On Storage Allocation for Maximum Service Rate in Distributed Storage Systems

Moslem Noori*, Emina Soljanin†, Masoud Ardakani*
*Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada
† Department of Electrical and Computer Engineering, Rutgers University, Piscataway, NJ, USA

***Abstract*—Storage allocation affects important performance measures of distributed storage systems. Most previous studies on the storage allocation consider its effect separately either on the success of the data recovery or on the service rate (time) where it is assumed that no access failure happens in the system. In this paper, we go one step further and incorporate the access model and the success of data recovery into the service rate analysis. In particular, we focus on quasi-uniform storage allocation and provide a service rate analysis for both fixed-size and probabilistic access models at the nodes. Using this analysis, we then show that for the case of exponential waiting time distribution at individuals storage nodes, minimal spreading allocation results in the highest system service rate for both access models. This means that for a given storage budget, replication provides a better service rate than a coded storage solution.**

I. Introduction

Cloud networks provide anywhere, anytime access to one's data, offer a high level of data safety (e.g., against hardware failure, theft, fire), and make sharing data easy. This functionality is achieved by storing chunks of a data entity (file) redundantly over multiple storage nodes. Distributed storage systems (DSSs), thus play a central role in cloud networks, and have been the focus of many ongoing diverse research activities [1]–[5].

A main concern for the consumers is to be able to download the data and, often more importantly, to do that quickly. Thus, the download service rate is the focus of this paper. Several studies have looked into how to allocate redundant chunks of data over the storage nodes to optimize some performance metrics (e.g., [6]–[10] and references therein). The constraints here are that the number of nodes and the level of redundancy are limited, and to download his file, the user can access all or some subset of (possibly unavailable) nodes in the system.

Existing studies on the storage allocation mostly focus on two performance aspects of DSSs. One of them is the probability of successful data recovery P_{s} when only a subset of possibly failed nodes are accessed. The other is the average service time T_{s} when a set of nodes from which the file can be recovered is accessed. Simply put, when a subset of storage nodes are assigned to serve a customer, P_{s} is the probability that these nodes jointly (under possible failures) have been allocated sufficient data to reconstruct and deliver the requested file to the customer. On the other hand, T_{s} represents the time needed to serve a customer's request to download the file. In other words, P_{s} reflects the reliability of the DSS in serving the customers' requests while T_{s} mostly represents the system's quality of service once the reliability has been provided. Finding these quantities has shown to be quite challenging, and optimal allocations are known only in some special cases.

In general, both these measures are of interest and should be simultaneously taken into account for devising the allocation strategy. For instance, assume a situation where several customers send a delay-sensitive request to access the stored data. While increasing the chance of successfully downloading the file by each of the customers is desirable, this should not come at the cost of unbearable delivery delay. Moreover, in practice, we may often want to partially sacrifice a successful (but possibly tardy) data delivery to some users in order to ensure that other users, that can receive the data, are indeed served fast.

The existing work does not address such scenarios. Papers concerned with P_{s} are not concerned with the delay or assume instantaneous (infinite rate) service . On the other hand, papers concerned with T_{s} assume that data is available on the accessed nodes and can be served to the customer at some finite rate.

In this work, we assume a finite service rate for storage nodes and the data (un)availability that depends on the used allocation scheme. We are interested in the entire system service rate, under certain access and/or node failure models. Note that, depending on the allocation, some subsets of nodes will not contain enough file chunks between them to recover the data, and accessing them will result in a zero system's service rate. On the other hand, again depending on the allocation, some subsets of nodes will contain redundant file chunks, and that redundancy can be exploited to increase the service rate.

Our analysis reveals that the allocation that maximizes the probability of successful data recovery is often not the one that maximizes the average service rate. The key to understanding this, perhaps unintuitive, phenomenon is to look into the role of redundancy. When the accessed nodes contain more data than necessary to reconstruct the file, this redundancy is superfluous for file recovery but could be exploited to speed up the download service rate since only a fraction of nodes have to deliver their chunks in a timely manner. Therefore, depending on the number of storage nodes and the allocated redundancy budget, it may be beneficial for recovery to maximally spread the redundant file chunks over the storage nodes, whereas concentrating the redundant chunks may increase the expected service rate. We show here that this is always the case for the DSS models considered in the literature.

The rest of the paper is organized as follows. In Section II, we introduce the considered DSS setup in more detail and formally define the considered problem in this paper. Service rate analysis considering the effect of access model and the success of serving a request is presented in Section III. Using this analysis, we then prove that minimal spreading maximizes the service rate of the system in Section IV-A and Section IV-B respectively for the fixed-size and probabilistic access models. Numerical examples are also provided in these two chapters. Finally, Section V concludes the paper.

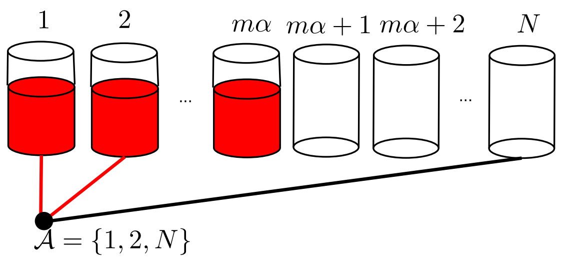

Fig. 1. An N-node DSS with quasi-symmetric allocation. While three nodes are successfully accessed, only two of them have (coded) data blocks.

II. System Model and Problem Definition

In this section, we describe the considered DSS in detail. Then, we formally define the storage allocation problem to maximize the service rate of the system.

A. Storage Model

We consider a DSS with N storage nodes, namely n_i's for $i \in \mathcal{N} = \{1,\ldots,N\}$. A file with F blocks is stored over these nodes that are to be accessed by the system's customers. To protect the data against nodes' failure, the file is encoded by a maximum distance separable (MDS) code to generate T encoded blocks (any F of them are sufficient to recover the original file). Here, we assume that the code rate is $1/m$, where m is a positive integer. Hence, $T = mF$. The encoded T blocks are then partitioned into N subsets, say $\mathcal{X}_i$'s for $i \in \mathcal{N}$ where $|\mathcal{X}_i| = x_i$, and thus $\sum_{i=1}^{N} x_i = T$. We call such partitioning an *allocation*. Now, the x_i blocks within $\mathcal{X}_i$ are stored at the node n_i. Note that $0 \le x_i \le F$ since storing more than F blocks on a node does not serve any purpose in our model.

Dealing with a general storage allocation optimization problem to maximize P_s or minimize T_s is computationally difficult for a general setup [6]. Here, we focus on the quasi-symmetric allocations [7] where for a positive integer α, the number of blocks stored in n_i, denoted by $x_\alpha(i)$, is either 0 or F/α. Details of the range of α will be discussed later. Here, we identify a quasi-symmetric allocation with a pair (α,β) where β represents the number of nodes that are not empty. Since $\beta\frac{F}{\alpha} = T$, we have $\beta = m\alpha$. Figure 1 depicts an example quasi-symmetric allocation for a DSS with N storage nodes.

A quasi-symmetric allocation where $\alpha = 1$ and $\beta = m$ is called a *minimal spreading* allocation [6]. Note that for a minimal spreading allocation, we can skip coding and replicate the whole F blocks of the file over m storage nodes without compromising the file protection. Similarly, an allocation with $\alpha = \frac{NF}{T}$ and $\beta = N$ is called a *maximal spreading* allocation.

B. Data Access Model

For the data access by the users, we consider the following two main models suggested for DSSs [6].

1) Fixed-size access: In this access model, when a download request is received, the request is forwarded to a random r-subset of the N nodes, i.e., a subset with cardinality r [6], [7]. Since an MDS code is used to store the data, the original file can be recovered if the accessed nodes contain at least F blocks. In other words, the access to a given r-subset $\mathcal{A}$ results in the successful recovery of the data iff

$$\sum_{i\in\mathcal{A}} x_i \ge F. \tag{1}$$

Note that for $\alpha > r$, it is impossible to recover the data. Thus, for the fixed-size access model we only consider $1 \le \alpha \le r$.

2) Probabilistic access: In this case, the download request is forwarded to all nodes that store the data. However, the request to access each of them fails with probability p and succeeds with probability $1-p$. Assuming that $\mathcal{A}$ represents the set of nodes that are successfully accessed, then the condition for data recovery is similar to (1). In this case, $1 \le \alpha \le \frac{NF}{T}$.

Regardless of the access model, for an arbitrary accessed subset of nodes $\mathcal{A}$, let us denote the number of nodes containing data by k. For instance, in Figure 1, three nodes ($|\mathcal{A}| = 3$) are accessed while only $k = 2$ of them have data. For an (α,β) quasi-symmetric allocation, data recovery from this subset is successful if and only if $k \ge \alpha$.

C. Service Model

Here, we assume that the arriving download requests follow a Poisson distribution. Each request is forwarded to a set of accessed nodes, called $\mathcal{A}$, to be served. At these nodes, we assume a multiple-fountain system [2] where the arriving request is forked into $|\mathcal{A}|$ tasks[1]. Each of these tasks then wait to be served by one of the accessed $|\mathcal{A}|$ nodes. For an (α,β) allocation over the nodes, the download request is successfully served when $k \ge \alpha$ and any α out of the k nodes with data successfully serve their assigned tasks. At this point, the remaining $|\mathcal{A}| - \alpha$ tasks are discarded and dropped from the rest of the accessed nodes. If the nodes in $\mathcal{A}$ do not contain enough data to reconstruct the file (i.e. less than F blocks), the download request cannot be served.

When a task is assigned to a storage node, it may not get served right away since the node is for example busy with serving another request. Thus, there is a waiting time associated with the time needed for the content inside the node to become available for download by the user. Here, for simplicity, we assume that the waiting time at all nodes are independent and identically distributed (i.i.d) random variables all following an exponential distribution with mean $\frac{1}{\mu}$. In other words, each storage node has a service rate of μ. Further, it is assumed that the download bandwidth is large enough so that the time needed to download the data is negligible compared to the waiting time at the servers. As a result, the overall service rate of the system is characterized only by the waiting time at the servers.

D. Problem Definition

For a given (α,β) quasi-symmetric allocation, the average service rate, denoted by $\mu_\mathrm{s}(\alpha)$ is the highest rate that the arriving download requests can be served by the system. (Since $\beta = m\alpha$, we do not consider β as a separate variable here.) As we discuss later in the paper, beside the service rate at each individual node, $\mu_\mathrm{s}(\alpha)$ also depends on the nodes' storage allocation. Our goal in this paper is to find the allocation that maximizes $\mu_\mathrm{s}(\alpha)$.

For a formal problem definition, we introduce the function

$$\mathbb{I}\Big\{\sum_{i\in\mathcal{A}} x_\alpha(i) \ge F\Big\}$$

[1]For the fixed-size access model $|\mathcal{A}| = r$ while for the probabilistic access $|\mathcal{A}|$ could be any number between 1 and N.

indicating whether the file can be recovered from the nodes in $\mathcal{A}$ or not. The probability of being able to successfully recover the file under an (α, β) allocation is, therefore, given by

$$P_{\mathrm{s}}(\alpha) = \sum_{\mathcal{A} \subseteq \mathcal{N}} P(\mathcal{A}) \mathbb{I}\Big\{ \sum_{i \in \mathcal{A}} x_\alpha(i) \geq F \Big\} \tag{2}$$

where $P(\mathcal{A})$ is the probability of choosing $\mathcal{A}$. Similarly, the average service rate under an (α, β) allocation is given by

$$\mu_{\mathrm{s}}(\alpha) = \sum_{\mathcal{A} \subseteq \mathcal{N}} P(\mathcal{A}) \mu_\alpha(\mathcal{A}) \mathbb{I}\Big\{ \sum_{i \in \mathcal{A}} x_\alpha(i) \geq F \Big\} \tag{3}$$

where $\mu_\alpha(\mathcal{A})$ is the service rate when the set of accessed nodes is $\mathcal{A}$.

Previous studies on finding the optimal storage allocations are focused on finding the allocation that maximizes P_{s} for a given storage budget T. For instance, it was shown in [7] that for a DSS with fixed-size access model, α that maximizes (2) depends on the ratio $m = T/F$. Similar claims are made in [6]. It is easy to see that $\mu_\alpha(\mathcal{A})$ is a decreasing function of α. Therefore, when $\alpha = 1$ maximizes (2), i.e. minimal spreading maximizes P_{s}, it also maximizes (3), term by term and thus maximizes $\mu_{\mathrm{s}}(\alpha)$. We devote the following sections to showing that (3) is maximized by $\alpha = 1$ even if, for some $\alpha > 1$, there are more sets $\mathcal{A} \subset \mathcal{N}$ that allow file reconstruction (more non-zero terms in (3)) than for $\alpha = 1$. That is, the service rate is always maximized by using minimal spreading allocation.

III. Analysis of $\mu_{\mathrm{s}}(\alpha)$

In this section, we study $\mu_{\mathrm{s}}(\alpha)$ considering the effect of storage allocation and access model. This study will then be used in the following sections to find the optimal allocation maximizing $\mu_{\mathrm{s}}(\alpha)$ for fixed-size and probabilistic access models.

The rate of serving incoming requests depends on how many nodes with data are successfully accessed. Thus,

$$\mu_{\mathrm{s}}(\alpha) = \sum_{k=1}^{m\alpha} P(k, \alpha) \mu_{\mathrm{s}}(\alpha|k). \tag{4}$$

where $P(k, \alpha)$ denotes the probability of having exactly k nodes with data in the set of accessed nodes $\mathcal{A}$. Also, $\mu_{\mathrm{s}}(\alpha|k)$ refers to the conditional service rate given that k nodes with data are accessed. Note that for any $k < \alpha$, recovering the data from the nodes in $\mathcal{A}$ is not possible, and $\mu_{\mathrm{s}}(\alpha|k) = 0$. Thus,

$$\mu_{\mathrm{s}}(\alpha) = \sum_{k=\alpha}^{m\alpha} P(k, \alpha) \mu_{\mathrm{s}}(\alpha|k). \tag{5}$$

It is easy to show that for the fixed-size access model

$$P(k, \alpha) = \frac{\binom{m\alpha}{k}\binom{N-m\alpha}{r-k}}{\binom{N}{r}} \tag{6}$$

and for the probabilistic access model

$$P(k, \alpha) = \binom{m\alpha}{k} (1-p)^k p^{m\alpha - k}. \tag{7}$$

Now that we have $P(k, \alpha)$, to evaluate $\mu_{\mathrm{s}}(\alpha)$, we present the following result on $\mu_{\mathrm{s}}(\alpha|k)$.

Lemma 1: For a given (α, β) quasi-symmetric allocation,

$$\mu_{\mathrm{s}}(\alpha|k) = \mu \frac{\prod_{j=k-\alpha+1}^{k} j}{\sum_{i=k-\alpha+1}^{k} \prod_{\substack{j=k-\alpha+1 \\ j \neq i}}^{k} j}. \tag{8}$$

Proof: To find $\mu_{\mathrm{s}}(\alpha|k)$, we start by considering the conditional service time of the requests, denoted by $T_{\mathrm{s}}(\alpha|k)$, which is the inverse of $\mu_{\mathrm{s}}(\alpha|k)$. As discussed before, a request is served when the first α storage nodes with data, out of the accessed k nodes with data, start serving the request. That said, $T_{\mathrm{s}}(\alpha|k)$ is the αth order statistics of k waiting times at the storage nodes. Considering that all waiting times have an exponential distribution with mean $\frac{1}{\mu}$, we have

$$T_{\mathrm{s}}(\alpha|k) = \frac{1}{\mu} \sum_{i=1}^{\alpha} \frac{1}{k - \alpha + i} = \frac{1}{\mu} \frac{\sum_{i=k-\alpha+1}^{k} \prod_{\substack{j=k-\alpha+1 \\ j \neq i}}^{k} j}{\prod_{j=k-\alpha+1}^{k} j}. \tag{9}$$

Since $\mu_{\mathrm{s}}(\alpha|k) = \frac{1}{T_{\mathrm{s}}(\alpha|k)}$, (8) is simply inferred from (9). ■

While the $\mu_{\mathrm{s}}(\alpha|k)$ expression in (8) looks rather complicated, it is used in the following sections to simplify the service rate analysis in the form of the following corollary.

Corollary 1: For $\alpha > 1$

$$\sum_{i=k-\alpha+1}^{k} \prod_{\substack{j=k-\alpha+1 \\ j \neq i}}^{k} j > \sum_{i=k-\alpha+1}^{k} \prod_{j=k-\alpha+1}^{k-1} j = \alpha \prod_{j=k-\alpha+1}^{k-1} j. \tag{10}$$

Now, using (10) and (8) we have

$$\mu_{\mathrm{s}}(\alpha|k) < \mu \frac{\prod_{j=k-\alpha+1}^{k} j}{\alpha \prod_{j=k-\alpha+1}^{k-1} j} = \mu \frac{k}{\alpha}. \tag{11}$$

IV. Optimal Storage Allocation

A. Fixed-Size Access Model

In this section, we find the optimal allocation to maximize $\mu_{\mathrm{s}}(\alpha)$ for fixed-size access model. For this, we start by the following lemma.

Lemma 2: For minimal spreading, i.e. $\alpha = 1$, service rate is

$$\mu_{\mathrm{s}}(1) = \mu \frac{mr}{N}. \tag{12}$$

Proof: First of all, note that for $\alpha = 1$, we have

$$\mu_{\mathrm{s}}(1|k) = \mu k. \tag{13}$$

Thus[2],

$$\mu_{\mathrm{s}}(1) = \mu \sum_{k=1}^{\min(m,r)} k P(k, 1) = \mu \sum_{k=1}^{r} k P(k, 1). \tag{14}$$

On the other hand, $\alpha = 1$ and

$$k \binom{m\alpha}{k} = k \binom{m}{k} = m \binom{m-1}{k-1}. \tag{15}$$

As a result,

$$\mu_{\mathrm{s}}(1) = \frac{m\mu}{\binom{N}{r}} \sum_{k=1}^{r} \binom{m-1}{k-1} \binom{N-m}{r-k}. \tag{16}$$

Using Vandermonde's convolution, one can show that

$$\sum_{k=1}^{r} \binom{m-1}{k-1} \binom{N-m}{r-k} = \binom{N-1}{r-1}. \tag{17}$$

Now, plugging (17) into (16) completes the proof. ■

[2]Recall that for $k > m\alpha$, $\binom{m\alpha}{k} = 0$, and hence, $P(k, \alpha) = 0$.

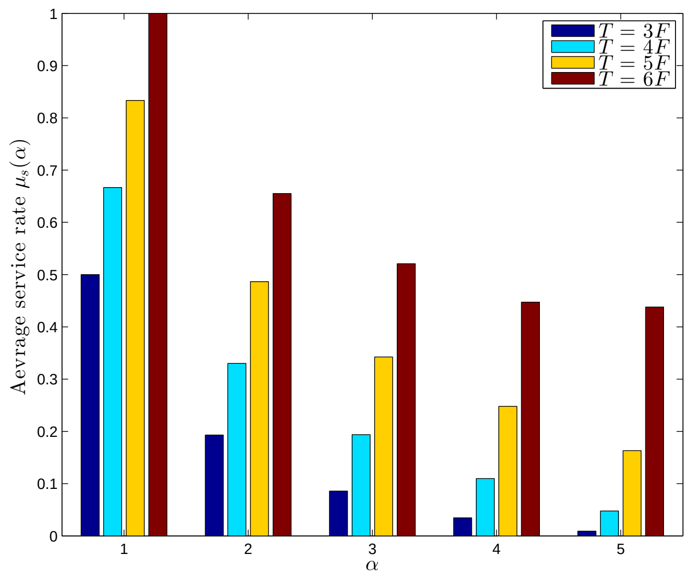

Fig. 2. Average service rate for fixed-sized access model with $N = 30$ and $r = 5$.

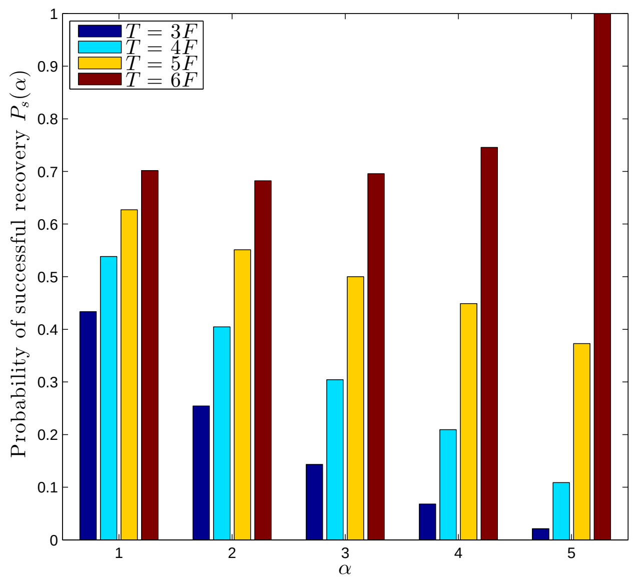

Fig. 3. Probability of successful recovery for fixed-sized access model with $N = 30$ and $r = 5$.

Now that we have $\mu_{\mathrm{s}}(1)$, the next step is to find an upper bound on $\mu_{\mathrm{s}}(\alpha)$ for any $2 \le \alpha \le r$ and compare this bound with $\mu_{\mathrm{s}}(1)$. First, we present the following lemma.

Lemma 3: For any $2 \le \alpha \le r$,

$$\mu_{\mathrm{s}}(\alpha) < \mu\frac{mr}{N}. \tag{18}$$

Proof: Using Corollary 1,

$$\mu_{\mathrm{s}}(\alpha) < \mu\sum_{k=\alpha}^{r}\frac{k}{\alpha}P(k,\alpha) \tag{19}$$

$$= \frac{\mu}{\alpha\binom{N}{r}}\sum_{k=\alpha}^{r}k\binom{m\alpha}{k}\binom{N-m\alpha}{r-k} \tag{20}$$

$$= \frac{m\mu}{\binom{N}{r}}\sum_{k=\alpha}^{r}\binom{m\alpha-1}{k-1}\binom{N-m\alpha}{r-k}. \tag{21}$$

In addition,

$$\sum_{k=\alpha}^{r}\binom{m\alpha-1}{k-1}\binom{N-m\alpha}{r-k} < \sum_{k=0}^{r-1}\binom{m\alpha-1}{k}\binom{N-m\alpha}{r-1-k} = \binom{N-1}{r-1}. \tag{22}$$

Hence,

$$\mu_{\mathrm{s}}(\alpha) < \frac{m\mu}{\binom{N}{r}}\binom{N-1}{r-1} = \mu\frac{mr}{N}. \tag{23}$$

■

Now, using Lemma 2 and 3, we have the following theorem on the optimal storage allocation maximizing $\mu_{\mathrm{s}}(\alpha)$.

Theorem 1: Minimal spreading maximizes the service rate for a DSS with fixed-size access model.

To verify the results of Theorem 1, we present some numerical examples in Figure 2 and 3. These figures depict the average service rate $\mu_{\mathrm{s}}(\alpha)$ and the probability of successful recovery $P_{\mathrm{s}}(\alpha)$ for a DSS with $N = 30$ nodes, a fixed-size access model with $r = 5$, and $\mu = 1$. Here, $\alpha = 1$ and $\alpha = 5$ are associated with minimal and maximal spreading allocation respectively. As seen in these figures, for smaller allocation budget T, minimal spreading maximizes both $P_{\mathrm{s}}(\alpha)$ and $\mu_{\mathrm{s}}(\alpha)$. However, as we increase T, while $P_{\mathrm{s}}(\alpha)$ is maximum when $\alpha = 5$ (in fact, any download request is successfully served and $P_{\mathrm{s}}(5) = 1$), $\mu_{\mathrm{s}}(\alpha)$ is always maximized by $\alpha = 1$.

B. Probabilistic Access Model

In this section, we study the service rate for a DSS with probabilistic access model. The goal is to find the optimal storage allocation maximizing the average service rate.

Lemma 4: The service of a DSS with probabilistic access and minimal spreading allocation is

$$\mu_{\mathrm{s}}(1) = m\mu(1-p). \tag{24}$$

Proof: Using (5), (7) and (13), we have

$$\mu_{\mathrm{s}}(1) = \mu\sum_{k=1}^{m}k\binom{m}{k}(1-p)^k p^{m-k} \tag{25}$$

$$= m\mu\sum_{k=1}^{m}\binom{m-1}{k-1}(1-p)^k p^{m-k} \tag{26}$$

$$= m\mu(1-p)\sum_{k=0}^{m-1}\binom{m-1}{k}(1-p)^k p^{m-k-1} \tag{27}$$

$$= m\mu(1-p). \tag{28}$$

■

Similar to the case of the fixed-size access model, we find an upper bound on the service rate of the system when $2 \le \alpha$.

Lemma 5: For a quasi-symmetric allocation where $2 \le \alpha$, the service rate of the system is bounded as

$$\mu_{\mathrm{s}}(\alpha) < m\mu(1-p). \tag{29}$$

Proof: Using Corollary 1, we have

$$\mu_{\mathrm{s}}(\alpha) < \frac{\mu}{\alpha}\sum_{k=\alpha}^{m\alpha}k\binom{m\alpha}{k}(1-p)^k p^{m\alpha-k} \tag{30}$$

$$= \frac{\mu}{\alpha}\sum_{k=\alpha}^{m\alpha}m\alpha\binom{m\alpha-1}{k-1}(1-p)^k p^{m\alpha-k}$$

$$= m\mu(1-p)\sum_{k=\alpha-1}^{m\alpha-1}\binom{m\alpha-1}{k}(1-p)^k p^{m\alpha-k-1}$$

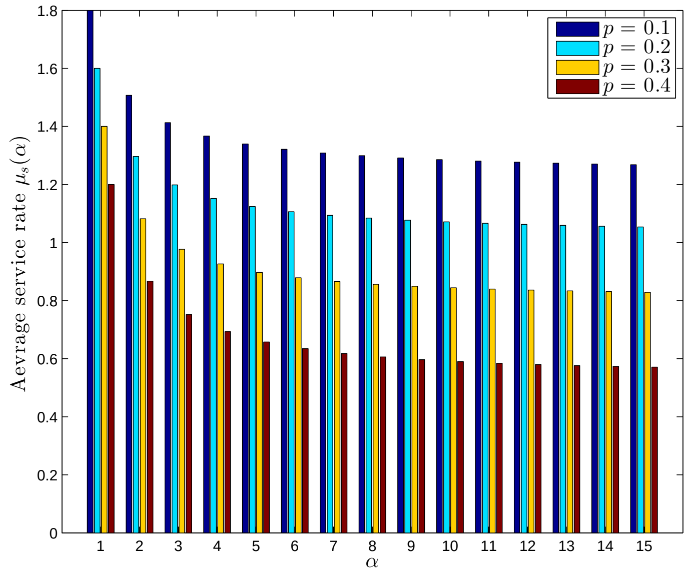

Fig. 4. Average service rate for probabilistic access model when $N = 30$ and $T = 2F$.

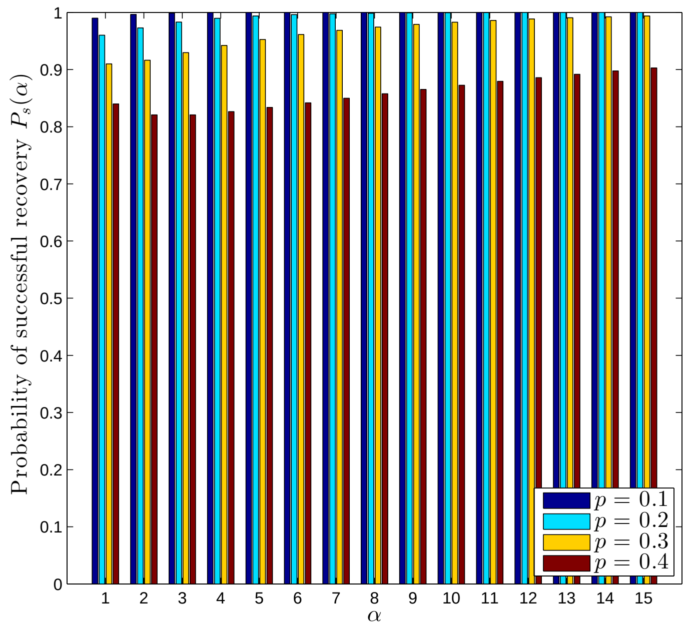

Fig. 5. Probability of successful recovery for probabilistic access model when $N = 30$ and $T = 2F$.

On the other hand

$$\sum_{k=\alpha-1}^{m\alpha-1} \binom{m\alpha-1}{k} (1-p)^k p^{m\alpha-k-1} \leq 1. \tag{31}$$

Thus,

$$\mu_{\mathrm{s}}(\alpha) < m\mu(1-p). \tag{32}$$

■

Now, using the results of Lemma 4 and 5, we have the following theorem on the optimal storage allocation for the probabilistic access model.

Theorem 2: In a DSS with probabilistic access model, minimal spreading results in the maximum service rate.

Numerical examples to verify the results of Theorem 2 are presented in Figure 4 and 5. The results are for a DSS with $N = 30$ storage nodes, a storage budget of $T = 2F$, and $\mu = 1$. As seen in Figure 5, maximal spreading allocation results in the highest probability of successful recovery for all considered probabilities of access failure. However, the average service rate always reaches its maximum for $\alpha = 1$, i.e. minimal spreading, as depicted in Figure 4.

V. Conclusion

Content allocation throughout a distributed storage system affects the probability that the content can be recovered when there is uncertainty in the number, identity, and/or availability of the storage nodes queried for service. So far the concern has been only that the stored data can eventually be downloaded, and not how long that process might take. To the best of our knowledge, this paper is the first attempt to understand how content allocation affects the download service rate. We showed that under certain assumptions, the minimal spreading allocation maximizes the service rate for the commonly assumed content access models specified by the number, identity, and/or availability of the storage nodes queried for service. Therefore, storing data through replication results in faster service for the incoming download requests than a coded storage with the same storage budget. Our assumption was that the service time at the storage nodes follows an exponential distribution, and is identically distributed and independent for all users. A more advanced model should involve other distributions (in particular, the shifted exponential as in [3] and [2]) as well as fork-join queuing considerations.

Acknowledgment

The authors were in part supported by Alberta Innovates Technology Futures (AITF) and Natural Sciences and Engineering Research Council of Canada (NSERC), and would also like to thank A. Badr, G. Joshi, and K. Mahdaviani for valuable discussions at the Banff International Research Station (BIRS).

References

[1] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Trans. Inform. Theory*, vol. 56, no. 9, pp. 4539–4551, 2010.

[2] G. Joshi, Y. Liu, and E. Soljanin, "Coding for fast content download," in *Communication, Control, and Computing (Allerton), 2012 50th Annual Allerton Conference on*, 2012, pp. 326–333.

[3] S. Chen, Y. Sun, U. C. Kozat, L. Huang, P. Sinha, G. Liang, X. Liu, and N. B. Shroff, "When queueing meets coding: Optimal-latency data retrieving scheme in storage clouds," in *IEEE Conf. on Computer Communications (INFOCOM)*, 2014, pp. 1042–1050.

[4] R. Tandon and S. Mohajer, "New bounds for distributed storage systems with secure repair," in *Allerton Conf. on Communication, Control, and Computing*, 2014, pp. 431–436.

[5] S. Kadhe, E. Soljanin, and A. Sprintson, "Analyzing download time for availability codes," in *Information Theory Proceedings (ISIT), 2015 IEEE International Symposium on*, July 2015.

[6] D. Leong, A. G. Dimakis, and T. Ho, "Distributed storage allocations," *IEEE Trans. Inform. Theory*, vol. 58, no. 7, pp. 4733–4752, 2012.

[7] M. Sardari, R. Restrepo, F. Fekri, and E. Soljanin, "Memory allocation in distributed storage networks," in *IEEE Intl. Symp. on Information Theory (ISIT)*, June 2010, pp. 1958–1962.

[8] M. Noori and M. Ardakani, "Allocation for heterogeneous storage nodes," *IEEE Commun. Lett.*, vol. 19, no. 12, pp. 2102–2105, 2015.

[9] B. Hong and W. Choi, "Asymptotic analysis of failed recovery probability in a distributed wireless storage system with limited sum storage capacity," in *IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 2014, pp. 6459–6463.

[10] D. Leong, A. G. Dimakis, and T. Ho, "Distributed storage allocations for optimal delay," in *IEEE Intl. Symp. on Information Theory (ISIT)*, 2011, pp. 1447–1451.