High Performance Centralized Content Delivery Infrastructure: Models and Asymptotics

Virag Shah and Gustavo de Veciana

***Abstract*—We consider a centralized content delivery infrastructure where a large number of storage-intensive files are replicated across several collocated servers. To achieve scalable mean delays in file downloads under stochastic loads, we allow multiple servers to work together as a pooled resource to meet individual download requests. In such systems basic questions include: How and where to replicate files? What is the impact of dynamic service allocation across request types, and whether such allocations can provide substantial gains over simpler load balancing policies? What are tradeoffs amongst performance, reliability and recovery costs, and energy? This paper provides a simple performance model for large systems towards addressing these basic questions.**

***Index Terms*—Performance models, delays, content delivery infrastructure, resource pooling.**

I. Introduction

In near future, high volume file transfers such as those involved in downloading scientific datasets/visualization, 3D videos, software updates, and other immersive technologies may dominate internet traffic. We consider a centralized infrastructure which stores and delivers large files such that delay to serve a download request is scalable with traffic loads. Such centralized infrastructure could, for example, be part of a larger distributed content delivery network, where requests not currently available at distributed sites are forwarded to the centralized infrastructure which in turn delivers the files to the remote sites and/or users. Performance in such systems is the result of a complex interaction among requests that come and go dynamically and the pools of resources that are able to serve them. As traffic loads increase, one can make the following design choices to meet performance requirements: 1) dimensioning of system's server and network resources; 2) (possibly random) placement of data across servers; and 3) policy for routing/servicing requests. One of the goals of this paper is to develop robust large-scale performance models to enable system-level optimization with respect to these design choices.

We also aim to study tradeoffs among conflicting goals in such systems, e.g., 1) service capacity available to end users and the resulting perceived performance; 2) reliability and recovery costs; and, 3) energy costs. For example, by increasing the total number of active servers, or scaling the speed of individual servers, one can tradeoff energy cost with performance. A more subtle example, discussed further in the sequel, involves spreading multiple copies of files across pools of servers so as to trade off the cost in recovery from large-scale server loss events, e.g., power outages [7], with performance.

Authors are with ECE Department at The University of Texas at Austin, Austin, TX 78712 USA (e-mail: virag@utexas.edu, gustavo@ece.utexas.edu).

This is an extended version of a paper presented at INFOCOM 2014, Toronto, Canada.

Our contributions. The key challenge we tackle in this paper is the performance evaluation of large scale storage systems wherein multiple file copies are placed across pools of servers and are subject to stochastic loads. We consider a system model where arriving file requests/jobs/flows can be collectively served by servers, i.e., different chunks of each file can be downloaded in parallel from servers currently storing the file – this is akin to peer-to-peer systems. Since each server can store multiple files, which are themselves replicated across sets of servers, the service capacities available to serve requests for different files are dynamically coupled. Indeed, as explained in the sequel, ongoing file requests can share server capacity subject to various possible 'fairness' objectives rendering performance evaluation quite challenging.

The main analytical contributions of this paper can be summarized as follows. Firstly, we propose a file-server model and show that the overall service capacity set has polymatroid structure. We combine this structural result of an achievable capacity region with dynamic balanced fair rate allocations (described later) to develop an explicit expression for the mean file transfer delay experienced by file requests. Secondly, we prove a new asymptotic result for *symmetric* large-scale systems wherein the distribution of the number of waiting file requests concentrates at its mean. This result provides an easily computable approximation for the mean delay which is used to quantify system tradeoffs.

Finally, these analytical results are used to develop and quantify three key insights regarding large file-server systems:

a) We show how dynamic service capacity allocation across ongoing demands is impacted by the structure of overlapping resource pools (file placement) and quantify the substantial performance benefits over simpler load balancing strategies such as those assigning file requests at random or to least loaded servers.
b) We show that performance gains resulting from the overlapping of server pools, although significant, quickly saturate as one increases the overlap. This enables engineering of such systems to realize close to optimal performance while simultaneously achieving high reliability and thus low recovery costs.
c) For a simple speed scaling policy where the processor runs at low speed (or halts) when idle and a high but fixed speed when busy, we show that dynamic service capacity allocation can achieve up to 70% energy saving

as compared to simpler policies.

Related work. There are several large-scale performance models applicable to content delivery systems. For example, the super-market queueing model studied in [6], [22], [23], [33] captures a policy where each arriving request is assigned to the least loaded server among those able to serve it. It is known to have better mean delay performance and tail decay for the distribution of the waiting jobs as compared to the policy of routing requests randomly among the possible servers. Alternatively, one can make centralized scheduling decisions as servers become available [19], [36]. In [19] a greedy policy is shown to be optimal over all scheduling disciplines in a heavy-traffic regime. A centralized policy is studied in [36] and is shown to have robustness properties with respect to limited heterogeneity in loads across different file types. The key difference between these works and ours is that, rather than assigning a file request to a single server, we allow it to be served by multiple servers simultaneously. In the sequel, we evaluate the benefits of doing so.

Pooling of server resources is similar in spirit to multipath routing in wireline networks, see e.g. [11]–[14], [35]. A multipath TCP architecture is proposed in [35] to achieve network wide resource pooling. Studies of the benefits of multipath routing have been previously carried out, e.g., in [14] the authors show the benefits of coordinating rate over multiple paths in terms of the worst case rate achieved by users in a static setting. For networks with stochastic loads, performance analysis under multipath transport is in general hard; [12], [13] study role of resource pooling in such a setting and provide performance bounds/approximations. Resource pooling in networks via multipath, and that in content delivery infrastructure via pooling of servers may eventually complement each other to achieve scalable performance gains.

There has also been previous work considering file placement across servers [16], [17], [25], [39]. For example, [16] studies file placement across servers so as to minimize 'bandwidth inefficiency' when there is a fixed set of file transfer requests. Further, [17], [25] consider the problem of adaptive replication of files for a loss network model where each server can serve one file request at a time, thus avoiding queuing. The focus of these works is on caching popular files via distributed content delivery networks. In turn, they rely on a centralized infrastructure to handle cache misses and request denials arising when all associated servers are busy. Another line of work has focused on online packing/placement of dynamically arriving files/objects under constraints on available resources, e.g., [29]. By contrast with these works, we assume file placements across servers are fixed and we examine the performance impact of this when the system is subject to stochastic loads with no loss.

There are several works in the literature studying energy-performance tradeoffs, see e.g., [10], [18] and citations therein. In [10], the authors provide an approximation to the number of servers that should be active so as to optimize the energy-delay product. Similarly, [34] investigates speed scaling so as to optimize a weighted average of energy and mean delay for a single server system. In [18], the authors consider energy costs of switching servers on and off and provide an optimal online algorithm to optimize overall convex cost functions that can include performance and energy costs. In these works a server can handle any job request. By contrast in this paper we are particularly interested in the situations where servers' capabilities are constrained (e.g., by the files they have available) and the coupling across server pools critically impacts energy-performance tradeoffs.

As will be discussed in more detail below this paper draws on, and extends, previous work on bandwidth sharing models; in particular "balanced fair" allocations, see e.g., [3]–[5]. Such allocations are a useful device in that they are amenable to analysis, are provably insensitive to job size distribution, and yet serve to approximate various forms of 'fair' resource sharing policies considered in the literature and in practice [2], [3], [21].

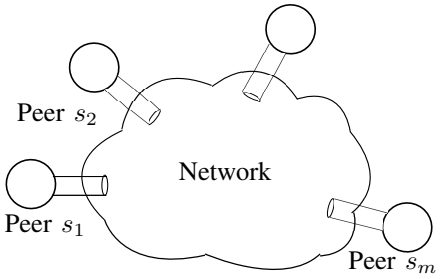
Organization of the paper. In Section II we develop our system model for file server systems under stochastic loads. In Section III we discuss fairness based resource allocation and provide an exact analysis for mean delay in file transfers under balanced fair service allocation. In Section IV we consider large scale systems and provide an asymptotic expression for the mean delay. In Section V we use our analysis to compare the performance of our policy with other resource allocation policies. In Section VI we discuss system tradeoffs involving mean delay, recovery costs and energy consumption. We conclude in Section VII. Proofs are provided in the Appendix.

II. System Model: file-server system, dynamics, and service capacity

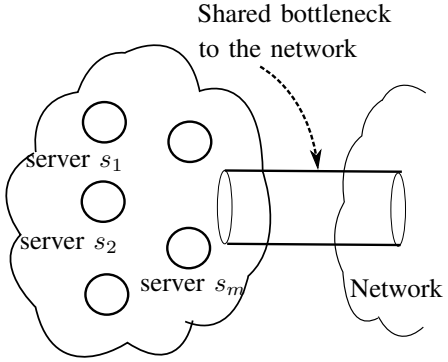
Let F denote a set of files and S a set of servers in a file-server system where $|F| = n$ and $|S| = m$. For each file $i \in F$ let $S_i \subset S$ denote the set of servers that store file i; thus $\mathcal{S} = (S_i : i \in F)$ captures a file placement policy. Suppose each server $s \in S$ has fixed service capacity of μ_s bits per second. For each $A \subset F$ let $S(A) \triangleq \cup_{i\in A} S_i$ and $\mu(A) \triangleq \sum_{s\in S(A)} \mu_s$ denote the set of servers capable of serving one or more of the files in A and the associated aggregate service capacity. In summary, $\big(F, S, \mu; \mathcal{S}\big)$ collectively define a *file-server system*.

Requests for file $i \in F$ arrive according to an independent Poisson process with rate λ_i. We shall use the terms request, flow and job interchangeably. Similarly, we refer to each file $i \in F$ as a file or a job class interchangeably. Each request has a service requirement corresponding to, for example, the number of bits it needs to download from the file-server system. Service requirements for a request for file $i \in F$ are i.i.d with mean ν_i bits. This can model, for example, requests for a part of a file. The requests for a file may even be of fixed size. Our model is insensitive to the service requirement distribution, i.e., the performance would depend on the distribution only through its mean. Let $\boldsymbol{\rho} = (\rho_i : i \in F)$, where $\rho_i = \lambda_i \nu_i$ denotes the load associated with class i.

Flows arrive to the system at total rate $\sum_{i\in F} \lambda_i$. Let u_k denote the flow corresponding to the k^{th} arrival after time $t = 0$. Let $q_i(t)$ denote the *set* of ongoing flows of class i at time t, i.e., flows which have arrived but have not completed service, and $\mathbf{q}(t) = (q_i(t) : i \in F)$. For each $A \subset F$, let $q_A(t) = \cup_{i\in A} q_i(t)$, i.e., the set of all active flows whose class



(a) A P2P system with m peers connected to each other via the Internet



(b) A centralized infrastructure with m dedicated servers connected to the network via a shared bottleneck

Fig. 1: Comparison between a P2P system and a centralized content delivery infrastructure.

is in A. Let $\mathbf{x}(t) = (x_i(t) : i \in F)$, where $x_i(t) \triangleq |q_i(t)|$, i.e., $\mathbf{x}(t)$ captures the *number of ongoing flows in each class*. We refer to $\mathbf{x}(t)$ as the state of the system at time t. Let $\mathbf{X}(t)$ correspond to the random vector describing the state of the system at time t.

For any $v \in q_i(t)$, let $b_v(t)$ be the rate in bits per second at which flow v is served at time t by the file-server system. At any time t, we assume that the rates $b_v(t)$ for all $v \in q_F(t)$ depend only on $\mathbf{x}(t)$ and the classes to which they belong. Thus for any $i \in F$ and $u, v \in q_i(t)$ we have $b_u(t) = b_v(t)$. Further, let $r_i(\mathbf{x}')$ be the total rate at which class i flows are served at time t when $\mathbf{x}(t) = \mathbf{x}'$, i.e., at any time t, $r_i(\mathbf{x}(t)) = \sum_{v \in q_i(t)} b_v(t)$. Let $\mathbf{r}(\mathbf{x}') = (r_i(\mathbf{x}') : i \in F)$. To visualize this system, think of the system as consisting of n queues, one corresponding to each file, with coupled service rates $\mathbf{r}(\mathbf{x}(t))$. Each queue in turn allocates its rate among its active users equally akin to processor sharing, i.e., $b_v(t) = r_i(\mathbf{x}(t))/x_i(t)$ for each $v \in q_i(t)$ if $x_i(t) \neq 0$. For any $\mathbf{x}(t)$, let $A_{\mathbf{x}(t)}$ denote the set of active classes, i.e., the classes with at least one ongoing flow. If flow v arrives at time t_v^a and has service requirement η_v, then it departs at time t_v^d such that $\eta_v = \int_{t_v^a}^{t_v^d} b_v(t)dt$.

Service Model: We consider a setting where files are stored on disk, as they may be too large and diverse to be held in main memory. However, they may be replicated across disks to enable high speed delivery by allowing multiple servers to work together as a pooled resource to serve download requests faster. Such a service model is reminiscent of service in P2P systems [37], [39] which consists of a set of users/peers connected through the Internet, collectively sharing their files/resources, see Fig. 1a. In this paper, however, we focus on modeling a centralized infrastructure aimed at serving large files very quickly. We abstract the maximum disk read capacity for each server $s \in S$ as server capacity μ_s, where S is the set of servers in a centralized infrastructure, see Fig. 1b.

Additionally, service may be constrained by a shared network bottleneck due to finite capacity of the link(s) which connect the servers S to the external network infrastructure, or by finite download speeds of the end users. We also study the impact of these bottlenecks in our upcoming works [27], [28]. For example, we show in [28] that for large systems with sufficient diversity in traffic and in the overlapping pools of servers if the shared network link capacity at the infrastructure is provisioned to be close to the average traffic demand, its impact on user performance becomes negligible as the system becomes large. Further, with increasing number of users being connected to a high speed broadband such as Google Fiber, the download capacity at the users may not be a significant bottleneck as compared to that of servers, especially when the load per server is large. We envisage a setting where the network is not the bottleneck, e.g., possibly through the use of multipath diversity and/or due to the availability of a high speed broadband, thus providing a user experience which is closer to or better than that of accessing data locally. Also see [9] for an overview on content delivery and network infrastructures, their interplay, and possibilities of joint optimization through collaboration between the two.

To abstract our service model, let $b_{v,s}(t)$ be the rate at which server s serves request v at time t. A request v for file i, i.e., $v \in q_i(t)$, can only be served by servers which have that file, thus $b_{v,s}(t) = 0$ if $s \notin S_i$, subject to the following assumption.

Assumption 1. *Sharing of system service capacity among ongoing flows is such that:*

1) *Each server can concurrently serve multiple requests as long as $\sum_v b_{v,s}(t) \leq \mu_s$ for all t.*
2) *Multiple servers can concurrently serve a request v at time t giving a total service rate $b_v(t) = \sum_s b_{v,s}(t)$.*
3) *The service rate $b_{v,s}(t)$ allocated to a flow v at server s at time t depends only on its flow's class and the numbers of ongoing flows $\mathbf{x}(t)$. Thus a flow's overall service rate $b_v(t)$ as well as the aggregate service rate allocated to flows in each class $\mathbf{r}(\mathbf{x}(t)) = (r_i(\mathbf{x}(t)) : i \in F)$ depend only on the number of ongoing flows.*

Note that service rate allocations depend only on the queue length $\mathbf{x}(t)$ and thus cannot depend on the residual file sizes of ongoing flows. This dependence will be made precise in the next section.

Under Assumption 1 we now show that the set of feasible service-rate allocations across classes, i.e., the *capacity region*, is a polymatroid. We say a polytope $\hat{\mathcal{C}}$ is a *polymatroid* if there exists a set function $\hat{\mu}$ on F such that

$$\hat{\mathcal{C}} = \left\{ \mathbf{r} \geq \mathbf{0} : \sum_{i \in A} r_i \leq \hat{\mu}(A),\ \forall A \subset F \right\},$$

and if $\hat{\mu}$ satisfies the following properties:
1) Normalized: $\hat{\mu}(\emptyset) = 0$.

2) Monotonic: if $A \subset B$, $\hat{\mu}(A) \leq \hat{\mu}(B)$.
3) Submodular: for all $A, B \subset F$,

$$\hat{\mu}(A) + \hat{\mu}(B) \geq \hat{\mu}(A \cup B) + \hat{\mu}(A \cap B).$$

A function $\hat{\mu}$ satisfying the above properties is called a *rank function*. Polymatroids and submodular functions are well studied in the literature, see e.g., [26]. Each polymatroid $\hat{\mathcal{C}}$ has a special property that for any $\mathbf{r} \in \hat{\mathcal{C}}$, there exists $\mathbf{r}' \geq \mathbf{r}$ such that $\mathbf{r}' \in \hat{\mathcal{D}} \triangleq \{\mathbf{r} \in \hat{\mathcal{C}} : \sum_{i \in F} r_i = \mu(F)\}$ [8]. Also, as evident from the definition, for any $A \subset F$ the set $\{\mathbf{r} \in \hat{\mathcal{C}} : r_i = 0, \forall i \notin A\}$ is a polymatroid, with a rank function which is the restriction of μ to subsets of A. A proof of the following theorem is provided in the Appendix.

Theorem 1. *Consider a file-server system defined by* $\big(F, S, \mu; \mathcal{S}\big)$ *and let*

$$\mathcal{C} \triangleq \{\mathbf{r} \geq \mathbf{0} : \sum_{i \in A} r_i \leq \mu(A),\ \forall A \subset F\}.$$

Then, the following hold
1) μ is a rank function.
2) Under Assumption 1, $\mathcal{C}$ is the polymatroid capacity region associated with the file server system.

We say that a polymatroid capacity region is *symmetric* if $\mu(A) = h(|A|)$ for any $A \subset F$ where $h : \mathbb{Z}_+ \to \mathbb{R}_+$ is a non-decreasing function, i.e., $\mu(A)$ depends on A only through $|A|$. Conversely, it is easy to show that if $\mu(A) = h(|A|)$ for some non-decreasing concave function $h : \mathbb{R}_+ \to \mathbb{R}_+$ with $h(0) = 0$, then the capacity region is a symmetric polymatroid.

III. Fairness Based Rate Allocation

There are several ways in which the capacity of a file-server system can be shared among a set of ongoing flows leading to potentially different user performance. For example, one may assign a *fixed* service capacity to each file to be exclusively shared by ongoing requests for that file. While this simplifies analysis by decoupling the dynamics across files, it results in wasted resources and poor performance. A better approach is to *dynamically* share service capacity across flow classes based on their load, e.g., queue lengths capturing the number of active flows.

Given the state $\mathbf{x}$ of the system at time t, one can consider allocating service capacity in various ways. For example, α-*fair rate allocation*, introduced in [24], allocates capacity based on maximizing a concave sum utility function subject to the system's capacity region. In our setting we can consider α-fair service rate allocation to flows subject to the the capacity region $\mathcal{C}$ given in Theorem 1. Formally, for any $\mathbf{x}$, the rate vector $\mathbf{r}(\mathbf{x})$ under α-fair allocation is given by

$$\mathbf{r}(\mathbf{x}) = \begin{cases} \arg\max_{\hat{\mathbf{r}} \in \mathcal{C}} \sum_{i \in F} \frac{x_i^\alpha \hat{r}_i^{1-\alpha}}{1-\alpha} & \text{for } \alpha \in (0, \infty) \backslash \{1\}, \\ \arg\max_{\hat{\mathbf{r}} \in \mathcal{C}} \sum_{i \in F} x_i \log(\hat{r}_i) & \text{for } \alpha = 1. \end{cases} \tag{1}$$

Note this generalizes various notions of fairness, e.g., *max-min fair* (MMF) and *proportional fair* (PF) allocations. Indeed PF and MMF are equivalent to α-fair policy for $\alpha = 1$ and $\alpha \to \infty$, respectively [24]. However, Theorem 2 below shows that on polymatroid capacity regions such allocations are equivalent.

Theorem 2. *All α-fair rate allocations are equivalent for polymatroid capacity regions.*

A proof is provided in the Appendix. Note that while this is clear for a single server system where α-fair allocations reduce to equal share, it may, at the first sight, be surprising in the multidimensional setting. Unfortunately, this does not characterize the performance users would see in a stochastic system and such results have been quite limited. What has been shown is that for such allocations, the performance is *sensitive* to the distribution of service requirements [4]. Thus, it is hard to make useful general claims.

By contrast, the *balanced fair* (BF) allocations introduced in [4] are 'insensitive', i.e., performance depends on the service distribution only through its mean. Moreover, BF has close structural relationship with proportional fairness, see e.g., [4], [21]. Additionally [3] studies several networks and shows a remarkable closeness in performance for balanced and proportional fairness motivating the use of BF as a mathematical tool for performance evaluation of stochastic networks under PF allocations.

Let us define BF rate allocation for our file-server system. Balanced fair rate allocation [4] for a polymatroid capacity region $\mathcal{C}$ can be defined as the service rate allocation $\mathbf{r}(\mathbf{x})$, where for any $\mathbf{x}$,

$$r_i(\mathbf{x}) = \frac{\Phi(\mathbf{x} - \mathbf{e}_i)}{\Phi(\mathbf{x})},\ \forall i \in F \tag{2}$$

where function Φ is called a balance function and is defined recursively as follows: $\Phi(\mathbf{0}) = 1$, and $\Phi(\mathbf{x}) = 0\ \forall \mathbf{x}$ s.t. $x_i < 0$ for some i, otherwise,

$$\Phi(\mathbf{x}) = \max_{A \subset F} \left\{ \frac{\sum_{i \in A} \Phi(\mathbf{x} - \mathbf{e}_i)}{\mu(A)} \right\}, \tag{3}$$

where $\mathbf{e}_i$ is a vector with 1 at i^{th} position and 0 elsewhere. As shown in [4], (2) ensures the important property of insensitivity, while (3) ensures that $\mathbf{r}(\mathbf{x})$ for each $\mathbf{x}$ lies in the capacity region, i.e., the constraints $\sum_{i \in A} r_i(\mathbf{x}) \leq \mu(A)$ are satisfied for each A. It also ensures that there exists a set $B \subset A_{\mathbf{x}}$ for which $\sum_{i \in B} r_i(\mathbf{x}) = \mu(B)$. In fact the BF allocation is the unique policy satisfying the above properties.

It was shown in [3], [4] that as long as the load vector lies in the interior of the capacity region, i.e., $\boldsymbol{\rho} \in \text{Interior}(\mathcal{C})$, the random process $(\mathbf{X}(t) : t \geq 0)$ is asymptotically stationary. Further, under this condition, its stationary distribution is given by

$$\pi(\mathbf{x}) = \frac{\Phi(\mathbf{x})}{G(\boldsymbol{\rho})} \prod_{i \in F} \rho_i^{x_i} \quad \text{where} \quad G(\boldsymbol{\rho}) = \sum_{\mathbf{x}'} \Phi(\mathbf{x}') \prod_{i \in F} \rho_i^{x_i'}.$$

An allocation of resources is said to be Pareto efficient if for any state $\mathbf{x}$, there does not exist an $\mathbf{r}' \in \mathcal{C}$ such that $r_i' \geq r_i(\mathbf{x}),\ \forall i \in A_{\mathbf{x}}$ with a strict inequality for at least one $i \in A_{\mathbf{x}}$. Pareto efficiency is a desirable property since it implies that the resource allocation is less wasteful. BF may not satisfy this property in general, e.g., see triangle networks studied in [4]. However, Theorem 3 below shows that BF is Pareto efficient

when the capacity region is a polymatroid. For a polymatroid capacity $\mathcal{C}$, showing Pareto efficiency is equivalent to showing $\sum_{i\in A_{\mathbf{x}}} r_i(\mathbf{x}) = \mu(A_{\mathbf{x}})$. A proof of the following theorem is provided in the Appendix.

Theorem 3. *For balanced fair rate allocations on polymatroid capacity regions we have $\sum_{i\in A_{\mathbf{x}}} r_i(\mathbf{x}) = \mu(A_{\mathbf{x}})$ for all $\mathbf{x}$.*

A similar result was proved in [5] for the special case of wireline networks with tree topology. Theorem 3 serves as a basis to obtain a recursive expression for the mean delay in our file-server system under BF rate allocation as given in the following theorem which is proven in the Appendix.

Theorem 4. *Consider a file-server system $(F, S, \mu; \mathcal{S})$ with load $\boldsymbol{\rho}$ and under balanced fair resource allocation. The mean delay for requests/flows of class i is given by*

$$E[D_i] = \frac{\nu_i \frac{\partial}{\partial \rho_i} G(\boldsymbol{\rho})}{G(\boldsymbol{\rho})} = \nu_i \frac{\partial}{\partial \rho_i} \log G(\boldsymbol{\rho}), \tag{4}$$

where $G(\boldsymbol{\rho})$ is given by,

$$G(\boldsymbol{\rho}) = \sum_{A\subset F} G_A(\boldsymbol{\rho}), \tag{5}$$

and where $G_\emptyset(\boldsymbol{\rho}) = 1$ and $G_A(\boldsymbol{\rho})$ can be computed recursively as

$$G_A(\boldsymbol{\rho}) = \frac{\sum_{i\in A} \rho_i G_{A\setminus\{i\}}(\boldsymbol{\rho})}{\mu(A) - \sum_{j\in A}\rho_j}. \tag{6}$$

Also, $\frac{\partial}{\partial \rho_i} G(\boldsymbol{\rho})$ can be recursively computed, without actually computing derivatives, as follows:

$$\frac{\partial}{\partial \rho_i} G(\boldsymbol{\rho}) = \sum_{A\subset F} \frac{\partial}{\partial \rho_i} G_A(\boldsymbol{\rho}), \tag{7}$$

where $\frac{\partial}{\partial \rho_i} G_\emptyset(\boldsymbol{\rho}) = 0$, and,

$$\frac{\partial}{\partial \rho_i} G_A(\boldsymbol{\rho}) = \frac{G_A(\boldsymbol{\rho}) + G_{A\setminus\{i\}}(\boldsymbol{\rho}) + \sum_{j\in A}\rho_j \frac{\partial}{\partial \rho_i} G_{A\setminus\{j\}}(\boldsymbol{\rho})}{\mu(A) - \sum_{j\in A}\rho_j}, \tag{8}$$

if $i \in A$ and 0 otherwise.

While the mean delay for systems with polymatroid capacity can be computed using (4) - (8), an exact computation has a complexity which grows exponentially in the number of files n. If, however, the capacity region is given by a symmetric polymatroid and the load vector $\boldsymbol{\rho}$ is homogenous, the complexity is linear in n. The following corollary, proved in the Appendix, details this result.

Corollary 1. *Consider a symmetric file-server system $(F, S, \mu; \mathcal{S})$ with homogenous load $\boldsymbol{\rho}$ and under balanced fair resource allocation, i.e., for each $A \subset F$, the rank function $\mu(A) = h(|A|)$ for some non-decreasing function $h : \mathbb{Z}_+ \to \mathbb{R}_+$ and for all $j \in F$ $\rho_j = \rho = \lambda\nu$. Then, the mean delay to serve the requests/flows of class i is given by,*

$$E[D_i] = \frac{\nu \hat{F}(\rho)}{F(\rho)}, \tag{9}$$

where, $F(\rho)$ and $\hat{F}(\rho)$ can be recursively obtained as follows:

$$F(\rho) = \sum_{k=0}^{n} F_k(\rho), \tag{10}$$

where, $F_0(\rho) = 1$, and for $k \geq 1$,

$$F_k(\rho) = \frac{(n-k+1)\rho F_{k-1}(\rho)}{h(k) - k\rho}. \tag{11}$$

Also,

$$\hat{F}(\rho) = \sum_{k=0}^{n} \frac{k}{n} \hat{F}_k(\rho), \tag{12}$$

where, $\hat{F}_0(\rho) = 0$, and for $k \geq 1$,

$$\hat{F}_k(\rho) = \frac{F_k(\rho) + \frac{n-k+1}{k} F_{k-1}(\rho) + \frac{(n-k+1)(k-1)}{k}\rho \hat{F}_{k-1}(\rho)}{h(k) - k\rho}. \tag{13}$$

IV. Large-Scale Asymptotics

In this section we consider asymptotics for large file-server systems wherein the number of files n and the number of servers m become large. Our focus is on systems where there is increased overall demand for increasingly diverse content, and thus one must scale server resources. The number of files in a content delivery infrastructure can be huge, e.g., a study in [38] estimated that Youtube had 5×10^8 videos in 2011, and the number has been steadily increasing since then. For now, we assume that the load across files is symmetric. We relax this assumption later.

Formally, consider a system with a given m and n. Let c copies of each file be placed independently and uniformly at random without replacement in c different servers. Let $\left(F^{(n)}, S^{(m)}, \mu^{(m,n)}; \mathcal{S}_c^{(m,n)}\right)$ represent a realization of such random file-server system. Further, let the $\mu_s^{(m,n)} = \xi$ for each server $s \in S^{(m)}$. Let the resulting capacity region be $\mathcal{C}^{(m,n)}$. Also, let the total request rate in the system be $m\lambda$, i.e., it grows linearly with m, resulting in a total traffic load $m\rho = m\lambda\nu$ where ν is the mean service requirement per request. Let the traffic load across files be symmetric, and thus equal to $\rho_i^{(m,n)} = m\rho/n$ for each file $i \in F^{(n)}$.

Further, we assume the number of files n to be orders of magnitude larger than m. To model this, we first fix m, and consider a sequence of systems wherein the number of files n increases to infinity. Then, to model the fact that m itself can be large, we consider a sequence of such sequences where m itself increases to infinity. This is a good model towards approximating systems with say $m \sim 10^3$, but with $n \sim 10^7$ or greater. For a given m and n, we let the total load on the system be ρm, with a fixed load per server ρ. Thus, for a given m, the load per file is equal to $\frac{\rho m}{n}$. As we will see in the sequel, this asymptotic regime is similar in spirit to that considered in the study of the super-market model [6], [22], [33].

For each realization, the service capacity is allocated dynamically according to balanced fair allocations over the associated capacity region, see Sec. III. We shall refer to the

file-server systems with resource allocation as described above as one with *Random Placement with Balanced Fairness (RP-BF)*.

A. Performance asymptotics for symmetric 'averaged' capacity region

For a given realization of the random file placement, the associated rank function $\mu^{(m,n)}$ need not be symmetric. Exact performance computations for such a system would require computation of the associated capacity region and evaluating the recursions developed in Sec. III both of which have exponential complexity in n. However, a key insight we develop below is that realizations of large RP-BF systems exhibit the same performance.

To that end consider the averaged RP-BF system having the "averaged capacity region". Let $M^{(m,n)}(.)$ denote the random rank function associated with an (m,n) RP-BF file placement. Given a set of files A where $|A| = k \leq n$ one can show that

$$\bar{\mu}^{(m,n)}(A) \triangleq E[M^{(m,n)}(A)] = \xi m(1-(1-c/m)^k).$$

Indeed the probability that none of the c copies of a file are stored on a given server is $(1-c/m)$. Thus the probability that none of A' s k files is stored at the server is $(1-c/m)^k$. So $m(1-(1-c/m)^k)$ is the mean number of servers that can serve *at least* one file in A, and the above is their associated service capacity. The averaged capacity region is thus given by a *symmetric* polymatroid with rank function $\bar{\mu}^{(m,n)}(A) = h^{(m,n)}(|A|)$ where

$$h^{(m,n)}(k) \triangleq \xi m(1-(1-c/m)^k) \text{ for } k = 0,1,\ldots,n. \quad (14)$$

Below we let $\pi^{(m,n)}(\mathbf{x})$ denote the stationary distribution of the queue length process for the averaged RP-BF system, i.e., using balanced fair allocations over the average capacity region. Also, let $E[D^{(m,n)}]$ be the expected delay for a typical request in this system. The following result gives a simple expression for the expected delay in the asymptotic regime of interest. Its proof is provided in the Appendix.

Theorem 5. *Consider a sequence of (m,n) averaged RP-BF file-server systems with symmetric polymatroid capacity with the rank function $\bar{\mu}^{(m,n)}(\cdot)$ given above and symmetric traffic load $\rho_i^{(m,n)} = m\rho/n$ for each file i where $\rho = \lambda\nu < \xi$. For given (m,n), let $\pi_k^{(m,n)} = \sum_{\mathbf{x}:|A_\mathbf{x}|=k} \pi^{(m,n)}(\mathbf{x})$ for $k = 0,1,2,\ldots,n$, and let*

$$\alpha^* \triangleq \frac{1}{c}\log\left(\frac{1}{1-\rho/\xi}\right). \quad (15)$$

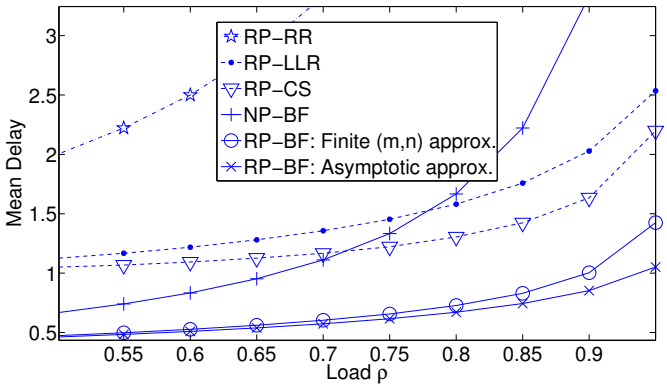
Then, for each $\epsilon > 0$, we have:

$$\lim_{m\to\infty}\lim_{n\to\infty} \sum_{k=\lfloor\alpha^* m(1-\epsilon)\rfloor}^{\lfloor\alpha^* m(1+\epsilon)\rfloor} \pi_k^{(m,n)} = 1 \quad (16)$$

Also, under the same limits, the expected delay is given by

$$\lim_{m\to\infty}\lim_{n\to\infty} E[D^{(m,n)}] = \frac{\alpha^*}{\lambda} = \frac{1}{\lambda c}\log\left(\frac{1}{1-\rho/\xi}\right). \quad (17)$$

The intuition underlying this result is as follows. For large systems, the probability measure $\pi^{(m,n)}(\mathbf{x})$ concentrates on states $\mathbf{x}$ such that $h^{(m,n)}(|A_\mathbf{x}|) \approx \rho m$. From (14), for any $\alpha > 0$, we have $\lim_{m\to\infty}\lim_{n\to\infty}\frac{1}{m}h^{(m,n)}(\alpha m) = \xi(1-e^{-c\alpha})$, which is equal to ρ for $\alpha = \alpha^*$.



(a) Mean delay comparison: $\nu_i = 1$ and $\rho_i = \rho m/n$ for each $i \in F$, $\xi = 1$, and $c = 3$. For finite (m,n) approximations: $m = 30$, $n = 2\times 10^4$.

File placement options	Service policies
RP: Randomized Placement	RR: Randomized Routing
NP: Non-overlapping Pools	LLR: Least-loaded Routing
	CS: Centralized Scheduling
	BF: Balanced Fairness

(b) Abbreviations

	Pooling of servers	Better load-balancing
RP-RR	×	×
RP-LLR	×	✓
RP-CS	×	✓
NP-BF	✓	×
RP-BF	✓	✓

(c) Qualitative comparision

Fig. 2: Comparision of different resource allocation policies.

Fig. 2 exhibits plots for mean delay as a function of load for *averaged* RP-BF systems. The plot for the approximation for a finite (m,n) system was computed using Corollary 1. The closeness of asymptotic expression to that for finite (m,n) depends on the value of ρ. Suppose n is orders of magnitude larger than m. For ρ less than or equal to 0.8 the asymptotic expression is remarkably close even for m as small as 30. Although not shown in the figure, for $\rho = 0.9$ the expression is close for m equal to 60 or larger. In next section we discuss why these expressions are good approximations for the actual performance in RP-BF realizations.

B. Approximating the performance of RP-BF file-server system via 'averaged' RP-BF.

In this subsection, we argue that the expression for mean delay given in Theorem 5 based on the averaged RP-BF system can be used to approximate the performance of realization of a large RP-BF file server system. In fact, we conjecture that the mean delay expression given in Theorem 5 holds for *almost all* sequences of RP-BF file placement realizations.

Recall that $M^{(m,n)}(.)$ denotes random rank function for our (m,n) RP-BF system, and $\bar{\mu}^{(m,n)}(.)$ its mean over all random file placements, and $\mu^{(m,n)}(.)$ denotes a (likely asymmetric) realization of $M^{(m,n)}(.)$. Our informal argument involves two steps.

Step 1: For large set of files A such that $|A| \approx \alpha m$ (integer) we have that

$$\frac{1}{m}\mu^{(m,n)}(A) \approx \frac{1}{m}h^{(m,n)}_{\text{avg}}(|A|),$$

where

$$h^{(m,n)}_{\text{avg}}(|A|) \triangleq \frac{\sum_{B:|B|=\alpha m}\mu^{(m,n)}(B)}{\binom{n}{\alpha m}}$$

This results from a general concentration property for c-Lipschitz monotonic submodular functions [32].

Step 2: With high probability, for most sets A such that $|A| = \alpha m$, we have

$$\frac{1}{m}\mu^{(m,n)}(A) \approx \frac{1}{m}\bar{\mu}^{(m,n)}(A) = \frac{1}{m}h^{(m,n)}(\alpha m),$$

where $h^{(m,n)}(.)$ is given by (14). This can be shown as follows.

Recall that $M^{(m,n)}(A) = \xi\sum_{s\in S^{(m)}}\mathbf{1}_{\{s\in S^{(m,n)}(A)\}}$, where $S^{(m)}$ and $S^{(m,n)}(A)$ are respectively the set of m servers, and the (random) set of servers where a copy of at least one of the files in A is stored. Suppose, for each (m,n), a subset of files $A^{(m,n)}_{\alpha}$ is selected uniformly at random from all $A \subset F^{(m,n)}$ such that $|A| = \alpha m$. Suppose $S^{(m)} = \{s_1, s_2, \ldots, s_m\}$. Consider a random process

$$X^{(m,n)} = \left(X^{(m,n)}_1, X^{(m,n)}_2, \ldots, X^{(m,n)}_m\right)$$

where

$$X^{(m,n)}_i = \mathbf{1}_{\left\{s_i\in S^{(m,n)}\left(A^{(m,n)}_{\alpha}\right)\right\}},\ \forall i \le m.$$

Then,

$$M^{(m,n)}\left(A^{(m,n)}_{\alpha}\right) = \xi\sum_{i=1}^{m}X^{(m,n)}_i.$$

We now study $\lim_{m\to\infty}\lim_{n\to\infty}\frac{1}{m}M^{(m,n)}\left(A^{(m,n)}_{\alpha}\right)$.

It can be checked that for each n, $X^{(m,n)}$ is a process of m exchangeable Bernoulli$(1-(1-c/m)^{\alpha m})$ random variables, and so is $X^{(m,\infty)} \triangleq \lim_{n\to\infty}X^{(m,n)}$. Also, for any fixed set of l servers, say $\{s_1, s_2, \ldots, s_l\}$, $X^{(m,\infty)}_i$ for $i \in \{1,2,\ldots,l\}$ can be shown to become independent in the limit as $m\to\infty$. As was shown in [1], [30], such asymptotic independence implies that a law of large numbers would hold for a sequence of exchangeable random processes which for our case implies that $\lim_{m\to\infty}\frac{1}{m}\sum_{i=1}^{m}X^{(m,\infty)}_i = 1-e^{-\alpha c}$ in probability. This shows that for most realizations, $\frac{1}{m}\mu^{(m,n)}\left(A^{(m,n)}_{\alpha}\right) \approx \frac{1}{m}h^{(m,n)}(\alpha m)$ for almost all sets A of size αm, thus showing the claim in Step 2.

Step 1 and Step 2 jointly imply that for each A such that $|A| \approx \alpha m$,

$$\frac{1}{m}\mu^{(m,n)}(A) \approx \frac{1}{m}h^{(m,n)}_{\text{avg}}(\alpha m) \approx \frac{1}{m}h^{(m,n)}(\alpha m),$$

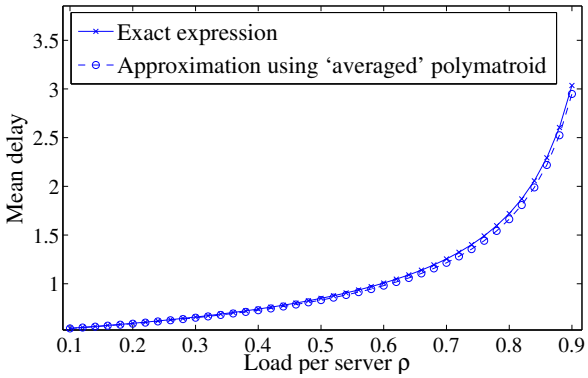


Fig. 3: Approximating performance of a file server system by using the 'averaged' polymatroid capacity: $m=4$, $n=6$, service rate $\mu_s = 1$ for each server s, $\rho_i = m\rho/n$ and $\nu_i = 1$ for each class i.

which further suggests that Theorem 5 holds for almost all file placement realizations of RP-BF systems.

Note that, for a given realization, there might still be few sets A of large enough size such that $\mu^{(m,n)}(A)$ is not close to $h^{(m,n)}(|A|)$. For example, consider set A of size m/c where each file in A is stored in disjoint set of servers. Here, $\mu(A) = m$ and is not close to $h^{(m,n)}(m/c)$. The above argument only shows that such outliers are small in number. A more rigorous argument is needed to show that the small number of outliers do not impact the overall performance a lot. We defer such analysis to a possible future work.

Let us numerically check the goodness of the approximation using an 'averaged' polymatroid capacity for a file-server system with $m=4$ servers and $n=6$ files, with each file stored on a distinct set of $c=2$ servers. The mean delay in such a system can be shown to be equivalent to a system with $m=4$ severs and number of files $n\to\infty$, as follows. A system with $m=4$ servers has $\binom{m}{c}=6$ distinct server-pools. For a given set of servers, one may view the group of files stored on each of them a distinct file-class. Since the files are distributed randomly, the load across these file-classes (equivalently server-pools) becomes homogeneous asymptotically.

Note, however, the rank function $\mu^{(4,6)}(.)$ is asymmetric. For example, $\mu^{(4,6)}(A)$ takes values 3 or 4 for different sets A of size 2, which is a difference of about 30%. We numerically compute $\mu^{(4,6)}(A)$ for each of the 2^6 subsets A of F, as well as an 'averaged' capacity region with the associated 'averaged' rank function $\mu^{(4,6)}_{\text{avg}}(A) = h^{(4,6)}_{\text{avg}}(|A|)$ for each $A\subset F$, where $h^{(4,6)}_{\text{avg}}(k) = \frac{\sum_{A:|A|=k}\mu^{(4,6)}(A)}{\binom{n}{k}}$ for $k=0,1,\ldots,6$. Fig. 3 exhibits the exact performance for both capacity regions using Theorem 4 and Corollary 1. It can be seen that the exact and the averaged systems are remarkably close.

C. Heterogeneity in Demand

The heterogeneity in load across files may not be seen at all the elements of a content delivery infrastructure. For example, if a centralized infrastructure is being used on the back end

to deliver files that are not available at distributed sites, i.e., requests correspond to 'cache misses', then the heterogeneity in demand might be less pronounced. The Poisson assumption on such demands might be justified when there are large numbers of files and the misses are relatively rare.

We now show that our model above involving a large number of files which are stored at a random set of c servers may also handle a limited variability/heterogeneity in demand. Consider a groups of files which are stored at the *same* c servers – if such groups are sufficiently large the overall load per group would be roughly the same. In fact as the number of files n increases the load for groups of files sharing the same c servers would become homogenous.

Thus, in the limit, we get $\binom{m}{c}$ file-groups with symmetric traffic load. Such groupings can now take the place of files in our original model, with homogeneous loads, and mean service required averaged across the files in the group. Note that our mean delay results are insensitive, i.e, they depend only on the mean service requirement. As m becomes large, one can thus show that our asymptotic analysis for an averaged capacity region would still hold.

V. Performance Comparison

We now compare RP-BF with several other resource allocation policies. For a given set of files and servers, the key components of a resource allocation policy that impact user-performance are the following:

1) **File placement**: Options include: (a) partitioning the set of servers and constraining each partition to store a distinct set of files, thus creating independent 'non-overlapping' pools of servers; (here, by pools of servers we mean the subsets of servers which can jointly serve file requests due to common files they store); and (b) randomly storing files across the servers, resulting into overlapping pools of servers. Option (a) was proposed in [7] as having a desirable property of higher reliability against correlated failures. We will explore this further in Section VI-A as well. Option (b), as we will see below, opens opportunities to better balance the load across servers and improve performance.

2) **Service policy**: A naive service policy is to route a file request randomly upon arrival to one of the servers that stores the corresponding file. The requests thus get queued at the servers and are served in, e.g., round-robin or processor sharing fashion. A simple modification to this policy which makes routing a function of the current load at servers, e.g., the number of queued requests at the servers, can provide significant performance improvement [6], [33]. An even better approach is that considered in [19], [31] where the requests are queued centrally and their service is scheduled dynamically based upon the availability of the servers. In each of these policies, a request is constrained to be served by a single server. Our work departs from these approaches, in that we allow each request to be served jointly by a pool of servers. As explained in Section II, we constrain service only through Assumption 1, or equivalently through capacity region $\mathcal{C}^{(m,n)}$. Under these constraints, we balance the load across servers through a fairness based rate allocation as explained in Section III.

We now compare four different resource allocation policies with RP-BF, each of which is characterized by a choice of file placement and of service policy.

Randomized Placement with Random Routing (RP-RR): Files are stored uniformly at random in c servers as with RP-BF. Upon arrival of a file request, it is randomly routed to one of the c servers that stores the corresponding file. Each server serves its request in processor sharing fashion. As $n \to \infty$, the total load of ρm is eventually balanced across the m servers and the system is equivalent to m independent $M/GI/1$ systems with load ρ and service rate ξ.

Random Placement with Least-loaded Routing (RP-LLR): Files are stored uniformly at random. Upon arrival, requests are routed to a server with least number of ongoing jobs among c servers which store the corresponding file. Each server serves its request in a processor sharing fashion. In the limit as $n \to \infty$, this system is equivalent to the super-market model studied in [6], [33]. Let p_k be the fraction of servers having k waiting requests in equilibrium. When the service-requirement distribution for each request is exponential, it was shown in [33] that as the number of servers $m \to \infty$, the fraction p_k is given by

$$p_k = (\rho/\xi)^{\frac{c^k-1}{c-1}} - (\rho/\xi)^{\frac{c^{k+1}-1}{c-1}},$$

where ρ is the load per server. Thus, by Little's law, the mean delay for a typical request in the asymptotic regime of interest is given by,

$$E[D_{\text{RP-LLR}}] = \frac{1}{\lambda}\sum_{k=1}^{\infty} k p_k = \frac{1}{\lambda}\sum_{k=1}^{\infty} (\rho/\xi)^{\frac{c^k-1}{c-1}}. \qquad (18)$$

Random Placement with Centralized Scheduling (RP-CS): Files are stored uniformly at random. Unlike the previous policies each server serves a maximum of one request at a time, and there is no service preemption. Upon arrival of a request, if there exist idle servers which store a copy of the corresponding file, it is assigned and served by one of them at random, else, it is queued at a central queue. Upon completion of service of a request at a server, if there exists a waiting request which the server can serve, it gets assigned to that server. If there exist multiple such requests, the choice is made as follows. Among all the files which the available server stores, one of the files with maximum number of waiting requests is chosen at random. Among the waiting requests of the chosen file, a request is chosen at random for service.

Non-overlapping Pools with Balanced Fairness (NP-BF): The m servers are divided into m/c groups, each of size c. Each server group stores a mutually exclusive subset with nc/m files. Within a group, each server stores the same set of files. Each file is thus stored at c servers. Under balanced fairness, each group behaves as an independent pool of servers which serves its requests in processor sharing fashion. The system is equivalent to m/c independent $M/GI/1$ queues with load ρc and service rate ξc, with mean delay given by

$$E[D_{\text{NP-BF}}] = \frac{\nu}{c\xi(1-\rho/\xi)}. \qquad (19)$$

Contrast this with Theorem 5 where the mean delay increase is logarithmic in $1/(1-\rho/\xi)$.

In Fig. 2, we compare the performance of these resource allocation policies. RP-BF's performance is plotted using the approximations described in Section IV. The performance of RP-RR, RP-LLR and NP-BF is plotted using corresponding asymptotic expressions for mean delay described above. For RP-CS, the service requirement distribution was assumed exponential and we built a simulator for the underlying Markov Chain. For each point in the plot, the average number of requests waiting in the queue or in service was measured over a period of time of up to 10^6 events and the mean delay was computed using Little's law.

All the above policies are stable for any value of ρ less than 1. As expected, RP-RR performs poorly as it does not exploit pooling or load dependent routing. RP-CS outperforms RP-LLR at higher loads since requests are queued centrally in the former and its service policy uses global state information. NP-BF outperforms both RP-CS and RP-LLR at lower loads since pooling of servers works to its advantage. However, due to creation of independent non-overlapping pools, its ability to balance the load across servers is limited and it performs significantly worse at higher loads.

RP-BF outperforms all of the policies since it enjoys the best of both worlds. At higher loads, one might expect that the gains of RP-BF over RP-LLR and RP-CS due to pooling may be limited since load balancing of the later policies would ensure that most of the servers are busy serving requests most of the time and are utilized well. However, even for $\rho = 0.9$, the mean delay for RP-LLR and RP-CS is over 2 and 1.6 times that of RP-BF for $c = 3$, respectively.

For larger values of c, the improvements are even greater. For any value of c, mean delay for RP-LLR and RP-CS is lower bounded by 1. However, from Theorem 5, mean delay for RP-BF is inversely proportional to c. The significant performance improvement by RP-BF shows that server pooling and fairness based resource allocation is worthwhile towards optimizing the performance of file-server systems.

VI. System Tradeoffs

A. Recovery costs on correlated failure v/s performance

We consider the cost of recovering files when there are large-scale correlated failures such as those occurring after power outages, see [7] for an extensive discussion. It is not uncommon in datacenters that about 1% of servers fail to reboot after a power outage. The system then needs to recover data in these servers by retrieving copies from the servers that successfully rebooted. However, there might be some files for which no copy exists in the datacenter due to the failure of all servers in which it was stored. The probability of such an event occurring can be significant especially when the total number of files in the system is large.

When this occurs the system needs to locate and recover the lost files from 'cold' storage. Recovery of the files from cold storage may incur a high fixed cost but may not be greatly affected by the number of files lost. Thus in practice (as argued in [7]) it is desirable that the probability that one or more files are lost during power outage events be low. This can be achieved by constraining randomness in how files are copied across servers. The intuition from Section V suggests that randomly 'spreading' the files across the servers so that the server pools overlap improves the user perceived performance. However, this may increase the probability of a file loss. To study how these quantities are related, we consider a storage policy that divides m servers into independent groups of smaller size and restricts the copies of each file to be placed within a single group, as follows.

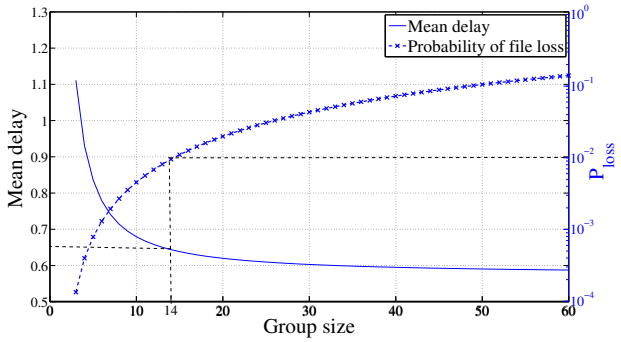


Fig. 4: Delay v/s reliability $n = 2 \times 10^6$, $m = 400$, $c = 3$, $\gamma = 0.01$, $\rho = 0.7$, and $\nu = 1$.

Fix an integer κ such that $c \leq \kappa \leq m$. Suppose, for now, that number of servers m is divisible by κ and that number of files n is divisible by m/κ. Divide the set S of m servers into m/κ number of groups each of size κ. Similarly, divide the set F of n files into disjoint m/κ groups of size $n\kappa/m$. Associate each group of files with a distinct group of servers. Then, for each file, independently store c copies by selecting c servers uniformly at random from the corresponding group.

Suppose that upon a power outage, each server fails to reboot with probability γ independently. Then, for a group of size κ, the probability that l servers fail is $\binom{\kappa}{l}\gamma^l(1-\gamma)^{\kappa-l}$, so the probability that one or more files are lost can be given by

$$P_{\text{loss}} = 1 - \left(\sum_{l=0}^{c-1} \binom{\kappa}{l} \gamma^l (1-\gamma)^{\kappa-l} + \sum_{l=c}^{\kappa} \binom{\kappa}{l} \gamma^l (1-\gamma)^{\kappa-l} \left(1 - \frac{\binom{l}{c}}{\binom{\kappa}{c}} \right)^{n\kappa/m} \right)^{m/\kappa}$$

For the general case where m is not divisible by κ or n is not divisible by m/κ, we can create non-uniform groups and compute the corresponding loss probability. We use the above expression as a simpler approximation by using $\lfloor m/\kappa \rfloor$ and $\lfloor n\kappa/m \rfloor$ appropriately. Also, the performance within each group can be computed using the expression of Corollary 1 for symmetric capacity systems, which gives a reasonable approximation as explained in Sec. IV-A.

Fig. 4 exhibits the mean delay and P_{loss} for $\gamma = 0.01$ for a system with $n = 2 \times 10^6$, $m = 400$, and $c = 3$ copies. The load per server is $\rho = 0.7$, i.e., the total load on the system is $m\rho = 280$ and is distributed uniformly across files. Also, $\nu_i = 1$ for all $i \in F$ and $\mu_s = 1$ for all $s \in S$. As can be seen, varying κ trades off performance with file loss

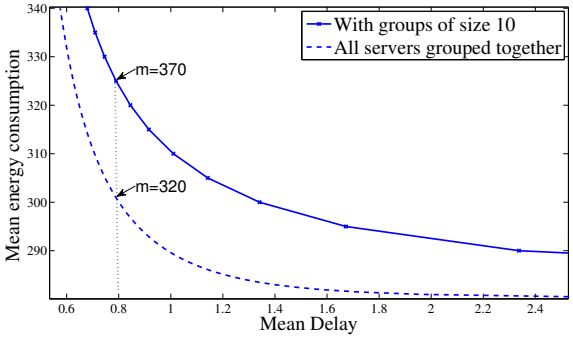


Fig. 5: Energy-delay tradeoff for system with $n = 2 \times 10^6$ and varying m: $\nu = 1$, $c = 3$, $\xi = 1$, and total load $\rho m = 280$.

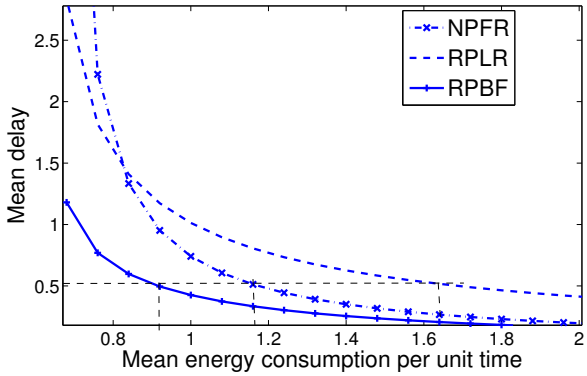


Fig. 6: Energy-delay tradeoff with varying server speed ξ: load per server fixed at $\rho = 0.8$, $\nu = 1$, and $c = 3$.

probability. As κ increases mean delay decreases but quickly saturates at 0.57, which matches with the asymptotic limit as given by Theorem 5. At $\kappa = 14$, mean delay is 0.64 which is about 12% greater than the asymptotic value, while P_{loss} is less than 1%. Decreasing κ can further lower P_{loss} but at the cost of a significant increase in mean delay.

B. Energy-delay tradeoffs

We now consider RP-BF systems where for each server $s \in S$, we have $\mu_s = \xi$. Energy consumption per unit time by a server is fixed when it is busy and is denoted by e_b. Similarly, even when a server is idle, its energy consumption per unit time is fixed and denoted by e_i. If the system is stable, the sum of the fraction of time each server is busy is equal to $\frac{\sum_{i \in F} \rho_i}{\xi}$. Thus, the mean energy spent by the system per unit time is given by

$$E = e_b \frac{\sum_{i \in F} \rho_i}{\xi} + e_i \left(m - \frac{\sum_{i \in F} \rho_i}{\xi} \right).$$

Thus, one can trade of energy consumption for performance by varying m.

Fig. 5 exhibits the energy-delay curve for a system with 2×10^6 files with a fixed total load of 280, $e_b = 1$ units and $e_i = 0.5$ units. Points in the plot are obtained by varying m and computing the performance using Corollary 1. The figure also exhibits tradeoff for the case when the total number of servers are divided into smaller independent groups of size 10, as in Section VI-A. The tradeoff curve worsens in this case. For example, to obtain a mean delay of 0.8, it requires $m = 370$ servers while the former system that groups all the servers together requires 320 servers; the corresponding mean energy consumption being 325 units and 300 units, respectively. Thus, creating smaller independent groups of size 10 increases the energy consumption by about 8%.

Next, we consider RP-BF systems where servers' processing speed is a bottleneck. The processing speed can be improved by increasing clock frequency and voltage supply, which in turn increases energy consumption. This dependence is typically modeled through a polynomial relationship of power with ξ, i.e., when the service rate of a server is ξ the power consumption is given by $f(\xi) = \xi^\alpha/\beta$ per unit time where $\alpha > 1$ and β is a positive constant [18]. In practice, even when ξ is set to 0, there is non-negligible leakage power consumption. Since our focus is on dynamic power, we ignore leakage power here. The choice of ξ trades off performance for energy consumption. Here, we consider a simple semi-static policy where each server operates at a fixed rate ξ when busy and rate 0 when idle, thus consuming negligible power when idle. For $M/GI/1$ queues, it was shown in [18] that such a simple policy, with ξ chosen judiciously, is close to an optimal policy for minimizing a weighted average of the mean delay and energy consumption across all dynamic policies where ξ is allowed to vary with the queue state.

Fig. 6 compares the energy-performance tradeoff for NP-BF, RP-LLR, and RP-BF where the plots are obtained by varying values of ξ. For RP-BF, Theorem 5 is used to compute dependence of performance on ξ, whereas for NP-BF and RP-LLR, (19) and (18), respectively, are used. Also, we assume that the power consumption as a function of ξ is given by $f(\xi) = \xi^2$. Since the fraction of time a server is busy in each system is ρ/ξ, the mean energy consumption is given by $E = \rho\xi$. To obtain a mean delay of 0.5 for $\rho = 0.8$, the energy consumption for NP-BF and RP-LLR systems is 20% and 70% more than that for RP-BF, respectively.

VII. Conclusions

Service models which allow pooling of servers to serve dynamically arriving file download requests provide favorable performance properties such as insensitivity to the distribution of service requirement and inverse relation of mean delay to the number of stored copies for each file, or equivalently, the size of pools of servers. Further, if file-placement across servers is designed such that there is enough overlap across these pools then fairness based load balancing policy mitigates the impact of server utilization on mean delay. Overall, the gains of such resource allocation over those which do not jointly exploit pooling and dynamic load balancing across servers are significant.

This work represents a first step towards developing the performance models needed for a disciplined engineering and optimization of such systems.

REFERENCES

[1] D. J. Aldous. *Exchangeability and related topics*. Springer, 1985.

[2] T. Bonald and L. Massoulie. Impact of fairness on internet performance. In *Proceedings of ACM Sigmetrics*, pages 82–91, 2001.

[3] T. Bonald, L. Massoulié, A. Proutiere, and J. Virtamo. A queueing analysis of max-min fairness, proportional fairness and balanced fairness. queueing systems: Theory and applications. *Queueing Systems*, 53:65–84, 2006.

[4] T. Bonald and A. Proutiere. Insensitive bandwidth sharing in data networks. *Queueing Systems*, 44:69–100, 2003.

[5] T. Bonald and J. Virtamo. Calculating the flow level performance of balanced fairness in tree networks. *Perform. Eval.*, 58(1):1–14, Oct. 2004.

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

[7] A. Cidon, S. Rumble, R. Stutsman, S. Katti, J. Ousterhout, and M. Rosenblum. Copysets: Reducing the frequency of data loss in cloud storage. In *Usenix Advanced Technical Conference*, 2013.

[8] J. Edmonds. Submodular functions, matroids, and certain polyhedra. In *Proceedings of Calgary International Conference on Combinatorial Structures and Applications*, pages 69–87, 1969.

[9] B. Frank, I. Poese, G. Smaragdakis, A. Feldmann, B. M. Maggs, S. Uhlig, V. Aggarwal, and F. Schneider. Collaboration opportunities for content delivery and network infrastructures. In H. Haddadi and O. Bonaventure, editors, *Recent Advances in Networking*, pages 305–377. 2013.

[10] A. Gandhi, V. Gupta, M. Harchol-Balter, and M. A. Kozuch. Optimality analysis of energy-performance trade-off for server farm management. *Perform. Eval.*, 67(11):1155–1171, Nov. 2010.

[11] H. Han, S. Shakkottai, C. V. Hollot, R. Srikant, and D. Towsley. Multi-path tcp: a joint congestion control and routing scheme to exploit path diversity in the internet. *IEEE/ACM Trans. Netw.*, 14(6):1260–1271, Dec. 2006.

[12] V. Joseph and G. de Veciana. Stochastic networks with multipath flow control: Impact of resource pools on flow-level performance and network congestion. In *Proceedings of the ACM Sigmetrics*, pages 61–72, 2011.

[13] F. Kelly, L. Massoulie, and N. Walton. Resource pooling in congested networks: proportional fairness and product form. *Queueing Systems*, 63(1-4):165–194, 2009.

[14] P. Key, L. Massoulié, and D. Towsley. Path selection and multipath congestion control. *Commun. ACM*, 54(1):109–116, Jan. 2011.

[15] T. Lan, D. Kao, M. Chiang, and A. Sabharwal. An axiomatic theory of fairness in network resource allocation. In *Proceedings of IEEE Infocom*, pages 1–9, March 2010.

[16] M. Leconte, M. Lelarge, and L. Massoulié. Bipartite graph structures for efficient balancing of heterogeneous loads. In *Proceedings of ACM Sigmetrics/Performance*, pages 41–52, 2012.

[17] M. Leconte, M. Lelarge, and L. Massoulié. Adaptive replication in distributed content delivery networks. *arXiv preprint arXiv:1401.1770*, 2014.

[18] M. Lin, A. Wierman, L. Andrew, and E. Thereska. Dynamic right-sizing for power-proportional data centers. In *Proceedings of IEEE Infocom*, pages 1098–1106, 2011.

[19] A. Mandelbaum and A. L. Stolyar. Scheduling flexible servers with convex delay costs: Heavy-traffic optimality of the generalized cμ-rule. *Operations Research*, 52(6):836–855, 2004.

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

[21] L. Massoulié. Structural properties of proportional fairness: Stability and insensitivity. *Annals of Applied Probability*, 17(3):809–839, 2007.

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

[23] M. D. Mitzenmacher, A. W. Richa, and R. Sitaraman. The power of two random choices: A survey of techniques and results. In P. Pardalos, S. Rajasekaran, and J. Rolim, editors, *in Handbook of Randomized Computing*, pages 255–312. Springer US, 2001.

[24] J. Mo and J. Walrand. Fair end-to-end window-based congestion control. *IEEE/ACM Transactions on Networking*, 8(5):556 –567, Oct. 2000.

[25] S. Moharir, J. Ghaderi, S. Sanghavi, and S. Shakkottai. Serving content with unknown demand: The high-dimensional regime. In *Proceedings of ACM Sigmetrics*, pages 435–447, 2014.

[26] G. L. Nemhauser and L. A. Wolsey. *Integer and combinatorial optimization*, volume 18. Wiley New York, 1988.

[27] V. Shah. *Centralized Content Delivery Infrastructure Exploiting Resource Pools: Performance Models and Asymptotics*. PhD thesis, The University of Texas at Austin, 2015.

[28] V. Shah and G. de Veciana. Asymptotic independence of servers' activity in queuing systems with limited resource pooling. *In preparation*.

[29] A. Stolyar. An infinite server system with customer-to-server packing constraints. In *Proceedings of Allerton Conference*, 2012.

[30] A. S. Sznitman. Topics in propagation of chaos. In *Ecole d'Eté de Probabilités de Saint-Flour XIX1989*, pages 165–251. Springer, 1991.

[31] J. N. Tsitsiklis and K. Xu. Queueing system topologies with limited flexibility. In *Proceedings of ACM Sigmetrics*, pages 167–178, 2013.

[32] J. Vondrák. A note on concentration of submodular functions. *arXiv preprint arXiv:1005.2791*, 2010.

[33] 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*, 32(1):20–34, 1996.

[34] A. Wierman, L. L. H. Andrew, and A. Tang. Power-aware speed scaling in processor sharing systems: Optimality and robustness. *Perform. Eval.*, 69(12):601–622, Dec. 2012.

[35] D. Wischik, M. Handley, and M. B. Braun. The resource pooling principle. *SIGCOMM Comput. Commun. Rev.*, 38(5):47–52, Sept. 2008.

[36] K. Xu. *On the power of (even a little) flexibility in dynamic resource allocation*. PhD thesis, Massachusetts Institute of Technology, 2014.

[37] X. Yang and G. de Veciana. Performance of peer-to-peer networks: Service capacity and role of resource sharing policies. *Perform. Eval.*, 63(3):175–194, Mar. 2006.

[38] J. Zhou, Y. Li, V. K. Adhikari, and Z.-L. Zhang. Counting youtube videos via random prefix sampling. In *Proceedings of ACM Sigcomm Conference on Internet Measurement Conference*, pages 371–380, 2011.

[39] Y. Zhou, T. Fu, and D. M. Chiu. A unifying model and analysis of P2P VoD replication and scheduling. In *Proceedings of IEEE Infocom*, pages 1530–1538, March 2012.

APPENDIX

A. Proof of Theorem 1

We first show that μ is a rank function. By definition it is clear that $\mu(\emptyset) = 0$ and that μ is monotonic. To show that $\mu(.)$ is submodular we use the inclusion-exclusion principle to obtain

$$\begin{aligned}\mu(A) &= \sum_{s\in S(A)} \mu_s = \sum_{s\in S(A\cap B)\cup S(A\setminus B)} \mu_s \\ &= \sum_{s\in S(A\cap B)} \mu_s + \sum_{s\in S(A\setminus B)} \mu_s - \sum_{s\in S(A\cap B)\cap S(A\setminus B)} \mu_s.\end{aligned}$$

Similarly,

$$\mu(B) = \sum_{s\in S(B\cap A)} \mu_s + \sum_{s\in S(B\setminus A)} \mu_s - \sum_{s\in S(B\cap A)\cap S(B\setminus A)} \mu_s$$

Again using inclusion-exclusion principle, we further have,

$$\begin{aligned}\mu(A\cup B) &= \sum_{s\in S(A\cup B)} \mu_s = \sum_{s\in S(A\cap B)\cup S(A\setminus B)\cup S(B\setminus A)} \mu_s \\ &= \sum_{s\in S(A\cap B)} \mu_s + \sum_{s\in S(A\setminus B)} \mu_s + \sum_{s\in S(B\setminus A)} \mu_s \\ &- \sum_{s\in S(A\cap B)\cap S(A\setminus B)} \mu_s - \sum_{s\in S(B\cap A)\cap S(B\setminus A)} \mu_s \\ &- \sum_{s\in S(A\setminus B)\cap S(B\setminus A)} \mu_s + \sum_{s\in S(B\cap A)\cap S(A\setminus B)\cap S(B\setminus A)} \mu_s\end{aligned}$$

Also, $\mu(A\cap B) = \sum_{s\in S(A\cap B)} \mu_s$. Thus,

$$\begin{aligned}&\mu(A) + \mu(B) - \mu(A\cup B) - \mu(A\cap B) \\ &\quad= \sum_{s\in S(A\setminus B)\cap S(B\setminus A)} \mu_s - \sum_{s\in S(B\cap A)\cap S(A\setminus B)\cap S(B\setminus A)} \mu_s \\ &\quad\geq 0\end{aligned}$$

which shows that μ is submodular.

We now show that $\mathcal{C}$ is the capacity region. We first show that if $\mathbf{r}$ is feasible then $\mathbf{r} \in \mathcal{C}$, and later show the converse.

Suppose $\mathbf{r} \notin \mathcal{C}$. Then, we show that $\mathbf{r}$ violates the capacity constraints in Assumption 1 for any set of active flows $\mathbf{q}$ such that for all i, $|q_i| > 0$ iff $r_i > 0$. By definition of $\mathcal{C}$, there exists $A \subset F$ such that $\sum_{i\in A} r_i > \mu(A)$. Now suppose $\sum_{v\in q_i, s\in S_i} b_{v,s} = r_i$ for all $i \in F$. Then, we get, $\sum_{i\in A}\sum_{v\in q_i, s\in S_i} b_{v,s} > \mu(A)$ which further gives $\sum_{s\in S(A)}\sum_{v\in \cup_{i\in A} q_i} b_{v,s} > \mu(A)$. Thus, there exists s such that $\sum_{v\in\cup_{i\in A} q_i} b_{v,s} > \mu_s$. Thus, $\mathbf{r}$ is not feasible.

We now show the converse, i.e., $\mathbf{r} \in \mathcal{C}$ implies that $\mathbf{r}$ is feasible. Recall that, for a polymatroid capacity $\mathcal{C}$, for all $\mathbf{r} \in \mathcal{C}$ there exists $\mathbf{r}' \geq \mathbf{r}$ such that $\mathbf{r}' \in \mathcal{D}$, where $\mathcal{D} = \{\mathbf{r} \in \mathcal{C} : \sum_{i\in F} r_i = \mu(F)\}$. Thus, it is sufficient to show that if $\mathbf{r} \in \mathcal{D}$, then $\mathbf{r}$ is feasible. Let P be set of all permutations on F. For each $p \in P$, let $\mathbf{r}^{(p)} = (r_i^{(p)} : i \in F)$ such that $r^{(p)}_{p(k)} = \mu(\{p(1),\ldots,p(k)\}) - \mu(\{p(1),\ldots,p(k-1)\})$, for all $k \in \{1,2,\ldots,n\}$. It can be shown that $\{\mathbf{r}^{(p)} : p \in P\}$ is the set of all extreme points of $\mathcal{D}$, see [8]. Thus, it is sufficient to show that $\mathbf{r}^{(p)}$ for each $p \in P$ is feasible. Remaining points can be obtained using time sharing over arbitrarily smaller time scale. For each s, find the smallest k such that $s \in S_{p(k)}$ and set $b_{(v,s)} = \mu_s/|q_{p(k)}|$ if $v \in q_{p(k)}$ and 0 otherwise, thus satisfying Assumption 1. Then, for each k, $\sum_{s\in S_{p(k)}} b_{(v,s)} = \mu(\{p(1),\ldots,p(k)\}) - \mu(\{p(1),\ldots,p(k-1)\}) = r^{(p)}_{p(k)}$. Thus, $\mathbf{r}^{(p)}$ is feasible.

B. Proof of Theorem 2

Clearly, for any α, α-fair rate allocations $\mathbf{r}(\mathbf{x})$ are Pareto efficient, i.e., for any state $\mathbf{x}$, there does not exist an $\mathbf{r}' \in \mathcal{C}$ such that $r'_i \geq r_i(\mathbf{x}),\ \forall i \in A_{\mathbf{x}}$ with a strict inequality for at least one $i \in A_{\mathbf{x}}$. Due to the existence of dominant face $\mathcal{D} = \{\mathbf{r} \in \mathcal{C} : \sum_{i\in F} r_i = \mu(F)\}$, α-fair rate allocation over capacity region $\mathcal{C}$ is equivalent to that over region $\mathcal{D}$.

We will show that α-fair rate allocations for any $\alpha \in (0,\infty)\backslash\{1\}$ are equivalent to Max-Min Fair (MMF) rate allocations. The result then follows immediately for $\alpha = 1$ as well since it is equivalent to the limiting α-fair allocation as $\alpha \to 1$.

Fix an $\alpha \in (0,\infty)\backslash\{1\}$. Without loss of generality, consider a state $\mathbf{x}$ such that $A_{\mathbf{x}} = F$. Consider the corresponding set of flows q_F. It is easy to show that an α-fair rate allocation over $\mathcal{D}$ is equivalent to assigning rates $(b_u : u \in q_F)$ as given by the unique solution to the following optimization problem:

$$\begin{aligned}
\text{maximize} \quad & \text{sign}(1-\alpha)\sum_{u\in q_F} \hat{b}_u^{1-\alpha} \\
\text{subject to} \quad & \sum_{u\in q_A} \hat{b}_u \leq \mu(A),\ \forall A \subset F \\
& \sum_{u\in q_F} \hat{b}_u = \mu(F) \\
& \hat{b}_u \geq 0,\ \forall u \in q_F
\end{aligned}$$

The objective function for the above problem is strictly concave, and thus Schur-concave, in $(\hat{b}_u : u \in q_F)$ [15], [20].

Now, suppose $(b_u : u \in q_F)$ is not max-min fair. Then, there exist flows u and v and a constant $\epsilon > 0$ such that $b_v \geq b_u$ and by increasing the rate of the flow u by ϵ and decreasing that of flow v by ϵ the feasibility for the above problem is not lost. However, due to Schur-concavity, this operation only increases the value of the objective function which contradicts with optimality and uniqueness of $(b_u : u \in q_F)$. Thus, $(b_u : u \in q_F)$ is max-min fair, and α-fair policy is equivalent to MMF.

C. Proof of Theorem 3

We prove this by induction on $|\mathbf{x}| \triangleq \sum_i x_i$. Clearly, the result is true when $|\mathbf{x}| = 1$. Lets assume that the claim is true for all $\mathbf{x}'$ such that $|\mathbf{x}'| < |\mathbf{x}|$ for a given $\mathbf{x}$. We show that it holds for $\mathbf{x}$ as well.

By definition of balanced fairness, i.e., by (2) and (3), there exists a B such that $\sum_{i\in B} r_i(\mathbf{x}) = \mu(B)$. Also, by monotonicity of $\mu(.)$, $B \subset A_{\mathbf{x}}$. If $B = A_{\mathbf{x}}$, then we are done. Suppose this is not the case. Then, from (2) and definition of B, we have

$$\Phi(\mathbf{x}) = \frac{\sum_{i\in B}\Phi(\mathbf{x}-\mathbf{e}_i)}{\mu(B)}. \tag{20}$$

Since the capacity condition $\sum_{i\in B} r_i(\mathbf{x}') \leq \mu(B)$ is satisfied for all states, we have $\sum_{i\in B} r_i(\mathbf{x}-\mathbf{e}_j) \leq \mu(B)$ for all $j \in A_{\mathbf{x}}\backslash B$. Using this in (20), we get

$$\Phi(\mathbf{x}) \leq \frac{\sum_{i\in B}\Phi(\mathbf{x}-\mathbf{e}_i)}{\sum_{i\in B} r_i(\mathbf{x}-\mathbf{e}_j)},\ \forall j \in A_{\mathbf{x}}\backslash B. \tag{21}$$

We now use this bound to compute one on the sum of all rates as follows:

$$\begin{aligned}
\sum_{i\in A_{\mathbf{x}}} r_i(\mathbf{x}) &= \sum_{i\in B} r_i(\mathbf{x}) + \sum_{j\in A_{\mathbf{x}}\backslash B} r_j(\mathbf{x}), \\
&= \mu(B) + \sum_{j\in A_{\mathbf{x}}\backslash B} \frac{\Phi(\mathbf{x}-\mathbf{e}_j)}{\Phi(\mathbf{x})}, \\
&\geq \mu(B) + \sum_{j\in A_{\mathbf{x}}\backslash B} \frac{\sum_{i\in B} r_i(\mathbf{x}-\mathbf{e}_j)\Phi(\mathbf{x}-\mathbf{e}_j)}{\sum_{i\in B}\Phi(\mathbf{x}-\mathbf{e}_i)}, \\
&= \mu(B) + \sum_{j\in A_{\mathbf{x}}\backslash B} \frac{\sum_{i\in B}\Phi(\mathbf{x}-\mathbf{e}_j-\mathbf{e}_i)}{\sum_{i\in B}\Phi(\mathbf{x}-\mathbf{e}_i)}, \\
&= \mu(B) + \frac{\sum_{i\in B}\sum_{j\in A_{\mathbf{x}}\backslash B}\Phi(\mathbf{x}-\mathbf{e}_j-\mathbf{e}_i)}{\sum_{i\in B}\Phi(\mathbf{x}-\mathbf{e}_i)}, \\
&\geq \mu(B) + \frac{\sum_{j\in A_{\mathbf{x}}\backslash B}\Phi(\mathbf{x}-\mathbf{e}_j-\mathbf{e}_{i^*})}{\Phi(\mathbf{x}-\mathbf{e}_{i^*})},
\end{aligned} \tag{22}$$

where $i^* = \arg\min_{i\in B}\left\{\frac{\sum_{j\in A_{\mathbf{x}}\backslash B}\Phi(\mathbf{x}-\mathbf{e}_j-\mathbf{e}_i)}{\Phi(\mathbf{x}-\mathbf{e}_i)}\right\}$. In the last inequality (22), we have used the identity $\frac{a+b}{c+d} \geq \frac{a}{c}$ if $\frac{a}{c} \leq \frac{b}{d}$. Thus, we get the following inequality.

$$\sum_{i\in A_{\mathbf{x}}} r_i(\mathbf{x}) \geq \mu(B) + \sum_{j\in A_{\mathbf{x}}\backslash B} r_j(\mathbf{x}-\mathbf{e}_{i^*}). \tag{23}$$

We now only need to show $\mu(B) + \sum_{j\in A_{\mathbf{x}}\backslash B} r_j(\mathbf{x}-\mathbf{e}_{i^*}) \geq \mu(A_{\mathbf{x}})$. The following two cases are possible for the given $\mathbf{x}$.

Case 1 $x_{i^*} = 1$: Then, in state $\mathbf{x} - \mathbf{e}_{i^*}$, only classes in $A_{\mathbf{x}} \backslash \{i^*\}$ are active. Thus, we have,

$$\begin{aligned}\sum_{j \in A_{\mathbf{x}} \backslash B} & r_j(\mathbf{x} - \mathbf{e}_{i^*}) + \mu(B) \\ &= \mu(A_{\mathbf{x}} \backslash \{i^*\}) - \sum_{k \in B \backslash \{i^*\}} r_k(\mathbf{x} - \mathbf{e}_{i^*}) + \mu(B), \\ &\geq \mu(A_{\mathbf{x}} \backslash \{i^*\}) - \mu(B \backslash \{i^*\}) + \mu(B), \\ &\geq \mu(A_{\mathbf{x}}),\end{aligned}$$

where the equality follows from induction hypothesis, the first inequality follows from the capacity constraint on set $B \backslash \{i^*\}$, and the last inequality follows from the submodularity of $\mu(.)$.

Case 2 $x_{i^*} > 1$: Here, all the classes in $A_{\mathbf{x}}$ are active in state $\mathbf{x} - \mathbf{e}_{i^*}$ as well, i.e., $A_{\mathbf{x}} = A_{\mathbf{x} - e_{i^*}}$. Thus, we have,

$$\begin{aligned}\sum_{j \in A_{\mathbf{x}} \backslash B} r_j(\mathbf{x} - \mathbf{e}_{i^*}) + \mu(B) &\geq \sum_{i \in A_{\mathbf{x}}} r_i(\mathbf{x} - \mathbf{e}_{i^*}) \\ &= \mu(A_{\mathbf{x}}),\end{aligned}$$

where the inequality follows from the capacity constraint on set B, and the equality follows from induction hypothesis. Thus, the result holds for both the cases.

D. Proof of Theorem 4

By Little's law,

$$E[D_i] = \frac{\sum_{\mathbf{x}} x_i \pi(\mathbf{x})}{\lambda_i} = \frac{\nu_i \frac{\partial}{\partial \rho_i} G(\boldsymbol{\rho})}{G(\boldsymbol{\rho})}. \tag{24}$$

Thus, to prove the result we only need to show (5). Equation (7) follows by taking derivative of (5) w.r.t. ρ_i. From Theorem 3 and (3) we have,

$$\Phi(\mathbf{x}) = \frac{\sum_{i \in A_{\mathbf{x}}} \Phi(\mathbf{x} - \mathbf{e}_i)}{\mu(A_{\mathbf{x}})}. \tag{25}$$

Since $G_A(\boldsymbol{\rho}) = \sum_{\mathbf{x}: A_{\mathbf{x}} = A} \Phi(\mathbf{x}) \prod_{i \in F} \rho_i^{x_i}$, we get , $G(\boldsymbol{\rho}) = \sum_{A \subset F} G_A(\boldsymbol{\rho})$ and

$$\begin{aligned}G_A(\boldsymbol{\rho}) &= \sum_{\mathbf{x}: A_{\mathbf{x}} = A} \frac{\sum_{i \in A} \Phi(\mathbf{x} - \mathbf{e}_i)}{\mu(A)} \prod_{j \in F} \rho_j^{x_j}, \\ &= \frac{\sum_{i \in A} \sum_{\mathbf{x}: A_{\mathbf{x}} = A} \Phi(\mathbf{x} - \mathbf{e}_i) \prod_{j \in F} \rho_j^{x_j}}{\mu(A)},\end{aligned}$$

Rearranging terms, we get,

$$\begin{aligned}\mu(A) G_A(\boldsymbol{\rho}) &= \sum_{i \in A} \rho_i \sum_{\mathbf{x}: A_{\mathbf{x}} = A \backslash \{i\}} \Phi(\mathbf{x}) \prod_{j \in F} \rho_j^{x_j} \\ &\quad + \sum_{i \in A} \rho_i \sum_{\mathbf{x}: A_{\mathbf{x}} = A} \Phi(\mathbf{x}) \prod_{j \in F} \rho_j^{x_j}, \\ &= \sum_{i \in A} \rho_i G_{A \backslash \{i\}}(\boldsymbol{\rho}) + G_A(\boldsymbol{\rho}) \sum_{i \in A} \rho_i,\end{aligned}$$

further simplification of which gives the desired result.

E. Proof of Corollary 1

From symmetry it follows that $G_A(\boldsymbol{\rho})$ depends on A only through $|A|$. For each $k \geq 0$, let $H_k(\rho) = G_A(\boldsymbol{\rho})$ for A such that $|A| = k$. Similarly, let $\hat{H}_k(\rho) = \frac{\partial}{\partial \rho_i} G_A(\boldsymbol{\rho})$ for A such that $|A| = k$ and $i \in A$.

Thus, from (5), we get

$$H_k(\rho) = \frac{k \rho H_{k-1}(\rho)}{h(k) - k\rho}.$$

Similarly, from (8), we get

$$\hat{H}_k(\rho) = \frac{H_k(\rho) + H_{k-1}(\rho) + (k-1)\rho \hat{H}_{k-1}(\rho)}{h(k) - k\rho}.$$

Then, the result follows from Theorem 4 by letting $F_k(\rho) = \binom{n}{k} H_k(\rho)$ and $\hat{F}_k(\rho) = \binom{n}{k} \hat{H}_k(\rho)$, and noting that for a given file $i \in F$, $\frac{\partial}{\partial \rho_i} G_A(\boldsymbol{\rho})$ is non zero only for $\binom{n-1}{k-1}$ sets of size k for each of which i is an element.

F. Proof of Theorem 5

We prove (16) first and then (17).

Proof of (16): We first prove the following lemma by finding an explicit expression for $\pi_k^{(m,n)}$ for each k for given m and n and then taking the limit as $n \to \infty$ for a fixed m. Let $\lim_{n \to \infty} \pi_k^{(m,n)} = \pi_k^{(m,\infty)}$. Also let $h^{(m,\infty)}(k) = \xi m (1 - (1 - c/m)^k)$ for $k = 0, 1, 2, \ldots, \infty$.

Lemma 1. *For any fixed integers k_1 and k_2 such that $k_1 > k_2$, we have*

$$\frac{\pi_{k_1}^{(m,\infty)}}{\pi_{k_2}^{(m,\infty)}} = \frac{(m\rho)^{k_1 - k_2}}{\prod_{l=k_2+1}^{k_1} h^{(m,\infty)}(l)} \tag{26}$$

Proof. Fix m and n. From definition of $F_k(.)$ in the proof of Corollary 1 one can show that

$$\pi_k^{(m,n)} = \frac{F_k(m\rho/n)}{F(m\rho/n)} \text{ for } k = 1, \ldots, n \tag{27}$$

where $F_k(m\rho/n)$ and $F(m\rho/n)$ are given by recursive expressions in the statement of Corollary 1. Thus, from (11), we get $\pi_0^{(m,n)} = 1/F(m\rho/n)$ and

$$\pi_k^{(m,n)} = \frac{(n-k+1)\frac{m\rho}{n} \pi_{k-1}^{(m,n)}}{h^{(m,n)}(k) - k\frac{m\rho}{n}}, \text{ for } k = 1, \ldots, n.$$

Thus, for any $k_1 > k_2$ we get

$$\begin{aligned}\frac{\pi_{k_1}^{(m,n)}}{\pi_{k_2}^{(m,n)}} &= \frac{(n-k_2)!(\frac{m\rho}{n})^{k_1 - k_2}}{(n-k_1)! \prod_{l=k_2+1}^{k_1} (h^{(m,n)}(l) - l\frac{m\rho}{n})} \\ &\xrightarrow[n \to \infty]{} \frac{(m\rho)^{k_1 - k_2}}{\prod_{l=k_2+1}^{k_1} h^{(m,\infty)}(l)}\end{aligned}$$

□

Now let us study $h^{(m,\infty)}$ and $\pi_k^{(m,\infty)}$ in the limit as $m \to \infty$. For any $\alpha > 0$, we have

$$\lim_{m \to \infty} \frac{1}{m} h(\lfloor \alpha m \rfloor) = \xi(1 - e^{-\alpha c}).$$

Let $k^{(m)}$ be the largest k such that $h^{(m,\infty)}(k) \leq m\rho$. Thus, it is easy to show that $k^{(m)}/m \to \alpha^*$ as $m \to \infty$ where α^* is given by (15).

Now for some large enough γ, consider the following four cases: (1) $0 \leq k < (1-2\epsilon)k^{(m)}$, (2) $(1-2\epsilon)k^{(m)} \leq k \leq (1+2\epsilon)k^{(m)}$, (3) $(1-2\epsilon)k^{(m)} < k \leq \gamma m$, and (4) $k > \gamma m$. Our approach now onwards can be summarized as follows. We first consider the case (4) and show that by choosing γ large enough the tail probability $\sum_{l:l>\gamma m} \pi_l^{(m,\infty)}$ can be made arbitrarily small, independent of m. For the remaining three cases, we then show that $\pi_k^{(m,\infty)}$ concentrates on the second case as m increases to ∞.

Lemma 2. *For any $\delta > 0$, there exists a constant γ such that*

$$\sum_{l:l>\gamma m} \pi_l^{(m,\infty)} \leq \pi_{k^{(m)}}^{(m,\infty)} \delta$$

for all m.

Proof. Find the smallest α such that αm is an integer and $h^{(m,\infty)}(\alpha m) \geq m\rho(1+\epsilon')$ for some fixed $\epsilon' > 0$. Since $\alpha m \geq k^{(m)}$, we have $\pi_{\alpha m}^{(m,\infty)} \leq \pi_{k^{(m)}}^{(m,\infty)}$. Also, it is easy to check that α is $O(1)$, i.e., it does not scale with m. By monotonicity of h, $h^{(m,\infty)}(k) \geq m\rho(1+\epsilon')$ for each $k \geq \alpha m$. From (26), for each $k \geq \alpha m$, we get

$$\pi_k^{(m,\infty)} \leq \pi_{\alpha m}^{(m,\infty)} (\frac{1}{1+\epsilon'})^{k-\alpha m}.$$

Also, for each $k > \alpha m$,

$$\begin{aligned}
\sum_{l:l\geq k} \pi_l^{(m,\infty)} &\leq \pi_k^{(m,\infty)} \sum_{l=1}^{\infty} \left(\frac{1}{1+\epsilon'}\right)^l \\
&= \pi_k^{(m,\infty)} \frac{1}{1-1/(1+\epsilon')} \\
&\leq \pi_{\alpha m}(\frac{1}{1+\epsilon'})^{k-\alpha_m} \frac{1}{1-1/(1+\epsilon')} \\
&\leq \pi_{k^{(m)}}^{(m,\infty)} c' \left(\frac{1}{1+\epsilon'}\right)^{k-\alpha m},
\end{aligned}$$

for some constant c'. Putting $k = \gamma m$, we get,

$$\sum_{l:l\geq\gamma m} \pi_l^{(m,\infty)} \leq c' \pi_{k^{(m)}}^{(m,\infty)} \left(\frac{1}{1+\epsilon'}\right)^{(\gamma-\alpha)m}$$

Thus, for any $\delta > 0$, by choosing γ large enough one can ensure that $\sum_{l\geq\gamma m} \pi_l^{(m,\infty)} \leq \pi_{k^{(m)}}^{(m,\infty)} \delta$ for all m. □

We now prove the following lemma from which (16) follows since ϵ can be chosen arbitrarily and $k^{(m)}/m \to \alpha^*$ as $m \to \infty$.

Lemma 3. *For any $\epsilon > 0$, we have*

$$\lim_{m\to\infty} \frac{\sum_{k=0}^{\infty} \pi_k^{(m,\infty)}}{\sum_{k=(1-2\epsilon)k^{(m)}}^{(1+2\epsilon)k^{(m)}} \pi_k^{(m,\infty)}} = 1$$

Proof. By monotonicity of $h^{(m,\infty)}$, $h^{(m,\infty)}(k) \leq h^{(m,\infty)}((1-2\epsilon)k^{(m)})$ for all $k \leq (1-2\epsilon)k^{(m)}$. Using (26) with $k_1 = (1-\epsilon)k^{(m)}$ and with any $k_2 \leq (1-2\epsilon)k^{(m)}$, we get,

$$\begin{aligned}
\frac{\pi_{(1-\epsilon)k^{(m)}}^{(m,\infty)}}{\pi_{k_2}^{(m,\infty)}} &= \frac{(m\rho)^{(1-\epsilon)k^{(m)}-k_2}}{\prod_{l=k_2+1}^{(1-\epsilon)k^{(m)}} h^{(m,\infty)}(l)} \\
&\geq \left(\frac{m\rho}{h^{(m,\infty)}((1-2\epsilon)k^{(m)})}\right)^{(1-\epsilon)k^{(m)}-k_2} \\
&\geq \left(\frac{m\rho}{h^{(m,\infty)}((1-2\epsilon)k^{(m)})}\right)^{(1-\epsilon)k^{(m)}-(1-2\epsilon)k^{(m)}} \\
&\geq \left(\frac{m\rho}{h^{(m,\infty)}((1-2\epsilon)k^{(m)})}\right)^{\epsilon k^{(m)}}
\end{aligned}$$

Similarly, $h^{(m,\infty)}(k) \geq h^{(m,\infty)}((1+2\epsilon)k^{(m)})$ for all $k \geq (1+2\epsilon)k^{(m)}$. Using (26) with any $k_1 \geq (1+2\epsilon)k^{(m)}$ and with $k_2 = (1+\epsilon)k^{(m)}$, we get,

$$\begin{aligned}
\frac{\pi_{k_1}^{(m,\infty)}}{\pi_{(1+\epsilon)k^{(m)}}^{(m,\infty)}} &= \frac{(m\rho)^{k_1-(1+\epsilon)k^{(m)}}}{\prod_{(1+\epsilon)k^{(m)}}^{k_1} h^{(m,\infty)}(l)} \\
&\leq \left(\frac{m\rho}{h^{(m,\infty)}((1+2\epsilon)k^{(m)})}\right)^{k_1-(1+\epsilon)k^{(m)}} \\
&\leq \left(\frac{m\rho}{h^{(m,\infty)}((1+2\epsilon)k^{(m)})}\right)^{\epsilon k^{(m)}}
\end{aligned}$$

Thus, we get,

$$\begin{aligned}
&\frac{\sum_{k=0}^{\infty} \pi_k^{(m,\infty)}}{\sum_{k=(1-2\epsilon)k^{(m)}}^{(1+2\epsilon)k^{(m)}} \pi_k^{(m,\infty)}} \\
&= \left(\sum_{k=(1-2\epsilon)k^{(m)}}^{(1+2\epsilon)k^{(m)}} \pi_k^{(m,\infty)}\right)^{-1} \left(\sum_{k<(1-2\epsilon)k^{(m)}} + \sum_{k=(1-2\epsilon)k^{(m)}}^{(1+2\epsilon)k^{(m)}}\right. \\
&\quad \left. + \sum_{k=(1+2\epsilon)k^{(m)}+1}^{\gamma m} + \sum_{k>\gamma m}\right) \pi_k^{(m,\infty)} \\
&\leq \frac{\sum_{k<(1-2\epsilon)k^{(m)}} \pi_k^{(m,\infty)}}{\pi_{(1-\epsilon)k^{(m)}}^{(m,\infty)}} + 1 + \frac{\sum_{k=(1+2\epsilon)k^{(m)}+1}^{\gamma m} \pi_k^{(m,\infty)}}{\pi_{(1+\epsilon)k^{(m)}}^{(m,\infty)}} + \delta \\
&\leq (1-2\epsilon)k^{(m)} \left(\frac{h^{(m,\infty)}((1-2\epsilon)k^{(m)})}{m\rho}\right)^{\epsilon k^{(m)}} + 1 \\
&\quad + (\gamma m - (1+2\epsilon)k^{(m)}) \left(\frac{m\rho}{h^{(m,\infty)}((1+2\epsilon)k^{(m)})}\right)^{\epsilon k^{(m)}} + \delta \\
&\underset{m\to\infty}{\longrightarrow} 0 + 1 + 0 + \delta
\end{aligned}$$

Where the last limit can be shown to hold as follows. Using $k^{(m)}/m \to \alpha^*$ as $m \to \infty$, one can show that $\lim_{m\to\infty} h^{(m,\infty)}((1-2\epsilon)k^{(m)})/(\rho m) = \xi(1-e^{-(1-2\epsilon)\alpha^* c}) < \xi(1-e^{-\alpha^* c}) = 1$. Thus, there exists $c_1 < 1$ and $m' > 0$ such that the inequality $\frac{h^{(m,\infty)}((1-2\epsilon)k^{(m)})}{m\rho} < c_1$ holds for all $m > m'$. Similarly, there exists $c_2 < 1$ and $m'' > 0$ such that the inequality $\frac{m\rho}{h^{(m,\infty)}((1+2\epsilon)k^{(m)})} < c_2$ holds for all $m > m'$. Thus, terms $\left(\frac{h^{(m,\infty)}((1-2\epsilon)k^{(m)})}{m\rho}\right)^{\epsilon k^{(m)}}$ and $\left(\frac{m\rho}{h^{(m,\infty)}((1+2\epsilon)k^{(m)})}\right)^{\epsilon k^{(m)}}$ tend to 0 geometrically fast. Since

$\epsilon > 0$ and $\delta > 0$ where chosen arbitrarily, the lemma holds. □

Proof of (17): To find mean delay, we cannot use Little's law just yet, since we have shown concentration in $\pi_k^{(m,n)}$ which is the probability measure for number of active classes and not number of waiting requests. However, intuitively, by increasing n while keeping ρ fixed, we are thinning the arrival process of each class so that the probability of having more than one waiting job for any given class at any given point in time goes to 0. By taking the limit as $n \to \infty$, $\pi_k^{(m,n)}$ then becomes a proxy for the number of waiting jobs. To prove the result formally, we use expression for mean delay in Corollary 1. Define

$$\tau_k^{(m,n)} = \frac{\hat{F}_k(m\rho/n)}{nF(m\rho/n)}.$$

Then, using (9) and (12) from Corollary 1 and using $\nu_i = \nu$ for all i, the mean delay for a given n and m is given by

$$E\left[D^{(m,n)}\right] = \nu \sum_{k=0}^{n} k\tau_k^{(m,n)}. \tag{28}$$

Let $\lim_{n\to\infty} \tau_k^{(m,n)} = \tau_k^{(m,\infty)}$. We now prove the following lemma by induction on k.

Lemma 4.

$$\tau_k^{(m,\infty)} = \frac{\pi_k^{(m,\infty)}}{m\rho} \text{ for } k = 1, 2, \ldots$$

Proof. For a given n, from (11), (12) and (27) we get

$$\tau_k^{(m,n)} = \frac{\frac{1}{n}\pi_k^{(m,n)} + \frac{n-k+1}{nk}\pi_{k-1}^{(m,n)} + \frac{(n-k+1)(k-1)m\rho}{nk}\tau_{k-1}^{(m,n)}}{h^{(m,n)}(k) - km\rho/n}$$

for $k = 1, 2, \ldots, n$ and $\tau_0^{(m,n)} = 0$. By taking limits as $n \to \infty$, we get

$$\tau_k^{(m,\infty)} = \frac{\frac{1}{k}\pi_{k-1}^{(m,\infty)} + \frac{(k-1)m\rho}{k}\tau_{k-1}^{(m,\infty)}}{h^{(m,\infty)}(k)},$$

for any $k \geq 1$, and $\tau_0^{(m,\infty)} = 0$ Now we prove the lemma by induction using the above recursion. First, we prove the result for the base case of $k = 1$. By direct substitution we get,

$$\begin{aligned}\tau_1^{(m,\infty)} &= \frac{\pi_0^{(m,\infty)} + 0}{h(1)} \\ &= \frac{\pi_1^{(m,\infty)}\frac{h(1)}{m\rho}}{h(1)},\end{aligned}$$

where the last equality follows from (26). Thus, we get $\tau_1^{(m,\infty)} = \pi_1^{(m,\infty)}/(m\rho)$. Now, assume the result is true for $\tau_{k-1}^{(m,\infty)}$. Thus we get,

$$\begin{aligned}\tau_k^{(m,\infty)} &= \frac{\frac{1}{k}\pi_{k-1}^{(m,\infty)} + \frac{(k-1)}{k}\pi_{k-1}^{(m,\infty)}}{h^{(m,\infty)}(k)} \\ &= \frac{\pi_{k-1}^{(m,\infty)}}{h^{(m,\infty)}(k)} \\ &= \frac{\pi_k^{(m,\infty)}}{m\rho},\end{aligned}$$

where the last equality again follows from (26). □

Thus from (28), we get,

$$\lim_{n\to\infty} E\left[D^{(m,n)}\right] = \frac{\sum_{k=1}^{\infty} k\pi_k^{(m,\infty)}}{\lambda m}.$$

Proofs of Lemma 2 and 3 show that the probability $\pi_k^{(m,\infty)}$ for $k < (1-2\epsilon)k^{(m)}$ or $k > (1+2\epsilon)k^{(m)}$ decreases to 0 geometrically fast with m. Thus, proceeding along similar lines, one can show that

$$\lim_{m\to\infty}\lim_{n\to\infty} E\left[D^{(m,n)}\right] \in \lambda^{-1}[\alpha^* - 2\epsilon, \alpha^* + 2\epsilon].$$

for any $\epsilon > 0$. Hence, the result.

Virag Shah is a PhD student in Electrical and Computer Engineering department at The University of Texas at Austin. He received his B.E. degree from University of Mumbai in 2007. He received his M.E. degree from Indian Institute of Science, Bangalore in 2009. He was a Research Fellow at Indian Institute of Technology, Bombay from 2009 to 2010. His research interests include designing algorithms for content delivery systems, cloud computing systems, and internet of things; performance modeling; applied probability and queuing theory. He is a recipient of two best paper awards: IEEE INFOCOM 2014 at Toronto, Canada; National Conference on Communications 2010 at Chennai, India.

Gustavo de Veciana (S'88-M'94-SM'01-F'09) received his B.S., M.S, and Ph.D. in electrical engineering from the University of California at Berkeley in 1987, 1990, and 1993 respectively, and joined the Department of Electrical and Computer Engineering where he is currently a Cullen Trust Professor of Engineering. He served as the Director and Associate Director of the Wireless Networking and Communications Group (WNCG) at the University of Texas at Austin, from 2003-2007. His research focuses on the analysis and design of communication and computing networks; data-driven decision-making in man-machine systems, and applied probability and queueing theory. Dr. de Veciana served as editor and is currently serving as editor-at-large for the IEEE/ACM Transactions on Networking.

He was the recipient of a National Science Foundation CAREER Award 1996 and a co-recipient of five best paper awards including: IEEE William McCalla Best ICCAD Paper Award for 2000, Best Paper in ACM TODAES Jan 2002-2004, Best Paper in ITC 2010, Best Paper in ACM MSWIM 2010, and Best Paper IEEE INFOCOM 2014. In 2009 he was designated IEEE Fellow for his contributions to the analysis and design of communication networks. He currently serves on the board of trustees of IMDEA Networks Madrid.