & 0 & 0 & -\mu & \mu \\ (3,0,0) & 0 & 0 & 0 & 0 & 0 & 0 \end{array}.$$

Finally, the matrices $S_{\text{full-(not-full)}}$ and $S_{\text{(not-full)-(not-full)}}$ are respectively given by

$$\begin{array}{c|ccccc} & (2,(1,0,2)) & (2,(1,1,1)) & (2,(1,2,0)) & (1,(2,0,1)) & (1,(2,1,0)) \\ \hline (3,(1,0,2)) & \mu & 0 & 0 & 0 & 0 \\ (3,(1,1,1)) & \mu & 0 & 0 & 0 & 0 \\ (3,(1,2,0)) & \mu & 0 & 0 & 0 & 0 \\ (3,(2,0,1)) & 0 & 2\mu & 0 & 0 & 0 \\ (3,(2,1,0)) & 0 & 2\mu & 0 & 0 & 0 \\ (3,(3,0,0)) & 0 & 0 & 3\mu & 0 & 0 \end{array},$$

$$\begin{array}{c|ccccc} & (2,(1,0,2)) & (2,(1,1,1)) & (2,(1,2,0)) & (1,(2,0,1)) & (1,(2,1,0)) \\ \hline (2,(1,0,2)) & -2\mu & 2\mu & 0 & 0 & 0 \\ (2,(1,1,1)) & 0 & -2\mu & \mu & \mu & 0 \\ (2,(1,2,0)) & 0 & 0 & -2\mu & 0 & 2\mu \\ (1,(2,0,1)) & 0 & 0 & 0 & -\mu & \mu \\ (1,(2,1,0)) & 0 & 0 & 0 & 0 & -\mu \end{array}.$$

References

[1] A. Thomasian, Analysis of fork/join and related queueing systems, ACM Comput. Surv. 47 (2014) 17.
[2] R. Serfozo, Basics of Applied Stochastic Processes, Springer Science & Business Media, 2009.
[3] L. Flatto, S. Hahn, Two parallel queues created by arrivals with two demands I, SIAM J. Appl. Math. 44 (1984) 1041–1053.
[4] R. Nelson, A.N. Tantawi, Approximate analysis of fork/join synchronization in parallel queues, IEEE Trans. Comput. 37 (1988) 739–743.
[5] C. Kim, A.K. Agrawala, Analysis of the fork-join queue, IEEE Trans. Comput. 38 (1989) 250–255.
[6] S. Balsamo, L. Donatiello, N.M. Van Dijk, Bound performance models of heterogeneous parallel processing systems, IEEE Trans. Parallel Distrib. Syst. 9 (10) (1998) 1041–1056.
[7] J.C.-S. Lui, R.R. Muntz, D. Towsley, Computing Performance Bounds for Fork-join Queueing Models, University of California (Los Angeles), Computer Science Department, 1994.
[8] S. Varma, A.M. Makowski, Interpolation approximations for symmetric fork-join queues, Perform. Eval. 20 (1994) 245–265.
[9] S.-S. Ko, R.F. Serfozo, Response times in M/M/s fork-join networks, Adv. Appl. Probab. (2004) 854–871.
[10] S.-S. Ko, R.F. Serfozo, Sojourn times in G/M/1 fork-join networks, Naval Res. Logist. 55 (2008) 432–443.
[11] G. Latouche, V. Ramaswami, Introduction to Matrix Analytic Methods in Stochastic Modeling, SIAM, 1999.
[12] S. Asmussen, J.R. Møller, Calculation of the steady state waiting time distribution in GI/PH/c and MAP/PH/c queues, Queueing Syst. 37 (2001) 9–29.
[13] B. Sengupta, Markov processes whose steady state distribution is matrix-exponential with an application to the GI/PH/1 queue, Adv. Appl. Probab. 21 (1989) 159–180.
[14] R.A. Horn, The Hadamard product, Proc. Sympos. Appl. Math. 40 (1990) 87–169.
[15] W. Whitt, Approximating a point process by a renewal process, I: Two basic methods, Oper. Res. 30 (1982) 125–147.
[16] A. Heindl, Inverse characterization of hyperexponential MAP(2)s, in: ASMTA, 2004.
[17] Q. He, Analysis of a continuous time SM[K]/PH[K]/1/FCFS queue: Age process, sojourn times, and queue lengths, J. Syst. Sci. Complex. 25 (2012) 133–155.
[18] G. Golub, S. Nash, C. Van Loan, A Hessenberg–Schur method for the problem $AX + XB = C$, IEEE Trans. Automat. Control 24 (1979) 909–913.
[19] D. Bini, B. Iannazzo, B. Meini, Numerical Solution of Algebraic Riccati Equations, in: SIAM Book Series Fundamentals of Algorithms, SIAM, 2012.
[20] H.A. David, H.N. Nagaraja, Order Statistics, Wiley Online Library, 1970.

[21] E.J. Gumbel, J. Lieblein, Statistical Theory of Extreme Values and Some Practical Applications: A Series of Lectures, US Government Printing Office, Washington, 1954, p. 33.
[22] P.G. Harrison, N.M. Patel, Performance Modelling of Communication Networks and Computer Architectures, Addison-Wesley Longman Publishing Co., Inc., 1992.
[23] S. Kang, R.F. Serfozo, et al., Extreme values of phase-type and mixed random variables with parallel-processing examples, J. Appl. Probab. 36 (1999) 194–210.
[24] J.F. Pérez, J. Van Velthoven, B. Van Houdt, Q-MAM: A tool for solving infinite queues using matrix-analytic methods, in: VALUETOOLS, 2008.
[25] M. Sipser, Introduction to the Theory of Computation, Cengage Learning, 2012.
[26] V. Ramaswami, A duality theorem for the matrix paradigms in queueing theory, Stoch. Models 6 (1990) 151–161.

Zhan Qiu is a Ph.D. student in Computing at Imperial College London. She received a M.Sc. in Computer Science from Imperial College London in October 2012. Her research interests include performance and reliability modeling and evaluation of computer and communication systems, parallel processing, performance optimization and resource provisioning.

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 interests center around the performance analysis of computer systems, especially on cloud computing and optical networking.

Peter G. Harrison is Professor of Mathematical Modeling in the Department of Computing at Imperial College London, where he became a lecturer in 1983. He graduated at Christ's College Cambridge as a Wrangler in Mathematics in 1972 and went on to gain Distinction in Part III of the Mathematical Tripos in 1973, winning the Mayhew prize for Applied Mathematics. He obtained his Ph.D. in Computing Science at Imperial College in 1979. He has researched into stochastic performance modeling and algebraic program transformation for some thirty five years, visiting IBM Research Centers during two summers. He has written two books, had over 200 research papers published and held a series of research grants, both national and international. The results of his research have been exploited extensively in industry, forming an integral part of commercial products such as Metron's Athene Client–Server capacity planning tool. Currently, his main research interests are in stochastic modeling, where he has developed the RCAT methodology for finding separable solutions, Hidden Markov Models, response time analysis and modulated fluid models, together with applications such as storage systems, resource virtualization and energy-saving. He has taught a range of subjects at undergraduate and graduate level, including Operating Systems: Theory and Practice, Functional Programming, Parallel Algorithms and Performance Analysis.