Wasiur R. KhudaBukhsh, Amr Rizk, Alexander Frömmgen, Heinz Koeppl:
Optimizing Stochastic Scheduling in Fork-Join Queuing Models: Bounds and Applications - accepted for publication. In: IEEE INFOCOM 2017, April 2017

Optimizing Stochastic Scheduling in Fork-Join Queuing Models: Bounds and Applications

Wasiur R. KhudaBukhsh*, Amr Rizk†, Alexander Frömmgen†, and Heinz Koeppl*
*Bioinspired Communication Systems Lab (BCS), E-Mail: {wasiur.khudabukhsh | heinz.koeppl}@bcs.tu-darmstadt.de,
†Multimedia Communications Lab (KOM), E-Mail: {amr.rizk | alexander.froemmgen}@kom.tu-darmstadt.de,
Technische Universitaet Darmstadt, Germany

***Abstract*—Fork-Join (FJ) queuing models capture the dynamics of system parallelization under synchronization constraints, for example, for applications such as MapReduce, multipath transmission and RAID systems. Arriving jobs are first split into tasks and mapped to servers for execution, such that a job can only leave the system when all of its tasks are executed.**

In this paper, we provide computable stochastic bounds for the waiting and response time distributions for heterogeneous FJ systems under general parallelization benefit. Our main contribution is a generalized mathematical framework for probabilistic server scheduling strategies that are essentially characterized by a probability distribution over the number of utilized servers, and the optimization thereof. We highlight the trade-off between the scaling benefit due to parallelization and the FJ inherent synchronization penalty. Further, we provide optimal scheduling strategies for arbitrary scaling regimes that map to different levels of parallelization benefit. One notable insight obtained from our results is that different applications with varying parallelization benefits result in different optimal strategies. Finally, we complement our analytical results by applying them to various applications showing the optimality of the proposed scheduling strategies.

I. Introduction

Fork-Join (FJ) queuing models naturally capture the dynamics of system parallelization under synchronization constraints. They have seen a rise of interest as a modeling tool in the wake of massive improvement of the infrastructure for cloud computing and large-scale data processing. The emergence of parallel data processing frameworks such as MapReduce [10], [27] and its implementation Hadoop [15] has significantly contributed to the modern IT infrastructure.

Fig. 1 presents a MapReduce abstraction that closely resembles an FJ system. Arriving jobs are first split into tasks each of which is then mapped exactly to one work-conserving server that executes the *map* operation. An optional *combine* operation compresses the intermediate result to reduce the amount of data that is transferred through the network. The compression efficiency depends on the application and, in particular, on the input the data size. A job finally leaves the system when all of its tasks are executed.

In order to design better parallelized systems we require tractable models that connect system dynamics to corresponding key performance metrics. However, until today an exact analysis of FJ queuing systems in a general setup remains elusive [5], [7]. It is particularly hard to find closed form expressions for the steady-state distributions of key quantities

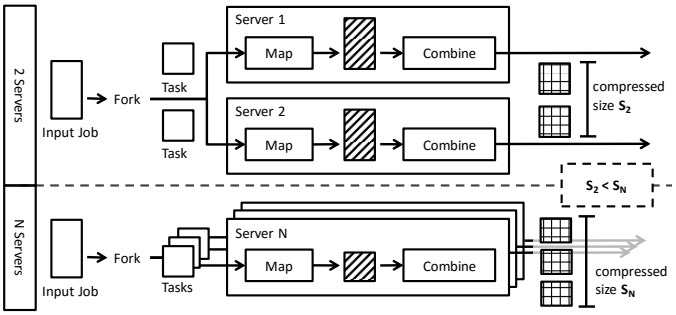


Fig. 1: MapReduce as Fork-Join system: The output size of the *combine*-phase may not scale linearly with the input size.

in FJ systems such as the waiting and response times. In this paper, we contribute computable bounds for heterogeneous FJ systems under a fairly general setup. Our main contribution is a generalized mathematical framework that allows the optimization of probabilistic server scheduling strategies that are shown to save server costs.

In this work, we model one of the main advantages of parallel systems, namely, the application specific parallelization benefit. To this end, we use the notion of service time scaling at each server of the FJ system. Since a job can only leave the system when all of its tasks are executed, we observe a naturally arising synchronization penalty in FJ systems. In this paper, we analytically highlight this trade-off for arbitrary parallelization benefit regimes. We also show the impact of heterogenous servers on this trade-off.

Since in large pools of cloud resources, or, in general, in many parallelized systems, jobs are not mapped to *all* available resources, and given the performance trade-off mentioned above, it is important to select the number of utilized servers from a given pool of available ones in an informed way. In the context of FJ systems, we define a scheduling strategy to be a probabilistic strategy of server selection. Clearly, a deterministic strategy is hence a degenerate case. In this work, we formalize scheduling strategies in FJ systems, derive corresponding stochastic bounds on the waiting and response times, and minimize them to provide optimal strategies under arbitrary application specific parallelization benefits.

Our key contributions in this paper include: **(1)** Computable stochastic bounds for the steady-state distributions of the waiting and response times for a broad class of heterogeneous

FJ systems for various scaling regimes.[1] **(2)** A generalized mathematical framework for scheduling strategies that highlights the trade-off between parallelization benefit and the synchronization penalty, and enables finding optimal scheduling strategies for arbitrary scaling regimes. **(3)** Application of our model to different scenarios showing their efficiency.

We organize the paper with a view to developing the concepts gradually and naturally, and to conveying the intuitions. Starting from the simplest case, we build up to the most general one. The remainder of the paper is structured as follows: Sect. II lays the mathematical foundation of our model of heterogeneous FJ systems. In Sect. III, we introduce scheduling in FJ systems. Our main discussion on application specific scaling and scheduling under arbitrary scaling regimes is given in Sect. IV. In Sect. V, we consider concrete applications of our model and show corresponding findings. Finally, we discuss related work in Sect. VI and then conclude the paper with a short discussion in Sect. VII.

II. Heterogeneous Fork-Join queuing systems

This section introduces FJ systems and provides stochastic bounds on the steady state waiting and response time distributions for a general heterogeneous setting. We denote the set of natural numbers by $\mathbb{N}$. Let $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$. For an event A, $\mathbb{1}(A)$ is its indicator function.

A. System description

Consider a single stage FJ queuing system with N parallel servers as depicted in Fig. 1. The servers are indexed on the set $[N] := \{1, 2, \ldots, N\}$. Jobs arrive at the input station according to some point process with inter-arrival time T_i between the i-th and $(i+1)$-th job, $i \in \mathbb{N}$. In the basic model a job is split into N tasks each of which is assigned to exactly one server. The service time for the task of job i at the n-th server is denoted by the random variable $X_{n,i}$. We shall assume independence of the families $\{X_{n,i}\}$ and $\{T_i\}$ throughout the course of this work. For lack of space, we only consider work-conserving servers in this paper. We assume that the families $\{X_{n,i}\}$ and $\{T_i\}$ admit finite moment generating function (MGF) and Laplace transform, defined as $\alpha_n(\theta) := \mathrm{E}[e^{\theta X_{n,1}}], \beta(\theta) := \mathrm{E}[e^{-\theta T_1}]$, respectively, for some $\theta > 0$ and for all $n \in [N]$. We also assume the job arrival process is a renewal process.

B. Waiting and response times for heterogeneous FJ Systems

In an FJ queuing system the waiting time W_j is defined as 0 for $j = 1$ and $\max\{0, \sup_{k\in[j-1]}\{\sup_{n\in[N]}\{\sum_{i=1}^{k} X_{n,j-i} - \sum_{i=1}^{k} T_{j-i}\}\}\}$, for $j > 1$ [30]. Intuitively a job is considered to be waiting until its last task starts being serviced. The waiting time for the first job is assumed to be zero. Similarly the response time R_j of job j is defined as $\max_{n\in[N]} X_{n,1}$ for $j = 1$ and $\sup_{k\in[j-1]\cup\{0\}}\{\sup_{n\in[N]}\{\sum_{i=0}^{k} X_{n,j-i} - \sum_{i=1}^{k} T_{j-i}\}\}$ for $j > 1$. In order to get steady state representations of the above two random quantities, we require the stability condition $\max_{n\in[N]} \mathrm{E}[X_{n,1}] < \mathrm{E}[T_1]$. Then, by stationarity of the system, we have the following steady state representations of the waiting time W and the response time R:

$$\begin{aligned} W &=_{\mathcal{D}} \sup_{k\in\mathbb{N}_0}\{\sup_{n\in[N]}\{\sum_{i=1}^{k} X_{n,i} - \sum_{i=1}^{k} T_i\}\}, \\ R &=_{\mathcal{D}} \sup_{k\in\mathbb{N}_0}\{\sup_{n\in[N]}\{\sum_{i=0}^{k} X_{n,i} - \sum_{i=1}^{k} T_i\}\}, \end{aligned} \tag{1}$$

where $=_{\mathcal{D}}$ denotes equality in distribution. Now, we provide our first result giving stochastic bounds on the tail probabilities of W and R upon which we build the rest of the paper.

Theorem 1. *Consider an FJ system with N parallel work-conserving servers fed by renewal job arrivals with inter-arrival times T_i, for $i \in \mathbb{N}$. Assuming iid service times $X_{n,i}$ and pairwise independence of the servers, the steady state waiting and response time distributions are bounded by*

$$\begin{aligned} \mathrm{P}(W \geq \sigma) &\leq \exp(-\tilde{\theta}\sigma) \sum_{n\in[N]} \exp(-(\theta_n - \tilde{\theta})\sigma), \\ \mathrm{P}(R \geq \sigma) &\leq \exp(-\tilde{\theta}\sigma) \sum_{n\in[N]} \alpha_n(\theta_n) \exp(-(\theta_n - \tilde{\theta})\sigma), \end{aligned}$$

where θ_n is the positive solution of $\alpha_n(x)\beta(x) = 1$ for $n \in [N]$ and $\tilde{\theta} := \min_{n\in[N]} \theta_n$.

The key steps involved in the proof of the above theorem are: 1) constructing separate martingales for each of the servers; and 2) applying Doob's sub- and supermartingale inequalities (see [3]) to arrive at the bounds. The detailed proof is provided in [20]. Note that the stability condition guarantees the existence of $\theta_n > 0$ such that $\alpha_n(\theta_n)\beta(\theta_n) = 1$ for all $n \in [N]$ (see [7], [28]). Hence, $\tilde{\theta} > 0$ is well defined.

Example: Hedging using revocable cloud resources. We consider a mixed cloud service consisting of both highly guaranteed and revocable resources. This service could be supplied by infrastructure providers such as Amazon EC2 [1], or by a virtual provider on top using, e.g., on-demand or revocable spot market machines [32].

Consider an application of parallel computation under synchronization such as MapReduce [1] or Spark [2] requiring N machines. In this example, we consider the case of exchanging on-demand machines with spot machines to save costs. In general, for a fixed budget the user obtains *faster* spot machines in comparison to on-demand machines. The price difference arises naturally since spot machines are at risk of revocation [32]. We abstract the characteristics of these two classes of machines (*on-demand* and *spot*) through different job service time distributions. Through revocation and application checkpointing procedures [32] that are associated with spot machines, we generally model the tail of the corresponding job service time distributions to decay slower than in the case of on-demand machines. For illustration we assume that the tail of the job service times decays exponentially in case of spot machines while in the case of on-demand machines we model the service times by a uniform distribution. Note that the following argument only requires that the tail of the service times decays slower for spot machines.

[1] We will use the terms *scaling* and *parallelization benefit* interchangeably.

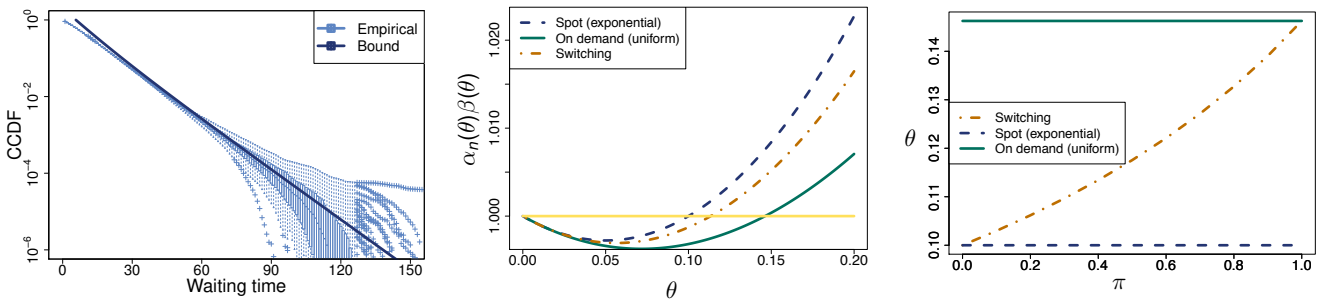


Fig. 2: Example of a heterogeneous FJ system. (Left) Waiting time performance in a MapReduce cloud scenario with $N = 2$ partially volatile servers. One server is on an average faster representing a revocable checkpointed spot server with an exponential tail of service time. The second server provides on average slower service with uniformly distributed service times representing an on-demand server with stronger guarantees. The bound is calculated using Thm. 1. CCDF denotes the complementary cumulative distribution function. (Middle) The FJ system is constrained by the (on an average) faster spot server due to its larger higher moments. This is apparent in the MGF condition $\alpha_n(x)\beta(x) = 1$. Observe that the constraining decay rate is given by $\tilde{\theta} := \min_{n\in[N]} \theta_n$. (Right) A system that switches between spot and on-demand servers with π being the fraction of time where on-demand servers are used. Observe the improvement in the decay rate θ with increasing π. Simulation parameters: spot exponential service rate $\mu = 1$, inter-arrival exponential rate $\lambda = 0.9$ and uniform service time over $[0.001, 2.009]$.

Fig. 2 (left) shows the waiting time distribution in the case of exchanging an on-demand machine by an - on an average faster - spot machine. At first sight this seems to be a good idea, however, looking at Fig. 2 (middle) we clearly see that the system is constrained by the spot machine which has lower average service time, however, a thicker tail. The figure on the right shows the utility of trading an on-demand machine with a spot one. While a greater usage of the on-demand machine incurs greater cost, it also increases the decay rate of the waiting and response times, θ which in turn leads to monetary saving due to faster job execution times.

III. Scheduling tasks in heterogeneous FJ systems

In this section, we study basic scheduling mechanisms that decide on the number of servers to be used from a pool of available servers[2]. Since in large pools of cloud resources (in general for parallelized systems) an arriving job is not scheduled on *all* available resources, we consider for each server if it is selected to execute a task of an arriving job or not. Specifically, when a job arrives we consider that each server n is selected with a probability π_n. This server selection probability π_n can be used to model different aspects of parallelized systems, such as the server failure rate in cloud computing facilities, a quality of service differentiation parameter for different applications, and a tuning parameter to control the degree of replication. Hence, different π_n may exist for different classes of users. Mathematically, the revised task service times $\tilde{X}_{n,i}$ are defined as $X_{n,i}$ with probability π_n and 0 with probability $1 - \pi_n$. The MGF of $\tilde{X}_{n,i}$ is given by $\alpha_n^*(\theta) = (1 - \pi_n) + \pi_n\alpha_n(\theta)$. The stability condition $\max_{n\in[N]} \mathrm{E}[X_{n,i}] < \mathrm{E}[T_1]$ ensures the existence of the decay rate $\theta_n > 0$ from Thm. 1 for each $n \in [N]$ such that $\alpha_n^*(\theta_n)\beta(\theta_n) = 1$. Define $\tilde{\theta} := \min_{n\in[N]} \theta_n > 0$. We retain the same mathematical setup as before except for X being replaced by $\tilde{X}$.

Theorem 2. *Consider an FJ system with N parallel work-conserving servers fed by renewal job arrivals with inter-arrival times T_i, for $i \in \mathbb{N}$. The probability that the n-th server is selected at the arrival of a job is π_n. Assuming iid service times $X_{n,i}$ and pairwise independence of the servers, the steady state waiting and response time distributions are bounded by*

$$\mathrm{P}(W \geq \sigma) \leq \exp\big(-\tilde{\theta}\sigma\big) \sum_{n\in[N]} \exp\big(-(\theta_n - \tilde{\theta})\sigma\big),$$

$$\mathrm{P}(R \geq \sigma) \leq \exp\big(-\tilde{\theta}\sigma\big) \sum_{n\in[N]} \alpha_n(\theta_n) \exp\big(-(\theta_n - \tilde{\theta})\sigma\big),$$

where θ_n is the positive solution of $\alpha_n^*(x)\beta(x) = 1$, for $n \in [N]$ and $\tilde{\theta} := \min_{n\in[N]} \theta_n$.

The proof is provided in [20].

Example: Mixed server pool with different availability. Consider a pool of heterogeneous servers that are available according to some probability π_i. For simplicity, we consider only three heterogeneous servers used for parallel processing. Note that this scenario can be easily generalized to N servers using Thm. 2. For the sake of simplicity, we assume that the task service times are exponentially distributed with server specific rates μ_i and that jobs arrive according to some renewal process with exponentially distributed inter-arrival times with parameter λ. Note that the probability π_i also signifies the fraction of time server i is used, hence, it is directly related to the computation cost in case of time priced resources.

Fig. 3 shows the change in the mean and the percentile of the waiting time due to the addition of a server with a selection probability π_i to a system of two permanently used servers each with $\pi_j = 1$. For example, the lowest curve in Fig. 3 (left) shows the increase in the average waiting time if the slowest server is added with increasing probability π_i.

[2]Note that our notion of scheduling differs from traditional scheduling algorithms such as the Shortest-Remaining-Processing-Time-first (SRPT).

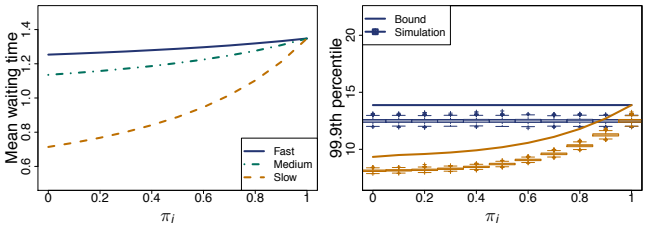


Fig. 3: Impact of the degree of usage of a server on the mean (left) and the 99.9-th percentile (right) of the steady-state waiting times. We consider a pool of three heterogeneous servers (fast, medium, slow), where tasks are always scheduled on two servers and the third server is included with probability π_i. Parameters: service exponential rates $(\mu_1, \mu_2, \mu_3) = (1.5, 1.25, 1)$ and inter-arrival exponential rate $\lambda = 0.5$.

Optimal Strategy. It can be shown that the bound in Thm. 2 is an increasing function of the number of servers N and that the decay rate $\tilde{\theta}$ can be maximized, *i.e.*, the bound can be minimized by choosing only the the strongest server.

IV. Scheduling under application specific scaling

In this section, we analyze scheduling in FJ systems under application specific workloads. We build on the fact that different applications receive different gains from parallelization that lies in the nature of the application itself. Consider for example a Monte-Carlo simulation and a video transcoding application. In the first case, the gain from parallelization is strong and apparent, while in the second case, the gain from parallelization may vary depending on different factors such as the dependency between video macroblocks [9], [24]. We capture these varying gains using the notion of scaled service times. Moreover, in Fork-Join systems (e.g. MapReduce) there is a synchronization price that increases with the number of servers N [5], [30]. We make the case that given these two opposing forces, the scheduling strategy that chooses the number of utilized servers in an FJ system can be optimized to minimize the waiting and response times in the system. We begin with the initial case of homogeneous servers before discussing the more general case of heterogeneous servers.

A. Homogeneous Servers - Linear scaling

The first natural scaling that we analyze is what we call *linear scaling*[3]. This is motivated by examples of FJ systems where incoming jobs are equally divided among the servers. Consider an FJ system with N parallel, identical servers fed by renewal job arrivals with inter-arrival times T_i. We choose the servers probabilistically and once chosen, stick to them for a long time. This allows us to write down steady-state representations conditional on the chosen set. Let the random variable $S \sim f_S \in \mathcal{P}([N], 2^{[N]})$ denote the number of servers chosen to split an incoming job into, where $\mathcal{P}([N], 2^{[N]})$ is the class of all probability distributions on $([N], 2^{[N]})$[4]. Let the service times at the n-th server $X_{n,i}$ be iid for all $i \in \mathbb{N}$ and $n \in [N]$. Suppose the unscaled service time at each server is distributed as X, *i.e.*, $X_{n,i} \mid \{S = 1\} =_{\mathcal{D}} X$ for some X with MGF $\alpha(x)$. We model the reduction of the average amount of work to be performed by each server when we use multiple servers using the following scaling of service times

$$X_{n,i} \mid \{S = s\} =_{\mathcal{D}} \frac{X}{s}. \tag{2}$$

Now, conditional on the given number of used servers $\{S = s\}$ for some $s \in [N]$, the steady-state waiting times W and the response times R can be represented as in (1) with $[N]$ replaced by $[s]$. We have the following result.

Theorem 3. *Consider a stable FJ system with N parallel work-conserving servers and renewal job arrivals with inter-arrival times T_i, for $i \in \mathbb{N}$. Let $S \sim f_S \in \mathcal{P}([N], 2^{[N]})$ denote the number of servers chosen to split an incoming job into. Let the unscaled service times X and the inter-arrival times T be exponentially distributed with parameters μ and λ, respectively. For service times $X_{n,i}$ at the n-th server that are scaled as in (2) independently for all $n \in [S], i \in \mathbb{N}_0$, the steady state waiting and response times are bounded as*

$$\mathrm{P}(W \geq \sigma) \leq e^{\lambda\sigma}\mathrm{E}[Se^{-\mu\sigma S}],$$
$$\mathrm{P}(R \geq \sigma) \leq \frac{e^{\lambda\sigma}}{\rho}\mathrm{E}[S^2e^{-\mu\sigma S}],$$

where $\rho = \frac{\lambda}{\mu}$ is the unscaled utilization level and the optimal strategy with respect to the bound for the waiting time is

$$S_{opt} \sim f_{opt} = \underset{f_S \in \mathcal{P}([N], 2^{[N]})}{\arg\min} \mathrm{E}[Se^{-\mu\sigma S}].$$

The proof is provided in [20]. For a given choice of the distribution of S, which we call a *strategy*, the bounds in Thm. 3 can be computed exactly, for it involves only a summation of finitely many terms. Note that the optimization is essentially over a probability N-simplex $\Delta_N := \{(p_1, p_2, \ldots, p_N) \in [0,1]^N \mid \sum_{k=1}^N p_k = 1\}$.

Interpretation of the server selection strategy: A strategy can be interpreted in two ways: *(i)* it actively arises through users' selection of different numbers of servers to utilize, or *(ii)* it passively arises through a variable number of provided servers that are price volatile, e.g., spot instances at a given budget. In the following, we mainly take the former as an example for strategy derivations.

Note that different strategies lead to varying performance bounds, e.g., consider the case where we select the number of used servers uniformly at random from the pool of N servers, *i.e.*, $\mathrm{P}(S = s) = (1/N)\mathbb{1}(s \in [N])$. Then, for $a > 0$, $\mathrm{E}[Se^{-aS}] = \frac{e^{-a}}{N(1-e^{-a})}[\frac{1-e^{-(N+1)a}}{(1-e^{-a})} - (N+1)e^{-aN}]$, and $\mathrm{E}[S^2e^{-aS}] = \frac{e^{-2a}}{N(1-e^{-a})}[2\frac{(1-e^{-(N+1)a})}{(1-e^{-a})^2} - \frac{2(N+1)e^{-Na}-(1-e^{-(N+1)a})}{(1-e^{-a})} - (N+1)(Ne^{-(N-1)a} + e^{-aN})]$. Setting $a = \mu\sigma$, closed-form

[3]Linear scaling has been introduced in [31] for a fixed number of homogeneous servers N without considering scheduling strategies.

[4]We use the symbol 2^A to denote the power set of a set A.

expressions for the bounds in Thm. 3 are obtained. The uniform distribution allows little control over the number of selected servers. To control the average number of utilized servers $\mathrm{E}[S]$ we employ what we call a *Binomial strategy*, i.e., we let S follow a truncated binomial distribution on $[N]$ with parameters N and $p \in (0,1]$,

$$\mathrm{P}(S=s) = \frac{\binom{N}{s} p^s q^{N-s}}{1-q^N} \mathbb{1}(s \in [N]),$$

writing $q := 1-p$. With abuse of notation, we write $S \sim \underline{\text{Binomial}}(N,p)$. Given the total number of available servers $N \in \mathbb{N}$, the binomial strategy allows us to vary p to control the desired number of on average utilized servers $Np/(1-q^N)$.

Computing the expectations in Thm. 3 for $S \sim \underline{\text{Binomial}}(N,p)$, we get the following bounds

$$\mathrm{P}(W \geq \sigma) \leq N e^{-\theta\sigma} \left[\frac{p}{1-q^N}(pe^{-\mu\sigma}+q)^{N-1}\right] \tag{3}$$

$$\mathrm{P}(R \geq \sigma) \leq \frac{N e^{-\theta\sigma}}{\rho} \left[\frac{p}{1-q^N}(Npe^{-\mu\sigma}+q)(pe^{-\mu\sigma}+q)^{N-2}\right].$$

The proof is provided in [20].

Optimizing the Binomial strategy: Our next goal is to minimize the waiting times given a binomial strategy for server selection. Precisely, given N available servers we look for p that minimizes the right hand side of (3) at some percentile σ, *e.g.*, the 99.9-th percentile. First, we rewrite the right hand side of (3) as $Ne^{-\theta\sigma}\left[(\epsilon q+1-\epsilon)^{N-1}/\sum_{k=0}^{N-1} q^k\right]$ where we define $\epsilon := 1-e^{-\mu\sigma}$. Next, we define $\psi : [0,1) \to \mathbb{R}_+$ as $\psi(q) := (\epsilon q+1-\epsilon)^{N-1}/\sum_{k=0}^{N-1} q^k$ and study its behavior. Taking derivative with respect to q, we get

$$\frac{d}{dq}\psi(q) = \frac{(\epsilon q+1-\epsilon)^{N-2}}{(\sum_{k=0}^{N-1} q^k)^2} \sum_{k=0}^{N-2} (N\epsilon - 1 - k) q^k.$$

Since $(\epsilon q+1-\epsilon)^{N-2}/(\sum_{k=0}^{N-1} q^k)^2 > 0$, the sign of the derivative is dictated by sign of the polynomial $Q(q) := \sum_{k=0}^{N-2}(N\epsilon - 1 - k)q^k$. Note that the coefficients $\{N\epsilon - 1 - k\}_{k\in\{0\}\cup[N-2]}$ of the polynomial are monotonically decreasing, implying there is only one change of sign of the coefficients so that by *Descartes' rule of signs*, there is at most one real root of $Q(q) = 0$. Consequently, the same holds true for $\frac{d}{dx}\psi(x)$. Now, observe that $Q(0) = N\epsilon - 1 > 0$ if $\epsilon > 1/N$. On the other hand, $Q(1) = N(N-1)(\epsilon - 1/2) > 0$ if $\epsilon > 1/2 \iff \sigma > (1/\mu)\ln(2)$. This condition on the 99.9-th percentile of the waiting time holds except for corner cases with nearly no queuing. This gives us a sufficient condition for $\frac{d}{dx}\psi(x) > 0$ implying that $\psi(q)$ is an increasing function of q on $\epsilon > 1/2$. In other words, the tail bound is a decreasing function of p. *Therefore, the optimal strategy would be to set $p_{opt} = 1$ and use all N available servers to make the most of the scaling benefit.* Our analytic arguments are also numerically validated using simulations, e.g., Fig. 5.

Optimization under budget constraint: In the interesting scenario of an application with a budget constraint on the average number of servers it uses, the above reduces to a constrained optimization problem. Precisely, if we have a budget constraint of the form $\mathrm{E}[S] \leq S^*$, the optimization problem can be stated as

$$\min N e^{-\theta\sigma}\left[\frac{p}{1-q^N}(pe^{-\mu\sigma}+q)^{N-1}\right] \quad \text{s. t. } \frac{Np}{1-q^N} \leq S^*,$$

leading to $p^* = \sup\{p \in (0,1] \mid \sum_{k=0}^{N-1}(1-p)^k \geq \frac{N}{S^*}\}$ so that $f_{opt} = \underline{\text{Binomial}}(N,p^*)$, In general, the given bound can always be numerically optimized for any σ.

Generalization to Power series strategies: To obtain bounds in the more general setup of a power series strategy, we assume

$$\mathrm{P}(S=s) := \frac{a_s \kappa^s}{\zeta(\kappa)} \mathbb{1}(s \in \mathbb{N}), \tag{4}$$

where $\zeta(\kappa) := \sum_{k\in\mathbb{N}} a_k \kappa^k < \infty$ for some $\kappa > 0$ and $a_k \geq 0\ \forall k \in \mathbb{N}$. We denote this distribution by $\underline{\text{Pow}}(\kappa,\zeta)$ and the corresponding bounds on the waiting and response time distributions in Thm. 3 evaluate to

$$\mathrm{P}(W \geq \sigma) \leq e^{\lambda\sigma} \frac{\kappa e^{-\mu\sigma}\zeta'(\kappa e^{-\mu\sigma})}{\zeta(\kappa)},$$

$$\mathrm{P}(R \geq \sigma) \leq \frac{e^{\lambda\sigma}}{\rho} \frac{\kappa e^{-\mu\sigma}}{\zeta(\kappa)} [\kappa e^{-\mu\sigma}\zeta''(\kappa e^{-\mu\sigma}) + \zeta'(\kappa e^{-\mu\sigma})].$$

The proof is provided in [20]. For a given form of ζ, the strategy can be optimized to minimize the waiting times. We skip this optimization due to the lack of space. Please note that the above is the *most* general result of this kind.

B. Homogeneous Servers - Partial scaling

In the previous subsection we considered linear scaling of the form (2) that models a perfect work division over s utilized servers in the sense of $\mathrm{E}[X_{n,i}] = \mathrm{E}[X]/s$. In this section, we analyze the general case of application specific scaling, i.e., where the parallelization benefit due to using more servers depends on the application itself. Two prominent examples are: *(i)* MapReduce scenarios where the servers have to separately calculate a state before starting the task executions, and *(ii)* parallelized video transcoding, where some involved decoding operations have a diminishing return on parallelization [9], [24].

Mathematically, we assume that for a certain application with scaling coefficient $\varphi \in [0,1]$, the following scaling down of service times holds,

$$X_{n,i} \mid \{S=s\} =_{\mathcal{D}} \frac{X}{s^\varphi}. \tag{5}$$

Given $\{S=s\}$, the steady-state waiting times W and the response times R have the same representation as in (1) where we need to replace N with s. Now, we present our bounds in the partial scaling regime.

Theorem 4. *Consider a stable FJ system with N parallel work-conserving servers and renewal job arrivals with inter-arrival times T_i, for $i \in \mathbb{N}$. Let the random variable $S \sim f_S \in \mathcal{P}([N], 2^{[N]})$ denote the number of servers chosen to split an incoming job into. Let the unscaled service times X and the inter-arrival times T be exponentially distributed with*

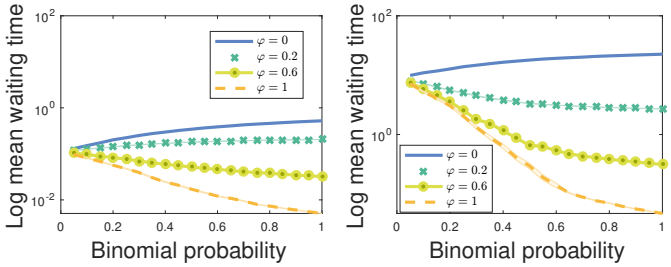


Fig. 4: The impact of the scheduling strategy (given by probability p) together with the parallelization benefit (given by increasing φ) on the mean waiting time in given FJ systems. Simulation parameters: $N = 10$ servers, (Left) low utilization: $\lambda = 0.1$. (Right) high utilization: $\lambda = 0.9$.

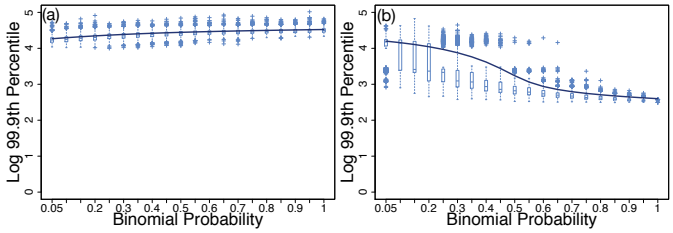


Fig. 5: The impact of the scheduling strategy on the waiting time percentiles. Simulation parameters: $N = 10, \lambda = 0.9$, parallelization benefit: (a) $\varphi = 0$ (b) $\varphi = 0.2$.

parameters μ and λ, respectively. For service times $X_{n,i}$ at the n-th server that are scaled as in (5) *for some $\varphi \in [0,1]$ the steady state waiting and response times are bounded as*

$$\mathrm{P}(W \geq \sigma) \leq e^{\lambda\sigma}\mathrm{E}[S\exp\big(-\mu\sigma S^{\varphi}\big)],$$
$$\mathrm{P}(R \geq \sigma) \leq \frac{e^{\lambda\sigma}}{\rho}\mathrm{E}[S^2\exp\big(-\mu\sigma S^{\varphi}\big)],$$

where $\rho = \frac{\lambda}{\mu}$ is the unscaled utilization level. The optimal strategy with respect to the bound for the waiting time is

$$S_{opt} \sim f_{opt} = \underset{f_S \in \mathcal{P}([N],2^{[N]})}{\arg\min}\ \mathrm{E}[Se^{-\mu\sigma S^{\varphi}}].$$

The proof is provided in [20].

Insights into partial parallelization benefit: Fig. 4 conveys multiple insights into scheduling strategies under different application specific scaling φ. It depicts the mean waiting time in a given FJ system for different scheduling strategies given by the Binomial probability p for various parallelization benefits given by the coefficient φ. The first insight from Fig. 4 is the trade-off between the FJ inherent synchronization penalty and the parallelization benefit due to scaled service times. For a given scheduling strategy in an FJ system, i.e., the probability p, we observe a decrease in the mean waiting time with increasing scaling benefit φ. Second, for low parallelization benefit φ, the synchronization penalty predominates leading to an increase in mean waiting times. We note that this phenomenon also depends on the utilization. Finally, for high parallelization benefit φ, we observe a decay of the mean waiting times with p, i.e., essentially increasing the average number of utilized servers $Np/(1-q^N)$. We observe a general diminishing behavior with p. Hence, for larger φ substantial savings in server cost can be obtained by sacrificing a little in terms of the average waiting time. Fig. 5 shows a similar behavior for the percentiles of the waiting time distribution.

Remarkably, we find that for any fixed stochastic strategy, i.e., $p \in (0,1]$ under no parallelization benefit, the percentiles of the waiting times grow as $O(\log \mathrm{E}[S])$. In case of no stochastic scheduling, i.e., $p = 1$, we recover the behavior of $O(\log N)$ known from [5], [30].

Optimal strategy under partial scaling: The prime motive of the analysis above is to gain analytic insights into the impact of the chosen number of servers on the waiting times for an application with a given scaling φ in a fixed FJ system. In particular, given a $\varphi \in [0,1]$, we find the optimal stochastic scheduling strategy by minimizing the bound obtained in Thm. 4. Observe that as $\varphi \to 0$, the scaling benefit diminishes to zero yielding the unscaled case from Sect. II. Further, as $\varphi \to 1$, we get greater scaling benefit. The optimal strategy, therefore, would be to choose all the servers if the scaling benefit outweighs the synchronization cost, and to choose only the strongest server if it does not. However, this depends on the parallelization benefit φ specific to the given application.

C. Heterogeneous Servers - Hierarchical Model

In this section, we generalize our scaling discussion to the heterogeneous case, building on the analytic intuitions gained in the previous section. We argue that the average service times at different servers are not identical, but rather follow some suitable probability distribution (see Fig. 6). Here, we assume a randomly drawn server has an exponential service rate with parameter μ where μ itself is drawn from an underlying hierarchical distribution f_μ. We present the following result for such a setup, assuming the strict stability $\max_{n\in[N]} \mathrm{E}[X_{n,1}] < \mathrm{E}[T_1]$.

Theorem 5. *Consider an FJ system with N parallel work-conserving servers fed by renewal job arrivals with iid exponentially distributed inter-arrival times T_i with parameter λ, for $i \in \mathbb{N}$. Let the random variable $S \sim f_S \in \mathcal{P}([N], 2^{[N]})$ denote the number of servers chosen to split an incoming job into and the unscaled service time X_n at the n-th server be exponentially distributed with parameter $\mu_n \sim f_\mu$ For service times $X_{n,i}$ at the n-th server that are scaled as*

$$X_{n,i} \mid \{S = s\} =_{\mathcal{D}} \frac{X_n}{s^{\varphi}},$$

independently for all $n \in [s], i \in \mathbb{N}_0$, $\varphi \in [0,1]$, the steady state waiting and response times are bounded as

$$\mathrm{P}(W \geq \sigma) \leq e^{\lambda\sigma}\mathrm{E}[S\exp\big(-\min_{n\in[S]}\mu_n\sigma S^{\varphi}\big)],$$
$$\mathrm{P}(R \geq \sigma) \leq \frac{e^{\lambda\sigma}}{\lambda}\mathrm{E}[S^{\varphi}(\sum_{n\in[S]}\mu_n)\exp\big(-\min_{n\in[S]}\mu_n\sigma S^{\varphi}\big)].$$

The optimal strategy with respect to the bound above for the waiting time is given by

$$S_{opt} \sim f_{opt} = \underset{f_S \in \mathcal{P}([N],2^{[N]})}{\arg\min} \; \mathrm{E}[S \exp\big(-\min_{n\in[S]} \mu_n \sigma S^{\varphi}\big)].$$

The proof is provided in [20].

Example: A two-class system: Consider the case where there are only two types of servers in the system, *fast* and *slow*. In a cloud computing infrastructure, these two types would correspond to different monetary prices. Suppose the exponential service rates of the two types of servers are κ_1 and κ_2, respectively, and the arrival rate is λ with $\lambda < \kappa_1 < \kappa_2$. Denote the probability that a randomly drawn server is of type-1, *i.e.*, has exponential service rate κ_1, by π. Hence, the service rate distribution is given by

$$f_\mu(x) := \pi^{\mathbb{1}(x=\kappa_1)}(1-\pi)^{\mathbb{1}(x=\kappa_2)}. \tag{6}$$

Given n random samples $\mu_1, \mu_2, \ldots, \mu_n$ from the above distribution, we require the first order statistic of the sample $Y_n := \min_{i\in[n]} \mu_i$ to compute the bounds in Thm. 5. The distribution of Y_n is given by $\mathrm{P}(Y_n = \kappa_1) = 1-(1-\pi)^n = 1-\mathrm{P}(Y = \kappa_2)$, such that its MGF is $\mathrm{E}[e^{aY}] = \exp\big(a\kappa_1\big) - \big(\exp\big(a\kappa_1\big) - \exp\big(a\kappa_2\big)\big)(1-\pi)^n$, whence we can compute the bounds obtained in Thm. 5 for different choices of distributions of the number of used servers S. In particular, when $S \sim \underline{\mathrm{Binomial}}(N,p)$ and we receive linear scaling $\varphi = 1$, the upper bounds on the tail probabilities can be explicitly written as

$$\mathrm{P}(W \geq \sigma) \leq e^{\lambda\sigma} \frac{Np}{1-q^N} b_1(\sigma)[1-(1-\pi)(\frac{c_1(\sigma)-c_2(\sigma)}{b_1(\sigma)})],$$

where $b_i(\sigma) := \exp\big(-\sigma\kappa_i\big)(p\exp\big(-\sigma\kappa_i\big)+q)^{N-1}$ and $c_i(\sigma) := \exp\big(-\sigma\kappa_i\big)(p(1-\pi)\exp\big(-\sigma\kappa_i\big)+q)^{N-1}$ for $i = 1, 2$.

While the above example only considers two types of servers, it is worth mentioning that it can easily be extended to take into account finitely many types of servers.

The hierarchical hyper-parameter model: In view of the stability of the system, we take f_μ to be a truncated exponential with (hyper-) parameter μ_0, truncated at λ. That is, we take

$$f_\mu(x) := \mu_0 \exp\big(-\mu_0(x-\lambda)\big)\mathbb{1}(x > \lambda). \tag{7}$$

Given n random samples $\mu_1, \mu_2, \ldots, \mu_n$ from the above distribution, the first order statistic of the sample $Y_n := \min_{i\in[n]} \mu_i$ has a truncated exponential distribution with parameter $n\mu_0$, truncated at λ. The MGF of Y_n is given as

$$\mathrm{E}[e^{aY_n}] = \frac{n\mu_0}{n\mu_0 - a} \exp\big(a\lambda\big).$$

Taking the same approach as in Sect. IV-B, we can compute the waiting and response time bounds from Thm. 5 for different choices of distributions of S. In particular for the linear scaling case, *i.e.*, $\varphi = 1$ and when $S \sim \underline{\mathrm{Binomial}}(N,p)$, the upper bounds on the tail probabilities can be explicitly found as

$$\mathrm{P}(W \geq \sigma) \leq \frac{Np\mu_0}{(1-q^N)(\mu_0+\sigma)}(pe^{-\sigma\lambda}+q)^{N-1}.$$

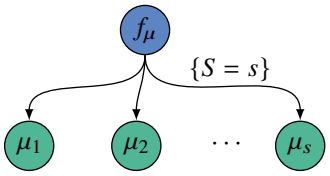
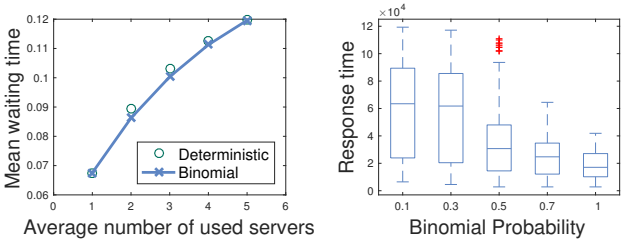


Fig. 6: The hierarchical model for the heterogeneous FJ systems. Conditional on $\{S = s\}$, the average service rates are drawn from a hierarchical distribution f_μ.



(a) Det. vs. Stochastic Strategy (b) MPTCP as FJ Application

Fig. 7: (Left) Deterministic vs. stochastic scheduling strategy for an application with specific φ in a heterogeneous FJ system. Thm. 5 shows that, in general, either strategy can be superior. (Right) The response time decreases with an increasing binomial probability, i.e., with increasing average number of Multipath TCP subflows.

The proof is provided in [20].

Heterogeneous FJ systems - Three forces: As shown above the hierarchical model extends our findings in the previous sections to a wide setting providing insights and lending greater applicability. Thm. 5 shows that *(i)* the first order statistic $Y_s := \min_{i\in[s]} \mu_i$ is decisive for the overall performance of the system, in addition, to the opposing forces from Sect. IV-A, i.e., *(ii)* scaling of service times at each server due to the parallelization, and *(iii)* the synchronization penalty at the output. In fact, the heterogeneous case provides less scaling benefit than the homogeneous case due to Y_s. This impact can be directly seen from the position of Y_s in the exponent in Thm. 5. The optimal strategy given all the relevant parameter values is obtained, as before, by optimizing the upper bound provided in Theorem 5.

V. Evaluation of Application Specific Scheduling in Fork-Join Systems

In this section, we provide evaluations for two exemplary Fork-Join scenarios, namely, *(i)* a comparison of deterministic and stochastic scheduling strategies, and *(ii)* stochastic scheduling results for the transport protocol Multipath TCP. We consider partial as well as linear scaling benefit as given in (5) and (2).

Evaluation of deterministic and stochastic strategies: In the following, we compare the average waiting times in a *heterogeneous* FJ system that uses a binomial scheduling strategy with one using a corresponding deterministic strategy.

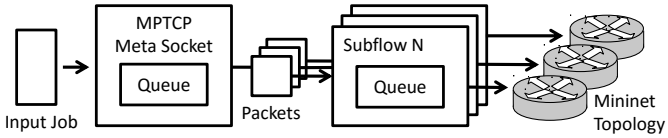


Fig. 8: Network transfer evaluation setup: Multipath TCP splits jobs on multiple subflows.

Our aim is to show the benefit of Thm. 5. We consider renewal job arrivals with exponentially distributed inter-arrival times with parameter $\lambda = 0.1$ at the ingress of an FJ system with $N = 5$ servers each of which can be in a *fast* or a *slow* state with probability 0.5. Hence, the service times are exponentially distributed with an average of $\mu = 1$ in the first state, and $\mu = 0.5$ in the second. We assume an application with a weak parallelization benefit $\varphi = 0.2$. The rationale here is to let the system switch between a regime where the synchronization cost outweighs the scaling benefit, and another regime where the opposite holds true. Given a pool of N available servers, Fig. 7a compares the mean waiting time under a deterministic strategy that uses $1 \leq S' \leq N$ servers to a stochastic strategy that uses an average number of servers $\mathrm{E}[S] = S'$. While this example shows that the stochastic strategy *can* be superior to a comparable deterministic one, we know that in general the superiority of either strategy depends on the number of available servers N, the application specific parallelization benefit φ and the utilization of the FJ system. This strengthens our arguments that for a known application that runs on a given FJ system Thm. 5 provides the optimal scheduling strategy.

Number of Subflows with Multipath TCP: We evaluate the binomial strategy for a network data transfer scenario with linear scaling, in which arriving jobs (*datasets*) of varying sizes are transmitted. Today's networks often provide several disjoint paths, e.g., for ECMP-based load balancing in data-center networks. The scheduling strategy chooses the number of utilized network paths. For a concrete evaluation, we use the Multipath TCP (MPTCP) transport protocol as Fork-Join system [13]. Multipath TCP splits the data on multiple *subflows* and joins them at the receiver side to ensure in-order data transfer for one logical TCP connection (see Fig. 8).

For the measurements, we use the MPTCP Linux kernel implementation [29] and *Mininet* to emulate topologies with disjoints paths. Fig. 7b shows the response time given a binomial strategy, where the response time decreases with increasing binomial probability. Clearly, in this case the higher number of subflows overwhelms the synchronization penalty. Remarkably, we observe diminishing returns in terms of response time with increasing p, which directly translates to the average number of subflows.

VI. Related Work

In this section, we review related work on the Fork-Join queuing systems and their applications. First inequalities for the stationary waiting time distribution in *GI/G/k* queues are shown in [22]. Martingale techniques have been used in queuing theory, in particular, for providing exponential upper bounds by means of maximal inequalities in [8], [11] and later on in [28]. The authors of [28] propose a characterization of queuing systems by bounding suitable martingale constructions, which allows embedding this queuing system characterization into the realm of stochastic network calculus.

An exact analysis of Fork-Join systems with more than two servers in a general setup remains elusive [5], [7], for it is hard to find closed-form expressions for the steady-state distributions. Several works derive exact analytical results for special cases. The authors of [21] obtain transient and steady-state solutions of the FJ queue in terms of virtual waiting times. The special case of an FJ system with two servers having exponential service times under Poissonian job arrivals is studied in [12]. Further, a multitude of useful approximations [18], [23], [25], [33] and bounds [5], [6], [19], [30], [31] are available in the literature. In [34], the authors study the scalability of a general FJ system with blocking, i.e., they study how the throughput of a general FJ system with blocking servers behaves as the number of nodes increases to infinity while the processing speed and buffer space of each node stay unchanged. Another interesting study of limiting behavior is done in [4] where the authors study FJ networks with non-exchangeable tasks under a heavy traffic regime and show asymptotic equivalence between this network and its corresponding assembly network with exchangeable tasks. From the perspective of choosing task assignment policies in distributed server systems, the authors of [14] study various policies and suggest different optimal policies in different situations. Similarly, the work in [16] seeks to quantify the benefits of splitting a task into different queues. It must be noted that the underlying premises in these works are quite dissimilar among themselves and from ours. We consider the works [5], [31] to be the closest to ours. While the basic instruments in deriving bounds in [31] are suitably constructed martingales, as they are in this work, the authors of [31] do not consider the notion of scheduling with respect to application specific scaling and only look at homogeneous servers. Further, [5] provides computable bounds for the expected response times in FJ systems under renewal Poissonian arrival and exponential service times. Their methodology differs from ours as they construct a tractable system to derive the bounds. We, on the contrary, concentrate on bounding the waiting time distributions and use these bounds to gain insights into application specific parallelization benefit and scheduling strategies therefrom.

Several contributions have been made to analyze the performance of applications that can be modeled by FJ systems such as MapReduce [10], [35]. Performance optimization problems that arise for MapReduce systems are surveyed in [15], [27]. In [26], the authors discuss different scheduling strategies regarding Hadoop MapReduce. Related work also considers the performance and pricing of EC2 instances such as on-demand instances (reliable, expensive) or spot instances (volatile, inexpensive). While there has been a number of articles studying spot pricing, mostly taking the provider's

viewpoint such as [17], [36], the authors of [37] take into account the user's standpoint too and explores bidding strategies analytically. These works can feed into our performance model as they essentially relate the obtained computing power, hence the service time distribution, to the monetary cost of computation. For instance, our model can thus be used to analyze a parallelized system where the number of utilized servers is modulated by the price curve of spot instances.

VII. Conclusions

In this paper we provide stochastic bounds on queuing performance metrics for heterogeneous Fork-Join systems under arbitrary level of parallelization benefit. Specifically, using a matching martingale construction we derive bounds on the waiting and response time distributions in this system. We model the application specific parallelization benefit in a given FJ system as a scaling parameter that affects the task service times and analytically show the impact of heterogeneity on this benefit. We highlight a fundamental trade-off between the parallelization benefit and the FJ intrinsic synchronization penalty. Finally, we propose optimal stochastic scheduling strategies in FJ systems for varying application specific parallelization benefits. We conclude our work with a simulation study that evaluates stochastic scheduling strategies in a Multipath TCP scenario while optimizing the number of used paths to improve the system response time.

References

[1] Amazon Elastic Compute Cloud EC2. [Online]. Available: https://aws.amazon.com/ec2/

[2] Apache Spark. [Online]. Available: http://spark.apache.org/

[3] R. B. Ash, *Real Analysis and Probability*. Academic Press, 1972.

[4] R. Atar, A. Mandelbaum, and A. Zviran, "Control of Fork-Join Networks in Heavy Traffic," in *Allerton*, Oct 2012, pp. 823–830.

[5] F. Baccelli, A. M. Makowski, and A. Shwartz, "The Fork-Join Queue and Related Systems with Synchronization Constraints: Stochastic Ordering and Computable Bounds," *Advances in Applied Probability*, pp. 629–660, 1989.

[6] S. Balsamo, L. Donatiello, and N. M. V. Dijk, "Bound Performance Models of Heterogeneous Parallel Processing Systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 9, no. 10, pp. 1041–1056, Oct 1998.

[7] O. J. Boxma, G. Koole, and Z. Liu, *Queueing-theoretic Solution Methods for Models of Parallel and Distributed Systems*. Centrum voor Wiskunde en Informatica, Department of Operations Research, Statistics, and System Theory, 1994.

[8] E. Buffet and N. Duffield, "Exponential Upper Bounds via Martingales for Multiplexers with Markovian Arrivals," *Journal of Applied Probability*, pp. 1049–1060, 1994.

[9] J. Chong, N. Satish, B. Catanzaro, K. Ravindran, and K. Keutzer, "Efficient Parallelization of H.264 Decoding with Macro Block Level Scheduling," in *IEEE ICME*, 2007, pp. 1874–1877.

[10] J. Dean and S. Ghemawat, "MapReduce: Simplified Data Processing on Large Clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[11] N. G. Duffield, "Exponential Bounds for Queues with Markovian Arrivals," *Queueing Systems*, vol. 17, no. 3, pp. 413–430, 1994.

[12] L. Flatto and S. Hahn, "Two Parallel Queues Created by Arrivals with Two Demands I," *SIAM Journal on Applied Mathematics*, vol. 44, no. 5, pp. 1041–1053, 1984.

[13] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure, "TCP Extensions for Multipath Operation with Multiple Addresses," RFC 6824, Internet Engineering Task Force. [Online]. Available: http://www.ietf.org/rfc/rfc6824.txt

[14] M. Harchol-Balter, M. E. Crovella, and C. D. Murta, "On Choosing a Task Assignment Policy for a Distributed Server System," *Journal of Parallel and Distributed Computing*, vol. 59, no. 2, pp. 204–228, 1999.

[15] I. A. T. Hashem, N. B. Anuar, A. Gani, I. Yaqoob, F. Xia, and S. U. Khan, "MapReduce: Review and Open Challenges," *Scientometrics*, pp. 1–34, 2016.

[16] E. Hyytiä and S. Aalto, "To Split or Not to Split: Selecting the Right Server with Batch Arrivals," *Operations Research Letters*, vol. 41, no. 4, pp. 325 – 330, 2013.

[17] H. Jin, X. Wang, S. Wu, S. Di, and X. Shi, "Towards Optimized Fine-grained Pricing of IAAS Cloud Platform," *IEEE Transactions on Cloud Computing*, vol. 3, no. 4, pp. 436–448, 2015.

[18] B. Kemper and M. Mandjes, "Mean Sojourn Times in Two-queue Fork-Join Systems: Bounds and Approximations," *OR Spectrum*, vol. 34, no. 3, pp. 723–742, 2012.

[19] G. Kesidis, Y. Shan, B. Urgaonkar, and J. Liebeherr, "Network Calculus for Parallel Processing," *SIGMETRICS Perform. Eval. Rev.*, vol. 43, no. 2, pp. 48–50, Sep. 2015.

[20] W. R. KhudaBukhsh, A. Rizk, A. Frömmgen, and H. Koeppl, "Optimizing Stochastic Scheduling in Fork-Join Queuing Models: Bounds and Applications," 2016, Extended version. [Online]. Available: https://arxiv.org/abs/1612.05486

[21] C. Kim and A. K. Agrawala, "Analysis of the Fork-join Queue," *IEEE Transactions on Computers*, vol. 38, no. 2, pp. 250–255, Feb 1989.

[22] J. Kingman, "Inequalities in the Theory of Queues," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 102–110, 1970.

[23] S.-S. Ko and R. F. Serfozo, "Sojourn Times in G/M/1 Fork-join Networks," *Naval Research Logistics (NRL)*, vol. 55, no. 5, pp. 432–443, 2008.

[24] M. A. Mesa, A. Ramírez, A. Azevedo, C. Meenderinck, B. Juurlink, and M. Valero, "Scalability of Macroblock-level Parallelism for H.264 Decoding," in *ICPADS*, 2009, pp. 236–243.

[25] R. Nelson and A. N. Tantawi, "Approximate Analysis of Fork/join Synchronization in Parallel Queues," *IEEE Transactions on Computers*, vol. 37, no. 6, pp. 739–743, Jun 1988.

[26] S. R. Pakize, "A Comprehensive View of Hadoop MapReduce Scheduling Algorithms," *International Journal of Computer Networks & Communications Security*, vol. 2, no. 9, pp. 308–317, 2014.

[27] I. Polato, R. Ré, A. Goldman, and F. Kon, "A Comprehensive View of Hadoop Research-A Systematic Literature Review," *Journal of Network and Computer Applications*, vol. 46, pp. 1–25, 2014.

[28] F. Poloczek and F. Ciucu, "Scheduling Analysis with Martingales," *Performance Evaluation*, vol. 79, pp. 56–72, 2014.

[29] C. Raiciu, C. Paasch, S. Barre, A. Ford, M. Honda, F. Duchene, O. Bonaventure, and M. Handley, "How Hard Can It Be? Designing and Implementing a Deployable Multipath TCP," in *USENIX NSDI*, 2012, pp. 29–29.

[30] A. Rizk, F. Poloczek, and F. Ciucu, "Computable Bounds in Fork-Join Queueing Systems," *SIGMETRICS Perform. Eval. Rev.*, vol. 43, no. 1, pp. 335–346, Jun. 2015.

[31] ——, "Stochastic bounds in Fork-Join queueing systems under full and partial mapping," *Queueing Systems*, vol. 83, no. 3, pp. 261–291, 2016.

[32] S. Subramanya, T. Guo, P. Sharma, D. E. Irwin, and P. J. Shenoy, "SpotOn: A Batch Computing Service for the Spot Market," in *SoCC*, 2015, pp. 329–341.

[33] S. Varma and A. M. Makowski, "Interpolation Approximations for Symmetric Fork-Join Queues," *Performance Evaluation*, vol. 20, no. 1-3, pp. 245–265, 1994.

[34] C. H. Xia, Z. Liu, D. Towsley, and M. Lelarge, "Scalability of fork/join queueing networks with blocking," *SIGMETRICS Perform. Eval. Rev.*, vol. 35, no. 1, pp. 133–144, Jun. 2007.

[35] M. Zaharia, A. Konwinski, A. D. Joseph, R. H. Katz, and I. Stoica, "Improving MapReduce Performance in Heterogeneous Environments," in *USENIX OSDI*, vol. 8, no. 4, 2008, p. 7.

[36] L. Zhang, Z. Li, and C. Wu, "Dynamic Resource Provisioning in Cloud Computing: A Randomized Auction Approach," in *IEEE INFOCOM*, 2014, pp. 433–441.

[37] L. Zheng, C. Joe-Wong, C. W. Tan, M. Chiang, and X. Wang, "How to Bid the Cloud," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 4, pp. 71–84, 2015.