Distributed Storage Allocation for High Reliability

Derek Leong
Department of Electrical Engineering
California Institute of Technology
Pasadena, California 91125, USA
derekleong@caltech.edu

Alexandros G. Dimakis
Department of Electrical Engineering
University of Southern California
Los Angeles, California 90089, USA
dimakis@usc.edu

Tracey Ho
Department of Electrical Engineering
California Institute of Technology
Pasadena, California 91125, USA
tho@caltech.edu

***Abstract*—We consider the problem of optimally allocating a given total storage budget in a distributed storage system. A source has a data object which it can code and store over a set of storage nodes; it is allowed to store any amount of data in each storage node, subject to a given total storage budget constraint. A data collector subsequently attempts to recover the original data object by accessing a random fixed-size subset of these storage nodes. Successful recovery of the data object occurs when the total amount of coded data in this subset of storage nodes is at least the size of the original data object. The goal is to determine the amount of data to store in each storage node so that the probability of successful recovery is maximized. We solve this problem in the *high recovery probability* regime. Our results can be applied to a variety of distributed storage systems, including delay tolerant networks (DTNs), content delivery networks (CDNs), and sensor networks.**

I. Introduction

How should we store a data object over a set of storage nodes so that it can be recovered with high reliability? Consider a distributed storage system consisting of n storage nodes that fail probabilistically. A source has a data object of unit size which it can code and store in a distributed manner over these storage nodes. For instance, it could split the data object into multiple chunks and then replicate them redundantly over the storage nodes. Let x_i be the amount of data stored in storage node $i \in \{1, \ldots, n\}$. Although any amount of data may be stored in each storage node, the total amount of storage used must not exceed a given budget T, that is, $\sum_{i=1}^{n} x_i \leq T$. This is a realistic constraint when there is limited transmission bandwidth or storage, or if it is too costly to mirror the data object in its entirety in every storage node.

At some time after the creation of this coded storage, a data collector attempts to recover the original data object by accessing only a random subset $\mathbf{r}$ of the storage nodes, where $\mathbf{r}$ is to be specified by the assumed failure model or access model. Fig. 1 depicts such a distributed storage system.

Two questions can be posed for this system: (i) how should we allocate the given total storage budget among the storage nodes, that is, determine the values of $x_1, \ldots, x_n$, and (ii) how do we design a coding scheme for such an allocation?

This material is based upon work under a subcontract #069153 issued by BAE Systems National Security Solutions, Inc. and supported by the Defense Advanced Research Projects Agency (DARPA) and the Space and Naval Warfare System Center (SPAWARSYSCEN), San Diego under Contract No. N66001-08-C-2013.

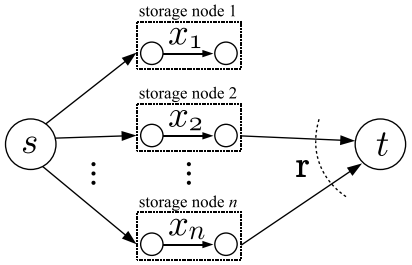


Fig. 1. Information flows in a distributed storage system. The source s has a data object of unit size which it can code and store over n storage nodes. Subsequently, a data collector t attempts to recover the original data object by accessing only a random subset $\mathbf{r}$ of the storage nodes.

It turns out that these two problems can be decoupled by using a good coding scheme that enables successful recovery whenever the amount of data accessed by the data collector is at least the size of the data object. This can be seen by considering the information flows in the network where the source is multicasting to some set of potential data collectors [1], [2]; successful recovery is possible for a data collector iff its max-flow or min-cut is at least the size of the data object. The storage of random linear combinations of data packets over a sufficiently large field, for example, would allow us to achieve such recovery with high probability [3], [4].

Therefore, assuming the use of an appropriate code, the probability of successful recovery for an allocation $\{x_1, \ldots, x_n\}$ can be written as

$$\mathbb{P}[\text{successful recovery}] = \mathbb{P}\left[\sum_{i \in \mathbf{r}} x_i \geq 1\right].$$

We can now consider the distributed storage allocation problem of finding an optimal allocation that maximizes this probability of successful recovery, subject to a given total storage budget constraint. The solution to this problem can be applied to a variety of distributed storage systems in which the storage allocation is an optimization variable, including delay tolerant networks (DTNs), content delivery networks (CDNs), and sensor networks [5]–[8].

Although the problem appears very basic, its theoretical analysis presents significant challenges. Consider a natural failure model in which each of the n storage nodes fails with some probability. We can define a random variable $Y_i \sim \text{Bernoulli}(p_i)$ for each storage node i, and choose the

random subset $\mathbf{r}$ to contain storage node i iff $Y_i = 1$. As a simplification, suppose that the n Bernoulli random variables are i.i.d. with $p_i = p$. The resulting problem is an open problem discussed by several people at UC Berkeley [9], and it is known that the optimal allocation that maximizes the probability of successful recovery can be quite complicated in general. One may expect to always find an optimal allocation that is *symmetric*, i.e. with all nonzero x_i being equal, but this intuition is incorrect. For instance, the following counterexample (originally from [9]) shows that symmetric allocations can be suboptimal: given $(n, p, T) = \left(5, 0.9, \frac{12}{5}\right)$, the nonsymmetric allocation $\left\{\frac{3}{5}, \frac{3}{5}, \frac{2}{5}, \frac{2}{5}, \frac{2}{5}\right\}$ yields a recovery probability of 0.99711, which is strictly greater than the corresponding probabilities for the five symmetric allocations, of which $\left\{\frac{3}{5}, \frac{3}{5}, \frac{3}{5}, \frac{3}{5}, 0\right\}$ achieves the highest recovery probability of 0.9963. The problem is nontrivial even if we restrict our optimization to only *symmetric* allocations [10].

Our Contribution: In this paper, we examine the case where the subset $\mathbf{r}$ is selected uniformly at random from the collection of all possible r-subsets, where r is a given fixed constant. This access model can be considered to be an approximation to the i.i.d. Bernoulli model if we set $r = n\,p$. Finding the optimal allocation in general for this case is still challenging; in fact, even the task of determining whether a given allocation yields a recovery probability that is at least a specified threshold can be shown to be computationally challenging.

Our main result is that we can find the optimal allocation for the *high recovery probability* regime. We do this by showing that a particular allocation remains optimal when the required recovery probability exceeds a specified threshold. This allocation is the optimal allocation that minimizes the total storage budget when we require the recovery probability to be exactly 1 (i.e. *all* r-subsets of $\{x_1, \ldots, x_n\}$ must have a sum of at least one), which is the solution to a linear program. We investigate how much we can lower the required recovery probability before the optimal solution changes; this corresponds to dropping constraints from the linear program. While the form of the derived lower bound on the recovery probability depends on n and r in a complicated way, we can conclude that for any r, this allocation is optimal if the recovery probability is to exceed $1 - \frac{1}{n}$.

This work builds on the basic results from our earlier paper [11]. In the next section, we define the problem formally and state our main results, which are then proved in the following section.

II. Problem Definition and Main Results

We adopt the following notation throughout the paper:

n	total number of storage nodes, $n \geq 1$
r	number of storage nodes accessed by the data collector, $1 \leq r \leq n$
N	the n-set $\{1, \ldots, n\}$
x_i	amount of data stored in storage node $i \in N$
X	storage allocation $\{x_i\}_{i \in N}$
$\mathbf{r}$	an r-subset of N
$\mathcal{R}$	a collection of distinct r-subsets of N, $0 \leq \lvert\mathcal{R}\rvert \leq \binom{n}{r}$
T	total storage budget, $\sum_{i \in N} x_i \leq T$
$\mathbf{I}[G]$	$\mathbf{I}[G] = 1$ if statement G is true, and 0 otherwise

We consider the storage allocation problem where the data collector accesses an r-subset of nodes selected uniformly at random from the collection of all $\binom{n}{r}$ r-subsets, where r is a given fixed constant. It follows that the probability of successful recovery for a given allocation $\{x_1, \ldots, x_n\}$ can be written as $\sum_{\mathbf{r}} \frac{1}{\binom{n}{r}} \cdot \mathbf{I}\left[\sum_{i \in \mathbf{r}} x_i \geq 1\right]$, where the outer summation is over all r-subsets of N.

There are two ways to express our optimization problem. We can treat the total storage budget T as given, and seek the optimal allocation that maximizes the probability of successful recovery P_{S}:

$$\mathbf{\Pi}(n, r, T): \tag{1}$$

$$\underset{X, \mathcal{R}, P_{\mathrm{S}}}{\text{maximize}} \quad P_{\mathrm{S}} \tag{1.1}$$

subject to

$$\lvert\mathcal{R}\rvert \;\geq\; P_{\mathrm{S}} \binom{n}{r} \tag{1.2}$$

$$\sum_{i \in \mathbf{r}} x_i \geq 1 \qquad \forall\ \mathbf{r} \in \mathcal{R} \tag{1.3}$$

$$\sum_{i \in N} x_i \leq T \tag{1.4}$$

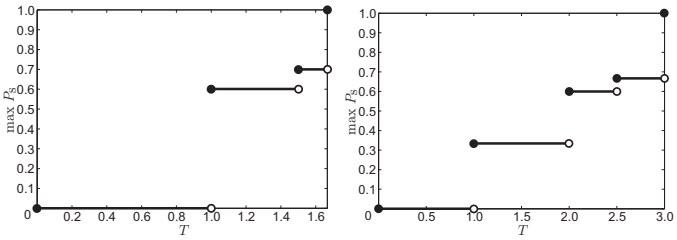
$$x_i \;\geq\; 0 \qquad \forall\ i \in N \tag{1.5}$$

Fig. 2 shows how the maximum value of P_{S} varies with T, for two instances of (n, r). Alternatively, we can treat the desired minimum probability of successful recovery P_{S} as given, and seek the optimal allocation that minimizes the required total storage budget T:

$$\mathbf{\Pi}(n, r, P_{\mathrm{S}}): \tag{2}$$

$$\underset{X, \mathcal{R}, T}{\text{minimize}} \quad T \tag{2.1}$$

$$\text{subject to (1.2)–(1.5)} \tag{2.2–2.5}$$



(a) $(n, r) = (5, 3)$ (b) $(n, r) = (6, 2)$

Fig. 2. Plot of maximum probability of successful recovery $\max P_{\mathrm{S}}$ against total storage budget T, for $(n, r) = (5, 3)$ and $(6, 2)$. These plots are generated by solving a mixed-integer linear program formulation of $\mathbf{\Pi}(n, r, P_{\mathrm{S}})$ at each possible value of P_{S}.

We begin by finding the optimal allocation for the special case of probability-1 recovery:

Theorem 1 (Theorem 1 in [11])**.** *Consider* $\mathbf{\Pi}(n, r, P_S)$ *given by* (2)*, with* $P_S = 1$*. The allocation*

$$x_i = \frac{1}{r}, \quad i \in N,$$

is optimal.

This solution is easy to determine since the budget minimization problem becomes a linear program when $P_S = 1$; we are simply requiring *all* $\binom{n}{r}$ r-subsets to have a sum of at least one. Naturally, we ask if this optimal allocation changes when we relax our recovery probability constraint so that $P_S \approx 1$, i.e. requiring *almost all* r-subsets to have a sum of at least one. Our attempts at specifying the values of P_S for which this allocation is optimal are met with varying degrees of success, depending on n and r. Theorem 2 deals with the case where n is a multiple of r, i.e. $r \mid n$, while Theorem 3 and Corollary 1 deal with the case where n is not a multiple of r, i.e. $r \nmid n$.

When $r \mid n$, we are able to state a necessary and sufficient condition on P_S for the allocation to be optimal:

Theorem 2. *Consider* $\mathbf{\Pi}(n, r, P_S)$ *given by* (2)*, with* $r \mid n$*. The allocation* $x_i = \frac{1}{r}$*,* $i \in N$*, is optimal if and only if*

$$P_S > 1 - \frac{r}{n}. \tag{3}$$

When $r \nmid n$, we are only able to state a sufficient condition on P_S for the allocation to be optimal:

Theorem 3. *Consider* $\mathbf{\Pi}(n, r, P_S)$ *given by* (2)*, with* $r \nmid n$*. Let* α *and* r' *be uniquely defined integers such that*

$$n = \alpha\, r + r', \quad \alpha \in \mathbb{Z}_0^+, \quad r' \in \{r+1, \ldots, 2r-1\}.$$

If

$$P_S > 1 - \frac{\gcd(r, r')}{\alpha \gcd(r, r') + r'},$$

then the allocation $x_i = \frac{1}{r}$*,* $i \in N$*, is optimal.*

If in addition we have $(n - r) \mid n$, then this sufficient condition becomes necessary too:

Corollary 1. *Consider* $\mathbf{\Pi}(n, r, P_S)$ *given by* (2)*, with* $(n - r) \mid n$*. The allocation* $x_i = \frac{1}{r}$*,* $i \in N$*, is optimal if and only if*

$$P_S > \frac{r}{n}. \tag{4}$$

The preceding result allows us to solve the problem *completely* when $(n - r) \mid n$. Fig. 3 describes the performance of the optimal allocation for each total storage budget T.

Corollary 2. *Consider* $\mathbf{\Pi}(n, r, T)$ *given by* (1)*, with* $(n - r) \mid n$*. We can specify an optimal allocation that maximizes* P_S *for each choice of total storage budget* T *as follows:*

$$\begin{aligned}
&\textit{For } 0 \le T < 1: && x_i = 0, \quad i \in N \\
&\textit{For } 1 \le T < \frac{n}{r}: && x_i = \begin{cases} 1 & \textit{if } i = 1, \\ 0 & \textit{if } i \in N \backslash \{1\} \end{cases} \\
&\textit{For } T \ge \frac{n}{r}: && x_i = \frac{1}{r}, \quad i \in N
\end{aligned}$$

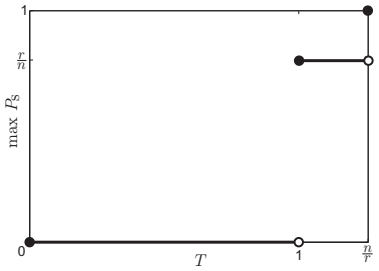


Fig. 3. Plot of maximum probability of successful recovery $\max P_S$ against total storage budget T, for $(n - r) \mid n$.

We can summarize our results with Fig. 4 which shows the intervals of P_S over which the allocation is optimal, for an instance of n.

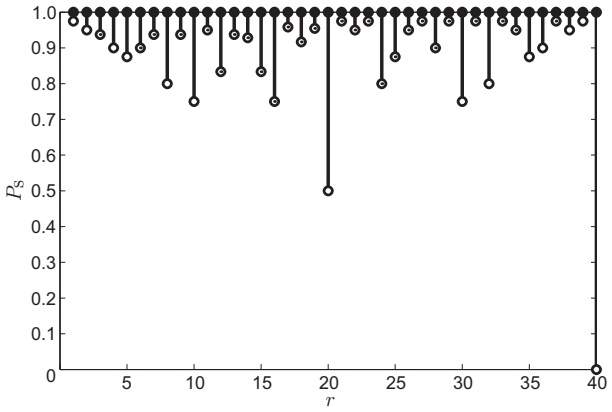


Fig. 4. Intervals of probability of successful recovery P_S over which the allocation $x_i = \frac{1}{r}$, $i \in N$, is optimal for $\mathbf{\Pi}(n, r, P_S)$, for $n = 40$. A dotted circle denotes an endpoint that may not be tight, i.e. we have not demonstrated that the allocation is suboptimal everywhere outside the interval.

III. Analysis

Before proving the theorems stated in the preceding section, we make a simple observation that allows us to lower-bound the amount of storage that must be used in order to satisfy some specific set of constraints:

Lemma 1. *Consider a set* $S \subseteq N$*, and* c *of its subsets* $\mathbf{r}_j \subseteq S$*,* $j = 1, \ldots, c$*. If*

$$\sum_{i \in \mathbf{r}_j} x_i \ge 1, \quad j = 1, \ldots, c, \tag{5}$$

and each element in S *appears the same number of times among the* c *subsets, that is, for some positive integer* b *we have*

$$\sum_{j=1}^{c} \mathbf{I}[i \in \mathbf{r}_j] = b \quad \forall\ i \in S, \tag{6}$$

then

$$\sum_{i \in S} x_i \ge \frac{c}{b}.$$

Proof of Lemma 1: We combine the inequalities in (5) and regroup the variables. Summing up the c inequalities in (5) gives

$$\sum_{j=1}^{c} \sum_{i \in \mathbf{r}_j} x_i \geq c \quad \Longrightarrow \quad \sum_{i \in S} \sum_{j=1}^{c} \mathbf{I}[i \in \mathbf{r}_j]\, x_i \geq c.$$

Substituting (6) into the above inequality produces

$$\sum_{i \in S} b\, x_i \geq c \quad \Longrightarrow \quad \sum_{i \in S} x_i \geq \frac{c}{b}.$$

■

Proof of Theorem 1: We employ a different approach from [11] by using Lemma 1. Let $\mathcal{R}$ be the collection of all $\binom{n}{r}$ r-subsets of N, which is the only feasible choice for $\mathcal{R}$. Observe that each element in N appears the same number of times among the $\binom{n}{r}$ r-subsets. Specifically, the number of distinct r-subsets in $\mathcal{R}$ that contain some element $i \in N$ is just the number of ways to choose the other $(r-1)$ elements in the r-subset from $(n-1)$ remaining elements in $N \backslash \{i\}$, that is,

$$\sum_{\mathbf{r} \in \mathcal{R}} \mathbf{I}[i \in \mathbf{r}] = \binom{n-1}{r-1} \quad \forall\ i \in N.$$

Therefore, applying Lemma 1 with $S = N$, $c = \binom{n}{r}$, and $b = \binom{n-1}{r-1}$ gives

$$\sum_{i \in N} x_i \geq \frac{\binom{n}{r}}{\binom{n-1}{r-1}} = \frac{n}{r}$$

for any feasible allocation X. Now, the allocation $x_i = \frac{1}{r}$, $i \in N$, is a feasible allocation since any r-subset yields a sum of 1; it follows that it is also optimal since it uses the minimum possible total storage $\frac{n}{r}$. ■

Proof of Theorem 2: First, we show that $P_{\mathrm{S}} > 1 - \frac{r}{n}$ is sufficient for the stated allocation to be optimal. Our approach is motivated by the realization that we can pick r-subsets $\mathbf{r}_1, \mathbf{r}_2, \ldots$ so that the corresponding r-subset constraints $\sum_{i \in \mathbf{r}_1} x_i \geq 1$, $\sum_{i \in \mathbf{r}_2} x_i \geq 1$, $\ldots$ forces $\sum_{i \in N} x_i \geq \frac{n}{r}$. Consider the following example for $(n, r) = (4, 2)$: the two (out of the possible six) constraints $\sum_{i \in \{1,2\}} x_i \geq 1$ and $\sum_{i \in \{3,4\}} x_i \geq 1$ would necessitate $\sum_{i \in N} x_i \geq 2 = \frac{n}{r}$. If these r-subsets are found in collection $\mathcal{R}$, then the stated allocation is optimal.

We begin by specifying such a collection of r-subsets, and showing that they result in the optimality of the stated allocation. For brevity, define positive integer $\alpha \triangleq n/r$. Let $Q = (\mathbf{v}_1, \ldots, \mathbf{v}_\alpha)$ be an ordered partition of N that comprises α parts, where $|\mathbf{v}_j| = r$, $j = 1, \ldots, \alpha$. For each ordered partition Q, we specify a collection of α distinct r-subsets $\mathcal{R}_Q = \{\mathbf{r}_1, \ldots, \mathbf{r}_\alpha\}$, where $\mathbf{r}_j = \mathbf{v}_j$, $j = 1, \ldots, \alpha$. Fig. 5 provides an example of how Q and $\mathcal{R}_Q$ are constructed.

We claim that for any such ordered partition Q, if

$$\sum_{i \in \mathbf{r}} x_i \geq 1 \quad \forall\ \mathbf{r} \in \mathcal{R}_Q,$$

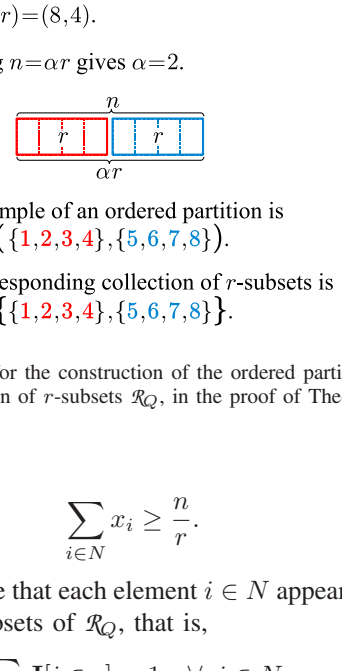


Fig. 5. An example for the construction of the ordered partition Q and its corresponding collection of r-subsets $\mathcal{R}_Q$, in the proof of Theorem 2.

then

$$\sum_{i \in N} x_i \geq \frac{n}{r}.$$

To see this, observe that each element $i \in N$ appears in exactly one of the α r-subsets of $\mathcal{R}_Q$, that is,

$$\sum_{\mathbf{r} \in \mathcal{R}_Q} \mathbf{I}[i \in \mathbf{r}] = 1 \quad \forall\ i \in N.$$

Therefore, applying Lemma 1 with $S = N$, $c = \alpha$, and $b = 1$ gives $\sum_{i \in N} x_i \geq \frac{\alpha}{1} = \frac{n}{r}$.

It follows that if $\mathcal{R}_Q \subseteq \mathcal{R}$ in $\mathbf{\Pi}(n, r, P_{\mathrm{S}})$, then the stated allocation is optimal since any r-subset yields a sum of 1, and the allocation uses the minimum possible total storage $\frac{n}{r}$.

We note that the total number of possible ordered partitions Q is

$$\underbrace{\binom{\alpha r}{r}\binom{(\alpha-1)r}{r}\binom{(\alpha-2)r}{r}\cdots\binom{r}{r}}_{\alpha \text{ terms}} = \frac{(\alpha r)!}{(r!)^\alpha} \triangleq A,$$

and for a given r-subset $\mathbf{r}$, the number of ordered partitions Q for which $\mathbf{r} \in \mathcal{R}_Q$ is

$$\alpha \underbrace{\binom{(\alpha-1)r}{r}\binom{(\alpha-2)r}{r}\cdots\binom{r}{r}}_{(\alpha-1) \text{ terms}} = \frac{\alpha((\alpha-1)r)!}{(r!)^{\alpha-1}} \triangleq B.$$

When $\mathcal{R}$ contains the complete collection of $\binom{n}{r}$ r-subsets, we can find all A collections $\mathcal{R}_Q$ in $\mathcal{R}$, i.e. $\mathcal{R}_{Q_1} \subseteq \mathcal{R}$, $\mathcal{R}_{Q_2} \subseteq \mathcal{R}$, $\ldots$, $\mathcal{R}_{Q_A} \subseteq \mathcal{R}$. With each removal of an r-subset from $\mathcal{R}$, we reduce the number of collections $\mathcal{R}_Q$ found in $\mathcal{R}$ by at most B. It follows that the minimum number of r-subsets that need to be removed from $\mathcal{R}$ so that no collections $\mathcal{R}_Q$ remain in $\mathcal{R}$ is at least $\lceil \frac{A}{B} \rceil$, where

$$\frac{A}{B} = \frac{(\alpha r)!}{\alpha\, r!((\alpha-1)r)!} = \frac{r}{n}\binom{n}{r};$$

removing any fewer r-subsets would result in at least one collection $\mathcal{R}_Q$ in $\mathcal{R}$, and consequently, the optimality of the stated allocation.

Therefore, if $P_{\mathrm{S}} > 1 - \frac{r}{n}$ in $\mathbf{\Pi}(n, r, P_{\mathrm{S}})$, then we have

$$|\mathcal{R}| > \binom{n}{r} - \frac{r}{n}\binom{n}{r},$$

which implies that the stated allocation is optimal.

Now, we proceed to show that $P_{\mathrm{S}} > 1 - \frac{r}{n}$ is also necessary for this allocation to be optimal. We do this by finding a feasible allocation that uses a total storage strictly less than $\frac{n}{r}$ for $P_{\mathrm{S}} = 1 - \frac{r}{n}$; such a feasible allocation would remain feasible for lower values of P_{S}.

For $r = n$, (3) becomes $P_{\mathrm{S}} > 0$. The empty allocation $x_i = 0$, $i \in N$, uses a total storage of $0 < \frac{n}{r}$, and achieves $P_{\mathrm{S}} = 0$ with $|\mathcal{R}| = 0 \geq P_{\mathrm{S}} \binom{n}{r}$.

For $r < n$, the allocation

$$x_i = \begin{cases} 0 & \text{if } i = 1, \\ \frac{1}{r} & \text{if } i \in N \backslash \{1\} \end{cases}$$

uses a total storage of $\frac{n-1}{r} < \frac{n}{r}$, and achieves $P_{\mathrm{S}} = 1 - \frac{r}{n}$ with

$$|\mathcal{R}| = \binom{n-1}{r} = \left(1 - \frac{r}{n}\right) \binom{n}{r} \geq P_{\mathrm{S}} \binom{n}{r},$$

where $\mathcal{R}$ is just the collection of all r-subsets in $N \backslash \{1\}$. ■

Proof of Theorem 3: We adopt the approach in the proof of Theorem 2, but change the construction of the ordered partition Q and its corresponding collection of r-subsets $\mathcal{R}_Q$. For brevity, we define the following positive integers:

$$d \triangleq \gcd(r, r'), \quad m \triangleq r/d, \quad m' \triangleq r'/d.$$

It follows that $n = (\alpha\, m + m')d$. For the moment, we will proceed with the assumption that $\alpha \geq 1$. Let

$$Q = (\mathbf{u}_1, \ldots, \mathbf{u}_{m'}, \mathbf{v}_1, \ldots, \mathbf{v}_\alpha)$$

be an ordered partition of N that comprises $(m' + \alpha)$ parts, where

$$\begin{aligned} |\mathbf{u}_j| &= d, & j &= 1, \ldots, m', \\ |\mathbf{v}_j| &= r = m\, d, & j &= 1, \ldots, \alpha. \end{aligned}$$

For each ordered partition Q, we specify a collection of $(m' + \alpha)$ distinct r-subsets

$$\mathcal{R}_Q = \{\mathbf{r}_1, \ldots, \mathbf{r}_{m'}, \mathbf{r}_{m'+1}, \ldots \mathbf{r}_{m'+\alpha}\},$$

$$\text{where } \mathbf{r}_j = \begin{cases} \bigcup_{\ell=0}^{m-1} \mathbf{u}_{j+\ell} & \text{if } j = 1, \ldots, m', \\ \mathbf{v}_{j-m'} & \text{if } j = m'+1, \ldots, m'+\alpha, \end{cases}$$

defining $\mathbf{u}_j = \mathbf{u}_{j-m'}$ for $j > m'$. Fig. 6 provides an example of how Q and $\mathcal{R}_Q$ are constructed.

We claim that for any such ordered partition Q, if

$$\sum_{i \in \mathbf{r}} x_i \geq 1 \quad \forall\ \mathbf{r} \in \mathcal{R}_Q,$$

then

$$\sum_{i \in N} x_i \geq \frac{n}{r}.$$

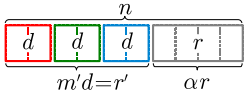
To see this, consider the partition of N formed by U and V, where

$$U = \bigcup_{j=1}^{m'} \mathbf{u}_j, \quad V = \bigcup_{j=1}^{\alpha} \mathbf{v}_j.$$

Let $(n,r)=(10,4)$.

Writing $n=\alpha r+r'$ gives $\alpha=1$ and $r'=6$.

We have $d=\gcd(r,r')=2$, $m=r/d=2$, and $m'=r'/d=3$.



An example of an ordered partition is

$$Q=\big(\{1,2\},\{3,4\},\{5,6\},\{7,8,9,10\}\big).$$

Its corresponding collection of r-subsets is

$$\mathcal{R}_Q=\big\{\{1,2,\ 3,4\},\ \{3,4,\ 5,6\},\ \{5,6,\ 1,2\},\ \{7,8,9,10\}\big\}.$$

Fig. 6. An example for the construction of the ordered partition Q and its corresponding collection of r-subsets $\mathcal{R}_Q$, in the proof of Theorem 3.

Correspondingly, we divide $\mathcal{R}_Q$ into two sets $\mathcal{R}_Q^U$ and $\mathcal{R}_Q^V$, where

$$\mathcal{R}_Q^U = \{\mathbf{r}_1, \ldots, \mathbf{r}_{m'}\}, \quad \mathcal{R}_Q^V = \{\mathbf{r}_{m'+1}, \ldots, \mathbf{r}_{m'+\alpha}\}.$$

Observe that each element $i \in U$ appears in exactly one $\mathbf{u}_j$, which in turn appears in exactly m of the m' r-subsets of $\mathcal{R}_Q^U$ (namely $\mathbf{r}_j, \mathbf{r}_{j-1}, \ldots, \mathbf{r}_{j-(m-1)}$, defining $\mathbf{r}_\ell = \mathbf{r}_{\ell+m'}$ for $\ell < 1$), that is,

$$\sum_{\mathbf{r} \in \mathcal{R}_Q^U} \mathbf{I}[i \in \mathbf{r}] = m \quad \forall\ i \in U.$$

Therefore, applying Lemma 1 with $S = U$, $c = m'$, and $b = m$ gives $\sum_{i \in U} x_i \geq \frac{m'}{m} = \frac{r'}{r}$. Likewise, observe that each element $i \in V$ appears in exactly one of the α r-subsets of $\mathcal{R}_Q^V$, that is,

$$\sum_{\mathbf{r} \in \mathcal{R}_Q^V} \mathbf{I}[i \in \mathbf{r}] = 1 \quad \forall\ i \in V.$$

Therefore, applying Lemma 1 with $S = V$, $c = \alpha$, and $b = 1$ gives $\sum_{i \in V} x_i \geq \alpha$. Combining the sums of U and V gives $\sum_{i \in N} x_i = \sum_{i \in U} x_i + \sum_{i \in V} x_i \geq \frac{r'}{r} + \alpha = \frac{n}{r}$.

As in the proof of Theorem 2, it follows that if $\mathcal{R}_Q \subseteq \mathcal{R}$ in $\mathbf{\Pi}(n, r, P_{\mathrm{S}})$, then the stated allocation is optimal.

The total number of possible ordered partitions Q is

$$\frac{\underbrace{\binom{(\alpha m + m')d}{d} \binom{(\alpha m + m' - 1)d}{d} \cdots \binom{(\alpha m + 1)d}{d}}_{m' \text{ terms}}}{\underbrace{\binom{\alpha m d}{md} \binom{(\alpha - 1)md}{md} \cdots \binom{md}{md}}_{\alpha \text{ terms}}} = \frac{((\alpha m + m')d)!}{(d!)^{m'}\,((md)!)^{\alpha}} \triangleq A,$$

and for a given r-subset $\mathbf{r}$, the number of ordered partitions Q for which $\mathbf{r} \in \mathcal{R}_Q$ is

$$\underbrace{\binom{((\alpha-1)m+m')d}{d}\binom{((\alpha-1)m+m'-1)d}{d}\cdots\binom{((\alpha-1)m+1)d}{d}}_{m' \text{ terms}}\cdot$$
$$\alpha\underbrace{\binom{(\alpha-1)md}{md}\binom{(\alpha-2)md}{md}\cdots\binom{md}{md}}_{(\alpha-1) \text{ terms}}$$
$$+\quad m'\underbrace{\binom{md}{d}\binom{(m-1)d}{d}\cdots\binom{d}{d}}_{m \text{ terms}}\cdot$$
$$\underbrace{\binom{((\alpha-1)m+m')d}{d}\binom{((\alpha-1)m+m'-1)d}{d}\cdots\binom{(\alpha m+1)d}{d}}_{(m'-m) \text{ terms}}\cdot$$
$$\underbrace{\binom{\alpha md}{md}\binom{(\alpha-1)md}{md}\cdots\binom{md}{md}}_{\alpha \text{ terms}}$$
$$= \quad \alpha\frac{(((\alpha-1)m+m')d)!}{(d!)^{m'}((md)!)^{\alpha-1}} \quad + \quad m'\frac{(((\alpha-1)m+m')d)!}{(d!)^{m'}((md)!)^{\alpha-1}}$$
$$= \quad (\alpha+m')\frac{(((\alpha-1)m+m')d)!}{(d!)^{m'}((md)!)^{\alpha-1}} \quad \triangleq \quad B,$$

which together give

$$\frac{A}{B}=\frac{1}{\alpha+m'}\frac{((\alpha m+m')d)!}{(((\alpha-1)m+m')d)!(md)!}=\frac{1}{\alpha+m'}\binom{n}{r}.$$

Repeating this argument for the special case of $\alpha=0$ produces $\frac{A}{B}=\frac{1}{m'}\binom{n}{r}$, which is consistent with the above expression. Therefore, if $P_{\mathrm{S}}>1-\frac{\gcd(r,r')}{\alpha\gcd(r,r')+r'}$ in $\mathbf{\Pi}(n,r,P_{\mathrm{S}})$, then we have

$$|\mathcal{R}|>\binom{n}{r}-\frac{\gcd(r,r')}{\alpha\gcd(r,r')+r'}\binom{n}{r}=\binom{n}{r}-\frac{1}{\alpha+m'}\binom{n}{r},$$

which implies that the stated allocation is optimal. ■

Proof of Corollary 1: If $(n-r)\mid n$, then $(n-r)\mid r$, since n, r, and $(n-r)$ are positive integers. We consider the following two cases:
(i) If $n-r=r$, then we also have $r\mid n$. By Theorem 2, the stated allocation is optimal if and only if $P_{\mathrm{S}}>1-\frac{r}{n}=\frac{r}{n}$.
(ii) If $\beta(n-r)=r$ for some integer $\beta\geq 2$, then $(n-r)<r\Rightarrow n<2r$. We can write $n=0\cdot r+r'$, where $r'\in\{r+1,\ldots,2r-1\}$. Applying Theorem 3 with $\alpha=0$, $r'=n$, and $\gcd(r,r')=\gcd(r,n)=(n-r)$ gives $P_{\mathrm{S}}>1-\frac{n-r}{n}=\frac{r}{n}$ as a sufficient condition for the optimality of the stated allocation. We proceed to show that (4) is also necessary by finding a feasible allocation that uses a total storage strictly less than $\frac{n}{r}$ for $P_{\mathrm{S}}=\frac{r}{n}$; such a feasible allocation would remain feasible for lower values of P_{S}.
The allocation

$$x_i=\begin{cases}1 & \text{if } i=1,\\ 0 & \text{if } i\in N\backslash\{1\}\end{cases}$$

uses a total storage of $1<\frac{n}{r}$, and achieves $P_{\mathrm{S}}=\frac{r}{n}$ with

$$|\mathcal{R}|=\binom{n-1}{r-1}=\frac{r}{n}\binom{n}{r}\geq P_{\mathrm{S}}\binom{n}{r},$$

where $\mathcal{R}$ is just the collection of all r-subsets that contain element 1. ■

Proof of Corollary 2: If $0\leq T<1$, then there can be no r-subset that yields a sum of at least 1; it follows that $\mathcal{R}=\emptyset$ is the only feasible choice, and the empty allocation gives the optimal value of $P_{\mathrm{S}}=0$.

For $1\leq T<\frac{n}{r}$, the stated allocation uses a total storage of $1\leq T$, and achieves $P_{\mathrm{S}}=\frac{r}{n}$ with

$$|\mathcal{R}|=\binom{n-1}{r-1}=\frac{r}{n}\binom{n}{r}\geq P_{\mathrm{S}}\binom{n}{r},$$

where $\mathcal{R}$ is just the collection of all r-subsets that contain element 1. By Corollary 1, higher values of P_{S} can only be achieved with $T\geq\frac{n}{r}$, and so this allocation is optimal.

For $T\geq\frac{n}{r}$, the stated allocation uses a total storage of $\frac{n}{r}\leq T$, and achieves $P_{\mathrm{S}}=1$ with $|\mathcal{R}|=\binom{n}{r}\geq P_{\mathrm{S}}\binom{n}{r}$, where $\mathcal{R}$ is just the complete collection of all r-subsets. It follows that this allocation is optimal since P_{S} is maximal. ■

IV. Conclusion

We described a distributed storage allocation problem and examined the optimal allocation that minimizes the total storage budget for the case of probability-1 recovery. We showed that this allocation remains optimal when the required probability of successful recovery exceeds a specified threshold. Stronger results were also obtained for a number of special cases.

Acknowledgment

We would like to thank Brighten Godfrey and Robert Kleinberg for introducing the problem to us and for sharing their insights.

References

[1] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. J. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Trans. Inf. Theory*, to appear.

[2] A. Jiang, "Network coding for joint storage and transmission with minimum cost," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2006.

[3] T. Ho, M. Médard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Trans. Inf. Theory*, vol. 52, no. 10, pp. 4413–4430, Oct. 2006.

[4] C. Fragouli, J.-Y. L. Boudec, and J. Widmer, "Network coding: An instant primer," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 1, pp. 63–68, Jan. 2006.

[5] A. G. Dimakis, V. Prabhakaran, and K. Ramchandran, "Ubiquitous access to distributed data in large-scale sensor networks through decentralized erasure codes," in *Proc. Int. Symp. Inf. Process. Sensor Netw. (IPSN)*, Apr. 2005.

[6] S. A. Aly, Z. Kong, and E. Soljanin, "Fountain codes based distributed storage algorithms for large-scale wireless sensor networks," in *Proc. ACM/IEEE Int. Conf. Inf. Process. Sensor Netw. (IPSN)*, Apr. 2008.

[7] A. Kamra, V. Misra, J. Feldman, and D. Rubenstein, "Growth codes: Maximizing sensor network data persistence," in *Proc. ACM SIGCOMM*, Sep. 2006.

[8] Y. Lin, B. Liang, and B. Li, "Data persistence in large-scale sensor networks with decentralized fountain codes," in *Proc. INFOCOM*, May 2007.

[9] R. Karp, R. Kleinberg, C. Papadimitriou, and E. Friedman, *Personal communication*, 2006.

[10] S. Jain, M. Demmer, R. Patra, and K. Fall, "Using redundancy to cope with failures in a delay tolerant network," in *Proc. ACM SIGCOMM*, Aug. 2005.

[11] D. Leong, A. G. Dimakis, and T. Ho, "Distributed storage allocation problems," in *Proc. Workshop Netw. Coding, Theory, and Appl. (NetCod)*, Jun. 2009.