Delay Comparison of Delivery and Coding Policies in Data Clusters

Virag Shah, Anne Bouillard, François Baccelli

ABSTRACT

A key function of cloud infrastructure is to store and deliver diverse files, e.g., scientific datasets, social network information, videos, etc. In such systems, for the purpose of fast and reliable delivery, files are divided into chunks, replicated or erasure-coded, and disseminated across servers. It is neither known in general how delays scale with the size of a request nor how delays compare under different policies for coding, data dissemination, and delivery.

Motivated by these questions, we develop and explore a set of evolution equations as a unified model which captures the above features. These equations allow for both efficient simulation and mathematical analysis of several delivery policies under general statistical assumptions. In particular, we quantify in what sense a workload aware delivery policy performs better than a workload agnostic policy. Under a dynamic or stochastic setting, the sample path comparison of these policies does not hold in general. The comparison is shown to hold under the weaker increasing convex stochastic ordering, still stronger than the comparison of averages.

This result further allows us to obtain insightful computable performance bounds. For example, we show that in a system where files are divided into chunks of equal size, replicated or erasure-coded, and disseminated across servers at random, the job delays increase sub-logarithmically in the request size for small and medium-sized files but linearly for large files.

1. INTRODUCTION

Modern cloud computing infrastructures feature several clusters each of which consists of thousands of highly interconnected servers which collectively run and serve diverse computing applications [1–3]. An important aspect of these clusters is to collectively store and deliver Internet scale data/files. Key design challenges for such systems include placement of files across servers and an algorithm for the swift delivery of dynamically arriving file requests. A common practice towards file placement is to divide each file into chunks of fixed size, which could then potentially be replicated/coded, and to disseminate them across the servers [4,5]. This can potentially reduce delays in delivering large files since the delivery algorithm could now aggregate the service rate from multiple servers.

To gain intuition, consider some hypothetical scenario where, for the placement of each file, one is allowed to use variable and arbitrarily small (possibly fractional) chunk sizes. Then, for a system of m servers, one could divide each file of size ν bits into m different chunks, each of size ν/m bits. Suppose that the service/delivery rate at each sever is μ bits/sec and that there is no other network bottleneck. Then, the minimum achievable delay in serving a download request for a file of size ν is $\frac{\nu}{m\mu}$, which is possible only if no other request is present in the system.

However, delays which scale inverse linearly with m clearly cannot be achieved for each file if there is a limit to the minimum chunk size. For example, suppose that each file of size ν is divided into $\lceil \frac{\nu}{c} \rceil$ chunks of size c. Then, if there is only one download request in the system at a given time, for any file of size less than cm, a download delay equal to $\frac{c}{\mu}$ can be achieved. Whereas for files several times larger than cm bits, the delay is still of the order of $\frac{\nu}{m\mu}$ under isolation. However, for a system with diverse files and fixed chunk size, it is not directly clear what the delays are under stochastic loads. In such a setting, how do delays relate with the size of a requested file? Do replication of chunks or erasure coding help in reducing delays? What is the impact of dynamic load-balancing? These are some of the questions we address in this work.

Contributions: We provide a stochastic model which encapsulates the key features of the content delivery process in a highly interconnected cluster of servers and allows us to compare several different policies as well as to obtain explicit performance bounds. In particular, our model captures the following aspects:

Dissemination policy: We allow each file to be divided into chunks of a given size. Thus, a larger file is divided into larger number of chunks. These chunks are most often encoded to obtain code blocks of the given size, as explained below. The code blocks are then disseminated across servers in a randomized fashion to ensure that the load across servers is balanced.

Coding policy: Suppose that a file is divided into k chunks. For each $k \geq 1$, these chunks are coded into α_k code blocks for some $\alpha_k \geq k$ via MDS (maximum distance separable) erasure codes [6, 7]. These codes are designed such that the original k chunks can be exactly recovered from any k

out of the α_k code blocks. This allows additional flexibility towards dynamically balancing load across servers as the requests arrive over time, as explained below.

Delivery policy: Upon the arrival of a request for a file with k chunks, a request is sent to a subset of servers to obtain k out α_k associated code blocks. The servers serve the block requests in FCFS fashion. We allow dynamic load balancing policies such as *Water-filling* and *Batch Sampling* policies (defined below) which favor a subset of the set of servers with lower instantaneous loads to balance the server workload as well as to achieve lower request delays.

We propose a comprehensive model for this class of systems, with the potential of representing all such policies under certain diversity and symmetry assumptions on the file sizes and the loading policy. This model consists of a set of evolution equations which allow for both efficient simulation and mathematical analysis under general statistical assumptions. In particular, we are able to show the following:

1. We compare the evolution of workloads under three different delivery policies: namely, water-filling ($\mathcal{WF}$), batch sampling ($\mathcal{BS}$), and a randomized policy called *Balanced Random* ($\mathcal{BR}$). We show that, for a given workload at each server, $\mathcal{WF}$ is optimal in the sense that upon a new arrival, it achieves 'the most balanced' workload as compared to any other policy. Further, $\mathcal{BS}$ is somewhere in between $\mathcal{WF}$ and $\mathcal{BR}$ in this respect.

2. We show that $\mathcal{WF}$ and $\mathcal{BS}$ achieve more favorable workload distributions and lower delay distributions as compared to $\mathcal{BR}$ in the sense of 'increasing convex order', which in turn implies that the former policies achieve better performance not only in expectation but in higher moments as well.

3. We provide an upper bound for the delay in delivering a file as a function of its size, under a scenario where the requests form a mix of diverse file sizes. Our bound reveals the relative impact of the local dynamics at an individual server and that of the global view of server workloads seen by an arrival. We also provide new scaling laws on the behavior of delays under such a scenario.

4. Using simulations we analyze the impact of the key options and parameters, including the delivery policy, the coding options and the chunck size. We identify two fundamental regimes, the logaritmic regime when the file sizes are such that no two chunks are stored on the same server, and the linear regime when files have a number of chunks that exceeds the number of servers. We show that in the logarithmic regime, the gains of dynamic load balancing via $\mathcal{WF}$ and $\mathcal{BS}$ are significant even when the coding rate is small. We also show that our product form bound on the delivery latency is tight when requests have a moderate size.

Related Work: Recently there has been significant interest towards developing scalable performance models and analysis for content delivery systems with low delays. For example, the work in [8–10] exploits server parallelism via "resource pooling", that is multiple servers are allowed to work together as a pooled resource to meet individual download requests. The pools of servers associated with different requests may overlap, so the sharing of server resources across classes is done via a fairness criterion. Under a scenario where the size of resource pools is *limited* (i.e.$o(m)$), it is shown that the gains of resource pooling and load-balancing can be achieved simultaneously.

An alternate approach considered in the literature is to split a download request into multiple parts, for example, into requests for individual chunks, and achieve server parallelism by employing different servers for different parts [11, 12]. Further, sophisticated coding policies are employed to achieve flexibility in server choices [12]. Under the assumption that the number of servers available for each request is *limited*, the works in [11, 12] are able to use mean-field based arguments to study performance as the number of servers m tends to infinity. Several other works also study queuing models under coding based techniques via heuristics or bounds, e.g. [13–16], but these are not scalable for our purposes.

We depart from the above approaches in that we are interested in developing performance models for a regime where we obtain a maximum gain from server parallelism without restricting ourselves to limited resource pools or a limited number of available servers for each request. Given a lower bound on the chunk size, we divide each file into a maximum number of chunks and disseminate them across several servers, potentially $\Omega(m)$ servers for large files. Thus, we allow $\Omega(m)$ servers to take care of a request in parallel. We allow diverse file sizes and provide a delay bound which is a function of the file size.

In terms of tools used, we model the system dynamics via an evolution equation which is a generalization of the Kiefer and Wolfowitz recursion for workloads in $G/G/s$ queues [17], which allows us to go beyond exponentiality assumptions for file-size requests. We use coupling arguments to compare different policies. Coupling has been used to compare several queueing systems in past, e.g., see for example [10, 18]. Further, to provide explicit bounds on delays, we use the notion of association of random variables, which is a property that has had several applications in queueing systems and beyond [17, 19].

Organization: In Section 2 we provide our system model and develop the evolution equations. In Section 3 we provide results comparing various dynamic load balancing policies via coupling arguments. In Section 4 we give performance bounds based on the notion of association of random variables. In Section 5 we consider a scenario where the chunk size may be different for different files. In Section 6 we provide simulation results and numerical evaluations. We conclude in Section 7. Some proofs of technical nature are provided in the Appendix.

2. SYSTEM MODEL

We consider a system with m servers, indexed $1, 2, \ldots, m$. The system consists of a very large number (several orders of magnitude larger than m) of diverse files. We assume that the size of each file is an integer multiple of c bits. Each file is divided into chunks of size c bits each. These chunks are encoded before being placed across servers, as explained below.

For each positive integer k we use an MDS erasure code of rate k/α_k, where α_k is an integer greater than or equal to k. Such a code is called (α_k, k) MDS code in coding theory [6]. Thus, equivalently, each file of size kc is divided into k chunks and encoded into α_k code blocks of size c bits each. The MDS erasure codes may serve various practical purposes. Only the following property is relevant for our purposes: for a file of size kc, it is possible to recover the entire file by downloading any k out of the α_k code blocks.

For each file of size kc, the associated α_k code blocks are placed across servers as follows. If $\alpha_k < m$, then we choose α_k among the m servers uniformly at random and place a distinct code block across each of these servers. Else, we place $\left\lfloor \frac{\alpha_k}{m} \right\rfloor$ distinct blocks on each server and for the remaining $\alpha_k - m \left\lfloor \frac{\alpha_k}{m} \right\rfloor$ blocks we choose that many servers uniformly at random.

We assume that the blocks are placed across servers as described above at time $t = -1$. The placement of blocks is kept fixed since then. From time $t = 0$, file download requests arrive as per an independent Poisson point process Π with rate λ. Let $\{t_0, t_1, \ldots\}$ be the points of Π.

Consider a probability mass function $\pi = (\pi_k : k \in \mathbb{Z}_+)$. Each request arrival corresponds to a file of size ck bits with probability π_k independently of all other arrivals. Let κ_n be the number of chunks for the file requested at time t_n. Thus, $\{\kappa_n\}_0^\infty$ is a sequence of discrete i.i.d. random variable with p.m.f. π.

For each n, let $a_n \in \mathbb{Z}_+^m$ represent the placement of the file requested at time t_n, in the following sense: for each server i the entry a_n^i represents the number of coded blocks placed on server i that correspond to the file requested upon the n^{th} arrival. Thus, for each $n, k \in \mathbb{Z}_+$, if $\kappa_n = k$, then $|a_n| = \alpha_k$. We call $\{a_n\}_0^\infty$ the sequence of placement vectors.

Let $\nu = c \sum_{k=0}^\infty k\pi_k$ denote the mean file-size in bits. Let $\rho = \lambda\nu/m$ denote the per server load in bits/sec.

Assumption 1 (Symmetry in load across servers). *Due to the randomized placement of blocks, for a very large number of files, the load across servers is approximately symmetric. Thus we model the symmetry in load via symmetry in request arrivals as follows: given $\kappa_n = k$, a_n is chosen uniformly at random from each of its feasible realizations. Equivalently, given $\kappa_n = k$, the entry a_n^i is equal to $\left\lfloor \frac{\alpha_k}{m} \right\rfloor + 1$ for $\alpha_k - m \left\lfloor \frac{\alpha_k}{m} \right\rfloor$ servers chosen uniformly at random and it is equal to $\left\lfloor \frac{\alpha_k}{m} \right\rfloor$ for the rest of the servers.*

Making such a symmetry assumption to obtain insightful results is a common practice, see e.g. [8, 11, 12]. While, in general, a system with a finite number of files may not be symmetric, we believe that this is a good approximation especially when the number of files is an order of magnitude larger than the number of servers.

We will not discuss server memory capacity issues here as this is not needed. Note however that such a randomized placement results into concentration of memory usage at each server.

Delivery policy: Upon each arrival, we load servers with requests for coded blocks via a delivery/routing policy as described below. Each server serves its block requests in FCFS fashion at rate μ, i.e., it delivers a code block at the rate of μ bits per second. Recall, due to our use of MDS codes, if $\kappa_n = k$ then the system only needs to deliver k out of the α_k associated blocks for the n^{th} arrival. We let s_n denote the $\mathbb{Z}_+^m$ valued random variable where s_n^i is the number of blocks requested from server i upon the n^{th} arrival. Thus, we have $|s_n| = \kappa_n$ and $s_n \leq a_n$ for each n. We call $\{s_n\}$ the sequence of *routing vectors*. Following are some of the admissible routing policies, each resulting into possibly different sequences of routing vectors.

Balanced Random Policy ($\mathcal{BR}$): For each n, k, if $\kappa_n = k$, then request $\left\lfloor \frac{k}{m} \right\rfloor$ blocks from each server and, for the remaining $k - m \left\lfloor \frac{k}{m} \right\rfloor$ blocks, choose the same number of servers at random from the remaining $\min\left(\alpha_k - m \left\lfloor \frac{k}{m} \right\rfloor, m\right)$ servers having an additional block. More formally, suppose $\kappa_n = k$. Let $k' = k - m \left\lfloor \frac{k}{m} \right\rfloor$ and $a'_n = a_n - \left\lfloor \frac{k}{m} \right\rfloor \mathbf{1}$. From the set $\{i : {a'}_n^i > 0\}$ choose a subset of size k' at random. Let s_n^i be equal to $\left\lfloor \frac{k}{m} \right\rfloor + 1$ for each i in this subset and $\left\lfloor \frac{k}{m} \right\rfloor$ for others.

The following two policies take a routing decision upon the n^{th} arrival based on the instantaneous workloads at different servers at time t_n^-.

Batch Sampling Policy ($\mathcal{BS}$): This is a workload dependent policy. The workload at a server at any given time is the number of bits requested from the server and which are not yet served. Of the required k blocks, request $\left\lfloor \frac{k}{m} \right\rfloor$ blocks from each server and for the remaining $k' = k - m \left\lfloor \frac{k}{m} \right\rfloor$ blocks, choose the k' servers with least instantaneous workload from the remaining $\min\left(\alpha_k - m \left\lfloor \frac{k}{m} \right\rfloor, m\right)$ servers having an additional block. More formally, suppose that the workload at the servers at time t_n^- is $w = (w^i : i = 1, \ldots, m)$ and that $\kappa_n = k$. Let $k' = k - m \left\lfloor \frac{k}{m} \right\rfloor$ and $a'_n = a_n - \left\lfloor \frac{k}{m} \right\rfloor \mathbf{1}$. Let $i_1, i_2, \ldots, i_{k'}$ be given recursively as follows: let $i_1 = \arg\min_{i : {a'}_n^i > 0} w^i$, and for $l = 2, \ldots, k'$ let $i_l = \arg\min_{i : {a'}_n^i > 0, i \neq i_1, \ldots, i_{l-1}} w^i$. Then, we have $s_n^i = \left\lfloor \frac{k}{m} \right\rfloor + 1$ for each $i \in \{i_1, i_2, \ldots, i_{k'}\}$ and $s_n^i = \left\lfloor \frac{k}{m} \right\rfloor$ for $i \notin \{i_1, i_2, \ldots, i_{k'}\}$.

Water-filling Policy ($\mathcal{WF}$): This is also a workload dependent policy. If $\kappa_n = k$, then at time t_n, we take a routing decision for k block requests defined sequentially as follows. Among the servers which store at least one of the α_k blocks for the associated file, choose the server with minimum workload. If there are multiple such servers, choose one at random. Request a block from this server and update its workload, i.e., add c to its existing value. We now have to choose $k - 1$ blocks among the $\alpha_k - 1$ remaining code blocks, for which we repeat the above procedure, see Fig. 1.

More formally, suppose that the workload at the servers at time t_n^- is w and that $\kappa_n = k$. Then, let $j_1, j_2, \ldots, j_k$ be recursively given as follows: $j_1 = \arg\min_{i : a_n^i > 0} w^i$, and for $l = 2, \ldots, k$ let

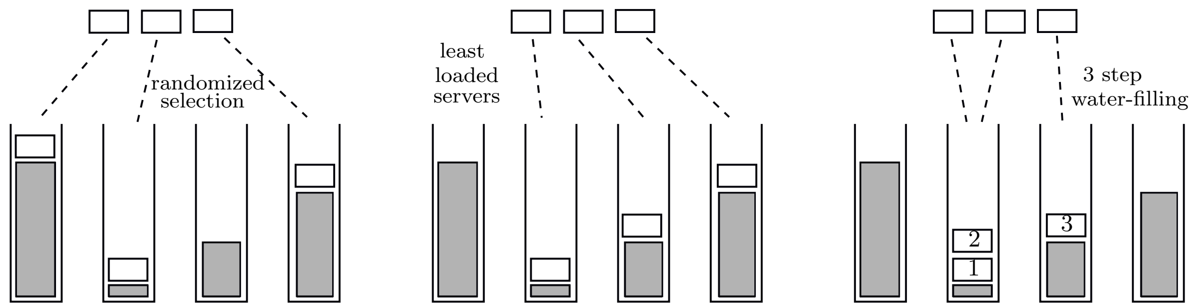
$$j_l = \underset{i : a_n^i - \sum_{l'=1}^{l-1} 1_{\{i = j_{l'}\}} > 0}{\arg\min} w^i + c \sum_{l'=1}^{l-1} 1_{\{i = j_{l'}\}}.$$

For $i = 1, \ldots, m$, let e_i represent the vector in $\mathbb{R}^m$ with i^{th} entry equal to 1 and other entries equal to 0. Then, under the $\mathcal{WS}$ policy we have $s_n = \sum_{l=1}^m e_{j_l}$.

One would guess that $\mathcal{WF}$ is the most egalitarian policy, i.e., it attempts at spreading the arriving load to servers with lower instantaneous workloads, and $\mathcal{BS}$ is somewhere in between $\mathcal{WF}$ and $\mathcal{BR}$ in egalitarianism. We will corroborate these intuitions in the next section.

Note that we do not allow policies which depend explicitly on the server indices. More concretely, if server indices are permuted at time $t = 0^-$, the choice of servers upon each arrival is permuted in the corresponding fashion.

Recall that the routing vector s_n for each n is such that $|s_n|$ is chosen independently with distribution $(\pi_k : k \in \mathbb{N})$, while its entries depend on the workload at the servers at time t_n^- and on the delivery policy. Due to symmetry in



(a) Balanced Random (b) Batch Sampling (c) Water-filling

Figure 1: Illustration of different dynamic delivery policies upon the n^{th} arrival; $m = 4$, $k = 3$, $\alpha_k = 5$, $a_n = (1, 2, 1, 1)$.

file placement (modeled via symmetry in request arrivals) and the above mentioned restriction on the delivery policies, we have that $\{s_n\}_0^\infty$ are exchangeable random vectors in the following sense: upon permutation of server indices the distribution of the sequence $\{s_n\}$ remains unchanged.

Let $\{\tau_n\}_0^\infty$ be inter-arrival times, i.e., $\tau_n = t_{n+1} - t_n$ for each n. Let $\{W_n\}_0^\infty$ be a sequence of $\mathbb{R}_+^m$ valued random variables representing the workload seen by n^{th} arrival, i.e., the workload at different servers at time $t = t_n^-$. Then we have $W_0 = \mathbf{0}$ and

$$W_{n+1} = (W_n + cs_n - \mu\tau_n\mathbf{1})^+, \quad n = 0, 1, \ldots \tag{1}$$

where $\mathbf{1} = (1, 1, \ldots, 1)$, and

$$(x^1, \ldots, x^m)^+ = (\max(x^1, 0), \ldots, \max(x^m, 0)).$$

The *delay* of the n^{th} request is then:

$$D_n = \max_{i:s_n^i > 0} W_n^i + cs_n^i, \quad n = 0, 1, \ldots$$

As mentioned earlier, we are mainly interested in the case where c is a constant since we want to obtain a maximum gain from server parallelism. However, one can envisage a scenario where different requests/files use different chunk sizes. This can be incorporated in our model as follows: we have $W_0 = \mathbf{0}$ and

$$W_{n+1} = (W_n + c_n s_n - \mu\tau_n\mathbf{1})^+, \quad n = 0, 1, \ldots, \tag{2}$$

where the random variables $\{c_n\}_0^\infty$ are $\mathbb{R}_+$ valued and i.i.d.. Note that in this extension, for a given file, the chunks are still of equal sizes. For most parts of the paper, we will use recursion (1). We will nevertheless discuss and analyze (2) in Section 5.

3. COMPARISON OF DELIVERY POLICIES

In this section we compare the server workloads and the request delays under different delivery policies. We use coupling arguments to compare systems adopting different delivery policies. In particular, we couple the request arrival process as well as the sequence of routing vectors in each system. We then study and compare the evolution of server workloads $\{W_n\}_0^\infty$ in the respective systems.

For comparing the workloads of different systems, we use stochastic submajorization and stochastic dominance in the increasing convex order sense, which are briefly introduced in the first subsection. While the former is more amenable to compare the loading under different policies subject to a given initial condition, the later allows us to propagate the comparison result and also to compare delays (recall that the delay of a request is the max of the delays in downloading individual blocks).

3.1 Order statistics and stochastic orders

The notation and concepts listed below are borrowed from [20] and [19].

For all vectors $z \in \mathbb{R}^m$, let $z^{(1)}, z^{(2)}, \ldots, z^{(m)}$ represent its entries in increasing order.

We say that a function $\phi : \mathbb{R}^m \to \mathbb{R}$ is symmetric if for all $x \in \mathbb{R}^m$ and its permutation $x' \in \mathbb{R}^m$, we have $\phi(x) = \phi(x')$.

For two vectors $x, y \in \mathbb{R}^m$, we say that x is majorized by y, which is denoted by $x \prec y$, if $\sum_{i=1}^m x^i = \sum_{i=1}^m y^i$ and $\sum_{i=1}^l x^{(i)} \geq \sum_{i=1}^l y^{(i)}$ for $l = 1, 2, \ldots, m-1$. Intuitively, if $x \prec y$, then x is 'more balanced' than y. For example, in $\mathbb{R}^m$, we have $(1, 1, \ldots, 1) \prec (\frac{m}{2}, \frac{m}{2}, 0, \ldots, 0) \prec (m, 0, 0, \ldots, 0)$.

We say that x is submajorized by y, which is denoted by $x \prec_s y$, if $\sum_{i=l}^m x^{(i)} \leq \sum_{i=l}^m y^{(i)}$ for $l = 0, 1, 2, \ldots, m-1$.

We say that a function $\phi : \mathbb{R}^m \to \mathbb{R}$ is Schur-convex if, for all x and y such that $x \prec y$, we have $\phi(x) \leq \phi(y)$. One can check that a function ϕ is Schur-convex and increasing if and only if (iff), for all x and y such that $x \prec_s y$, we have $\phi(x) \leq \phi(y)$. Further, Schur-convex functions are symmetric since the property $x \prec_s y$ depends only on the ordered entries of x and y.

Consider two random vectors X and Y. We say that X is stochastically dominated by Y, which is denoted by $X \leq^{st} Y$, if, for all increasing functions g, we have $E[g(X)] \leq E[g(Y)]$. A classical result (Strassen's theorem) states that $X \leq^{st} Y$ iff there exist random vectors $\tilde{X}$ and $\tilde{Y}$ such that X and $\tilde{X}$ are identically distributed, Y and $\tilde{Y}$ are identically distributed, and $\tilde{X} \leq \tilde{Y}$ w.p. 1.

For two random vectors X and Y we say that X is stochastically submajorized by Y, which is denoted by $X \prec^{st} Y$ if, for all Schur-convex functions ϕ, we have $E[\phi(X)] \leq E[\phi(Y)]$. We have $X \prec^{st} Y$ iff there exist random vectors $\tilde{X}$ and $\tilde{Y}$ such that X and $\tilde{X}$ are identically distributed, Y and $\tilde{Y}$ are identically distributed, and $\tilde{X} \prec^{st} \tilde{Y}$ w.p. 1.

Similarly, for the random vectors X and Y we say that X is stochastically submajorized by Y, which is denoted by $X \prec_s^{st} Y$, if, for all increasing Schur-convex functions ϕ, we have $E[\phi(X)] \leq E[\phi(Y)]$. Again, $X \prec_s^{st} Y$ iff there exist random vectors $\tilde{X}$ and $\tilde{Y}$ such that X and $\tilde{X}$ are identically distributed, Y and $\tilde{Y}$ are identically distributed, and $\tilde{X} \prec_s^{st} \tilde{Y}$ w.p. 1.

For the random vectors X and Y, we say that X is stochastically dominated by Y in the increasing convex order sense, which is denoted by $X \leq^{icx} Y$, if, for all increasing convex functions g, we have $E[g(X)] \leq E[g(Y)]$.

For $i = 1, \ldots, m$, let e_i denote the vector in $\mathbb{R}^m$ with i^{th} entry equal to 1 and other entries equal to 0. For any vector x we let $|x|$ represent the sum of the absolute values of its entries.

The following lemma is proved in the Appendix.

Lemma 1. *Consider $\mathbb{R}^m$ valued exchangeable random variable X and Y. If $X \prec_s^{st} Y$ then we have $X \leq^{icx} Y$.*

3.2 Comparison of Policies

A delivery policy can be seen as a form of load balancing. Intuitively, a more egalitarian load balancing should achieve more balanced overall workloads. For instance, recall the policies $\mathcal{WF}$, $\mathcal{BS}$, and $\mathcal{BR}$ defined in the System Model. The following theorem says that, given a workload vector W, $\mathcal{WF}$ is the most egalitarian policy while $\mathcal{BS}$ is somewhere in between $\mathcal{WF}$ and $\mathcal{BR}$. For a proof, see the Appendix.

Theorem 1. *Suppose an arrival into the system sees the workload W, where W is an $\mathbb{R}^m$ valued random variable. Let $s^{\mathcal{WF}}$, $s^{\mathcal{BS}}$, $s^{\mathcal{BR}}$, and s' be the routing vectors associated with $\mathcal{WF}$, $\mathcal{BS}$, $\mathcal{BR}$, and an arbitrary routing policy, respectively. Then, the following holds.*

$$W + cs^{\mathcal{WF}} \prec^{st} W + cs' \quad \text{and} \quad W + cs^{\mathcal{BS}} \prec^{st} W + cs^{\mathcal{BR}}.$$

Further, if W is an exchangeable random vector, then we have

$$W + cs^{\mathcal{WF}} \leq^{icx} W + cs^{\mathcal{BS}} \leq^{icx} W + cs^{\mathcal{BR}}.$$

Thus, for a given workload at n, a system under $\mathcal{WF}$ or $\mathcal{BS}$ achieves a more balanced workload in the $\prec_s$ sense at $n+1$ as compared to $\mathcal{BR}$. However, the resulting workloads might be different. Starting with $W_0 = \mathbf{0}$, to be able to claim that an ordering holds for each n, one needs to argue that it propagates. For this we additionally need the monotonicity property of $\mathcal{BR}$ given in the lemma below. For a proof, see the Appendix.

Lemma 2. *Consider random vectors W and W' such that $W \leq^{icx} W'$. Let s and s' be the routing vectors as per the $\mathcal{BR}$ policy for W and W' respectively. Then, $W + cs \leq^{icx} W' + cs'$.*

The following theorem establishes that the $\mathcal{WF}$ and $\mathcal{BS}$ policies achieve 'more balanced and lower' workloads across servers as compared to $\mathcal{BR}$ in a strong sense. For a proof, see the Appendix.

Theorem 2. *Consider a system which starts empty. The workload under policies $\mathcal{WF}$, $\mathcal{BS}$, and $\mathcal{BR}$ satisfy the following:*

$$W_n^{\mathcal{WF}} \leq^{icx} W_n^{\mathcal{BR}} \text{ and } W_n^{\mathcal{BS}} \leq^{icx} W_n^{\mathcal{BR}} \text{ for } n = 0, 1, \ldots$$

Proof. We show the comparison result for a system with $\mathcal{BS}$ and a system with $\mathcal{BR}$; the argument for comparison for $\mathcal{WF}$ and $\mathcal{BR}$ is analogous.

Suppose that the two systems are fed with arrivals as given by the same point process Π. Thus, the sequence of interarrival times $\{\tau_n\}_0^\infty$ is the same for both systems.

For ease of notation let W_n, s_n represent the vectors associated with $\mathcal{BS}$ with their usual meaning, and let W'_n, s'_n represent those associated with $\mathcal{BR}$. $W_0 \leq^{icx} W'_0$ holds trivially since both systems start empty. Now suppose that $W_n \leq^{icx} W'_n$ for a given n. We show below that this implies $W_{n+1} \leq^{icx} W'_{n+1}$.

From Theorem 1 we have that $W_n + cs_n \leq^{icx} W_n + cs'_n$. Further, by Lemma 2 we have $W_n + cs'_n \leq^{icx} W'_n + cs'_n$. Thus, we have $W_n + cs_n \leq^{icx} W'_n + cs'_n$. Since $\mu\tau_n\mathbf{1}$ has equal entries and $\max(.,0)$ is an increasing and convex operation, we have $(W_n + cs_n - \mu\tau_n\mathbf{1})^+ \leq^{icx} (W'_n + cs'_n - \mu\tau_n\mathbf{1})^+$, i.e., $W_{n+1} \leq^{icx} W'_{n+1}$. Hence the result holds. □

The above theorem implies, for example, that each raw moment of the workload at given server under $\mathcal{WF}$ and $\mathcal{BS}$ is less than or equal to that under $\mathcal{BR}$. Similarly, each raw moment of the total workload in the system is lower or equal under $\mathcal{WF}$ and $\mathcal{BS}$ as compared to that under $\mathcal{BR}$.

However, the above theorem does not directly allow us to compare the delays of requests for each n. To see this, recall that delay seen by a request is the max of the delays in downloading individual blocks, which are random in number. Further, a more unbalanced workload W'_n may have more empty servers than W_n. The next arrival could, for example, have the associated blocks stored on the servers which are empty in W'_n and not in W_n.

The following theorem compares delays of requests under both the policies.

Theorem 3. *Consider a system which starts empty. The delays seen by requests under the $\mathcal{WF}$, $\mathcal{BS}$, and $\mathcal{BR}$ policies satisfy the following:*

$$D_n^{\mathcal{WF}} \leq^{icx} D_n^{\mathcal{BR}} \text{ and } D_n^{\mathcal{BS}} \leq^{icx} D_n^{\mathcal{BR}}, \quad n = 0, 1, \ldots$$

Proof. We show this for $\mathcal{BS}$; the argument for $\mathcal{WF}$ is analogous.

For ease of notation, we will use the notation W_n, s_n for random vectors associated with policy $\mathcal{BS}$ with their usual meaning, and W'_n, s'_n for those associated with policy $\mathcal{BR}$.

For a given $\mathbb{R}_+^m$ valued vector r, the function $\max_{i:r_i>0}(x^i + r^i)$ is increasing and convex in $x \in \mathbb{R}^m$. Thus, for any increasing convex function $g : \mathbb{R} \to \mathbb{R}$, $g\left(\max_{i:r_i>0}(x^i + r^i)\right)$ is an increasing convex function in x. Thus, from Theorem 2 we have

$$E_{s'_n} g\left(\max_{i:s_n'^i>0}(W_n^i + cs_n'^i)\right) \leq E_{s'_n} g\left(\max_{i:s_n'^i>0}(W_n'^i + cs_n'^i)\right),$$

where $E_{s'_n}$ denotes the conditional expectaion given s'_n. Note that on both sides of the above inequality we are conditioning on s'_n which is the routing vector associated with $\mathcal{BR}$.

Recall that under the $\mathcal{BR}$ policy, s_n is independent of the instantaneous workload W_n for each n. Using the coupling $\kappa_n = \kappa'_n$ and $a_n = a'_n$, and the definition, given instantaneous workload W_n one can additionally couple the routing vectors s_n and s'_n and the associated κ_n block requests under $\mathcal{BS}$ and $\mathcal{BR}$ policies such that the workload seen by the l^{th} block in front of it under $\mathcal{BS}$ is lower than that under $\mathcal{BR}$ for each $l \leq \kappa_n$ under W_n. Thus, we get

$$E_{s'_n} g\left(\max_{i:s_n^i>0}(W_n^i + cs_n^i)\right) \leq E_{s'_n} g\left(\max_{i:s_n'^i>0}(W_n^i + cs'^i_n)\right).$$

By combining the previous two inequalities we get

$$E_{s'_n} g\left(\max_{i:s_n^i>0}(W_n^i + cs_n^i)\right) \leq E_{s'_n} g\left(\max_{i:s_n'^i>0}(W'^i_n + cs'^i_n)\right),$$

from which the result follows by taking expectation on both sides. $\square$

Recall that $\rho = \lambda\nu/m = c\lambda\frac{\sum_k k\pi_k}{m}$ is the load factor per server. The overall system load is ρm. By exchangeability, the marginal dynamics of the workload at a given server under $\mathcal{BR}$ can be modeled via an $M/GI/1$ FCFS queueing system with load ρ bits/sec and service rate μ bits/sec. Since the number of servers m is finite, the system is stable (asymptotically stationary) if $\rho < \mu$. From Theorem 2 and the ergodicity of the arrival process, it follows that the system is stable under $\mathcal{WF}$ and $\mathcal{BS}$ as well if $\rho < \mu$.

Note that, for general α_k, the delays under the $\mathcal{BR}$ policy are statistically equivalent to the delays obtained when $\alpha_k = k$ for each k, i.e., when the code rate is equal to 1. There are prior works which study gains of erasure-coding via simulations [13,14], experiments [15,16], and analytically but under mean-filed type asymptotic approximations and under exponential service time assumptions [12]. To the best of our knowledge, Theorem 3 is the first rigorous analytical result which compares delays for finite systems employing erasure codes with different code rates. Further, we would like to stress that the result holds under general statistical assumptions for service requirements.

4. ASSOCIATION AND DELAY BOUNDS

In this section, we use the notion of association of random variables to obtain computable bounds on the delays of requests.

Definition 1. *The random variables $X_1, X_2, \ldots, X_k$ are* associated *if, with notation $X = (X_1, X_2, \ldots, X_k)$, the inequality*

$$E[f(X)g(X)] \geq E[f(X)]E[g(X)]$$

holds for each pair of increasing functions $f, g : \mathbb{R}^k \to R$ for which $E[f(X)]$, $E[g(X)]$, and $E[f(X)g(X)]$ exist.

We say that a random vector X is associated if its entries are associated. Similarly, we say that a set of random variables is associated if its elements are associated.

To understand the power of association, consider the following definition and subsequent proposition.

Definition 2. *Consider random variables $X_1, X_2, \ldots, X_k$. We say that $\tilde{X}_1, \tilde{X}_2, \ldots, \tilde{X}_k$ are independent versions of the random variables $X_1, \ldots, X_k$ if the $\tilde{X}_1, \ldots, \tilde{X}_k$ are mutually independent, and if X_i and $\tilde{X}_i$ are identically distributed for $1 \leq i \leq k$.*

Proposition 1 (see [17] Chap 4.3). *Suppose that random variables $X_1, \ldots, X_k$ are associated and that $\tilde{X}_1, \ldots, \tilde{X}_k$ are their independent versions. Then the following holds:*

$$\max_{1\leq i\leq k} X_i \leq^{st} \max_{1\leq i\leq k} \tilde{X}_i.$$

Now consider m different queues with dependent workloads, as in the previous section. If we can show that the arrival of a request sees associated workloads, then we can bound its delay by using the independent version of the workloads. Several works in the literature for large-scale systems, e.g. [12,21], consider the marginal distribution at a given server and study its properties by assuming that the dynamic at any other server is independent of that under the given server; an assumption which is justified in these works as a 'mean-field approximation'. In [21], the queue associated with a given server is called a 'queue at the cavity'. With the association property, can analyze a system without resorting to the mean-field approximation.

Recall that under the $\mathcal{BR}$ policy, the selection of servers s_n is independent of the workload W_n. Upon an arrival, a server gets no additional workload with probability $1 - \sum_{k=1}^{m} \frac{k}{m}\pi_k - \sum_{k=m+1}^{\infty} \pi_k$, and gets workload which is a multiple of c otherwise. One can show that, given that the request is of size kc, the server gets the load $c\left(\left\lfloor \frac{k}{m}\right\rfloor + 1\right)$ with probability $\frac{k}{m} - \left\lfloor \frac{k}{m}\right\rfloor$ and the load $c\left\lfloor \frac{k}{m}\right\rfloor$ with probability $1 - \frac{k}{m} + \left\lfloor \frac{k}{m}\right\rfloor$. Thus, for $i = 1, \ldots, m$, the workload process at i^{th} server, namely $\{W_n^i\}_{n=0}^{\infty}$, in isolation is stochastically equivalent to workload seen by arrivals in a *Cavity Queue* which is as defined below.

Definition 3. *A* Cavity Queue *is an $M/GI/1$ FCFS queue which starts empty at time $t = 0$, has Poisson arrivals with rate λm, service rate μ bits/sec, and service requirement in bits with probability mass function on set $\{0, c, 2c, \ldots\}$ given as follows:*

$$\tilde{\pi}(0) = 1 - \sum_{k=1}^{m} \frac{k}{m}\pi_k - \sum_{k=m+1}^{\infty} \pi_k,$$

and for $l = 1, 2, \ldots$

$$\tilde{\pi}(lc) = \sum_{k=(l-1)m+1}^{lm} \left(\frac{k}{m} - l + 1\right)\pi_k + \sum_{k=lm+1}^{(l+1)m-1} \left(1 - \frac{k}{m} + l\right)\pi_k.$$

The $M/GI/1$ FCFS queues are well studied in the literature. In particular, the following lemma well-known as Pollaczek-Khinchine formula describes the steady state workload distribution of jobs in these queues. Below, we view service time of a job as the ratio of its service requirement in bits and the service rate of the server in bits/sec.

Lemma 3 ([22]). *Consider an $M/GI/1$ FCFS queue with arrival rate $\tilde{\lambda}$. Let σ be a random variable with distribution equal to that of the service times of jobs. Let $\psi_\sigma(s) = E[e^{-s\sigma}]$. Suppose that $\tilde{\lambda}E[\sigma] < 1$. In steady state the workload W has Laplace Transform $\mathcal{G}(.)$ (i.e., $\mathcal{G}(s) = E[e^{-sW}]$) which can be given as:*

$$\mathcal{G}(s) = \frac{(1 - \tilde{\lambda}E[\sigma])s}{s - \tilde{\lambda}\left(1 - \psi_\sigma(s)\right)}. \tag{3}$$

Below, we use (3) to obtain performance bounds on the systems of our interest by using association property along with Proposition 1. The following subset of the many known properties of association can come handy in proving association of random variables (RVs).

Proposition 2 (see [17] Chap 4.3). *The following statements hold.*

(i) The set consisting of a single RV is associated.

(ii) The union of independent sets of associated RVs forms a set of associated RVs

(iii) Any subset of a set of associated RVs forms a set of associated RVs

(iv) *For a non-decreasing function $\phi : \mathbb{R}^m \to \mathbb{R}$ and associated RVs $\{X_1, \ldots, X_m\}$, the random variables*

$$\{\phi(X_1, \ldots, X_m), X_1, \ldots, X_m\}$$

are associated.

Before providing our main results for this section, we need the following additional notation.

Definition 4. *For each $k, n \in \mathbb{Z}_k$, let D_n^k denote the delay seen by the n^{th} arrival given that the size of the requested file is kc bits, that is,*

$$Pr\left(D_n^k \le t\right) = Pr\left(\max_{i:s_n^i>0} W_n^i + cs_n^i \le t \Big| \kappa_n = k\right), \quad t \in \mathbb{R} \text{ and } k, n \in \mathbb{Z}_+. \tag{4}$$

Recall that for each $k, n \in \mathbb{Z}_+$, the n^{th} request for a file is of size kc bits with probability π_k and the k requests for coded blocks are routed to different servers upon the n^{th} arrival as per the chosen policy.

Definition 5. *Let $\Theta(m)$ be the class of probability mass functions $\{\pi\}$ such that for each $\pi = (\pi_k : k \in \mathbb{Z}_+)$ in class Θ, a system with m servers operating under $\mathcal{BR}$ policy has the routing vector s_n which is associated for each n.*

In this paper we will be content to note that $\Theta(m)$ is a rich class of p.m.f.s which includes *Binomial*(p, m) distribution as well as *Geometric*(p) distribution for each $p \in [0, 1]$.

The following theorem, proved in the Appendix, says that for any π in $\Theta(m)$, we get an upper bound on the delay seen by the n^{th} arrival by pretending that the workloads at the m servers 'evolved independently in the past'.

Theorem 4. *Consider a system with m servers which starts empty. For each $k \in \mathbb{Z}_+$, requests for files of size kc bits, equivalently batch requests for k blocks of c bits each, arrive as per an independent point process with rate $\pi_k \lambda m$ and are routed to different servers upon arrival. Servers serve the block requests in FCFS fashion at rate μ bits per second.*

Suppose that $\pi = (\pi_k : k \in \mathbb{Z}_+)$ belongs to class $\Theta(m)$ (see Definition 5). Then the following statements hold:

1. *The workload $W_n \in \mathbb{R}^m$, at the m servers seen by the n^{th} arrival under $\mathcal{BR}$ is associated for each n.*

2. *For $i = 1, 2, \ldots, m$, let $\{\tilde{W}_n^i\}_0^\infty$ represent the workload seen by arrivals in an independent Cavity Queue as in Definition 3. Let ${}^k s$ be a typical routing vector s under $\mathcal{BR}$ subject to $|s| = k$. Under either $\mathcal{WF}$, $\mathcal{BS}$, or $\mathcal{BR}$, the conditional delay D_n^k of Definition 4 satisfies the following property: for each $k, n \in \mathbb{Z}_+$:*

$$D_n^k \le^{icx} \max_{i:\, {}^k s^i>0} \tilde{W}_n^i + c\, {}^k s^i. \tag{5}$$

Here is now a uniform bound in n.

Theorem 5. *Consider a system satisfying the assumptions of Theorem 4. Suppose that $\rho = c\lambda \sum_{k=0}^\infty k\pi_k/m < \mu$. For $i = 1, 2, \ldots, m$, let $\tilde{W}^i$ represent the stationary workload of an independent Cavity Queue. Then, under either $\mathcal{WF}$, $\mathcal{BS}$, or $\mathcal{BR}$, the conditional delay D_n^k satisfies the following property: for each $k, n \in \mathbb{Z}_+$:*

$$D_n^k \le^{icx} \max_{i:\, {}^k s^i>0} \tilde{W}^i + c\, {}^k s^i. \tag{6}$$

Proof. This follows from Theorem 4 and noting that, using standard coupling arguments, an $M/GI/1$ queue starting empty at time $t = 0$ and its version in equilibrium can be coupled in such a way that the former is always lower than the latter. □

The above bound clearly reflects the impact of the local dynamics at individual servers as well as the global view seen by arrivals. As we shall see, it can be computed using Lemma 3 and using extremal statistics.

In what follows, we focus on π such that $\pi_k = 0$ for each $k > m$. Such a case is perhaps meaningful for clusters with very large m since files which span each of the thousands of servers may be rare. Under this scenario, Corollary 2 below shows that delays admit a particularly simple bound.

Corollary 1. *Consider a system with m servers. Suppose that π belongs to class $\Theta(m)$ and that $\pi_k = 0$ for each $k > m$. Suppose that $\rho < \mu$. Let*

$$q = q(\lambda, \sigma) = \left|\lambda + \frac{W\left(-\lambda\sigma \exp(-\lambda\sigma)\right)}{\sigma}\right|, \tag{7}$$

where W denotes the principal branch of the Lambert W function. Then, under $\mathcal{WF}$, $\mathcal{BS}$ or $\mathcal{BR}$, the conditional steady state delay D^k satisfies

$$E[D^k] - c \le \frac{1}{q(\frac{\rho}{c}, \frac{c}{\mu})} \log k(1 + o(1)), \tag{8}$$

as k tends to infinity, where q is the function defined in (7).

Note that the last relation implies that

$$E[D^k] \le \frac{1}{q(\frac{\rho}{c}, \frac{c}{\mu})} \log k(1 + o(1)),$$

when k tends to infinity. However it turns out that the formulation in (10) is numerically more accurate in the prelimit.

Surprisingly, as long as π belongs to $\Theta(m)$ and the load per server is fixed, the above bound does not depend on π. However, note that the bound is for the conditional delay. The bound on the overall delay still depends on π.

This bound scales linearly with c but logarithmically with k. Thus, for small and medium files, it pays to have smaller chunk size (see Subsection 6.4 for a quantification of this gain). This insight also concurs with the results obtained in [12] under a mean field approximation.

5. RANDOM CHUNK SIZES

We now study the scenario where the chunk size may be different for different files, which is modeled via recursion (2). Suppose that the random variables $\{c_n\}_0^\infty$ are i.i.d. with distribution ψ. The results of Section 3 readily extend to this scenario. In particular the statement of Theorems 1, 2 and 3 can be shown to hold for this scenario as well, with minor modifications in the proofs. We skip the details for brevity.

We now extend the results of Section 4. We first modify the notion of Cavity Queue as follows.

Definition 6. *The* Modified Cavity Queue *is an $M/GI/1$ FCFS queue which starts empty at time $t = 0$, has Poisson arrivals with rate λm, service rate μ bits/sec, and where service requirement in bits are i.i.d. with distribution equal to* *that of the random variable X, where X can is generated as follows: first, generate a $\mathbb{Z}_+$ valued random variable Y with probability mass function given as follows:*

$$\tilde{\pi}(0) = 1 - \sum_{k=1}^{m} \frac{k}{m}\pi_k - \sum_{k=m+1}^{\infty} \pi_k,$$

and for $l = 1, 2, \ldots$

$$\tilde{\pi}(lc) = \sum_{k=(l-1)m+1}^{lm} (\frac{k}{m} - l + 1)\pi_k + \sum_{k=lm+1}^{(l+1)m-1} (1 - \frac{k}{m} + l)\pi_k.$$

Let Z be a random variable with distribution ψ. Then, $X = YZ$.

Recall that the steady state workload distribution of an $M/GI/1$ FCFS queue satisfies Lemma 3. By using the above notion of Modified Cavity Queue, analogues of Theorem 4 and 5 can be shown to hold with minor modifications in proofs. Here, we only reproduce the analogue of Theorem 5 for brevity.

Theorem 6. *Consider a system with m servers which start empty. The chunk sizes $\{c_n\}_0^\infty$ are i.i.d. with distribution ψ. For each $k \in \mathbb{Z}_+$, batch requests for k blocks (i.e., coded chunks) arrive as per an independent point process with rate $\pi_k \lambda m$ and are routed to different servers upon arrival. Servers serve the block requests in FCFS fashion at rate μ bits per second.*

Suppose that $\pi = (\pi_k : k \in \mathbb{Z}_+)$ belongs to class $\Theta(m)$ (see Definition 5). Suppose that $\rho = E[c_1]\lambda \sum_{k=0}^{\infty} k\pi_k/m < \mu$. For $i = 1, 2, \ldots, m$, let $\tilde{W}^i$ represent the stationary workload of an independent Modified Cavity Queue (see Definition 6). Then, under either $\mathcal{WF}$, $\mathcal{BS}$, or $\mathcal{BR}$, the conditional delay D_n^k satisfies the following property: for each $k, n \in \mathbb{Z}_+$:

$$D_n^k \leq^{icx} \max_{i:\, {}^k s^i > 0} \tilde{W}^i + c\, {}^k s^i, \qquad (9)$$

where c is a random variable with distribution ψ.

Again consider a scenario where $\pi_k = 0$ for each $k > m$. Suppose that ψ is exponential. Then the Modified Cavity Queue is an $M/M/1$ queue. Thus, the following corollary readily follows from the above theorem.

Corollary 2. *Consider a system with m servers. Suppose that π belongs to class $\Theta(m)$, and that $\pi_k = 0$ for each $k > m$. Suppose that the distribution ψ is exponential with mean c. Suppose that $\rho < \mu$. Then, under $\mathcal{WF}$, $\mathcal{BS}$ or $\mathcal{BR}$, the conditional steady state delay D^k satisfies*

$$E[D^k] - c \leq \frac{\mu}{\mu - \rho} \sum_{l=1}^{k} \frac{1}{l} < \frac{\mu}{\mu - \rho}(\log k + 1). \qquad (10)$$

6. SIMULATION AND PERFORMANCE EVALUATION

In this section we use our analysis and simulations in order to develop a better quantitative understanding of the relative performance and scaling laws under $\mathcal{WF}$, $\mathcal{BS}$, and $\mathcal{BR}$.

6.1 Simulation Methodology

The simulation methodology we selected is not based on the classical discrete event principles but rather on a direct use of the recurrence equations (1). The advantages of the latter on the former are multiple, in term of generality and of complexity. This recurrence relation setting is well adapted to handling deterministic service times and general routing vectors, whereas event driven Markov chain simulation would require exponentiality assumptions and make the handling of workload based routing policies cumbersome. The complexity of $\mathcal{BS}$ is that of a sorting algorithm. If the servers containing at most one chunk from the requested file are sorted in increasing order of their load, then it suffices to take the k smallest loads if $k \leq m$. When $k > m$, the complexity depends on m rather than k, as only $k - m\lfloor k/m \rfloor$ servers with the smallest load need to be searched. The complexity is then in $O(\min(k, m) \log \min(k, m))$. The complexity of $\mathcal{WF}$ depends on k whatever its value: one strategy is to first sort the servers containing at least one chunk of the requested file. Each time a chunk is requested from one server, its load increases by c, and this server has to be reinserted in the ordered list of servers. The complexity is then $O(k \log(m + k))$.

In several experiments, the size of the files is at most mc and $\alpha_k - k \leq 2$. When $\alpha_k \leq m$ for all k, then $\mathcal{BS}$ and $\mathcal{WF}$ are exacly the same: as each server contains at most one chunk of any given file, all the routing vectors are balanced, and Theorem 1 states the optimality of $\mathcal{BS}$ in this case. For this reason, we will only compare $\mathcal{BS}$ with $\mathcal{BR}$.

An important question is that of the steady state characterization. For this, we leverage Birkhoff's pointwise ergodic theorem, which shows that empirical averages based on iterates of the recurrence equations (1) converge to the steady state mean values. In practice, we perform 10^5 iterates to estimate each point of the following plots.

6.2 Impact of the Delivery Policy

The first numerical experiments illustrate the comparison results of Section 3 and more precisely Theorem 2. The setting is the following: there are $m = 200$ servers; the distribution π is *Binomial*(m,p), with $p = 0.1, 0.3, 0.5$ (which gives an average of 20, 60 and 100 chunks, respectively); recall that for each value of p this falls within the class of distribution $\Theta(m)$; the server speed is $\mu = 1$ and the chunk size is $c = 10$; the arrival rate is chosen in such a way that the load per server is always equal to 0.7; the coding assumptions are that $\alpha_k = k + 2$.

Figure 2 compares the mean delay under $\mathcal{BR}$ and $\mathcal{BS}$, for various values of p. The bound obtained in Corollary 2 is also plotted. Within the range considered in these plots, the mean delays increase logarithmically in k for $\mathcal{BR}$. The bound correctly captures the logarithmic increase w.r.t. the $\mathcal{BR}$, and is in fact approximate for small p. For these parameters, it is already a good heuristic for $p = 0.1$.

We observe that $\mathcal{BS}$ (or equivalently $\mathcal{WF}$) performs significantly better than $\mathcal{BR}$. Intuitively, this happens since the workload across servers is more balanced under $\mathcal{BS}$ and $\mathcal{WF}$. In particular, while they seem to increase as $\log \log k$ for $\mathcal{BR}$ and $\mathcal{WF}$. One may see this in the light of the well-known result on balanced allocations under balls and bins setting [23] where load-balancing is shown to achieve exponential improvement in load at the most-loaded bin. However, our setting is markedly different. Not only do we in-

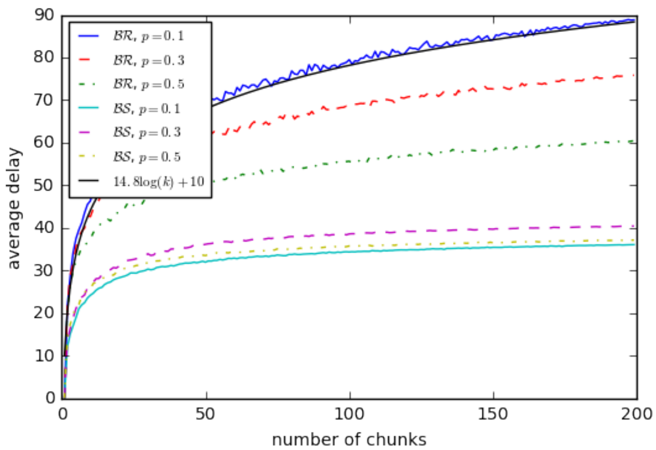


Figure 2: Mean delay as a function of the number of chunks.

corporate queuing dynamics (i.e., arrivals and services), but also batch arrivals. We are interested in studying the delay of a typical job which depends on the workload at a randomly chosen subset of servers, instead of the most-loaded server.

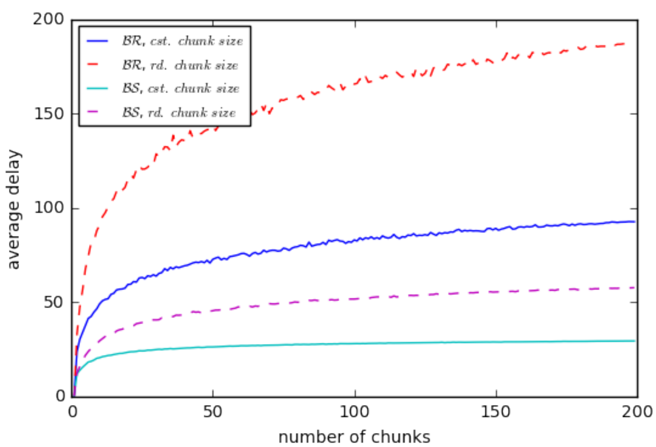


Figure 3: Mean delay as a function of the number of chunks: Comparison of scenarios with constant chunk size and random chunk size.

Interestingly, in our setting, the increase in delays seems logarithmic in k even for $\mathcal{BR}$ and $\mathcal{WF}$ policies under a scenario where the chunk size is assumed to be random, as exhibited in Figure 3. The setting is the following: there are $m = 200$ servers; the distribution π is Geometric with rate 0.25, the size of chunks are exponentially distributed with rate 0.1. The load per server is 0.7. The coding assumptions are that $\alpha_k = k + 2$. The plots show that, in the cases where the chunks sizes are exponentially distributed, the log growth in delays as exhibited by the upper bound of Corollary 2 is tight when the per-server load is sufficiently large.

Under the assumptions studied above, for each policy, the growth of delays is logarithmic or sub-logarithmic in the file size. This type of growth does not generalize to all cases. For instance, it is shown in Subsection 6.5 below that it can actually be linear.

6.3 Impact of Coding Rate

In order to evaluate the impact of coding rate, we consider a system under $\mathcal{BS}$ with m servers, where m varies. We take $\lambda = 0.1$, $p = 0.5$ and again π is *Binomial*(p, m), so that the load per server is constant. We take $c = 14$ and $\mu = 1$. Figure 4 gives the mean delay as a function of m for different choices of $\alpha_k - k$.

As expected, $\mathcal{WF}$ and $\mathcal{BS}$ perform significantly better than $\mathcal{BR}$ when $\alpha_k > k$. We observe that the delays increase logarithmically with m. This may be reasoned as follows: In the presence of small and medium sized files if $\alpha_k - k$ is a constant then the choice in load-balancing is limited and the unevenness in workload distribution across servers increases with m. Further, as we increase the code redundancy $\alpha_k - k$, we observe that the mean delays decrease as $\frac{1}{\log(\alpha_k - k)}$. This shows that the impact of increasing choice in load-balancing by improving coding rate is limited.

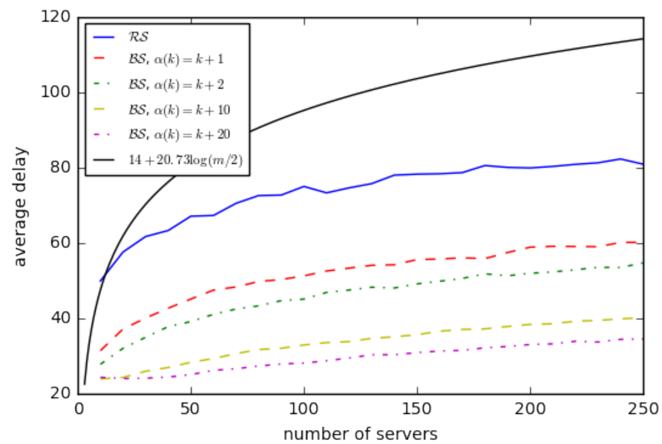


Figure 4: Mean delay under $\mathcal{BS}$ as a function of number of servers, for different coding rates.

6.4 Impact of the (Deterministic) Chunk Size

We now consider the impact of increasing chunk size on delays for the case with π being *Binomial*(p, m). Rather than taking chunks of size c, we take chunks of size c/a with a an integer larger than 1, and study mean delay as a function of a. Here, a file which had k chunks now has ak chunks. Consider the upper bound of Corollary 2 (this bound is generic in that it holds for all considered delivery policies). The bound in the new chunk definition is now $\frac{1}{|s^*(a)|}\ln(1+k)(1+o(1))$, when k tends to infinity, with $s^*(a)$ the only negative solution of the equation $s = \lambda pa\left(1 - \exp\left(-\frac{sc}{a}\right)\right)$. When a is large (but such that $pa < 1$), this root can be approximated as $|s^*(a)| = \frac{(1-\lambda pc)2a}{c^2\lambda p}$. so that we have the generic bound on requests of initial cardinality k:

$$E[D^k] \leq \frac{c^2\lambda p}{(1-\lambda pc)2a}\ln(ak)(1+o(k)),$$

when k tends to infinity. This shows that within the above Binomial setting, the mean delay of any policy can be decreased in such a way that the constant multiplying the logarithmic term is divided by a (provided $pa < 1$).

6.5 Beyond the Logarithmic Regime

The last three subsections were about the case where π has its support on the integers from 0 to m. In view of the results of these subsections, it makes sense to call this regime the *logarithmic regime*. There are some caveats with this terminology. This term is justified within the $Binomial(p, m)$ setting, if p is sufficiently separated from 1. As we saw above, for p constant and less than 1, the logarithmic regime prevails even when m tends to infinity. Note that this goes way beyond the regimes considered in the mean field approach. However, it should be clear that for fixed m and for p close to 1, the mean delay must be approximately a constant in k.

Note that when the support of π is not limited to the integers less than m with m fixed, it should be clear that for all delivery policies, when k tends to infinity, requests of cardinality k have a mean delay of order Ck with C a constant. This is the linear regime alluded to above.

7. CONCLUSIONS

One of the main motivations of this work was to derive scaling laws for job delays in data clusters. A primary difficulty in the analysis of job delays in multi-server systems comes from the stochastic coupling of the server dynamics. To simplify the analysis, research often resorts to an asymptotic 'mean-field' approximation which assumes an infinite number of servers and a static empirical distribution. This approximation allows for the decoupling of the dynamics at the servers attending a tagged job. However, such a decoupling does not hold when the total number of servers m is finite, or when certain jobs are attended by $O(m)$ servers. In the present paper, we developed a new machinery which utilizes the notion of association of random variables to obtain explicit bounds on delays for *finite systems*. We obtain these bounds via an 'independent version' of a coupled system but without requiring the decoupling of the servers. Further, we clarified the sense (increasing convex ordering) in which adaptive policies outperform workload oblivious policies. Our simulation results suggest that several quite different delay growths can be obtained in function of file size, from strictly sub-logarithmic to logarithmic to linear. While some specific examples of these behaviors are well explained by our machinery, there is still a need in the future for a full classification allowing one to predict which assumptions lead to each type of growth.

Our machinery is robust to statistical assumptions and to model specifics. In addition, various types of file updates/writes operations can be incorporated in the basic model while preserving the basic association and stochastic comparison properties. In the future, this model should hence also provide a first comprehensive setting for analyzing the impact of updates on job delays in data clusters.

8. REFERENCES

[1] A. Verma, L. Pedrosa, M. R. Korupolu, D. Oppenheimer, E. Tune, and J. Wilkes, "Large-scale cluster management at Google with Borg," in *Proceedings of the European Conference on Computer Systems (EuroSys)*, pp. 18:1–18:17, 2015.

[2] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The hadoop distributed file system," in *2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, pp. 1–10, 2010.

[3] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta, "VL2: A scalable and flexible data center network," in *Proceedings of the ACM SIGCOMM 2009 Conference on Data Communication*, pp. 51–62, 2009.

[4] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The google file system," in *Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles (SOSP)*, pp. 29–43, 2003.

[5] D. Ford, F. Labelle, F. Popovici, M. Stokely, V.-A. Truong, L. Barroso, C. Grimes, and S. Quinlan, "Availability in globally distributed storage systems," in *Proceedings of the 9th USENIX Symposium on Operating Systems Design and Implementation*, pp. 61–74, 2010.

[6] A. Dimakis, P. Godfrey, Y. Wu, M. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *Information Theory, IEEE Transactions on*, vol. 56, pp. 4539–4551, Sept 2010.

[7] J. S. Plank and C. Huang, "Tutorial: Erasure coding for storage applications." Slides presented at FAST-2013: 11th Usenix Conference on File and Storage Technologies: `http://web.eecs.utk.edu/~plank/plank/papers/FAST-2013-Tutorial.html`, February 2013.

[8] V. Shah and G. de Veciana, "High performance centralized content delivery infrastructure: Models and asymptotics," *IEEE/ACM Transactions on Networking*, vol. 23, pp. 1674–1687, Oct 2015.

[9] V. Shah and G. de Veciana, "Asymptotic independence of servers' activity in queueing systems with limited resource pooling," *Queueing Systems*, vol. 83, no. 1, pp. 13–28, 2016.

[10] V. Shah and G. de Veciana, "Impact of fairness and heterogeneity on delays in large-scale centralized content delivery systems," *Queueing Systems*, vol. 83, no. 3, pp. 361–397, 2016.

[11] N. D. Vvedenskaya, "Large queueing system where messages are transmitted via several routes," *Problemy Peredachi Informatsii*, vol. 34, no. 2, pp. 98–108, 1998.

[12] B. Li, A. Ramamoorthy, and R. Srikant, "Mean-field-analysis of coding versus replication in cloud storage systems," in *Proceedings of IEEE INFOCOM*, 2016.

[13] N. B. Shah, K. Lee, and K. Ramchandran, "The MDS queue: Analysing the latency performance of erasure codes," in *IEEE International Symposium on Information Theory (ISIT)*, pp. 861–865, June 2014.

[14] 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*, vol. 32, pp. 989–997, May 2014.

[15] S. Jain, M. Demmer, R. Patra, and K. Fall, "Using redundancy to cope with failures in a delay tolerant network," in *ACM SIGCOMM*, (New York, NY, USA), pp. 109–120, 2005.

[16] G. Liang and U. C. Kozat, "TOFEC: achieving optimal throughput-delay trade-off of cloud storage using erasure codes," in *IEEE INFOCOM 2014 -*

IEEE Conference on Computer Communications, pp. 826–834, April 2014.

[17] F. Baccelli and P. Brémaud, *Elements of Queueing Theory: Palm Martingale Calculus and Stochastic Recurrences*, vol. 26 of *Applications of Mathematics*. Berlin: Springer-Verlag, second ed., 2003.

[18] 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*, vol. 21, no. 3, pp. 629–660, 1989.

[19] A. Müller and D. Stoyan, *Comparison methods for stochastic models and risks*. Wiley, 2002.

[20] A. W. Marshall, I. Olkin, and B. C. Arnold, *Inequalities: Theory of Majorization and Its Applications*. Springer, 2nd ed., 2011.

[21] M. Bramson, Y. Lu, and B. Prabhakar, "Randomized load balancing with general service time distributions," in *Proceedings of the ACM Sigmetrics*, pp. 275–286, 2010.

[22] L. Takács, *Introduction to the theory of queues*. University texts in the mathematical sciences, Oxford University Press, 1962.

[23] Y. Azar, A. Z. Broder, A. R. Karlin, and E. Upfal, "Balanced allocations," *SIAM J. Comput.*, vol. 29, pp. 180–200, Feb. 2000.

[24] M. D. Mitzenmacher, *The Power of Two Choices in Randomized Load Balancing*. PhD thesis, University of California, Berkeley, 1996.

[25] N. D. Vvedenskaya, R. L. Dobrushin, and F. I. Karpelevich, "Queueing system with selection of the shortest of two queues: An asymptotic approach," *Problemy Peredachi Informatsii*, vol. 32, no. 1, pp. 20–34, 1996.

[26] L. Ying, R. Srikant, and X. Kang, "The power of slightly more than one sample in randomized load balancing," in *Proc. of IEEE INFOCOM*, 2015.

9. APPENDIX

9.1 Proof of Lemma 1

Recall that if $X \prec_s^{st} Y$ then $E[\phi(X)] \le E[\phi(Y)]$ for any increasing Schur-convex function ϕ. Now consider an increasing convex function $g : \mathbb{R}^m \to \mathbb{R}$. Let P be the set of all permutations of $(1, 2, \dots, m)$. One can check that for any $p \in P$, the function $g(p(x))$ is increasing and convex in x. Let function ϕ be given as follows:

$$\phi(x) = \frac{1}{m!} \sum_{p \in P} g(p(x)).$$

Then, ϕ is a symmetric, increasing, and convex function; hence an increasing Schur-convex function [20]. Further, by exchangeability of X, we have $E[g(X)] = E[g(p(X))]$ for any $p \in P$, which in turn implies $E[g(X)] = E[\phi(X)]$. Similarly, by exchangeability of Y, we have $E[g(Y)] = E[\phi(Y)]$. But as noted above, we have $E[\phi(X)] \le E[\phi(Y)]$. The result thus follows since g is chosen arbitrarily.

9.2 Proof of Theorem 1

We show $\prec^{st}$ comparisons below. The *icx* comparisons then follow from Lemma 1 and noting that $W + s^{\mathcal{WS}}$, $W + s^{\mathcal{BS}}$, and $W + s^{\mathcal{BR}}$ are exchangeable since each of these policies is exchangeable.

We will need below the following lemma, which says that a vector becomes more balanced if we decrease a larger entry by a 'small' amount and increase a smaller entry by the same amount.

Lemma 4. *Let $x \in \mathbb{R}^m$ such that $x^i \le x^j$ and $0 \le \delta \le x^j - x^i$. Then $x + \delta e_i - \delta e_j \prec x$.*

Proof. Set $y = x + \delta e_i - \delta e_j$. There exist k and l such that $x_i = x^{(k)}$ and $x_j = x^{(l)}$, with $k < l$, k' and l' such that $y_i = y^{(k')}$ and $y_j = y^{(l')}$, and as $\delta \le x^j - x^i$, $k \le k', l' \le l$.

For all $i' < k$ and $i' > l$, we have $\sum_{u \le i'} x^{(u)} = \sum_{u \le i'} y^{(u)}$: in the first case, exactly the same terms are involved, and in the second, $x^i + x^j = y^i + y^j$, and these terms are all involved.

If $k \le i' < \min(k', l')$, then $\sum_{u \le i'} y^{(u)} = \sum_{u<k} x^{(u)} + \sum_{k \le u \le i'} x^{(u+1)} \ge \sum_{u<k} x^{(u)} + \sum_{k \le u \le i'} x^{(u)}$.

If $\min(k', l') \le i' < \max(k', l')$, $\sum_{u \le i'} y^{(u)} = \sum_{u<k} x^{(u)} + \sum_{k \le u < \min(k',l')} x^{(u+1)} + \min(y^i, y^j) + \sum_{\min(k',l') < u \le i'} x^{(u)} \le \sum_{u \le i'} x^{(u)}$, as $\min(y^i, y^j) = \min(x^i + \delta, x^j - \delta) \ge x^i$ (because $\delta \le x^j - x^i$).

If $\max(k', l') \le i' \le l$, $\sum_{u \le i'} y^{(u)} = \sum_{u \le i'} x^{(u)} + x^{(l)} - x^{(i')} \ge \sum_{u \le i'} x^{(u)}$ (we have used that $x^{(l)} = x^j \ge x^{(i')}$).

Then $y \prec x$. $\square$

Optimality of $\mathcal{WS}$: We now show that $\mathcal{WS}$ achieves more balanced workload than any other policy. For ease of notation, let s, κ and a represent the vectors associated with $\mathcal{WS}$ with their usual meaning, and let s', κ' and a' represent those associated with any other policy. Recall that the number of chunks for the requested file κ and the placement vector a associated with an arrival have same distribution in each system and they are independent of the workload seen by the arrival. Thus, it is sufficient to prove that $W + cs \prec W + cs'$ w.p. 1 subject to the coupling $\kappa = \kappa'$ and $a = a'$.

We proceed as follows. For any routing vector s'', define its distance to s as $d(s, s'') = \sum_{i \mid s^i > s''^i} s^i - s''^i$. As s and s'' are integer-valued, $d(s, s'')$ is a non-negative integer.

Under the coupling $\kappa = \kappa'$ and $a = a'$, we show that for the routing vector $s' \ne s$, there exists another routing vector s'' such that $d(s, s'') < d(s, s')$ and $W + cs'' \prec W + cs'$. This means that for any routing vector s', we can construct a sequence $(s_0, \dots, s_d)$ such that $W + cs_0 \prec W + cs_1 \prec \cdots \prec W + cs_d = W + cs'$, with $d(s_0, s) = 0$, that is, $s = s_0$. In conclusion, for all routing vector s', $W + cs \prec W + cs'$, hence the optimality of water-filling.

Let us now prove the existence of the routing vector s''. Note that any routing vector less than $\max(s, s')$ is admissible, i.e., $\max(s, s') \le a$. As $s' \ne s$, there exists i and j such that $(W + cs)^i < (W + cs')^i$ and $(W + cs)^j > (W + cs')^j$, and as s and s' are integer-valued, $s^i \le s'^i - 1$ and $s^j \ge s'^j + 1$.

Consider the step of the water-filling algorithm where a chunk is sent to server j for the last time, and let $\tilde{s}$ be the routing vector obtained just before the chunk is sent to server j step: in particular $\tilde{s}^j = s^j - 1$, so

$$(W + cs')^j \le (W + cs)^j - c = (W + c\tilde{s})^j. \qquad (11)$$

Due to the water-filling algorithm, server j is chosen over i because $(W + c\tilde{s})^j \le (W + c\tilde{s})^i$. But we also have $(W + c\tilde{s})^i \le$

$(W+cs)^i \leq (W+cs')^i - c$. Thus, we get $(W+c\tilde{s})^j \leq (W+cs')^i - c$. Combining this with (11), we get $(W+cs')^j \leq (W+cs')^i - c$.

Now, consider the new vector where $s'' = s' + e_j - e_i$. We have $d(s, s'') < d(s, s')$ and from Lemma 4, we have $W + s'' \prec W + s'$, as required.

Comparing $\mathcal{BS}$ with $\mathcal{BR}$: We now show that $\mathcal{BS}$ achieves more balanced workload than $\mathcal{BR}$. For ease of notation, let s, κ and a represent the vectors associated with $\mathcal{BS}$ with their usual meaning, and let s', κ' and a' represent those associated with $\mathcal{BR}$. We again assume the coupling $\kappa = \kappa'$ and $a = a'$.

Under the coupling, we will show a statement which is somewhat stronger than required; in particular, we will show that the batch-sampling is optimal among all *balanced routing vectors*, i.e. the routing vectors s' such that $s'^i \in \{l, l+1\}$ where $l = \left\lfloor \frac{\kappa'}{m} \right\rfloor$, and $|\{i : s'^i = l+1\}| = \kappa' - m \left\lfloor \frac{\kappa'}{m} \right\rfloor$. Moreover, due to our coupling $\kappa = \kappa'$ and $a = a'$, we only need to focus on routing vectors s, s' of the form $\{0,1\}^m$.

We now proceed as follows: Take any balanced routing vector s'' and define its distance to s, the routing vector obtained with the water-filling policy as $d(s, s'') = \sum_{i \mid s^i > s''^i} s^i - s''^i$. As s and s' are integer-valued, $d(s, s')$ is a non-negative integer. We show that for any routing vector $s' \neq s$, there exists another balanced routing vector $s'' \in \{0,1\}^m$ such that $d(s, s'') < d(s, s')$ and $W + cs'' \prec W + cs'$. This means that for any routing vector s', we can construct a sequence $(s_0, \ldots, s_d)$ such that $W + cs_0 \prec W + cs_1 \prec \cdots \prec W + cs_d = W + cs'$, with $d(s_0, s) = 0$, that is, $s = s_0$. In conclusion, for all routing vector s', $W + cs \prec W + cs'$, hence the optimality of batch-sampling among the balanced routing vectors.

Let us now prove the existence of the routing vector s''. Note that any routing vector less than $\max(s, s')$ is admissible (there are enough chunks available). As $s' \neq s$, there exists i and j such that $s^i = 0$, $s'^i = 1$, $s^j = 1$ and $s'^j = 0$, and as s is obtained from the batch-sampling, one can always such an i and j such that that $W^j \leq W^i$, so $(W+cs')^j \leq (W+cs')^i$ and $(W+cs')^j = W^j \leq W^i = X^i + cs'^i - c = (W+cs')^i - c$.

Consider the routing vector $s'' = s' + e_j - e_i$. From Lemma 4, the above inequality implies that $W + s'' \prec W + s'$, and $d(s, s'') < d(s, s')$, as required.

9.3 Proof of Lemma 2

Since $W \leq^{icx} W'$, Strassen's theorem [19] says that there exists a coupling such that $E[W'|W] \geq W$. In addition, since s and s' are identical in distribution and independent of W and W', there exists a coupling (namely one with $s = s'$) such that

$$E\left[W' + cs' | W, s\right] \geq W + cs. \tag{12}$$

Consider an increasing convex function g. Under the above coupling, using Jensen's inequality we get

$$E[g(W' + cs')|W, s] \geq g\left(E[W' + cs'|W, s]\right).$$

Combining this with (12), we get

$$E[g(W' + cs')|W, s] \geq g(W + cs).$$

By taking expectation on both sides, we get $E[g(W'+cs')] \geq E[g(W+cs)]$. Hence the result holds.

9.4 Proof of Theorem 4

We first prove part (i) using induction. Clearly, W_0 is associated since all its entries are constant and equal to zero. Suppose W_n is associated for some n. We show below that this implies that W_{n+1} is associated as well.

Recall that under $\mathcal{BR}$ the random vectors W_n, s_n, and $-\mu\tau_n\mathbf{1}$ are mutually independent and are themselves associated. Hence, from part (ii) of Proposition 2, we have that the entries of W_n, s_n, and $-\mu\tau_n\mathbf{1}$ are mutually associated. Each entry of $(W_n + cs_n - \mu\tau_n\mathbf{1})^+$ is an increasing function of the entries of W_n, s_n, and $-\mu\tau_n\mathbf{1}$. From m applications of part (i) of Proposition 2, and then of its part (iii), we get that W_{n+1} is associated.

We now prove part (ii) of the theorem. We show that in fact for $\mathcal{BR}$ policy the stochastic dominance is in $\leq^{st}$ sense which is stranger than $\leq^{icx}$. By definition, the vector $\tilde{W}_n = (\tilde{W}_n^i : i = 1, \ldots, m)$ is an independent version of W_n for each n under $\mathcal{BR}$ policy. Since $\tilde{s}$ is an independent exchangeable vector, and since both W_n and $\tilde{W}_n$ are exchangeable, it is sufficient to assume that $\tilde{s}$ is deterministic. Then, the result follows for $\mathcal{BR}$ from Proposition 1.

For $\mathcal{WF}$ and $\mathcal{BS}$, the result then follows by arguing along the lines of Theorem 3 while additionally conditioning on $|s_n| = k$.

9.5 Proof of Corollary 2

The proof leverages the following two results:

Theorem 7 (Theorem 7.4. in [18]). *Let $\{Y_l\}_1^\infty$ be a family of i.i.d. $\mathbb{R}_+$-valued random variables whose common distribution function $G(\cdot)$ exhibits the tail behavior*

$$P[Y_1 > x] = 1 - G(x) = Ce^{-qx}(1 + o(1)), \quad x \geq 0,$$

for some $q > 0$ and $C > 0$. Then

$$E\Big[\max\{Y_1, ..., Y_k\}\Big] = \frac{1}{q}\log(k)(1 + o(1))$$

when k goes to infinity.

Lemma 5. *The steady state Y delay in the M/D/1 queue with arrival rate λ and service time σ, with $\lambda\sigma < 1$ has the tail behavior*

$$P[Y > x] = 1 - G(x) = Ce^{-qx}(1 + o(1)), \quad x \geq 0,$$

with q defined as in Equation (7).

Proof. The Pollaczek-Khinchine formula, of Lemma 3, when applied to the M/D/1 queue, gives a steady state delay with a Laplace transform having an isolated pole at the only solution other than 0 of the equation

$$s = \lambda(1 - \exp(-s\sigma)).$$

Elementary calculations show that this solution is precisely q given in (7). The shape of the tail then follows from classical complex analysis arguments. □

The fact that the delay of a request of size k is upper bounded by c plus the maximum of the workloads in k independent M/D/1 queues with arrival rate λp and service times c immediately leads to the announced result.