Practical Analysis of Replication-Based Systems

Florin Ciucu, *University of Warwick*
Felix Poloczek
Lydia Y. Chen, *Delft University of Technology*
Martin Chan, *University of Warwick*

***Abstract*—Task replication has been advocated as a practical solution to reduce response times in parallel systems. The analysis of replication-based systems typically rests on some strong assumptions: Poisson arrivals, exponential service times, or independent service times of the replicas. This study is motivated not only by several studies which indicate that these assumptions are unrealistic, but also by some elementary observations highlighting some contriving behaviour. For instance, when service times are not exponential, adding a replication factor can stabilize an unstable system, i.e., having infinite delays, but a tempting higher replication factor can push the system back in a perilous unstable state. This behaviour disappears however if the replicas are sufficiently correlated, in which case any replication factor would even be detrimental.**

Motivated by the need to dispense with such common yet unrealistic and misleading assumptions, we provide a robust theoretical framework to compute stochastic bounds on response time distributions in general replication systems subject to Markovian arrivals, quite general service times, and correlated replicas. Numerical results show that our bounds are accurate and improve state-of-the-art bounds in the case of Markovian arrivals by as much as three orders of magnitude. We apply our results to a practical application and highlight that correctly setting the replication factor crucially depends on both the service time distributions of the replicas and the degree of correlation amongst.

I. Introduction

Despite a significant increase in network bandwidth and computing resources, the revenue of online service providers (and not only) is very sensitive to the variability of latencies, especially in their tails (e.g., the 95^{th}-percentile). Several convincing studies reported significant potential revenue loss by Google, Bing, or Amazon, were the latencies higher [36], [23], [38]; a typical cited argument is that an additional 100ms in latency would cost Amazon 1% of sales.

Given the abundance of computing resources, a very simple way to improve latencies is *replication*, a concept which was traditionally used to improve the reliability of fault-tolerant systems [35]. In the context of a multi-server (parallel) system, the idea is merely to replicate a task into multiple copies/replicas, and to execute each replica on a different server. By leveraging the statistical variability of the servers it is expected that some replicas would finish much faster than others; for a discussion of various system/OS factors affecting execution times see [11]. The key gain of executing multiple replicas is not to reduce the average latency but its tail, which is recognized as critically important for ensuring a consistently fluid responsiveness of systems. Therefore, replication can be instrumental to the development of "latency tail-tolerant systems", similarly to its role in fault-tolerant systems [11].

While the idea of using redundant requests is not new, as it was used to speeding up parallel programs [20], [22], it has become very attractive with its implementation in the MapReduce framework through the so-called 'backup-tasks' [12]. Subsequent empirical work demonstrated significant benefits, e.g., to latency reductions in Google's distributed systems [10], in DNS queries and database servers [41], key-value storage systems [39], cloud storage systems [44], or speed-ups of small jobs in data-centers [2] or short TCP flows [45]. Complementary analytical studies (see the Related Work section) provided fundamental insight into the benefits of replication. Constrained by analytical tractability, most of these works rely on several strong assumptions: not only the arrivals are Poisson and the service times are exponentially distributed, but the service times of the replicas are statistically independent.

The motivation of this paper is to advance the understanding of replication systems beyond these three assumptions, which have been shown to be unrealistic. Indeed, the Poisson assumption was shown to fail in the case of a large-scale Google production trace, both in terms of the exponential distribution and the renewal property for the inter-arrival times [29]; for a similar and popular study in the case of (older) HPC workloads see [33]; for a recent related survey see [5]. In turn, analysis of Google, Facebook, and Bing traces indicated heavy-tailed distributions for the job sizes [2], whereas the assumption of independent replicas was shown to be unrealistic in [17] using a real trace.

In addition to these convincing empirical justifications, we provide some elementary analytical arguments, along with simulation results, highlighting that the benefits of replication crucially depend on both the distributional and correlation structures of the service times. A key observation is that the stability region of a system is not monotonous in the replication factor. For instance, an overloaded system (i.e., with infinite latencies/delays) can be stabilized by adding a replica server (i.e., latencies become finite), but adding even more replica servers can push the system back in the perilous unstable state.

To better capture and understand the behaviour of modern workloads in replication-based systems, our contribution is a *robust* analytical framework to compute stochastic bounds on response time distributions. In particular, our framework covers scenarios with Markovian arrivals, general service time distributions (subject to a finite moment generating function (MGF)), and a correlation model amongst the replicas. Using back-of-the-envelope calculations, our results can be imme-

diately used for engineering purposes (e.g., to determine the optimum number of replicated servers to minimize the top percentiles of latencies).

Our analysis relies on a powerful martingale methodology which was recently shown to provide remarkably accurate stochastic bounds in various and challenging queueing systems with non-Poisson arrivals (see, e.g., [7], [8]). Central to this methodology are concepts and techniques from both queueing theory (e.g., [30]) and the stochastic network calculus, a robust methodology yet prone to large numerical inaccuracies in the non-Poisson case [9]. According to several numerical/simulation illustrations, our results are highly accurate, including the case of Markovian arrivals. Moreover, our bounds improve upon the state-of-the-art bounds from [15], [16], relying on the standard network calculus approach, by as much as three orders of magnitude.

To demonstrate the applicability of our theoretical framework we study a scenario whereby replicas can be deferred for execution to better balance the tradeoff between resource usage and response times under replication; such a system has recently been addressed through Google and Bing empirical studies. The key finding is that "statistical independence matters", i.e., unless the replicas are 'sufficiently' independent then replication gains are either marginal or even negative.

The rest of the paper is organized as follows: In § II we introduce the analytical models and discuss related work. In § III we provide our robust theoretical framework dealing with both Poisson and Markovian arrivals, and independent and correlated replicas; we also validate the underlying correlated replica model using real traces. In § IV we investigate a case study and in § V we conclude the paper.

II. Replication Models and Related Work

We consider a parallel system with K homogeneous servers with identical speeds (see Figure 1). A stream of tasks arrives at a dispatcher according to some stationary and ergodic point process; the interarrival times are denoted by t_i with mean $\mathbb{E}[t_1] = \frac{1}{\lambda}$. The process $(t_i)_i$ can have a Markov structure, to be more precisely defined in § III-B.

The service times of the tasks are denoted by x_i and are drawn from some general distribution subject to a finite moment generating function; the average is set to $\mathbb{E}[x_1] = \frac{1}{\mu}$.

The *utilization* of one server, in a system without replicas where tasks are symmetrically distributed, is denoted by

$$\rho := \frac{\lambda}{K\mu} \ .$$

In general, it is assumed for stability that $\rho < 1$. However, in a system with replication, the expression of the utilization ρ may change depending on various factors (e.g., the distribution of tasks' service times) whereas the stability condition may fail (such occurrences will be specifically indicated).

A. Task Assignment Policies

A crucial design component in the parallel server system is the task assignment policy, i.e., how are the incoming tasks assigned to the K servers for processing? While many such policies have been analytically and empirically studied, we focus on few relevant ones in terms of both performance and overhead:

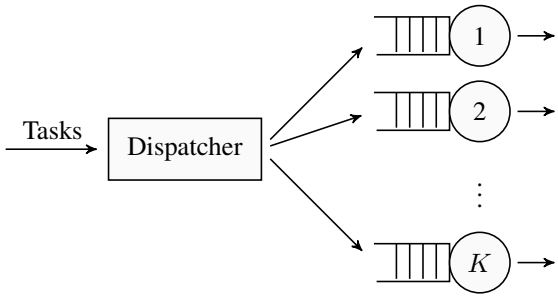


Fig. 1. A parallel system with K servers; tasks are dispatched to the servers in a possibly replicated manner (i.e., the same task to multiple servers)

- **Random**: Each task is dispatched, uniformly at random, to one of the K servers; in the particular case of a Poisson (overall) arrival stream, the task arrivals at each server follow a Poisson distribution with rate $\frac{\lambda}{K}$.
- **Round-Robin**: Tasks are deterministically dispatched in a circular fashion to the K servers, i.e., task i is assigned to server i mod K (with the convention that 0 stands for K); in the case of a Poisson stream, the interarrival times at some server follow an Erlang $E(K, \lambda)$ distribution.
- **G/G/K**: Unlike the previous two schemes, which immediately dispatch the incoming tasks, and whereby tasks enqueue at the assigned servers, in $G/G/K$ it is the responsibility of each server to fetch a single task, from a centralized queue at the dispatcher, once they become idle.
- **(Full-)Replication** (K-replication factor): Each incoming task i is replicated to all the K servers[1]; the corresponding service times are denoted by $x_{i,j}$ for $j = 1, \ldots, K$. Alike in *Random* and *Round-Robin*, each server maintains a local (FIFO) queue.
- **Partial-Replication** (k-replication factor): Besides *full* replication, a task may be replicated to only $k \leq K$ servers; for simplicity, we will assume that both K and k are powers of 2, and that consecutive blocks of k replicas are allocated to the K servers in a round-robin manner. We call the underlying strategy (strict) *Partial-Replication* when $1 < k < K$, and *No-Replication* when $k = 1$.

In terms of analytical tractability, *Random* and *Round-Robin* are significantly more amenable than $G/G/K$; in fact, exact results are known for $G/G/K$ only in the case of Poisson arrivals and exponential service times (in which case the model is denoted by $M/M/K$). However, $G/G/K$ yields significantly better performance (i.e., much smaller response times of the tasks) than *Random* and *Round-Robin*, especially in the case of high variability of the tasks' service times;

[1]For the sake of clarification, the original task is called a replica as well.

in turn *Round-Robin* slightly outperforms *Random* (for an excellent related discussion see [21], pp. 408–430).

It is to be noted however that the superiority of $G/G/K$ is (partly) due to the availability of additional system information, i.e., each task is 'informed' about which server is idle such that it can minimize its response time. Other similar policies, in the sense of relying on additional information, with superior performance include Join the Shortest Queue (JSQ) [1] and its variants. In turn, amongst policies which are oblivious to such information, *Round-Robin* was shown to be optimal for exponential [13], [42] and increasing failure rate distributions [32]; for a recent state-of-the-art queueing analysis of Round-Robin see [24].

A more sophisticated replication strategy was proposed in the context of massively parallel data processing systems in which (large) jobs are split into (smaller) tasks, each assigned to a server; once a fraction of the tasks finish their executions, each of the remaining (straggling) tasks are further replicated. This model appeared in the MapReduce specification [12], and was formally studied in terms of the underlying response time / resource usage tradeoff, albeit by disregarding queueing effects in [43]. Another strategy used by Google is to defer the start of executing the second replica for some suitable time, in order to reduce resource usage [11].

In a variant of *Replication* whereby only the fastest $l \leq K$ tasks are required to complete (whilst the residual tasks are purged) general lower and upper bounds on average response times appeared in [26] (the same model was *qualitatively* studied in [37]). In particular, it was shown that *Replication* outperforms the corresponding *M/M/K* model. Further upper bounds were derived in the case of general service time distributions, using existing bounds on the first two moments of the l^{th} order statistics.

Another set of qualitative results, on the superiority of *Full-Replication* for a specific type of service time distributions (including the exponential) is presented in [31]. Interestingly, under a discrete time model with geometric service time distributions, it is shown in [3] through quantitative results that *No-Replication* is optimal (for an explanation of the apparent contradiction between exponential and geometric service time distributions, with respect to the optimality of the replication model, see [31]). Related results, along with delay optimal scheduling, are given in a fork-join queueing system in which tasks can be replicated [40].

A generalized version of *Partial-Replication* considers the situation when the fastest $l \leq k$ replicas finish their execution; a practical use of this generalization is in coded distributed storage systems [37]. The key result is that under arrivals with *independent increments* and exponential (or '*heavier*') service times, *Full-Replication* minimizes average response times. In turn, in the case of '*lighter*' service times and 100% utilization, a replication factor greater than one is detrimental. The underlying proofs use an ingenious coupling argument, but do not provide quantitative results.

The perhaps most fundamental related result obtained so far is a recent exact analysis [18]. While the analysis critically relies on the Poisson/exponential models, a key analytical contribution is capturing multi-class arrivals (i.e., different arrival streams are served by different sets of (replicated) servers). The elegance of the results lends itself to several fundamental and contriving insights into the properties of replication, especially accounting for the multi-class feature of the model. A set of related exact results, for a partial-replication strategy with random selection of the servers, is presented in [19].

More general stochastic bounds in replication systems are obtained in [15], including the very challenging multi-stage case, by leveraging the analytical power of the stochastic network calculus methodology. While the underlying arrival and service models from [15] are more general than ours, the crucial difference is in handling the underlying correlation structures: concretely, while [15] deals with arbitrary correlation structures yielding stochastic bounds holding in great generality, we exploit the specific correlation structures through the martingale methodology.

B. Purging/Non-Purging Strategies

One commonality of the previous works is using a *purging* replication strategy to deal with residual resources:

- **Purging**: A task is considered to complete (and hence its response time is determined) when the fastest replica finishes its execution; at the same time, the residual replicas are all purged/cancelled from the system (with some negligible related cost).

The alternative is a non-purging strategy:

- **Non-Purging**: A task response time is determined as in the *Purging* case, but the remaining replicas leave the system no sooner than their execution end.

Purging is clearly more efficient from a purely *task response-time* perspective, as it frees resources once the first replica completes; this operation demands however synchronization overhead amongst the servers. One basic reason for this superiority is that in the *Non-Purging* model the utilization increases k-fold for a k-replication factor, for any task service time distribution; in particular, a 2-replication factor requires the replica-free system to have a utilization under 50% (otherwise the response times get unbounded). In turn, the growth of the utilization is less pronounced in the *Purging* model, depending on the type of distribution of the service times; in fact, and perhaps counterintuitively, there is no increase in the case of the exponential distribution regardless the replication factor.

Besides the advantage of a better queueing performance, the *Purging* model is much easier to analyze because the analysis is simplified to the analysis of a single queue with service times as first-order statistics. In turn, in the *Non-Purging* model, the execution times of the replicas are not synchronized, which poses significant technical complications. The only analytical study of *Non-Purging* is considered in [41]; besides the classical and simplifying assumptions of Poisson arrivals and exponential service times, the underlying queueing

analysis critically relies on an artificial statistical independence assumption amongst the queues. Using this assumption, it is shown that below a utilization threshold of 33%, a 2-replication factor strategy does improve the response time despite the inherent doubling of the utilization.

We finally mention an *Early Purging* model, in which residual replicas are purged once the first one starts its execution, and which appeared in [11] and further analyzed in [27]; besides reducing the resource usage, it was shown that this model can also significantly reduce response times despite the apparent loss of diversity, at high utilizations.

C. Runtime Models

What most of the previous analytical works have in common is the following model

- **Independent Runtimes (IR)** [17]: The service times of the replicas are statistically independent.

As it was pointed out in [17], the IR model is unrealistic from a practical point of view; in particular, while many theoretical results under the IR model suggest that increasing the number of replicas decreases the response times (for others see [28]), the same does not hold in practice. Therefore, a more realistic model is

- **Correlated Runtimes (CR)**: The service times of the replicas exhibit some form of correlation.

Such a model is introduced in [17], whereby the service times of the replicas have in common a common (random) multiplicative factor; under a Poisson arrival model and a specific replication strategy, the response times are computed in terms of approximations, which are however remarkably accurate when the replication factor is much smaller than the number of servers. In this paper we use instead an alternative model from [26] (see § III-C) whose additive rather than multiplicative structure is more convenient in our framework (essentially by leveraging the simple fact that the exponential of a sum is the product of exponentials).

III. A Robust Analysis of Replication-Based Systems

Here we present the main (theoretical) results of this paper for models with *Partial-Replication*, *Purging*, and both *Independent* and *Correlated Runtimes*. The accuracy of the proposed results will be evaluated numerically using both simulations and state-of-the-art results. The cases of correlated runtimes/replicas will be motivated through empirical measurements (see § III-D).

We point out that the proofs of the main results are standard and follow a martingale-based methodology (see, e.g., [7]). The key contribution, from an analytical point of view, lies in the construction of the underlying martingales in the Markovian case (see Theorem 3), which is instrumental for getting numerically accurate results[2].

[2]Using different martingales for the same problem can yield fundamentally (numerically) different results, see, e.g., [6] and [4].

We assume a queueing system with K servers and interarrival times between jobs i and $i+1$ denoted by t_i. Upon its arrival, job i is replicated to $k \leq K$ servers where they are processed with service times $x_{i,1}, \ldots, x_{i,k}$, respectively. For simplicity, we throughout assume that K is an integral multiple of k. Further, the jobs are assigned to the $\frac{K}{k}$ batches in a round robin scheme, i.e., the interarrival times for one batch can be described as:

$$\tilde{t}_i := \sum_{j=0}^{K/k-1} t_{(i-1)\frac{K}{k}+j} \ .$$

While other analytical works on replication-based systems chose a *Random* assignment policy (e.g., [17]), we choose the *Round-Robin* policy as it is best suitable in our framework from an analytical point of view, in particular to deal with the challenging Markovian case; we note that [17], as well as other works using the *Random* policy, are restricted to Poisson arrivals. Moreover, *Round-Robin* is in fact superior to *Random* in a system without replication (recall the discussion from § II-A), an advantage which is expected to hold in a system with replication as well.

The following recursion describes the response time r_{i+1} of job $i+1$, i.e., the time between the job's arrival and its service being complete:

$$r_1 := \min_{j\leq k} x_{1,j} \ , \quad r_{i+1} := \min_{j\leq k}\{x_{i+1,j}\} + \max\{0, r_i - \tilde{t}_i\} \ ,$$

resulting in a representation of the *steady-state* response time r as:

$$r =_{\mathcal{D}} \max_{n\geq 0} \left\{ \sum_{i=1}^{n+1} \min_{j\leq k}\{x_{i,j}\} - \sum_{i=1}^{n} \tilde{t}_i \right\} \ , \tag{1}$$

where $=_{\mathcal{D}}$ stands for equality in distribution, and the empty sum is by convention equal to 0.

Depending on the correlation between either the interarrival times and the service times, respectively, we consider four different scenarios: In § III-A, all random variables t_i, $x_{i,j}$ are assumed to be independent. In § III-B, the interarrival times are driven by a certain Markov chain, whereas in § III-C the service times are correlated through a common additive factor. Finally, in § III-E, a combination of both correlation models is considered; for the main proofs see Appendix § A.

A. Independent Arrivals, Independent Replication

As stated above, we consider the scenario of *independent replication*, i.e., $\{t_i, x_{i,j} \mid i \geq 1,\ j \leq k\}$ is an independent family of random variables.

The next Theorem provides an upper bound on the CCDF of r as defined in Eq (1):

Theorem 1. *Let θ_{ind} be defined by*

$$\theta_{ind} := \sup\left\{ \theta \geq 0 \;\middle|\; \mathbb{E}\left[e^{\theta \min_{j\leq k}\{x_{i,j}\}}\right] \mathbb{E}\left[e^{-\theta t_i}\right]^{\frac{K}{k}} \leq 1 \right\} \ . \tag{2}$$

Then the following bound on the response time holds for all $\sigma \geq 0$:

$$\mathbb{P}(r \geq \sigma) \leq \mathbb{E}\left[e^{\theta_{ind} \min_{j\leq k}\{x_{1,j}\}}\right] e^{-\theta_{ind}\sigma} \ .$$

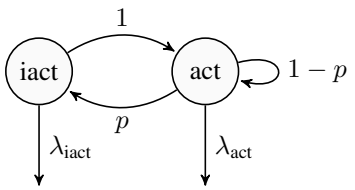


Fig. 2. Two-state Markov chain $Z(n)$

The key assumptions which considerably simplify the proof are the *Purging* model along with the batch allocation of replication in a round-robin manner. In this way the analysis simplifies to the analysis to a single queue with service times as first-order statistics, which is amenable to the bounding technique for GI/GI/1 queues from [30]; one technical complication is that the MGF of the first-order statistic may not be available in explicit form. The more sophisticated analysis in the non-renewal case will be addressed next.

B. Markovian Arrivals, Independent Replication

Motivated by the empirical observations from [29], on the inappropriateness of the Poisson model to fit real production traces, we now turn to the more realistic scenario where the interarrival times are correlated: A two-state Markov chain $Z(n)$ alternates between *active* and *inactive* periods; while in the active state, exponentially distributed interarrival times are generated with parameter λ_{act}, and the chain turns inactive with probability $p > 0$. In the inactive state, *one* interarrival time (exponentially distributed, parameter $\lambda_{\text{iact}} < \lambda_{\text{act}}$) is generated, and the chain jumps back to the active state (see Figure 2)[3]. Formally, let

$$t_{i,\text{act}} \sim \text{Exp}(\lambda_{\text{act}}) \ , \quad t_{i,\text{iact}} \sim \text{Exp}(\lambda_{\text{iact}})$$

be i.i.d. random variables and define the sequence of interarrival times t_i by

$$t_i := t_{i,Z(i)} \ .$$

The steady state distribution π of the Markov chain is given by

$$\pi_{\text{act}} = \frac{1}{1+p} \ , \quad \text{and} \quad \pi_{\text{iact}} = \frac{p}{1+p} \ ,$$

such that for the average of the interarrival times holds

$$\mathbb{E}[t_i] = \left(\lambda_{\text{act}}^{-1} + p\lambda_{\text{iact}}^{-1}\right) / \left(1+p\right) \tag{3}$$

Note that the transition matrix of $Z(n)$ is given by:

$$T := \begin{pmatrix} 0 & 1 \\ p & 1-p \end{pmatrix} \ .$$

In order to state the main result of this section, we need the following transform of matrix T:

[3]The underlying theory extends to more complex Markovian structures, see, e.g., [7], at the expense of notational complexity; the chosen 2-state Markov model is arguably sufficient for the paper's purposes.

Definition 2. *For $0 \le \theta < \lambda_{iact}$, let T_θ denote the following matrix:*

$$T_\theta := \begin{pmatrix} 0 & \frac{\lambda_{act}}{\lambda_{act}+\theta} \\ p\frac{\lambda_{iact}}{\lambda_{iact}+\theta} & (1-p)\frac{\lambda_{act}}{\lambda_{act}+\theta} \end{pmatrix} \ .$$

Further, let $\xi(\theta)$ denote the spectral radius of T_θ, and $h = (h_{act}, h_{iact})$ be a corresponding eigenvector.

Note that T_θ is an exponential transform of T which has the Laplacians of the respective arrival times as an additional factor in each column. In particular, with $\theta = 0$ we recover the transition matrix itself, i.e., $T_0 = T$.

The following Theorem is the analogous result to Theorem 1 (note that the service times $x_{i,j}$ are still assumed to be i.i.d.):

Theorem 3. *Let $1 \le k \le K$ and θ_{mkv} be defined by*

$$\theta_{mkv} := \sup\left\{\theta \ge 0 \;\middle|\; \mathbb{E}\left[e^{\theta \min_{j\le k} x_{1,j}}\right] \xi^{\frac{K}{k}}(\theta) \le 1\right\} \ .$$

Then, for the system with replication to k out of K servers, the following bound on the response time holds for all $\sigma > 0$:

$$\mathbb{P}(r \ge \sigma) \le \mathbb{E}\left[e^{\theta_{mkv} \min_{j\le k} x_{1,j}}\right] e^{-\theta_{mkv}\sigma} \ .$$

C. Independent Arrivals, Correlated Replication

We now address the more realistic scenario when the replicas $x_{i,j}$ are no longer independent; we consider the following correlation model from [26] (for additional numerical justification see § III-D):

$$x_{i,j} = \delta y_i + (1-\delta)\, y_{i,j} \ , \tag{4}$$

where the random variables y_i and $y_{i,j}$ are i.i.d. Here, the parameter δ describes the degree of correlation amongst the replicas: $\delta = 0$ corresponds to the i.i.d. case from § III-A, whereas for $\delta = 1$ the K servers are entirely synchronized in which case no replication gain could be achieved.

For simplicity, the interarrival times t_i are first assumed to be i.i.d. as in § III-A.

Theorem 4. *Let θ_{cor} be defined by*

$$\theta_{cor} := \sup\left\{\theta \ge 0 \;\middle|\; \mathbb{E}\left[e^{\theta\delta y_i}\right]\mathbb{E}\left[e^{\theta(1-\delta)\min_{j\le k} y_{1,j}}\right] \mathbb{E}\left[e^{-\theta t_1}\right]^{\frac{K}{k}} \le 1\right\} \ .$$

Then the following bound on the response time holds for all $\sigma \ge 0$:

$$\mathbb{P}(r \ge \sigma) \le \mathbb{E}\left[e^{\delta\theta_{cor} y_1}\right]\mathbb{E}\left[e^{(1-\delta)\theta_{cor}\min_{j\le k} y_{1,j}}\right] e^{-\theta_{cor}\sigma} \ .$$

Proof. Entirely analogous to the proof of Theorem 1. □

To illustrate the impact of the correlation parameter δ, we consider the special case when y_i and $y_{i,j}$ are exponentially distributed with parameter μ. Clearly,

$$\min_{j\le k} y_{1,j} \sim \text{Exp}(k\mu) \ ,$$

so that $\theta_{\text{cor}} > 0$ is the solution of

$$\frac{\mu}{\mu - \delta\theta} \ \frac{k\mu}{k\mu - (1-\delta)\,\theta} \ \frac{\lambda}{\lambda+\theta} = 1 \ .$$

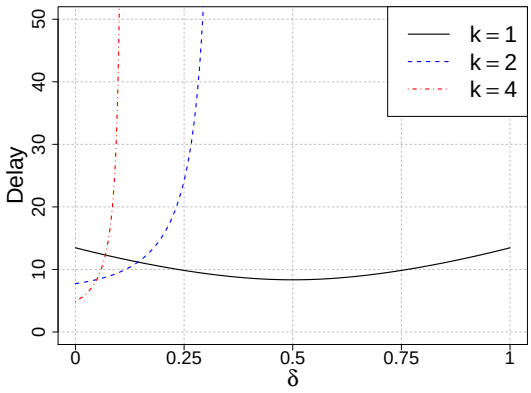


Fig. 3. Delay for the 99%-percentile, from Theorem 4, as a function of the degree of correlation δ ($\lambda = 4 * 0.75$, $\mu = 1$, $K = 4$, $k = 1, 2, 4$)

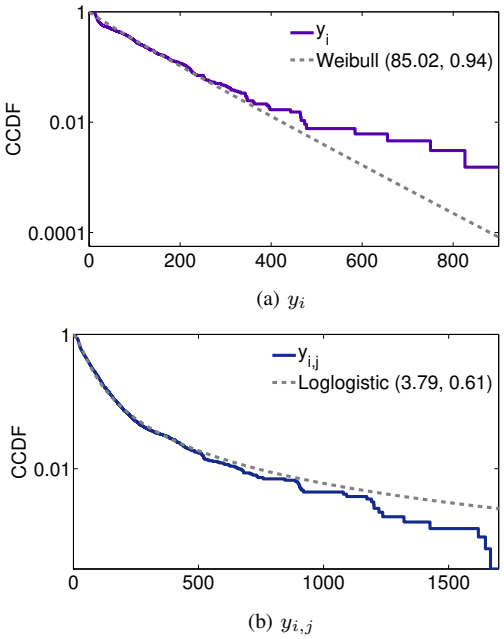
Further, Figure 3 illustrates the 99%-percentile of the delay as a function of the degree of correlation δ for several numbers of replicas k. Strictly from the point of view of the stability region replication (both $k = 2$ and $k = 4$) is detrimental as the corresponding systems quickly become unstable. In contrast, from the point of view of delays, replication can be beneficial within a subset of the corresponding stability region (e.g., to the left of where the '$k = 1$' and '$k = 4$' curves intersect) notwithstanding its strict inclusion in the stability region of the non-replicated system. This fundamental observation can be intuitively explained in that for larger values of the degree of correlation δ, the servers become more synchronized and consequently no significant *replication gain* can be achieved; alternatively, as $1-\delta$ gets smaller, the potential replication gain stemming from $\min_{j \leq k} y_{i,j}$ diminishes. As a side remark, the symmetry in the delay for $k = 1$ is due to the underlying Erlang distribution, which minimizes its variance at $\delta = .5$.

D. An Empirical Justification for the Correlated Service Model

Before presenting the results for correlations in both arrivals and replicas, we validate the earlier correlated service time model from Eq. (4).

We use a trace of service times collected from a large web service system that adopts replication to improve latency: Users' original and replicated requests are first sent to the front-end Apache servers and then pages are retrieved either from the memcache servers or the back-end database, depending on whether the content is cached at the memcache servers. We measured the time from a request's arrival at the Apache server until the request being returned, either from the memcache or the database servers. As the trace is collected at a low arrival rate, the measured times are exactly the service times and do not include a queueing delay.

Concretely, the trace is collected from a web system consisting of 16 homogeneous virtual servers and uses a replication factor of 5, i.e., a single request is processed at 5 different servers in parallel. The trace contains a total number of 1000 requests, and each request has 5 different service times, one from each of the replicas.



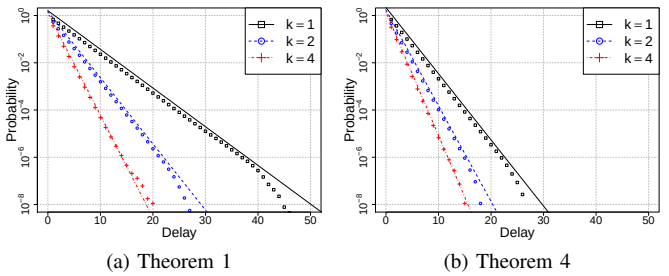
(a) y_i

(b) $y_{i,j}$

Fig. 4. Empirical and fitted distributions of service times

We use the following fitting procedure to obtain both the value of δ and the (empirical) distributions of y_i and $y_{i,j}$ corresponding to Eq. (4): For each δ within an exhaustive set of values, we first let y_i be a scaled median of all its 5 service times, reflecting the identical content of the replicated page requests. We then generate a set of $y_{i,j}$ according to the selected values of δ and y_i. Secondly, we fit the y_i and $y_{i,j}$ to one of the following statistical distributions: Exponential, Normal, Beta, Inverse Gaussian, Log-normal, Weibull, and Log-logistic. Using the Kolmogorov-Smirnov and Chi-squared tests, we compute the p-values for each fitting, and determine the goodness of fit at the significance level of 5%, against the hypothesis that the y_i, $y_{i,j}$ are from the fitted distribution. We repeat those two steps for every δ and choose the one that maximizes the p-value of the optimal fitting among all δ-values considered. To ease the analysis, we only considered δ such that the difference between $\mathbb{E}[y_i]$ and $\mathbb{E}[y_{i,j}]$ is within a 10% margin.

In Figure 4, the empirical distributions and the corresponding fitted distributions are depicted for y_i and $y_{i,j}$, respectively. The degree of correlation δ with the best fitting for y_i and y_{ij} is $\delta = 0.8$; the resulting means are $\mathbb{E}[y_i] = 87.13$ and $\mathbb{E}[y_{i,j}] = 82.05$, respectively. In terms of the distributions, the best fitting for y_i is the Weibull distribution with scale parameter $\lambda = 85.02$ and shape parameter $k = 0.94$. Although the exponential distribution can also pass both the Kolmogorov-Smirnov and Chi-square tests, the resulting p-value is lower than for the Weibull distribution. The best fitting for $y_{i,j}$ is the Log-logistic distribution with scale parameter



(a) Theorem 1 (b) Theorem 4

(c) Theorem 3 (d) Theorem 5

Fig. 5. Delay bounds from the four main Theorems (in (a): $\lambda = 4 * 0.75$, $\mu = 1$, in (b): $\lambda = 4 * 0.75$, $\delta = 0.5$, $\mu' := \delta k + (1 - \delta)$, in (c): $p = 0.1$, $\lambda_{\text{iact}} = 0.3$, $\lambda_{\text{act}} = 30$, $\mu = 1$, in (d): $p = 0.1$, $\lambda_{\text{iact}} = 0.3$, $\lambda_{\text{act}} = 30$, $\mu = 1$, $\delta = 0.5$, $\mu' := \delta k + (1 - \delta)$); some simulation points are above the bounds in the tail due to the insufficient length of the simulation.

$\alpha = 3.79$ and shape parameter $\beta = 0.61$. Both Weibull and Log-logistic distributions are, with the obtained parameters, heavy-tailed; fitting these models within our framework would require approximations of heavy-tailed distributions with hyperexponential distributions, which are subject to finite MGFs (see, e.g., [14]). Overall, our results show that service times of replicated requests from real systems are dependent (with $\delta = 0.8$) and have long tails (thus further confirming existing measurements [2]), strongly arguing for the correlated service model.

E. Markovian Arrivals, Correlated Replication

We briefly state the results for the combination of the scenario from § III-B and § III-C:

Theorem 5. *With the same notation as in § III-B and § III-C, let $\theta_{mkv,cor}$ be defined by*

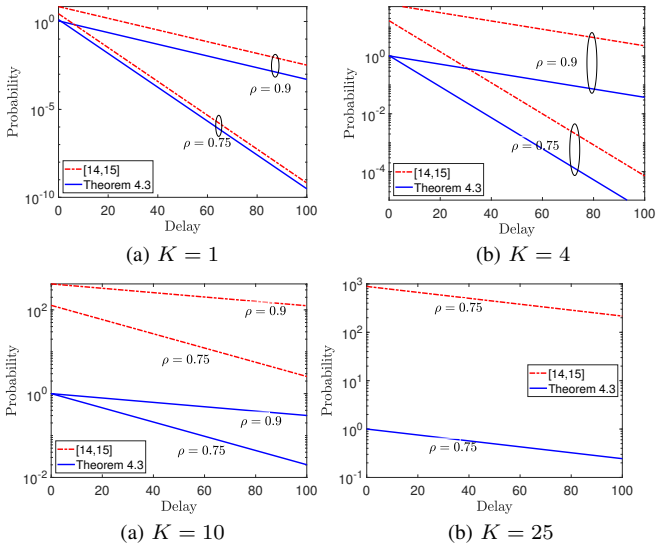
$$\theta_{mkv,cor} := \sup \left\{ \theta \geq 0 \;\middle|\; \mathbb{E}\left[e^{\theta\delta y_1}\right] \mathbb{E}\left[e^{\theta(1-\delta)\min_{j\leq k} y_{1,j}}\right] \xi^{\frac{K}{k}}(\theta) \leq 1 \right\} .$$

Then the following bound on the response time holds for all $\sigma \geq 0$:

$$\mathbb{P}(r \geq \sigma) \leq \mathbb{E}\left[e^{\delta\theta_{mkv,cor} y_1}\right] \mathbb{E}\left[e^{(1-\delta)\theta_{mkv,cor}\min_{j\leq k} y_{1,j}}\right] e^{-\theta_{mkv,cor}\sigma} .$$

Proof. Entirely analoguous to the proofs of Theorems 3-4. □

To check the numerical accuracy of the stochastic bounds from Theorems 1, 3, 4 and 5 we refer to Figure 5. In all scenarios, addressing combinations of correlated arrivals and replications, exponential jobs with parameter μ are replicated



(a) $K = 1$ (b) $K = 4$

(a) $K = 10$ (b) $K = 25$

Fig. 6. Delay bounds (Theorem 3 vs. Theorem 1 from [15], [16]) ($p = 0.1$, $\lambda_{\text{iact}} = 0.3$, $\mu = 1$, λ_{act} is determined from ρ)

to $k = 1, 2, 4$ out of a total number of $K = 4$ servers. The parameters of the respective models are chosen such that the (server) utilization remains constant, i.e., $\rho = 0.75$. We remark that in all scenarios the stochastic bounds are remarkably accurate.

F. Comparison against state-of-the-art

Here we compare our bounds against the state-of-the-art bounds from [15], [16]. We only consider the case of independent replication, considered in [15]. We also consider the case when $k = K$ (i.e., all servers process all replicas) and exponential service times for the replicas with parameter μ.

In the case of independent arrivals, our results from Theorem 1 exactly match those from [15], [16] (see Theorem 1 therein).

In the Markovian case, consider the Markov chain model from § III-B. The bound from [15], [16] (Theorem 1), which is used in [15] for the replication scenario, can be stated as

$$\mathbb{P}(r \geq \sigma) \leq \inf_{\theta} \frac{1}{1 - \xi(\theta)\frac{K\mu}{K\mu+\theta}} e^{-\theta\sigma} ,$$

where $\xi(\theta)$ is the spectral radius from § III-B and the infimum is taken for all values $\theta > 0$ such that $\xi(\theta)\frac{K\mu}{K\mu+\theta} < 1$.

In Fig. 6 we numerically compare our bound from Theorem 3 against the bound from Theorem 1 from [15], [16], for $K = 1, 4, 10, 25$; our bound for $K = 4$ and $\rho = 0.75$ is the same one depicted in Fig. 5.(c). While the two sets of bounds are reasonably similar in the case of a single server $K = 1$, we observe that our bounds improve by at least three orders of magnitude when $K = 25$ at utilization $\rho = 0.75$. Such improvements can become more pronounced for larger numbers of servers (e.g., at least four orders of magnitude when $K = 100$)[4]. Moreover, the bounds from [15], [16] are

[4]The corresponding figure is omitted for brevity.

trivial at high utilization $\rho = 0.9$ as the corresponding values (probabilities) are larger than 1. The numerical looseness of the results from [15], [16] is symptomatic of the underlying stochastic network calculus approach which does not properly capture the statistical correlations within arrival or service processes; for related discussions see [9], [7].

IV. Application: Resource Usage vs. Response Times

We investigate the analytical tradeoff between resource usage and response times under replication. This application is motivated by empirical observations from Google [11] and Bing [25] traces that a slight increase in the resource budget may yield substantial reductions of the upper quantiles of response times. For example, [25] reports that the 99^{th} percentile of the delay improves by as much as 40% under a 5% increase of the resource budget. To compensate for the inherent increase of resource usage under replication, the schemes from [11], [25] defer the execution time of the replicas until the original request has been outstanding for a given *replication offset* Δ. These schemes were recently substantially improved through a sophisticated algorithm which permits replication under some conditions [34].

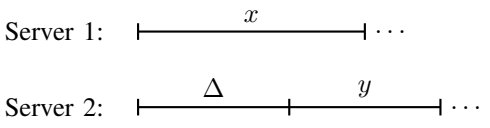


Fig. 7. Replication with deferred execution times: a replica (at Server 2) may start no sooner than ($\Delta \geq 0$) after the starting time of the original (at Server 1).

Consider a scenario with two servers. Jobs arrive with rate λ at the first server with interarrival times t_i and service times $x_i =_{\mathcal{D}} x$; if the processing time of a job is larger than some fixed Δ, then the job is replicated at the second server with service times $y_i =_{\mathcal{D}} y$ (see Figure 7 for a time-line illustration of a generic job with execution time x and its replica, should $x > \Delta$). Whenever either of the original job or its replica finishes execution, the residual service time of the other is cancelled (i.e., the purging replication model).

The utilization at the first server is thus given by

$$\rho_1 = \lambda E\left[\min\{x, \Delta + y\}\right] \ , \tag{5}$$

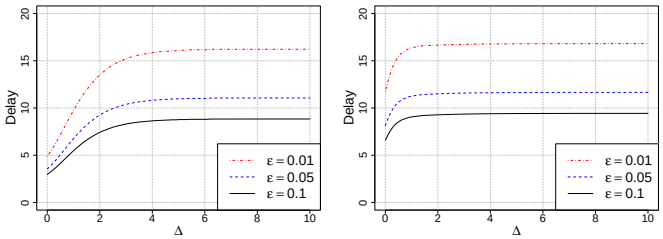
whereas the utilization at the second is

$$\rho_2 = \lambda E\left[\min\left\{|x - \Delta|, y\right\}\right] \ . \tag{6}$$

We note that unlike previous models, where the utilization is server independent, the current model is subject to different server utilizations due to the lack of symmetry in dispatching the load.

The measure for *resource usage* is the total utilization at the two servers and is denoted by u to avoid confusion

$$u := \rho_1 + \rho_2 \ .$$



(a)$\delta = 0.25$ (b) $\delta = 0.75$

Fig. 8. Quantiles of the response time vs. the replication offset Δ ($\lambda = 0.75$, $\mu = 1$); ε is the tail probability of the delay

Aiming for explicit results, we assume for convenience the exponential service model, i.e., $x \sim \exp(\mu)$ and $y \sim \exp(\mu)$, with $\mu = 1$. Moreover, we consider both the independent and correlated replication models.

1) Independent Replication: Given the statistical independence of x_i's and y_i's, straightforward computations of integrals yield

$$\begin{aligned} \rho_1 &= \frac{\lambda}{\mu} - \frac{\lambda}{2\mu} e^{-\mu\Delta} \text{ and} \\ \rho_2 &= \frac{\lambda}{2\mu} e^{-\mu\Delta} \ , \end{aligned}$$

which means that the resource usage $u = \frac{\lambda}{\mu}$ is invariant to the choice of Δ.

In turn, Δ can have a major impact on the response times: for instance, if $\mu < \lambda < 2\mu$ then the response times can be either unbounded for sufficiently large values of Δ, and in particular when $\Delta = \infty$ (i.e., no replicas are executed and the system running on a single server is unstable), or finite for some values of Δ (i.e., replication stabilizes the system).

In fact, an immediate application of Theorem 1 yields that the response time is non-decreasing in Δ. Thus, the optimal choice of Δ, which minimizes both the resource usage and the response times, is $\Delta = 0$. The explanation for the seemingly sharp contrast between this theoretical result and the empirical results from [11], [25] is the underlying independence assumption of the replication model.

2) Correlated Replication: A non-trivial tradeoff between resource usage and response times manifests itself under the more realistic correlated replication model from § III-C. The original and replica response times are modelled by

$$(1-\delta)x + \delta z \text{ and } (1-\delta)y + \delta z \ ,$$

where $x, y,$ and z are exponential with rate $\mu = 1$. The parameter δ sets the degree of correlation; in particular, small values of δ indicate a small degree of correlation.

Rather tedious computations of integrals, due to several conditions stemming from the absolute value operator in ρ_2, yield the individual utilizations

$$\begin{aligned} \rho_1 &= \frac{\lambda}{\mu}\left(1 - \frac{1-\delta}{2} e^{-\frac{\mu}{1-\delta}\Delta}\right) \text{ and} \\ \rho_2 &= \frac{\lambda}{\mu}\left(\frac{\delta^2}{2\delta - 1} e^{-\frac{\mu}{\delta}\Delta} - \frac{1-\delta}{2(2\delta-1)} e^{-\frac{\mu}{1-\delta}\Delta}\right) \ , \end{aligned}$$

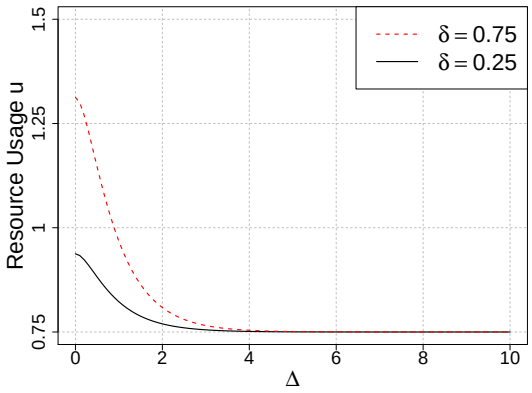


Fig. 9. Resource usage u from Eq. (7) ($\lambda = 0.75$, $\mu = 1$)

and further the resource usage

$$u = \frac{\lambda}{\mu}\left(1 + \frac{\delta^2}{2\delta - 1}e^{-\frac{\mu}{\delta}\Delta} - \frac{\delta(1-\delta)}{2\delta - 1}e^{-\frac{\mu}{1-\delta}\Delta}\right) \qquad (7)$$

under the assumptions that $\delta \in (0,1)$ and $\delta \neq .5$.

To illustrate a quantitative tradeoff between resource usage (Eq. (7)) and response times (Theorem 4), we refer to Figure 8 which shows the increase of the top percentiles of the response times (90^{th}, 95^{th}, and 99^{th}) as a function of the replication offset Δ. Both small ($\delta = 0.25$) and high ($\delta = 0.75$) correlation degrees are considered; in Figure 9, the resource usage u corresponding to Eq. (7) is shown. We observe that under the small correlation degree, a 20% decrease of resource usage from $u = \frac{\lambda}{\mu}(1+\delta) \approx 0.94$ (when $\Delta = 0$) to $u = \frac{\lambda}{\mu} = 0.75$ (when $\Delta = \infty$) yields a dramatic increase of the 99^{th} percentile of the response times of roughly 230%. In turn, under the high correlation degree, the same 20% decrease of resource usage from $u \approx 1.31$ (when $\Delta = 0$) to $u \approx 1.05$ (when $\Delta = 0.8$) yields an increase of the same response time percentile of only roughly 37%. These numerical results, which are clearly dependent on the model's assumptions and numerical values, indicate nevertheless that a drastic reduction of the top percentiles of response times at the expense of a small increase of resource usage [11], [25] is likely due to a low correlation (or almost independence) of the service times. Conversely, if the service times of the replicas are sufficiently correlated, increasing the resource usage only yields a marginal gain in response time reductions.

V. Conclusions

In this paper we have developed an analytical framework to compute stochastic bounds on response time distributions in quite general queueing systems with replication. Unlike existing models, we accounted for three main practical characteristics: non-Poisson inter-arrivals, general service time distributions, and correlated service times for the replicas. By employing a powerful methodology based on martingales, we obtained numerically accurate bounds by exploiting the specific correlation structures of the underlying processes. By further considering a practical case study of systems with replication, we highlighted that the benefits or drawbacks of replication are highly sensitive to the three features altogether.

Appendix

Proof of Theorem 1. Define the process $M(n)$ by

$$M(n+1) := e^{\theta_{\text{ind}}\left(\sum_{i=1}^{n+1}\min_{j\le k}\{x_{i,j}\} - \sum_{i=1}^{n}\tilde{t}_i\right)} .$$

$M(n)$ is a martingale:

$$\begin{aligned}
&\mathbb{E}[M(n+1)M(n)^{-1} \mid M(1),\ldots,M(n)] \\
&= \mathbb{E}\left[e^{\theta_{\text{ind}}\left(\min_{j\le k} x_{n+1,j} - \tilde{t}_n\right)}\right] \\
&= \mathbb{E}\left[e^{\theta_{\text{ind}}\min_{j\le k} x_{n+1,j}}\right]\mathbb{E}\left[e^{-\theta_{\text{ind}} t_n}\right]^{\frac{K}{k}} \\
&= 1 .
\end{aligned}$$

Now define the stopping time N as

$$N := \min\left\{ n \ge 0 \;\middle|\; \sum_{i=1}^{n}\min_{j\le k}\{x_{i,j}\} - \sum_{i=1}^{n-1}\tilde{t}_i \ge \sigma \right\} ,$$

and note that $\{N < \infty\} = \{r \ge \sigma\}$. With the optional stopping theorem for some $l \in \mathbb{N}$:

$$\begin{aligned}
&\mathbb{E}\left[e^{\theta_{\text{ind}}\min_{j\le k} x_{1,j}}\right] = \mathbb{E}\left[M(1)\right] = \mathbb{E}\left[M(N\wedge l)\right] \\
&\ge \mathbb{E}\left[M(N\wedge l)1_{N\le l}\right] \ge e^{\theta_{\text{ind}}\sigma}\mathbb{P}(N \le l)
\end{aligned}$$

Now let $l \to \infty$. □

Proof of Theorem 3. Proceeding similarly as in the proof of Theorem 1, define the process $M(n)$ by

$$M(n) := h_{Z(n\frac{K}{k}-1)}e^{\theta_{\text{mkv}}\left(\sum_{i=1}^{n}\min_{j\le k}\{x_{i,j}\} - \sum_{i=1}^{n-1}\tilde{t}_i\right)} .$$

$M(n)$ is a martingale: By induction over $\frac{K}{k} - 1$ one shows that:

$$\begin{aligned}
&\mathbb{E}\left[e^{-\theta_{\text{mkv}}\tilde{t}_{n+1}} \;\middle|\; Z\left(n\frac{K}{k} - 1\right)\right] \\
&= \left(T_{\theta_{\text{mkv}}}^{\frac{K}{k}}\right)_{Z(n\frac{K}{k}-1),\text{iact}} + \left(T_{\theta_{\text{mkv}}}^{\frac{n}{k}}\right)_{Z(n\frac{K}{k}-1),\text{act}} .
\end{aligned}$$

Now:

$$\begin{aligned}
&\mathbb{E}\left[h_{Z((n+1)\frac{K}{k}-1)}e^{\theta_{\text{mkv}}\left(\min_{j\le k}\{x_{n+1,j}\} - \tilde{t}_n\right)} \;\middle|\; S = \text{act}\right] \\
&= \mathbb{E}\left[e^{\theta_{\text{mkv}}\min_{j\le k}\{x_{n,j}\}}\right]\left(T_{\theta_{\text{mkv}}}^{\frac{K}{k}}h\right)_{\text{act}} \\
&= \mathbb{E}\left[e^{\theta_{\text{mkv}}\min_{j\le k}\{x_{n+1,j}\}}\right]\xi^{\frac{K}{k}}(\theta_{\text{mkv}})h_{\text{act}} \\
&= h_{\text{act}} ,
\end{aligned}$$

where $S := Z\left(n\frac{K}{k} - 1\right)$, and similarly one obtains:

$$\mathbb{E}\left[h_{Z((n+1)\frac{K}{k}-1)}e^{\theta_{\text{mkv}}\left(\min_{j\le k}\{x_{n+1,j}\} - \tilde{t}_n\right)} \;\middle|\; S = \text{iact}\right] = h_{\text{iact}} ,$$

so that:

$$\mathbb{E}\left[h_{Z((n+1)\frac{K}{k}-1)}e^{\theta_{\text{mkv}}\left(\min_{j\le k}\{x_{n+1,j}\} - \tilde{t}_n\right)} \;\middle|\; S\right] = h_{Z(n)} .$$

Now multiply both sides by $e^{\theta_{\text{mkv}}\left(\sum_{i=1}^{n}\min_{j\le k} x_{1,j} - \sum_{i=1}^{n-1} t_i\right)}$. The proof completes along the same kind of lines as in the proof of Theorem 1. □

REFERENCES

[1] I. J. B. F. Adan, G. J. van Houtum, and J. van der Wal. Upper and Lower Bounds for the Waiting Time in the Symmetric Shortest Queue System. *Annals of Operations Research*, 48(2):197–217, Apr. 1994.

[2] G. Ananthanarayanan, A. Ghodsi, S. Shenker, and I. Stoica. Effective straggler mitigation: Attack of the clones. In *10th USENIX Conference on Networked Systems Design and Implementation (NSDI)*, pages 185–198, 2013.

[3] S. Borst, O. Boxma, J. F. Groote, and S. Mauw. Task allocation in a multi-server system. *Journal of Scheduling*, 6(5):423–436, Sept. 2003.

[4] E. Buffet and N. G. Duffield. Exponential upper bounds via martingales for multiplexers with Markovian arrivals. *Journal of Applied Probability*, 31(4):1049–1060, Dec. 1994.

[5] M. C. Calzarossa, L. Massari, and D. Tessera. Workload characterization: A survey revisited. *ACM Comput. Surv.*, 48(3):48:1–48:43, Feb. 2016.

[6] C.-S. Chang. *Performance Guarantees in Communication Networks*. Springer Verlag, 2000.

[7] F. Ciucu and F. Poloczek. Two extensions of Kingman's GI/G/1 bound. *Proc. of the ACM on Measurement and Analysis of Computing Systems - ACM Sigmetrics / IFIP Performance*, 2(3):43:1–43:33, Dec. 2018.

[8] F. Ciucu, F. Poloczek, and A. Rizk. Queue and loss distributions in finite-buffer queues. *Proc. of the ACM on Measurement and Analysis of Computing Systems - ACM Sigmetrics / IFIP Performance*, 3(2):31:1–31:29, June 2019.

[9] F. Ciucu and J. Schmitt. Perspectives on network calculus - No free lunch but still good value. In *ACM Sigcomm*, 2012.

[10] J. Dean. [Online] Achieving rapid response times in large online services. Mar. 2012. Berkeley AMPLab Cloud Seminar, http://research.google.com/people/jeff/latency.html.

[11] J. Dean and L. A. Barroso. The tail at scale. *Communications of the ACM*, 56(2):74–80, Feb. 2013.

[12] J. Dean and S. Ghemawat. Mapreduce: Simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, Jan. 2008.

[13] A. Ephremides, P. Varaiya, and J. Walrand. A simple dynamic routing problem. *IEEE Transactions on Automatic Control*, 25(4):690–693, Aug. 1980.

[14] A. Feldmann and W. Whitt. Fitting mixtures of exponentials to long-tail distributions to analyze network performance models. *Performance Evaluation*, 31(3-4):245–279, Jan. 1998.

[15] M. Fidler and Y. Jiang. Non-asymptotic delay bounds for (k, l) fork-join systems and multi-stage fork-join networks. *CoRR*, abs/1512.08354, 2015.

[16] M. Fidler and Y. Jiang. Non-asymptotic delay bounds for (k, l) fork-join systems and multi-stage fork-join networks. In *IEEE INFOCOM*, 2016.

[17] K. Gardner, M. Harchol-Balter, A. Scheller-Wolf, undefined, undefined, undefined, and undefined. A better model for job redundancy: Decoupling server slowdown and job size. *IEEE MASCOTS*, 2016.

[18] K. Gardner, S. Zbarsky, S. Doroudi, M. Harchol-Balter, and E. Hyytiä. Reducing latency via redundant requests: Exact analysis. In *ACM Sigmetrics*, pages 347–360, 2015.

[19] K. Gardner, S. Zbarsky, M. Harchol-Balter, and A. Scheller-Wolf. The power of d choices for redundancy. In *ACM Sigmetrics*, New York, NY, USA, 2016. ACM.

[20] G. D. Ghare and S. T. Leutenegger. Improving speedup and response times by replicating parallel programs on a snow. In *10th International Conference on Job Scheduling Strategies for Parallel Processing (JSSPP)*, pages 264–287, 2004.

[21] M. Harchol-Balter. *Performance Modeling and Design of Computer Systems. Queueing Theory in Action*. Cambridge University Press, 2012.

[22] E. Heymann, M. Senar, E. Luque, and M. Livny. Evaluation of strategies to reduce the impact of machine reclaim in cycle-stealing environments. In *First IEEE/ACM International Symposium on Cluster Computing and the Grid*, pages 320–328, 2001.

[23] T. Hoff. [Online] Latency is everywhere and it costs you sales - how to crush it. July 2009. http://highscalability.com/blog/2009/7/25/latency-is-everywhere-and-it-costs-you-sales-how-to-crush-it.html.

[24] E. Hyytiä and S. Aalto. Round-robin routing policy: Value functions and mean performance with job- and server-specific costs. In *Proc. of the 7th International Conference on Performance Evaluation Methodologies and Tools (ValueTools)*, pages 69–78, 2013.

[25] V. Jalaparti, P. Bodik, S. Kandula, I. Menache, M. Rybalkin, and C. Yan. Speeding up distributed request-response workflows. In *ACM SIGCOMM 2013*, pages 219–230, 2013.

[26] G. Joshi, Y. Liu, and E. Soljanin. On the delay-storage trade-off in content download from coded distributed storage systems. *IEEE Journal on Selected Areas in Communications (JSAC)*, 32(5):989–997, May 2014.

[27] G. Joshi, E. Soljanin, and G. Wornell. Queues with redundancy: Latency-cost analysis. In *ACM Sigmetrics Workshop on MAthematical performance Modeling and Analysis (MAMA)*, 2015.

[28] G. Joshi, E. Soljanin, and G. W. Wornell. Efficient redundancy techniques for latency reduction in cloud systems. *CoRR*, abs/1508.03599, 2015.

[29] D.-C. Juan, L. Li, H.-K. Peng, D. Marculescu, and C. Faloutsos. *Beyond Poisson: Modeling Inter-Arrival Time of Requests in a Datacenter*, pages 198–209. Springer International Publishing, Cham, 2014.

[30] J. F. C. Kingman. A martingale inequality in the theory of queues. *Cambridge Philosophical Society*, 60(2):359–361, Apr. 1964.

[31] G. Koole and R. Righter. Resource allocation in grid computing. *Journal of Scheduling*, 11(3):163–173, June 2007.

[32] Z. Liu and D. Towsley. Optimality of the round-robin routing policy. *Journal of Applied Probability*, 31(2):466–475, 1994.

[33] U. Lublin and D. G. Feitelson. The workload on parallel supercomputers: Modeling the characteristics of rigid jobs. *J. Parallel Distrib. Comput.*, 63(11):1105–1122, Nov. 2003.

[34] X. Ren, G. Ananthanarayanan, A. Wierman, and M. Yu. Hopper: Decentralized speculation-aware cluster scheduling at scale. *ACM Sigcomm*, pages 379–392, 2015.

[35] F. B. Schneider. Implementing fault-tolerant services using the state machine approach: A tutorial. *ACM Computing Surveys*, 22(4):299–319, Dec. 1990.

[36] E. Schurman and J. Brutlag. The user and business impact of server delays, additional bytes and HTTP chunking in web search. *O'Reilly Velocity Web Performance and Operations Conference*, June 2009.

[37] N. Shah, K. Lee, and K. Ramchandran. When do redundant requests reduce latency ? In *51st Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 731–738, Oct. 2013.

[38] S. Souders. [Online] Velocity and the bottom line. July 2009. http://radar.oreilly.com/2009/07/velocity-making-your-site-fast.html.

[39] C. Stewart, A. Chakrabarti, and R. Griffith. Zoolander: Efficiently meeting very strict, low-latency slos. In *10th International Conference on Autonomic Computing (ICAC 13)*, pages 265–277, 2013.

[40] Y. Sun, C. E. Koksal, and N. B. Shroff. On delay-optimal scheduling in queueing systems with replications. *CoRR*, abs/1603.07322, 2016.

[41] A. Vulimiri, P. B. Godfrey, R. Mittal, J. Sherry, S. Ratnasamy, and S. Shenker. Low latency via redundancy. In *Proc. of the Ninth ACM Conference on Emerging Networking Experiments and Technologies (CoNext)*, pages 283–294, 2013.

[42] J. Walrand. *An introduction to queueing networks*. Prentice Hall, 1988.

[43] D. Wang, G. Joshi, and G. W. Wornell. Using straggler replication to reduce latency in large-scale parallel computing (extended version). *CoRR*, abs/1503.03128, 2015.

[44] Z. Wu, C. Yu, and H. V. Madhyastha. CosTLO: Cost-effective redundancy for lower latency variance on cloud storage services. In *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*, pages 543–557, May 2015.

[45] H. Xu and B. Li. Repflow: Minimizing flow completion times with replicated flows in data centers. In *IEEE Infocom*, pages 1581–1589, 2014.