# Reducing Delay of Flexible Download in Coded Distributed Storage System

Qiqi Shuai
The University of Hong Kong
Email: qqshuai@eee.hku.hk

Victor O.K. Li, *Fellow, IEEE*
The University of Hong Kong
Email: vli@eee.hku.hk

***Abstract*—Download delay is a crucial performance metric in distributed storage systems as it greatly impacts user experience. Recently, plenty of research has pointed out that coding can reduce delay compared with replication. However, almost all previous studies only focus on the case in which all of the users require the same size of files and hence must download all files of a codeword and ignore the case in which some users only need some of the files in the codeword. Moreover, they do not consider the advantage of codes with original data nodes, such as systematic codes. In this paper, based on a more general and practical case in which download requests may desire different sizes of files and hence only some of the files of a codeword in a systematic $(n, k)$ MDS-coded storage system, we propose the compound read method, characterize its mean download delay in low arrival rate scenario and derive upper and lower bounds on its mean download delay in high arrival rate scenario. We also compare the delay performance of compound and k-access reads and propose a scheme C & K to dynamically take advantage of them according to users' required size of files to reduce the mean download delay. In addition, with real service time traces from Amazon S3, we conduct trace-driven simulations to verify our theoretical analyses and the effectiveness of the C & K scheme.**

***Keywords*—*Systematic code, MDS property, k-access read, compound read, fork-join queues.***

## I. Introduction

In massive distributed storage systems, failure is the norm rather than the exception [1]. To tolerate frequent failures and provide sufficient reliability, we need to increase storage redundancy by replication or erasure coding.

A single codeword of an erasure code has $n$ nodes, $k$ of which are the original data nodes, and the other $n - k$ are parity nodes. If a code satisfies the maximum-distance-separable (MDS) property, any $k$ out of the $n$ nodes are sufficient to reconstruct all data in the $k$ original data nodes. As illustrated in Fig. 1, a codeword of (9,6) MDS code consists of 6 data nodes and 3 parity nodes. The code is systematic since $d_1$ to $d_6$ are in uncoded form.

Delay is a key performance metric for distributed storage systems and has great impact on user experience, especially for data retrieval applications. Compared with replication, erasure codes enjoy a higher degree of fault tolerance and storage efficiency and recently, there has been a significant interest in how erasure codes can reduce delay as well.

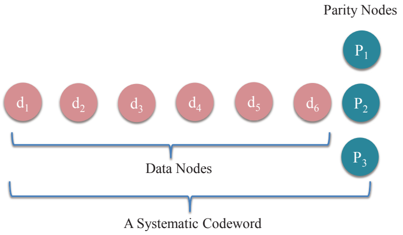


Fig. 1. A codeword of systematic (9,6) MDS code.

Till now, previous work has proved that coding can reduce delay compared with replication and redundant requests can further improve delay performance since they can make use of extra resources and abandon unnecessary tasks when the corresponding request has been met [2]. As illustrated in Fig. 2, based on a fork-join queueing model, a typical work flow in an $(n, k)$ MDS-coded storage system is given as follows: when a user request arrives, it is forked into $n$ tasks which are sent to each of the $n$ nodes and the results of any $k$ out of the $n$ tasks, constituting a completed request, are sent back to the user. At the same time, the other $n - k$ unfinished tasks are abandoned immediately.

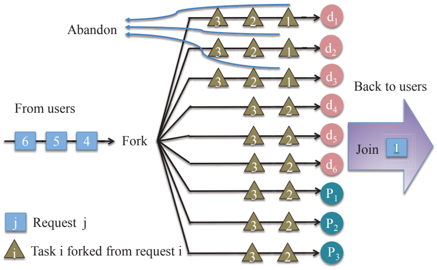


Fig. 2. A typical work flow.

Other than some work that mentioned direct reads [3], [4], almost all previous studies analyze download delay when a user is interested in downloading all files in a codeword, i.e., each request to the storage system needs to access at least $k$ storage nodes (k-access read). However, in practical storage systems such as Windows Azure storage system (WAS), only when files reach a certain size (e.g., 1 GB), will it be a candidate for erasure coding [3]. That is, in practice, files stored in a codeword are usually very large and users' requests may only desire part of these files, and this desired part can be directly read from the systematic part in the codeword. Therefore, it is significant to analyze the delay performance when users only request a subset of the erasure coded content, with a corresponding reduced delay.

In this paper we focus a case where different requests may desire different sizes of files from a codeword. That is, a request may only be interested in obtaining the files of $d$ nodes, where $1 \leq d \leq k$. For instance, consider when a request only desires $d_1$ in Fig. 1. A k-access read can meet the requirement since it gets all the information in the codeword by the MDS property. But we note that we can also satisfy the requirement by only sending a task to the $d_1$ node (direct read), possibly obtaining superior delay performance. Although Kadhe *et al.* [5] also considered direct read, they only analyzed the special case when $d = 1$ and they focused on availability codes rather than MDS codes.

*Our Contributions.* Based on a more general and practical case in which users may require different sizes of files in a systematic $(n, k)$ MDS-coded storage system, we propose the compound read method, characterize its mean download delay for the low arrival rate scenario and compare it with that of k-access read. We also propose a scheme C & K which dynamically takes advantage of compound and k-access reads according to users' required number of data nodes, $d$ to reduce download delay. For the high arrival rate scenario, we present both upper and lower bounds on the mean download time of compound read. Finally, with real service time traces from Amazon S3, we conduct trace-driven simulations and validate our theoretical analyses and verify the effectiveness of C & K.

## II. Theoretical Analyses

We consider a systematic $(n, k)$ MDS-coded storage system and let $T_i$ denote the service time of node $i$, $1 \leq i \leq n$. As with previous work [2], [5], [6], [7], we assume that $T_i$ is an independent and identically distributed ($i.i.d.$) exponential random variable for each node, say $T_i \sim \exp(\mu)$.

Similar to [5], we consider two specific cases.

(1) Low arrival rate case, in which the request arrival rate is so small that there is no overlap between different requests, i.e., nodes in a codeword will not simultaneously serve tasks from different requests.

(2) High arrival rate case, in which the request rate is high, and hence there is a need to queue the requests before they can be served. In this case, the download requests may overlap, i.e., nodes in a codeword can simultaneously serve tasks from different requests.

### A. Low Arrival Rate Scenario

When a download request which desires $d$ out of the $k$ original data nodes arrives, we first analyze the mean download time for direct read and k-access read.

*(1) Direct read*

Direct read means that the download request only accesses the $d$ desired nodes. Denote $T^{(d)}$ as the time required by all the $d$ nodes to complete their respective tasks. Then $T^{(d)}$ would be the maximum of $d$ exponential random variables, that is,

$$T^{(d)} = \max_{1 \leq i \leq d} T_i. \tag{1}$$

**Lemma 1:** For the low arrival rate scenario in a systematic $(n, k)$ MDS-coded storage system, the mean download time for $d$ nodes with direct read is given as

$$\mathrm{E}\left[T^{(d)}\right] = \frac{1}{\mu} \sum_{i=1}^{d} \binom{d}{i} \frac{(-1)^{i+1}}{i}, \tag{2}$$

where $1 \leq d \leq k$.

The proof of Lemma 1 is similar to that of Theorem 1 and due to the lack of space, we omit it here.

*(2) K-access read*

In an $(n, k)$ MDS-coded system, any $k$ out of the $n$ nodes can reconstruct the whole information in a codeword. With k-access read, when a download request arrives, no matter how many files it desires from a codeword, it is forked into $n$ tasks which are sent to each of the $n$ nodes. The request is complete when any $k$ out of $n$ nodes complete their services. Then, the download time $T^{(n,k)}$ is given as

$$T^{(n,k)} = T_{k:n}, \tag{3}$$

where $T_{k:n}$ denotes the $k$th order statistics out of $n$ $i.i.d.$ exponential random variables. According to [8], [9], the mean download time of k-access read is given as

$$\mathrm{E}[T^{(n,k)}] = \frac{1}{\mu} \sum_{i=0}^{k-1} \frac{1}{n-i}. \tag{4}$$

We note that direct read utilizes the systematic property while k-access read uses the MDS property. Hence, we will propose a compound read that can take advantage of both direct read and k-access read.

*(3) Compound read*

In a systematic $(n, k)$ MDS-coded storage system, when a download request which desires $d$ out of the $k$ original data nodes arrives, it is forked into $n$ tasks which are sent to each of the $n$ nodes. The request is complete if the $d$ desired data nodes complete their services or any $k$ out of the remaining $n - d$ nodes complete their services. Let $T^{(d,n-d,k)}$ be the random variable that denotes the time required by compound read, we have

$$T^{(d,n-d,k)} = \min\{\max_{1 \leq i \leq d} T_i, T_{k:n-d}\}. \tag{5}$$

This allows us to get the following result for the mean download time.

**Theorem 1:** For the low arrival rate scenario in a systematic $(n, k)$ MDS-coded storage system, the mean download time for $d$ nodes with compound read is given as

$$\mathrm{E}[T^{(d,n-d,k)}] = \frac{1}{\mu C(n,d)} \sum_{i=0}^{k-1} \frac{C(n,d) - C(i,d)}{n-d-i}, \tag{6}$$

where $C(n,d) = \prod_{j=0}^{d-1} (n-j)$, $C(i,d) = \prod_{j=1}^{d} (i+j)$, $1 \leq d \leq \min(k, n-k)$ and $\frac{C(n,d)-C(i,d)}{n-d-i}$ is a polynomial.

*Proof:* The complimentary CDF of $T^{(d,n-d,k)}$ can be given by

$$\begin{aligned}
& P\left(T^{(d,n-d,k)} > t\right) \\
= \; & P\left(\min\{\max_{1\leq i\leq d} T_i, T_{k:n-d}\} > t\right) \\
= \; & P\left(\{\max_{1\leq i\leq d} T_i > t\} \cap \{T_{k:n-d} > t\}\right) \\
\overset{(b1)}{=} \; & P\left(\{\max_{1\leq i\leq d} T_i > t\}\right) P\left(\{T_{k:n-d} > t\}\right) \\
= \; & (1 - P\left(\{\max_{1\leq i\leq d} T_i \leq t\}\right))(1 - P\left(T_{k:n-d} \leq t\right)) \\
\overset{(b2)}{=} \; & (1 - (P\left(T_i \leq t\right))^d)(P\left(T_{k:n-d} > t\right)) \\
\overset{(b3)}{=} \; & (1 - (1 - e^{-\mu t})^d) \sum_{i=0}^{k-1} \tbinom{n-d}{i} (1 - e^{-\mu t})^i e^{-\mu t(n-d-i)} \\
= \; & \sum_{i=0}^{k-1} ((\tbinom{n-d}{i} (1 - e^{-\mu t})^i e^{-\mu t(n-d-i)} \\
& - \sum_{i=0}^{k-1} \tbinom{n-d}{i} (1 - e^{-\mu t})^{d+i} e^{-\mu t(n-d-i)} \\
= \; & P_1(t) - P_2(t),
\end{aligned}$$

where (b1) follows from the independence of events $\{\max_{1\leq i\leq d} T_i > t\}$ and $\{T_{k:n-d} > t\}$, (b2) follows from the independence of the $d \; T_i's$. To obtain (b3), we use the standard expression for the CDF of the $k$th order statistics of $n-d$ independent random variables as

$$P\left(\{T_{k:n-d} \leq t\}\right) = \sum_{i=k}^{n-d} \tbinom{n-d}{i} [P\left(T_i \leq t\right)]^i [P\left(T_i > t\right)]^{n-d-i},$$

then, we have

$$\begin{aligned}
P\left(\{T_{k:n-d} > t\}\right) & = 1 - P\left(\{T_{k:n-d} \leq t\}\right) \\
& = \sum_{i=0}^{k-1} \tbinom{n-d}{i} [P\left(T_i \leq t\right)]^i [P\left(T_i > t\right)]^{n-d-i}
\end{aligned}$$

Finally, (b3) also follows from the fact that each $T_i \sim \exp(\mu)$.

Then, we can compute the mean download time as follows

$$\begin{aligned}
\mathrm{E}\left[T^{(d,n-d,k)}\right] & = \int_{t=0}^{\infty} P\left(T^{(d,n-d,k)} > t\right) dt \\
& = \int_{t=0}^{\infty} P_1(t) dt - \int_{t=0}^{\infty} P_2(t) dt \\
& = E\left[T_1\right] - E\left[T_2\right].
\end{aligned}$$

Then, we can compute $E\left[T_1\right]$ and $E\left[T_2\right]$, respectively.

$$\begin{aligned}
E\left[T_1\right] & = \sum_{i=0}^{k-1} \tbinom{n-d}{i} \int_{t=0}^{\infty} \left(1 - e^{-\mu t}\right)^i e^{-\mu t(n-d-i)} dt \\
& \overset{(c1)}{=} \frac{1}{\mu} \sum_{i=0}^{k-1} \tbinom{n-d}{i} \int_{x=0}^{1} x^i (1-x)^{n-d-i-1} dx \\
& \overset{(c2)}{=} \frac{1}{\mu} \sum_{i=0}^{k-1} \tbinom{n-d}{i} \beta(i+1, n-d-i) \\
& \overset{(c3)}{=} \frac{1}{\mu} \sum_{i=0}^{k-1} \frac{1}{n-d-i},
\end{aligned}$$

where (c1) can be obtained by substituting $1 - e^{-\mu t} = x$, (c2) follows from the definition of the beta function $\beta(x,y) = \int_0^1 v^{x-1}(1-v)^{y-1} dv$. Finally, c(3) is obtained by using $\beta(x,y) = \frac{(x-1)!(y-1)!}{(x+y-1)!}$, where $x$ and $y$ are positive integers.

Similarly, we can get

$$E\left[T_2\right] = \frac{1}{\mu} \frac{(n-d)!}{n!} \sum_{i=0}^{k-1} \frac{(i+d)!}{i!(n-d-i)}.$$

Accordingly, we have

$$\begin{aligned}
\mathrm{E}\left[T^{(d,n-d,k)}\right] & = E\left[T_1\right] - E\left[T_2\right] \\
& = \frac{(n-d)!}{\mu n!} \sum_{i=0}^{k-1} \left[\frac{n!}{(n-d)!(n-d-i)} - \frac{(i+d)!}{i!(n-d-i)}\right] \\
& = \frac{1}{\mu \prod_{j=0}^{d-1} (n-j)} \sum_{i=0}^{k-1} \frac{\prod_{j=0}^{d-1} (n-j) - \prod_{j=1}^{d} (i+j)}{n-d-i} \\
& = \frac{1}{\mu C(n,d)} \sum_{i=0}^{k-1} \frac{C(n,d) - C(i,d)}{n-d-i}.
\end{aligned}$$

In a systematic $(n,k)$ MDS-coded system, there are only $k$ original data nodes, so $d \leq k$. If $k > n-d$, then the remaining $n-d$ nodes are insufficient to reconstruct the information of the whole codeword, therefore, $d \leq n-k$. Hence, $1 \leq d \leq \min(k, n-k)$.

When $n = d + i$,

$$\begin{aligned}
C(n,d) - C(i,d) & = \frac{n!}{(n-d)!} - \frac{(i+d)!}{i!} \\
& = \frac{(i+d)!}{i!} - \frac{(i+d)!}{i!} \\
& = 0.
\end{aligned}$$

Therefore $i+d$ is a root of $C(n,d) - C(i,d)$ and by the Factor Theorem, $\frac{C(n,d)-C(i,d)}{n-d-i}$ is a polynomial. ■

*Corollary 1:* In the conditions of Theorem 1, when $d = 1$, $\mathrm{E}\left[T^{(1,n-1,k)}\right] = \frac{k}{\mu n}$; when $d = 2$, $\mathrm{E}\left[T^{(2,n-2,k)}\right] = \frac{k}{\mu n} \frac{2n+k+1}{2(n-1)}$; when $d = 3$, $\mathrm{E}\left[T^{(3,n-3,k)}\right] = \frac{k}{\mu n} \frac{6n^2+3(k-1)n+2(k+1)(k+2)}{6(n-1)(n-2)}$.

The results of Corollary 1 can be obtained from Theorem 1 and Corollary 1 obtains the same result as [5] for the case when $d = 1$. This demonstrates that the result on MDS codes in [5] is a special case of this paper.

Note that under the same conditions, the mean download time of direct read is not smaller than that of compound read since the extra k-access read will help reduce the mean download time. Next, we will compare the mean download time for compound read and k-access read.

*Corollary 2:* For the low arrival rate scenario in a systematic $(n,k)$ MDS-coded storage system, the mean download time for the case of $d = 1$ with compound read is smaller than that of k-access read, *i.e.*, $\mathrm{E}[T^{(1,n-1,k)}] < \mathrm{E}[T^{(n,k)}]$; while for $d = k$, $\mathrm{E}[T^{(k,n-k,k)}] \geq \mathrm{E}[T^{(n,k)}]$.

*Proof:* We can obtain $\mathrm{E}[T^{(n,k)}] = \frac{1}{\mu} \sum_{i=0}^{k-1} \frac{1}{n-i}$ by Eq. (4). Therefore, $\mathrm{E}\left[T^{(n,k)}\right] > \frac{1}{\mu} \sum_{i=0}^{k-1} \frac{1}{n} = \frac{k}{\mu n} = \mathrm{E}\left[T^{(1,n-1,k)}\right]$ by Corollary 1.

For $d = k$ with compound read, the download request completes when the $k$ data nodes complete their services or any $k$ out of the remaining $n-k$ nodes complete their services. While for k-access reads, any $k$ out of all $n$ nodes completing their services will result in the request completed. Obviously, $\mathrm{E}[T^{(k,n-k,k)}] \geq \mathrm{E}[T^{(n,k)}]$. ■

Note that, for fixed $n$ and $k$, as $d$ increases, $\mathrm{E}\left[T^{(d,n-d,k)}\right]$ will increase while $\mathrm{E}[T^{(n,k)}]$ remains the same. Moreover,

**Algorithm 1** C & K

```
In a systematic $(n,k)$ MDS-coded storage system, compute $d^*$ according to Eq. (4) and Theorem 1.
When a download request desiring $d$ data nodes arrives, run C & K().
function C & K()
    if $d \leq d^*$ then
        take compound read
    else
        take k-access read
    end if
end function
```

by Corollary 2, $\mathrm{E}[T^{(1,n-1,k)}] < \mathrm{E}[T^{(n,k)}] \leq \mathrm{E}[T^{(k,n-k,k)}]$. Hence, there must be two integers $i, i+1 \in [1,k]$, such that $\mathrm{E}[T^{(i,n-i,k)}] \leq \mathrm{E}[T^{(n,k)}] \leq \mathrm{E}[T^{(i+1,n-i-1,k)}]$. Unfortunately, by the results from Eq. (4) and Theorem 1, we cannot obtain the explicit solution of $d$ from $\mathrm{E}[T^{(d,n-d,k)}] = \mathrm{E}[T^{(n,k)}]$. However, given specific $n$ and $k$, by the results from Eq. (4) and Theorem 1, we can easily obtain explicit solutions of $d$ for the equation $\mathrm{E}[T^{(d,n-d,k)}] = \mathrm{E}[T^{(n,k)}]$. Consequently, there must be a number $d^*$ satisfying the definition as follows.

*Definition 1(Integer $d^*$):* For the low arrival rate scenario in a systematic $(n,k)$ MDS-coded storage system, for any download request for $d$ data nodes, when $d \leq d^*$, $\mathrm{E}[T^{(d,n-d,k)}] < \mathrm{E}[T^{(n,k)}]$ and when $d > d^*$, $\mathrm{E}[T^{(d,n-d,k)}] \geq \mathrm{E}[T^{(n,k)}]$.

Then, we can propose a scheme to reduce mean download time by Compound read and K-access read (C & K) as illustrated in Algorithm 1.

Given specific $(n,k)$, we can easily compute the corresponding $d^*$ by Eq. (4) and Theorem 1. Due to the lack of space, in Table I, we only give the values of $d^*$ for different $n$ and $k$, where $2 \leq n \leq 16$. As an example, for systematic (14,10) MDS code (used in HDFS-RAID in Facebook [10]), from Table I, we get its $d^* = 2$. With Algorithm 1, when any download request for $d$ data nodes arrives, if $d \leq 2$, we should take compound read, otherwise we should take k-access read.

### B. High Arrival Rate Scenario

As in the low arrival rate scenario, we suppose that the service time at each node is an *i.i.d.* exponential random variable with mean $\frac{1}{\mu}$. Similar to previous work [2], [5], [11], we assume that download requests arrive as a Poisson process with aggregate rate $\lambda$. As with existing research [2], [5], it is widely shared that fork-join queue is an excellent choice to describe the work flow in a systematic MDS-coded storage system as illustrated in Fig. 2. However, fork-join (FJ) queue is difficult to analyze and generally only bounds can be found for the mean download time [12].

Similar to previous work [2], [5], [9], we also analyze the upper bound with a more restricted queueing model, split-merge (SM) system. In an FJ queue, when a node completes serving a task, it can start serving the task of the next request immediately. However, in an SM system, only when the current request is completed, can idle nodes in the codeword begin to serve the task of the next request. Therefore, due to the blocking of idle nodes, the mean download time of SM system is an upper bound of the corresponding FJ system.

TABLE I. DIFFERENT VALUES OF $d^*$ FOR VARIOUS $(n,k)$.

| $(n,k)$ | $d^*$ | $(n,k)$ | $d^*$ | $(n,k)$ | $d^*$ | $(n,k)$ | $d^*$ |
|---|---|---|---|---|---|---|---|
| (2,1) | 1 | (9,3) | 1 | (12,6) | 1 | (14,13) | 1 |
| (3,1) | 1 | (9,4) | 1 | (12,7) | 1 | (15,1) | 1 |
| (3,2) | 1 | (9,5) | 1 | (12,8) | 2 | (15,2) | 1 |
| (4,1) | 1 | (9,6) | 1 | (12,9) | 2 | (15,3) | 1 |
| (4,2) | 1 | (9,7) | 2 | (12,10) | 2 | (15,4) | 1 |
| (4,3) | 1 | (9,8) | 1 | (12,11) | 1 | (15,5) | 1 |
| (5,1) | 1 | (10,1) | 1 | (13,1) | 1 | (15,6) | 1 |
| (5,2) | 1 | (10,2) | 1 | (13,2) | 1 | (15,7) | 1 |
| (5,3) | 1 | (10,3) | 1 | (13,3) | 1 | (15,8) | 1 |
| (5,4) | 1 | (10,4) | 1 | (13,4) | 1 | (15,9) | 2 |
| (6,1) | 1 | (10,5) | 1 | (13,5) | 1 | (15,10) | 2 |
| (6,2) | 1 | (10,6) | 1 | (13,6) | 1 | (15,11) | 2 |
| (6,3) | 1 | (10,7) | 2 | (13,7) | 1 | (15,12) | 2 |
| (6,4) | 1 | (10,8) | 2 | (13,8) | 1 | (15,13) | 2 |
| (6,5) | 1 | (10,9) | 1 | (13,9) | 2 | (15,14) | 1 |
| (7,1) | 1 | (11,1) | 1 | (13,10) | 2 | (16,1) | 1 |
| (7,2) | 1 | (11,2) | 1 | (13,11) | 2 | (16,2) | 1 |
| (7,3) | 1 | (11,3) | 1 | (13,12) | 1 | (16,3) | 1 |
| (7,4) | 1 | (11,4) | 1 | (14,1) | 1 | (16,4) | 1 |
| (7,5) | 1 | (11,5) | 1 | (14,2) | 1 | (16,5) | 1 |
| (7,6) | 1 | (11,6) | 1 | (14,3) | 1 | (16,6) | 1 |
| (8,1) | 1 | (11,7) | 1 | (14,4) | 1 | (16,7) | 1 |
| (8,2) | 1 | (11,8) | 2 | (14,5) | 1 | (16,8) | 1 |
| (8,3) | 1 | (11,9) | 2 | (14,6) | 1 | (16,9) | 2 |
| (8,4) | 1 | (11,10) | 1 | (14,7) | 1 | (16,10) | 2 |
| (8,5) | 1 | (12,1) | 1 | (14,8) | 1 | (16,11) | 2 |
| (8,6) | 1 | (12,2) | 1 | (14,9) | 2 | (16,12) | 2 |
| (8,7) | 1 | (12,3) | 1 | (14,10) | 2 | (16,13) | 2 |
| (9,1) | 1 | (12,4) | 1 | (14,11) | 2 | (16,14) | 2 |
| (9,2) | 1 | (12,5) | 1 | (14,12) | 2 | (16,15) | 1 |

Note: $2 \leq n \leq 16$ and $1 \leq k < n$.

**Theorem 2** *(Upper Bound on Mean Download Time).* For a fork-join queueing system using a systematic $(n,k)$ MDS code, the mean download time for $d$ nodes with compound read satisfies

$$\mathrm{E}[T^{(d,n-d,k)}] \leq \mathrm{E}[T] + \frac{\lambda \mathrm{E}[T^2]}{2(1-\lambda \mathrm{E}[T])}, \tag{7}$$

where $\mathrm{E}[T] = \frac{1}{\mu \prod_{j=0}^{d-1}(n-j)} \sum_{i=0}^{k-1} \frac{\prod_{j=0}^{d-1}(n-j) - \prod_{j=1}^{d}(i+j)}{n-d-i}$ and $\mathrm{E}[T^2] = \sum_{i=1}^{d} \binom{d}{i}(-1)^{i+1} \sum_{j=0}^{k-1} \binom{n-d}{j} \sum_{m=0}^{j} \binom{j}{m}(-1)^m \frac{2}{\mu^2(n+i+m-d-j)^2}$.

*Proof:* The SM queueing system is equivalent to an $M/G/1$ queue with Poisson arrival rate $\lambda$ and generalized service time which equals that of the low arrival scenario, i.e., $T^{(d,n-d,k)}$. Then, we can use the Pollaczek-Khinichin formula to obtain the mean download time $\mathrm{E}[T_{SM}]$ of an $M/G/1$ queue as follows:

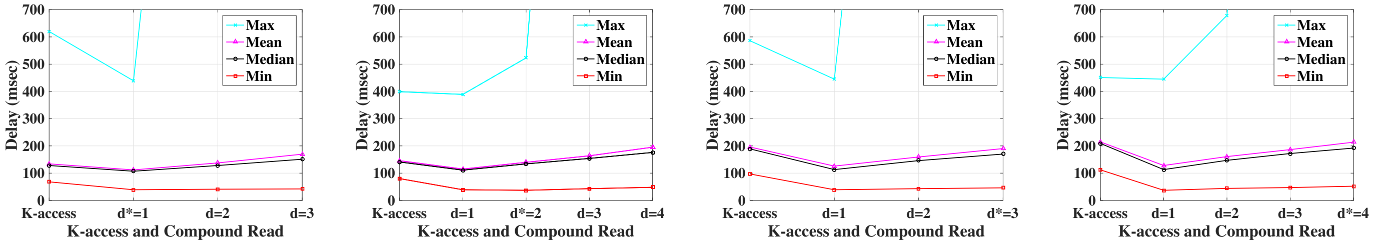
$$\mathrm{E}[T_{SM}] = \mathrm{E}[T] + \frac{\lambda \mathrm{E}[T^2]}{2(1-\lambda \mathrm{E}[T])}, \tag{8}$$

where $\mathrm{E}[T] = \mathrm{E}[T^{(d,n-d,k)}]$ in the low arrival rate scenario.

Then, we need to compute $\mathrm{E}[T^2]$ as follows.

$$\begin{aligned} \mathrm{E}[(T^{(d,n-d,k)})^2] &= \int_{t=0}^{\infty} P\left(T^{(d,n-d,k)} > t\right) d(t^2) \\ &= \int_{t=0}^{\infty} 2tP\left(T^{(d,n-d,k)} > t\right) dt. \end{aligned}$$

To obtain this integration, we need a new complimentary CDF



(a) MDS(9,6) d*=1 (b) MDS(14,10) d*=2 (c) MDS(21,18) d*=3 (d) MDS(34,30) d*=4

Fig. 3. Delay performance comparison of codes with k-access read and compound read with different $d$ and $d^*$.

of $T^{(d,n-d,k)}$ based on the proof of Theorem 1.

$$\begin{aligned}
&P\left(T^{(d,n-d,k)} > t\right)\\
\overset{(d1)}{=}\ & (1-(1-e^{-\mu t})^d)\sum_{i=0}^{k-1}\binom{n-d}{i}(1-e^{-\mu t})^i e^{-\mu t(n-d-i)}\\
\overset{(d2)}{=}\ & \sum_{i=1}^{d}\binom{d}{i}(-1)^{i+1}e^{-\mu it}\sum_{i=0}^{k-1}\binom{n-d}{i}(1-e^{-\mu t})^i e^{-\mu t(n-d-i)}\\
=\ & \sum_{i=1}^{d}\binom{d}{i}(-1)^{i+1}\sum_{j=0}^{k-1}\binom{n-d}{j}\sum_{m=0}^{j}\binom{j}{m}(-1)^m C(t),
\end{aligned}$$

where $C(t) = e^{-\mu(n+m+i-d-j)t}$, (d1) follows from the proof of Theorem 1. (d2) follows from the binomial expansion of $(1-(1-e^{-\mu t})^d)$.

Then, we have,

$$\begin{aligned}
&\mathrm{E}[(T^{(d,n-d,k)})^2]\\
\overset{(d3)}{=}\ & \sum_{i=1}^{d}\binom{d}{i}(-1)^{i+1}\sum_{j=0}^{k-1}\binom{n-d}{j}\sum_{m=0}^{j}\binom{j}{m}(-1)^m D(t),
\end{aligned}$$

where $D(t) = \int_{t=0}^{\infty} 2tC(t)dt = \frac{2}{\mu^2(n+i+m-d-j)^2}$ and (d3) follows from interchanging the order of integration and summation. Therefore, with the values of $\mathrm{E}[T^{(d,n-d,k)}]$ and $\mathrm{E}[(T^{(d,n-d,k)})^2]$, we can easily compute the upper bound $\mathrm{E}[T_{SM}]$ by Eq. (8). ■

**Theorem 3** *(Lower Bound on Mean Download Time).* For a fork-join queueing system using a systematic $(n,k)$ MDS code, the mean download time for $d$ nodes with compound read satisfies

$$\mathrm{E}[T^{(d,n-d,k)}] \geq \frac{1}{\mu}\sum_{i=0}^{d-1}\frac{1}{(n-i)-\rho}, \tag{9}$$

where $\rho = \frac{\lambda}{\mu}$.

*Proof:* We use a similar method as the proof of k-access read in [2]. For compound read, a download request is finished if the $d$ systematic nodes complete their services or any $k$ out of the remaining $n-d$ nodes complete their services. Since $d \leq k$, the lower bound is derived by considering that a request is finished by the first $d$ completed tasks, i.e., there are only $d$ stages and each stage completes a task at one of the desired data nodes. Moreover, the lower bound also considers that at first, the $0^{th}$ stage, the service rate of a request is equivalent to $(n-0)\mu$ and at the $i^{th}$ stage, the service rate of a request is equivalent to $(n-i)\mu$. Hence, the mean time for a request to move from the $i^{th}$ to $(i+1)^{th}$ stage is lower bounded by $\frac{1}{(n-i)\mu-\lambda}$. Therefore, the total mean download time is the sum of the mean time of the $d$ stages and is bounded as follows,

$$\mathrm{E}[T^{(d,n-d,k)}] \geq \sum_{i=0}^{d-1}\frac{1}{(n-i)\mu-\lambda} \geq \frac{1}{\mu}\sum_{i=0}^{d-1}\frac{1}{(n-i)-\rho}.$$

■

## III. Performance Evaluation

The results in the last section rely on the assumption of exponential service times. In this part, we evaluate them with real service time traces from Amazon S3. These traces are for read files of 1MB in size from an S3 bucket, located in Northern California.

### A. Low Arrival Rate Scenario

In systems with systematic $(n,k)$ MDS coding, we use different values of $(n,k)$ and find that they exhibit similar results. Due to the lack of space, we only display the results of (9,6), (14,10), (21,18) and (34,30) MDS coding, and their corresponding $d^*$ are 1, 2, 3, and 4, respectively. We take the average delay over 1 million sample paths for each experiment.

In Fig. 3 $(a)$, $(b)$, $(c)$ and $(d)$, we compare the delay performance of k-access read and compound read with different values of $d$. We observe that, when $d \leq d^*$, for each case, both mean and median delays of compound reads are lower than those of k-access reads. This validates our theoretical analyses for compound read and manifests the effectiveness of C & K scheme in practice.

From Fig. 3, we can also observe that the median delay is invariably smaller than the mean delay for both k-access and compound reads, and we can infer that the real service time distribution is positively skewed. It is also noted that the minimum delay of compound read is always smaller than that of k-access. This is in line with our analysis since compound read can make use of direct read from the $d$ required nodes, for $1 \leq d \leq k$, thus achieving lower minimum delay compared with k-access read. We also observe that other than the case when $d = 1$, the maximum delay of compound read is much higher than that of k-access read. This implies that even though compound read can realize lower mean, median and minimum delays when $d \leq d^*$, it suffers from a potentially much higher maximum delay compared with k-access read when $d \neq 1$. Besides, the maximum results for $d = 1$ further strengthen the result in Corollary 2 and demonstrate that compound read is superior to k-access read when $d = 1$.

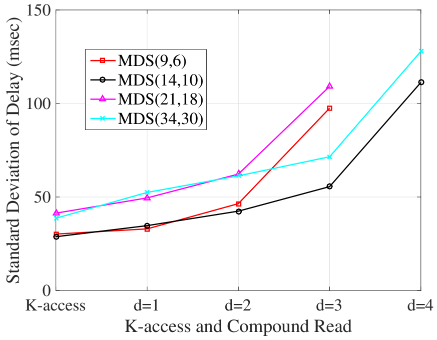


Fig. 4. The volatility of delay performance for codes with different read methods.

Next, we examine the volatility of delay performance with the standard deviation of each case. As illustrated in Fig. 4, k-access read can invariably achieve lower volatility in terms of delay performance compared with that of compound read. This is because compound read may achieve lower delay by direct read from the $d$ required nodes, thus resulting in higher volatility. It is also noted that the bigger $d$ is, the higher the volatility of compound read delay. In practice, when using compound read and k-access read, we need to consider this especially when users have requirements on delay volatility.

### *B. High Arrival Rate Scenario*

We suppose that the download requests to a codeword of MDS(10,$k$), where $2 \leq k \leq 4$, arrive as a Poisson process with parameter $\lambda$. In our simulations, we set $\lambda = 1$ request/sec, and the number of arrivals is set to be 10000. We assume the service time of each task is exponential with $\mu = 1$ task/sec. We take the average delay over 1000 sample paths for each experiment.

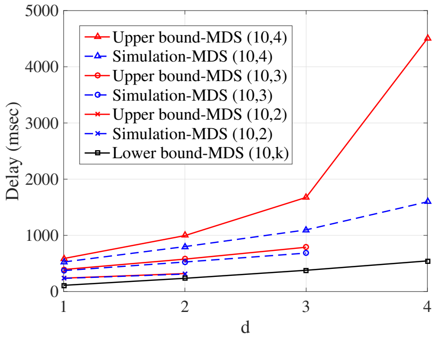


Fig. 5. Delay performance in high arrival rate scenario when $\lambda = 1$ and $\mu = 1$ as $d$ increases for MDS(10,4), MDS(10,3) and MDS(10,2) with compound read.

Fig. 5 illustrates the upper and lower bounds and simulations of delay performance in high arrival rate scenario. The upper and lower bounds are numerical results when $\lambda = 1$ and $\mu = 1$ by Theorem 2 and 3. By Theorem 3, the lower bound of the mean download time is independent of $k$, therefore, MDS(10,4), MDS(10,3) and MDS(10,2) share the same lower bound in Fig. 5 for the same $d$.

From Fig. 5, under the same conditions, the bigger $d$ is, the looser the bounds. It is also noted that for the same $n = 10$, the smaller $k$ is, the tighter the bounds. We know that for the same $n$, smaller $k$ means higher storage cost. Here we actually obtain a tradeoff between the storage cost and download delay for a systematic $(n, k)$ MDS-coded storage system with compound read.

## IV. Conclusion

We presented the compound read that combines direct read and k-access read for flexibly downloading different sizes of files from a systematic MDS-coded distributed storage system to achieve lower mean and median delays. We characterized its mean download delay in low arrival rate scenario and derived upper and lower bounds on its mean download delay in high arrival rate scenario. We also analyzed the differences in terms of delay performance between compound and k-access reads and developed a scheme C & K to dynamically take advantage of them according to users' required size of files to reduce download delay. Furthermore, via simulations using real service time traces from Amazon S3, our evaluations verified our theoretical analyses and demonstrated the effectiveness of the C & K scheme.

## Acknowledgment

The authors would like to thank Dr. Guanfeng Liang for sharing the traces used in the performance evaluation.

## References

[1] S. Ghemawat, H. Gobioff, and S.-T. Leung, "The Google file system," in *ACM SIGOPS Operating Systems Review*, vol. 37, no. 5. ACM, 2003, pp. 29–43.

[2] G. Joshi, Y. Liu, and E. Soljanin, "On the delay-storage trade-off in content download from coded distributed storage systems," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 5, pp. 989–997, 2014.

[3] C. Huang, H. Simitci, Y. Xu, A. Ogus, B. Calder, P. Gopalan, J. Li, S. Yekhanin *et al.*, "Erasure coding in Windows Azure storage," in *USENIX ATC*, 2012.

[4] Q. Shuai and V. O. K. Li, "Delay performance of direct reads in distributed storage systems with coding," in *IEEE 17th International Conference on High Performance Computing and Communications (HPCC)*, Aug 2015, pp. 184–189.

[5] S. Kadhe, E. Soljanin, and A. Sprintson, "Analyzing the download time of availability codes," in *Information Theory (ISIT), 2015 IEEE International Symposium on*. IEEE, 2015, pp. 1467–1471.

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

[7] Q. Shuai and V. O. K. Li, "HTSC and FH_HTSC: XOR-based codes to reduce access latency in distributed storage systems," *Journal of Communications and Networks*, vol. 17, no. 6, pp. 582–591, Dec 2015.

[8] R. Sheldon *et al.*, *A first course in probability*. Pearson Education India, 2002.

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

[10] D. Borthakur, R. Schmidt, R. Vadali, S. Chen, and P. Kling, "HDFS RAID," in *Hadoop User Group Meeting*, 2010.

[11] Q. Shuai, V. O. K. Li, and Y. Zhu, "Performance models of access latency in cloud storage systems," in *Fourth Workshop on Architectures and Systems for Big Data*, 2014.

[12] R. Nelson and A. N. Tantawi, "Approximate analysis of fork/join synchronization in parallel queues," *Computers, IEEE Transactions on*, vol. 37, no. 6, pp. 739–743, 1988.