On Erasure Coding for Distributed Storage and Streaming Communications

Thesis by

Derek Leong

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

California Institute of Technology

Pasadena, California, USA

2013

(Defended October 8, 2012)

To my parents and grandparents

Acknowledgments

I would like to express my gratitude to my research adviser Tracey Ho for her guidance and infinite wisdom, generosity, and patience, and also for her constant encouragement and prodding to tighten this bound and generalize that theorem. I would also like to extend my appreciation to my research mentor and collaborator Alex Dimakis for sharing his intuitions and insights on various problems, and for his candid tips on preventing paper rejections and audience narcolepsy. My thanks also go to my other thesis committee members Michelle Effros, Steven Low, Babak Hassibi, and Jehoshua (Shuki) Bruck for their feedback and comments on improving my work. I would also like to thank my research collaborators Edmund Yeh, Asma Qureshi, Michael Burd, Kyle Dumont, and Rebecca Cathey for the many enriching discussions.

The work in this thesis was funded in part by the Agency for Science, Technology and Research (A*STAR), Singapore; the Lee Center for Advanced Networking, California Institute of Technology; 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 N66001-08-C-2013; and the Air Force Office of Scientific Research (AFOSR) under grant FA9550-10-1-0166.

My Caltech graduate student experience has been an exciting and fulfilling one thanks in large part to the company of my research group mates and office mates Theodoros Dikaliotis, Svitlana Vyetrenko, Alex Gittens, Christopher Chang, Tao Cui, Mayank Bakshi, Ming Fai Wong, Jinghao Huang, and Hongyi Yao.

Lastly, I would like to thank my parents and grandparents for their unconditional love and support, and for the sacrifices they have made to give me the opportunity to pursue my ambitions. To them I dedicate my thesis with love.

Abstract

The work presented in this thesis revolves around erasure correction coding, as applied to distributed data storage and real-time streaming communications.

First, we examine the problem of allocating a given storage budget over a set of nodes for maximum reliability. The objective is to find an allocation of the budget that maximizes the probability of successful recovery by a data collector accessing a random subset of the nodes. This optimization problem is challenging in general because of its combinatorial nature, despite its simple formulation. We study several variations of the problem, assuming different allocation models and access models, and determine the optimal allocation and the optimal *symmetric* allocation (in which all nonempty nodes store the same amount of data) for a variety of cases. Although the optimal allocation can have nonintuitive structure and can be difficult to find in general, our results suggest that, as a simple heuristic, reliable storage can be achieved by spreading the budget maximally over all nodes when the budget is large, and spreading it minimally over a few nodes when it is small. Coding would therefore be beneficial in the former case, while uncoded replication would suffice in the latter case.

Second, we study how distributed storage allocations affect the recovery delay in a mobile setting. Specifically, two recovery delay optimization problems are considered for a network of mobile storage nodes: the maximization of the probability of successful recovery by a given deadline, and the minimization of the expected recovery delay. We show that the first problem is closely related to the earlier allocation problem, and solve the second problem completely for the case of symmetric allocations. It turns out that the optimal allocations for the two problems can be quite different. In a simulation study, we evaluated the performance of a simple data dissemination and storage protocol for mobile delay-tolerant networks, and observed that the choice of allocation can have a significant impact on the recovery delay under a variety of scenarios.

Third, we consider a real-time streaming system where messages created at regular time intervals at a source are encoded for transmission to a receiver over a packet erasure link; the receiver must subsequently decode each message within a given delay from its creation time. For erasure models containing a limited number of erasures per coding window, per sliding window, and containing erasure bursts whose maximum length is sufficiently short or long, we show that a time-invariant intrasession code asymptotically achieves the maximum message size among all codes that allow decoding under all admissible erasure patterns. For the bursty erasure model, we also show that diagonally interleaved codes derived from specific systematic block codes are asymptotically optimal over all codes in certain cases. We also study an i.i.d. erasure model in which each transmitted packet is erased independently with the same probability; the objective is to maximize the decoding probability for a given message size. We derive an upper bound on the decoding probability for any time-invariant code, and show that the gap between this bound and the performance of a family of time-invariant intrasession codes is small when the message size and packet erasure probability are small. In a simulation study, these codes performed well against a family of random time-invariant convolutional codes under a number of scenarios.

Finally, we consider the joint problems of routing and caching for named data networking. We propose a backpressure-based policy that employs virtual interest packets to make routing and caching decisions. In a packet-level simulation, the proposed policy outperformed a basic protocol that combines shortest-path routing with least-recently-used (LRU) cache replacement.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The work presented in this thesis revolves around erasure correction coding, as applied to distributed data storage and real-time streaming communications. In this chapter, we briefly discuss key motivations, related previous work, and our main contributions.

Distributed storage systems are widely used today as a means of keeping data safe and easily accessible. They can be as small as a RAID system [1] sitting on a desk, or as large as the Amazon S3 storage service [2] spanning across multiple data centers around the world. A *distributed* storage system introduces many new coding challenges that are not addressed in the single node case; chief among them are the cost and reliability of data recovery (or reconstruction) and node repairs in the event of device failures.

In Chapter 2, we explore the fundamental limits of the reliability of data recovery in a distributed storage system. Specifically, we examine the problem of allocating a given storage budget over a set of nodes for maximum reliability. The objective is to find an allocation of the budget that maximizes the probability of successful recovery by a data collector accessing a random subset of the nodes. This optimization problem is challenging in general because of its combinatorial nature, despite its simple formulation. The issue of storage allocations has rarely been addressed in previous work on distributed data storage; most storage schemes that apply coding, as opposed to uncoded replication or mirroring, would simply assume that the same amount of coded data should be stored in every node (e.g., see [3–7]). However, this common strategy of uniformly spreading the budget over all nodes can be suboptimal even for a simple homogeneous access model (or failure model) in which each node fails independently with the same probability [8]. We study several variations of

the problem, assuming different allocation models and access models, and determine the optimal allocation and the optimal *symmetric* allocation (in which all nonempty nodes store the same amount of data) for a variety of cases. Although the optimal allocation can have nonintuitive structure and can be difficult to find in general, our results suggest that, as a simple heuristic, reliable storage can be achieved by spreading the budget maximally over all nodes when the budget is large, and spreading it minimally over a few nodes when it is small. Coding would therefore be beneficial in the former case, while uncoded replication would suffice in the latter case. Although these results are framed in the context of distributed data storage, the fundamental nature of the problem lends them to many other applications, such as multipath routing over delay-tolerant networks [9–11], peer-to-peer networking [12], and streaming communications.

In Chapter 3, we study how distributed storage allocations affect the recovery delay in a mobile setting. Specifically, two recovery delay optimization problems are considered for a network of mobile storage nodes: the maximization of the probability of successful recovery by a given deadline, and the minimization of the expected recovery delay. We show that the first problem is closely related to the allocation problem of Chapter 2, and solve the second problem completely for the case of symmetric allocations. It turns out that the optimal allocations for the two problems can be quite different. We present a simple data dissemination and storage protocol that generalizes the Spray-and-Wait protocol [13] by using variable-size coded packets to create different symmetric allocations. In a simulation study, we evaluated the performance of the proposed protocol for delay-tolerant networks using a random waypoint mobility model and real-world mobility traces from vehicles, and observed that the choice of allocation can have a significant impact on the recovery delay under a variety of scenarios.

The ability to stream data reliably in real-time over packet erasure networks such as the Internet is essential for applications ranging from personal video conferencing to large-scale environmental monitoring or surveillance. In contrast to ordinary communications, *real-time streaming* imposes stringent constraints on what an encoder can access, and how long a decoder can wait.

In Chapter 4, we explore the fundamental limits of real-time streaming under various packet erasure models. Specifically, we consider a real-time streaming system where messages created at regular time intervals at a source are encoded for transmission to a receiver over a packet erasure link; the receiver must subsequently decode each message within a given delay from its cre-

ation time. Unlike previous work that aim to minimize the expected message decoding delay [14], or achieve a decoding failure probability that decays exponentially with delay [15–17], our real-time streaming model features hard message decoding deadlines. For erasure models containing a limited number of erasures per coding window, per sliding window, and containing erasure bursts whose maximum length is sufficiently short or long, we show that a time-invariant intrasession code asymptotically achieves the maximum message size among all codes that allow decoding under all admissible erasure patterns. This is interesting because intrasession coding is attractive due to its relative simplicity (it allows coding within the same message but not across different messages), but it is not known in general when intrasession coding is sufficient or when intersession coding is necessary. For the bursty erasure model, we also show that diagonally interleaved codes derived from specific systematic block codes are asymptotically optimal over all codes in certain cases. We also study an i.i.d. erasure model in which each transmitted packet is erased independently with the same probability; the objective is to maximize the decoding probability for a given message size. We derive an upper bound on the decoding probability for any time-invariant code, and show that the gap between this bound and the performance of a family of time-invariant intrasession codes is small when the message size and packet erasure probability are small. In a simulation study, these codes performed well against a family of random time-invariant convolutional codes, which are related to the codes in [14, 17], under a number of scenarios.

In Chapter 5, we consider the joint problems of routing and caching for named data networking. Named data networking (NDN), or content-centric networking (CCN), is a proposed network architecture for the Internet that replaces the traditional client-server model of communications with one based on the identity of data or content [18]. Requests for a data object in an NDN can be fulfilled not only by the origin server but also by any node with a copy of the object in its cache. Assuming the prevalence of caches, the usual approaches to routing and caching may no longer be effective for such a network. We propose a policy based on the backpressure algorithm [19, 20] that employs virtual interest packets to make routing and caching decisions. In a packet-level simulation, the proposed policy outperformed a basic protocol that combines shortest-path routing with least-recently-used (LRU) cache replacement [21].

In the final chapter, we conclude the thesis by summarizing our main contributions and proposing avenues for future work. Up-to-date information and resources on this thesis (e.g., links to

publications, source code, corrections) are available at `http://purl.org/net/phdthesis`.

Chapter 2

Distributed Storage Allocations

2.1 Introduction

Consider a distributed storage system comprising n storage nodes. A source has a single data object of normalized unit size that is to be coded and stored in a distributed manner over these nodes, subject to a given total storage budget T. Let x_i be the amount of coded data stored in node $i \in \{1, \ldots, n\}$. Any amount of data may be stored in each node, as long as the total amount of storage used over all nodes is at most the given budget T, i.e.,

$$\sum_{i=1}^{n} x_i \leq T.$$

This is a realistic constraint if there is limited transmission bandwidth or storage space, or if it is too costly to mirror the data object in its entirety in every node. At some time after the creation of this coded storage, a data collector attempts to recover the original data object by accessing only the data stored in a *random* subset $\mathbf{r}$ of the nodes, where the probability distribution of $\mathbf{r} \subseteq \{1, \ldots, n\}$ is specified by an assumed access model or failure model (nodes or links may fail probabilistically, for example). Figure 2.1 depicts such a distributed storage system.

The *reliability* of this system, which we define to be the probability of successful recovery (or recovery probability in short), depends on both the storage allocation and the coding scheme. For maximum reliability, we would therefore need to find

The material in this chapter was presented in part in [22–25].

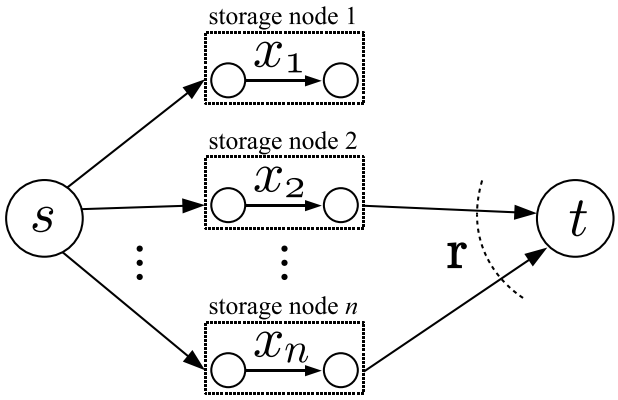


Figure 2.1. Information flows in a distributed storage system. The source s has a single data object of normalized unit size that is to be coded and stored over n storage nodes. Subsequently, a data collector t attempts to recover the original data object by accessing only the data stored in a random subset $\mathbf{r}$ of the nodes.

1) an optimal allocation of the given budget T over the nodes, specified by the values of $x_1, \ldots, x_n$; and

2) an optimal coding scheme

that jointly maximize the probability of successful recovery. It turns out that these two problems can be decoupled by using a good coding scheme, specifically one that enables successful recovery whenever the total amount of data accessed by the data collector is at least the size of the original data object. This can be seen by considering the information flows for a network in which the source is multicasting the data object to a set of potential data collectors [26, 27]: successful recovery can be achieved by a data collector if and only if its corresponding max-flow or min-cut from the source is at least the size of the original data object. Random linear coding over a sufficiently large field would allow successful recovery with high probability when this condition is satisfied [28, 29]. Alternatively, a suitable maximum distance separable (MDS) code for the given budget and data object size would allow successful recovery with certainty when this condition is satisfied.

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

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

Our goal is to find an optimal allocation that maximizes this recovery probability, subject to the

given budget constraint.

Although we have assumed coded storage at the outset, coding may ultimately be unnecessary for certain allocations. For example, if the budget is spread minimally such that each nonempty node stores the data object in its entirety (i.e., $x_i \geq 1$ for all $i \in S$, and $x_i = 0$ for all $i \notin S$, where S is some subset of $\{1, \ldots, n\}$), then uncoded replication would suffice since the data object can be recovered by accessing any *one* nonempty node; the data collector would not need to combine data accessed from different nodes in order to recover the data object. Thus, by solving for the optimal allocation, we will also be able to determine whether coding is beneficial for reliable storage.

We note that even though no explicit upper bound is imposed on the amount of data that can be stored in each node, it is never necessary to set $x_i > 1$ because $x_i = 1$ already allows the data object to be stored in its entirety in that node. The absence of a tighter per-node storage constraint $x_i \leq c_i < 1$ is reasonable for storage systems that handle a large number of data objects: we would expect the storage capacity of each node to be much larger than the size of a single data object, making it possible for a node to accommodate some of the data objects in their entirety. As such, it would be appropriate to apply a storage constraint for each data object via the budget T, without a separate *a priori* constraint for x_i. Furthermore, the simplifying assumption of x_i being a continuous variable is a reasonable one for large data objects: a large data object size would facilitate the creation of coded data packets with sizes (closely) matching that of a desired allocation. Incidentally, the overhead associated with random linear coding or an MDS code, which is ignored in our model, becomes proportionately negligible when the amount of coded data is large.

In spite of the simple formulation, this optimization problem poses significant challenges because of its combinatorial nature and the large space of feasible allocations. Different variations of this problem can be formulated by assuming different allocation models and access models; in this chapter, we will examine three such variations that are motivated by practical storage problems in content delivery networks, delay tolerant networks, and wireless sensor networks.

2.1.1 Independent Probabilistic Access to Each Node

In the first problem formulation, we assume that the data collector accesses each of the n nodes independently with constant probability p; in other words, each node i appears in subset $\mathbf{r}$ independently with probability p. The resulting problem can be interpreted as that of maximizing the

reliability of data storage in a system comprising n storage devices where each device fails independently with probability $1-p$. It is not hard to show that determining the recovery probability of a *given* allocation is computationally difficult (specifically, #P-hard). The intuitive approach of spreading the budget maximally over all nodes, i.e., setting $x_i = \frac{T}{n}$ for all i, turns out to be not necessarily optimal; in fact, the optimal allocation may not even be symmetric (we say that an allocation is *symmetric* when all nonzero x_i are equal). The following counterexample from [8] demonstrates that symmetric allocations can be suboptimal: for $(n, p, T) = \left(5, \frac{2}{3}, \frac{7}{3}\right)$, the nonsymmetric allocation

$$\left(\frac{2}{3}, \frac{2}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right),$$

which achieves a recovery probability of 0.90535, performs strictly better than any symmetric allocation; the maximum recovery probability among symmetric allocations is 0.88889, which is achieved by both

$$\left(\frac{7}{6}, \frac{7}{6}, 0, 0, 0\right) \quad \text{and} \quad \left(\frac{7}{12}, \frac{7}{12}, \frac{7}{12}, \frac{7}{12}, 0\right).$$

Evidently, the simple strategy of "spreading eggs evenly over more baskets" may not always improve the reliability of an allocation.

Our Contribution: We show that the intuitive symmetric allocation that spreads the budget maximally over all nodes is indeed *asymptotically* optimal in a regime of interest. Specifically, we derive an upper bound for the suboptimality of this allocation, and show that the performance gap vanishes asymptotically as the total number of storage nodes n grows, when $p > \frac{1}{T}$. This is a regime of interest because a high recovery probability is possible when $p > \frac{1}{T} \iff pT > 1$: The expected total amount of data accessed by the data collector is given by

$$\mathbb{E}\left[\sum_{i=1}^{n} x_i Y_i\right] = \sum_{i=1}^{n} x_i \mathbb{E}\left[Y_i\right] = p \sum_{i=1}^{n} x_i \leq pT, \tag{2.1}$$

where Y_i's are independent Bernoulli(p) random variables. Therefore, the data collector would be able to access a sufficient amount of data *in expectation* for successful recovery if $pT > 1$.

We also show that the symmetric allocation that spreads the budget minimally is optimal when p is sufficiently small. In such an allocation, the data object is stored in its entirety in each nonempty node, making coding unnecessary. Additionally, we explicitly find the optimal *symmetric* allocation

for a wide range of parameter values of p and T.

Related Work: This problem was introduced to us through a discussion at UC Berkeley [8]. We have since learned that variations of the problem have also been studied in several different fields.

In reliability engineering, the weighted-k-out-of-n system [30] comprises n components, each having a positive integer weight w_i and surviving independently with probability p_i; the system is in a good state if and only if the total weight of its surviving components is at least a specified threshold k. Related work on this system and its extensions has focused on the efficient computation of the reliability of a given weight allocation (e.g., [31]).

In peer-to-peer networking, the allocation problem deals with the recovery of a data object from peers that are available only probabilistically. Lin *et al.* [12] compared the performance of uncoded replication vs. coded storage, restricted to symmetric allocations, for the case where the budget is an integer.

In wireless communications, the allocation problem is studied in the context of multipath routing, in which coded data is transmitted along different paths in an unreliable network, exploiting path diversity to improve the reliability of end-to-end communications. Tsirigos and Haas [9, 10] examined the performance of symmetric allocations and noted the existence of a phase transition in the optimal symmetric allocation; approximation methods were also proposed by the authors to tackle the optimization problem, especially for the case where path failures occur with nonuniform probabilities and may be correlated. Jain *et al.* [11] evaluated the performance of symmetric allocations experimentally in a delay tolerant network setting, and presented an alternative formulation using Gaussian distributions to model partial access to nodes.

Our work generalizes these previous efforts by considering nonsymmetric allocations and non-integer budgets. We also correct some inaccurate claims about the optimal symmetric allocation in [11] and its associated technical report.

2.1.2 Access to a Random Fixed-Size Subset of Nodes

In the second problem formulation, we assume that the data collector accesses an r-subset of the n nodes selected uniformly at random from the collection of all $\binom{n}{r}$ possible r-subsets, where r is a given constant. The resulting problem can be interpreted as that of maximizing the recovery probability in a networked storage system of n nodes where the end user is able or allowed to

contact up to r nodes randomly. We can treat this access model as an approximation to the preceding independent probabilistic access model by picking $r \approx np$. Finding the optimal allocation in this case is still challenging. As in the first problem formulation, it is not hard to show that determining the recovery probability of a *given* allocation is computationally difficult (specifically, #P-complete).

The problem appears nontrivial even if we restrict the optimization to only *symmetric* allocations. Numerically, we observe that given n and r, either a minimal or a maximal spreading of the budget is optimal among symmetric allocations for most, if not all, choices of T. One example of an exception is $(n, r, T) = \left(14, 5, \frac{8}{3}\right)$ for which it is optimal to have 8 nonempty nodes in the symmetric allocation, instead of the extremes 2 or 13; another example is $(n, r, T) = \left(16, 4, \frac{7}{2}\right)$ for which it is optimal to have 7 nonempty nodes in the symmetric allocation, instead of the extremes 3 or 14. Furthermore, the number of nonempty nodes in the optimal symmetric allocation is not necessarily a nondecreasing function of the budget T; for instance, given $(n, r) = (20, 4)$, it is optimal to have $(4, 18, 14, 19, 20)$ nonempty nodes in the symmetric allocation for $T = (4.25, 4.5, 4.67, 4.75, 5)$, respectively.

Our Contribution: We show that the allocation $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$ is optimal in the *high recovery probability regime*. Specifically, we demonstrate that this allocation, which has a recovery probability of exactly 1, minimizes the budget T necessary for achieving any recovery probability exceeding a specified threshold $1 - \epsilon$. Although ϵ 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 $1 - \frac{1}{n}$.

We also make the following conjecture about the optimal allocation, based on our numerical observations:

Conjecture 2.1. *A* symmetric *optimal allocation always exists for any n, r, and T.*

Related Work: Sardari *et al.* [32] presented a method of *approximating* an optimal solution to this problem by considering a data collector that accesses r random nodes with replacement. More recently, Alon *et al.* [33] showed that this problem is related to an old conjecture by Erdős on the maximum number of edges in a uniform hypergraph [34].

2.1.3 Probabilistic Symmetric Allocations

In the third problem formulation, we assume a *probabilistic* allocation model in which the source selects a random allocation from a distribution of allocations, with the constraint that the *expected* total amount of storage used in an allocation is at most the given budget T. We specifically consider the case where each of the n nodes is selected by the source independently with constant probability $\min\left(\frac{\ell T}{n}, 1\right)$ to store a constant $\frac{1}{\ell}$ amount of data, thus creating a probabilistic *symmetric* allocation of the budget. The data collector subsequently accesses an r-subset of the n nodes selected uniformly at random from the collection of all $\binom{n}{r}$ possible r-subsets, where r is a given constant. The goal is to find an optimal allocation, specified by the value of parameter ℓ, that maximizes the recovery probability. This model was conceived as a simplification of the preceding fixed-size subset access model which assumes a deterministic allocation of the budget.

Our Contribution: We show that the choice of $\ell = r$, which corresponds to a maximal spreading of the budget, is optimal when the given budget T is sufficiently large, or equivalently, when a sufficiently high recovery probability (specifically, $\frac{3}{4}$ or higher) is achievable. We believe this is a reasonable operating regime for applications that require good reliability.

2.1.4 Other Related Work

Apart from the work done on the preceding problems, a variety of storage allocation problems have also been studied in a *nonprobabilistic* setting. For instance, the objective adopted in [35] and [36] is to minimize the total storage budget required to satisfy a given set of deterministic recovery requirements in a network. Incidentally, the use of network coding makes it easier to deal with the total cost of content delivery, which covers the initial dissemination, storage, and eventual fetching of data objects; this cost-minimization problem is considered in [27] and [37], subject to various deterministic constraints involving, for example, load balancing or fetching distance.

We note that in most of the literature involving reliable distributed storage, either the data object is assumed to be replicated in its entirety (e.g., [13]), or, if coding is used, every node is assumed to store the same amount of coded data (e.g., [3–7]). Allocations of a storage budget with nodes possibly storing different amounts of data are not usually considered.

In the following three sections, we define each problem formally and state our main results.

Table 2.1. Notation.

Symbol	Definition
n	total number of storage nodes, $n \geq 2$
x_i	amount of data stored in storage node i, $x_i \geq 0$, where $i \in \{1, \ldots, n\}$
T	total storage budget, $1 \leq T \leq n$
$\mathbf{r}$	subset of nodes accessed, $\mathbf{r} \subseteq \{1, \ldots, n\}$
p	access probability (Section 2.2), $0 < p < 1$
r	number of nodes accessed (Section 2.3), $1 \leq r \leq n$
$\frac{1}{\ell}$	amount of data stored in each nonempty node (Section 2.4), $\ell > 0$
$\mathcal{B}(n, p)$	binomial random variable with n trials and success probability p
$\mathbb{1}[G]$	indicator function; $\mathbb{1}[G] = 1$ if statement G is true, and 0 otherwise
$\mathbb{Z}_0^+$	the set of nonnegative integers, i.e., $\mathbb{Z}^+ \cup \{0\}$

Proofs of theorems are deferred to Section 2.6. Table 2.1 summarizes the notation used throughout this chapter.

2.2 Independent Probabilistic Access to Each Node

In the first variation of the storage allocation problem, we consider a data collector that accesses each of the n nodes independently with probability p; successful recovery occurs if and only if the total amount of data stored in the accessed nodes is at least 1. We seek an optimal allocation $(x_1, \ldots, x_n)$ of the budget T that maximizes the probability of successful recovery, for a given choice of n, p, and T. This optimization problem can be expressed as follows:

$$\mathbf{\Pi}_1(n, p, T):$$

$$\underset{x_1,\ldots,x_n}{\text{maximize}} \sum_{\mathbf{r}\subseteq\{1,\ldots,n\}} p^{|\mathbf{r}|}(1-p)^{n-|\mathbf{r}|} \cdot \mathbb{1}\left[\sum_{i\in\mathbf{r}} x_i \geq 1\right] \tag{2.2}$$

subject to

$$\sum_{i=1}^{n} x_i \leq T \tag{2.3}$$

$$x_i \geq 0 \qquad \forall\ i \in \{1, \ldots, n\}. \tag{2.4}$$

The objective function (2.2) is just the recovery probability, expressed as the sum of the probabilities corresponding to the subsets $\mathbf{r}$ that allow successful recovery. An equivalent expression for (2.2) is

$$\mathbb{P}\left[\sum_{i=1}^{n} x_i\, Y_i \geq 1\right],$$

where Y_i's are independent Bernoulli(p) random variables. Inequality (2.3) expresses the budget constraint, and inequality (2.4) ensures that a nonnegative amount of data is stored in each node. For the trivial budget $T = 1$, the allocation $(1, 0, \ldots, 0)$ is optimal; for $T = n$, the allocation $(1, \ldots, 1)$ is optimal. Incidentally, computing the recovery probability of a *given* allocation turns out to be #P-hard:

Proposition 2.2. *Computing the recovery probability*

$$\sum_{\mathbf{r}\subseteq\{1,\ldots,n\}} p^{|\mathbf{r}|}(1-p)^{n-|\mathbf{r}|} \cdot \mathbb{1}\left[\sum_{i\in\mathbf{r}} x_i \geq 1\right]$$

for a given allocation $(x_1, \ldots, x_n)$ and choice of p is #P-hard.

Table 2.2 lists the optimal allocations for $n = 2, 3, 4, 5$, covering all parameter values of $p \in (0, 1)$ and $T \in [1, n)$. These solutions are obtained by minimizing T for each possible value of the objective function (2.2). We observe that

1) for any T, the symmetric allocation $(1, \ldots, 1, 0, \ldots, 0)$, which corresponds to a minimal spreading of the budget (uncoded replication), appears to be optimal when p is sufficiently small; and

2) the optimal *symmetric* allocation appears to perform well despite being suboptimal in some cases, e.g., when $(n, T) = \left(4, \frac{5}{2}\right)$ and $p > \frac{1}{2}$.

We will proceed to show that the first observation is indeed true in Section 2.2.2; the opposite approach of spreading the budget maximally over all nodes turns out to be *asymptotically* optimal when p is sufficiently large, as will be demonstrated in Section 2.2.1. Motivated by the second observation, we examine the optimization problem restricted to symmetric allocations in Section 2.2.3.

For brevity, let $\bar{\mathbf{x}}(n, T, m)$ denote the *symmetric* allocation for n nodes that uses a total storage

Table 2.2. Optimal allocations for number of nodes $n = 2, 3, 4, 5$.

n	Budget T	Optimal allocation	Condition on access probability p (if any)
2	$1 \le T < 2$	$(1, 0)$	
3	$1 \le T < \frac{3}{2}$	$(1, 0, 0)$	
	$\frac{3}{2} \le T < 2$	$(1, 0, 0)$	if $p \le \frac{1}{2}$
		$\left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right)$	if $p \ge \frac{1}{2}$
	$2 \le T < 3$	$(1, 1, 0)$	
4	$1 \le T < \frac{4}{3}$	$(1, 0, 0, 0)$	
	$\frac{4}{3} \le T < \frac{3}{2}$	$(1, 0, 0, 0)$	if $p \le \frac{1+\sqrt{13}}{6} \approx 0.768$
		$\left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$	if $p \ge \frac{1+\sqrt{13}}{6} \approx 0.768$
	$\frac{3}{2} \le T < 2$	$(1, 0, 0, 0)$	if $p \le \frac{1}{2}$
		$\left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 0\right)$	if $p \ge \frac{1}{2}$
	$2 \le T < \frac{5}{2}$	$(1, 1, 0, 0)$	if $p \le \frac{2}{3}$
		$\left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right)$	if $p \ge \frac{2}{3}$
	$\frac{5}{2} \le T < 3$	$(1, 1, 0, 0)$	if $p \le \frac{1}{2}$
		$\left(1, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right)$	if $p \ge \frac{1}{2}$
	$3 \le T < 4$	$(1, 1, 1, 0)$	
5	$1 \le T < \frac{5}{4}$	$(1, 0, 0, 0, 0)$	
	$\frac{5}{4} \le T < \frac{4}{3}$	$(1, 0, 0, 0, 0)$	if $p \le \frac{1+\sqrt[3]{235-6\sqrt{1473}}+\sqrt[3]{235+6\sqrt{1473}}}{12} \approx 0.869$
		$\left(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}\right)$	if $p \ge \frac{1+\sqrt[3]{235-6\sqrt{1473}}+\sqrt[3]{235+6\sqrt{1473}}}{12} \approx 0.869$
	$\frac{4}{3} \le T < \frac{3}{2}$	$(1, 0, 0, 0, 0)$	if $p \le \frac{1+\sqrt{13}}{6} \approx 0.768$
		$\left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}, 0\right)$	if $p \ge \frac{1+\sqrt{13}}{6} \approx 0.768$
	$\frac{3}{2} \le T < \frac{5}{3}$	$(1, 0, 0, 0, 0)$	if $p \le \frac{1}{2}$
		$\left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 0, 0\right)$	if $p \ge \frac{1}{2}$
	$\frac{5}{3} \le T < 2$	$(1, 0, 0, 0, 0)$	if $p \le \frac{1}{2}$
		$\left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$	if $p \ge \frac{1}{2}$
	$2 \le T < \frac{7}{3}$	$(1, 1, 0, 0, 0)$	if $p \le \frac{2}{3}$
		$\left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 0\right)$	if $p \ge \frac{2}{3}$
	$\frac{7}{3} \le T < \frac{5}{2}$	$(1, 1, 0, 0, 0)$	if $p \le 2 - \sqrt{2} \approx 0.586$
		$\left(\frac{2}{3}, \frac{2}{3}, \frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right)$	if $p \ge 2 - \sqrt{2} \approx 0.586$
	$\frac{5}{2} \le T < 3$	$(1, 1, 0, 0, 0)$	if $p \le \frac{7-\sqrt{17}}{8} \approx 0.360$
		$\left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right)$	if $p \ge \frac{7-\sqrt{17}}{8} \approx 0.360$
	$3 \le T < \frac{7}{2}$	$(1, 1, 1, 0, 0)$	if $p \le \frac{2}{3}$
		$\left(1, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right)$	if $p \ge \frac{2}{3}$
	$\frac{7}{2} \le T < 4$	$(1, 1, 1, 0, 0)$	if $p \le \frac{1}{2}$
		$\left(1, 1, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}\right)$	if $p \ge \frac{1}{2}$
	$4 \le T < 5$	$(1, 1, 1, 1, 0)$	

of T and contains exactly $m \in \{1, 2, \ldots, n\}$ nonempty nodes:

$$\bar{\mathbf{x}}(n, T, m) \triangleq \Bigg(\underbrace{\frac{T}{m}, \ldots, \frac{T}{m}}_{m \text{ entries}}, \underbrace{0, \ldots, 0}_{(n-m) \text{ entries}} \Bigg).$$

Since successful recovery for the symmetric allocation $\bar{\mathbf{x}}(n, T, m)$ occurs if and only if at least $\left\lceil 1 \big/ \left(\frac{T}{m}\right) \right\rceil = \left\lceil \frac{m}{T} \right\rceil$ out of the m nonempty nodes are accessed, the corresponding probability of successful recovery can be written as

$$P_{\mathrm{S}}(p, T, m) \triangleq \mathbb{P}\left[\mathcal{B}\left(m, p\right) \geq \left\lceil \frac{m}{T} \right\rceil \right].$$

2.2.1 Asymptotic Optimality of Maximal Spreading

The recovery probability of the symmetric allocation $\bar{\mathbf{x}}\left(n, T, m{=}n\right)$, which corresponds to a maximal spreading of the budget over all nodes, is given by

$$P_{\mathrm{S}}(p, T, m{=}n) = \mathbb{P}\left[\mathcal{B}\left(n, p\right) \geq \left\lceil \frac{n}{T} \right\rceil \right]. \tag{2.5}$$

To establish the optimality of this allocation, we compare (2.5) to an upper bound for the recovery probability of an optimal allocation. Such a bound can be derived by conditioning on the number of accessed nodes:

Lemma 2.3. *The probability of successful recovery for an optimal allocation is at most*

$$\sum_{r=0}^{n} \left\lfloor \min\left(\frac{rT}{n}, 1 \right) \binom{n}{r} \right\rfloor p^r (1-p)^{n-r}. \tag{2.6}$$

The suboptimality of $\bar{\mathbf{x}}\left(n, T, m{=}n\right)$ is therefore bounded by the difference between (2.5) and (2.6), as given by the following theorem; when $p > \frac{1}{T}$, this allocation becomes asymptotically optimal since its suboptimality gap vanishes as n goes to infinity:

Theorem 2.4. *The gap between the probabilities of successful recovery for an optimal allocation and for the symmetric allocation $\bar{\mathbf{x}}\left(n, T, m{=}n\right)$, which corresponds to a maximal spreading of the*

budget over all nodes, is at most

$$p\,T\,\mathbb{P}\left[\mathcal{B}\left(n-1,p\right)\leq\left\lceil\frac{n}{T}\right\rceil-2\right].$$

If p and T are fixed such that $p>\frac{1}{T}$, then this gap approaches zero as n goes to infinity.

We note that the regime $p>\frac{1}{T}$ is particularly interesting because it corresponds to the regime of high recovery probability; the recovery probability would be bounded away from 1 if $p<\frac{1}{T}\iff pT<1$ instead. This follows from the application of Markov's inequality to the random variable W denoting the total amount of data accessed by the data collector, which produces

$$\mathbb{P}\left[W\geq 1\right]\leq\mathbb{E}\left[W\right].$$

Since $\mathbb{P}\left[W\geq 1\right]$ is just the probability of successful recovery, and $\mathbb{E}\left[W\right]\leq pT$ according to (2.1), we have

$$\mathbb{P}\left[\text{successful recovery}\right]\leq pT.$$

2.2.2 Optimality of Minimal Spreading (Uncoded Replication)

The recovery probability of the symmetric allocation $\bar{\mathbf{x}}\left(n,T,m{=}\lfloor T\rfloor\right)$, which corresponds to a minimal spreading of the budget, is given by

$$P_{\mathrm{S}}\left(p,T,m{=}\lfloor T\rfloor\right)=\mathbb{P}\left[\mathcal{B}\left(\lfloor T\rfloor,p\right){\geq}1\right]=1-\left(1{-}p\right)^{\lfloor T\rfloor}. \tag{2.7}$$

Recall that coding is unnecessary in such an allocation since the data object is stored in its entirety in each nonempty node. A sufficient condition for the optimality of this allocation can be found by comparing (2.7) to an upper bound for the recovery probabilities of all other allocations. Our approach is to classify each allocation according to the number of individual nodes that store at least a unit amount of data. We then find a bound for allocations containing exactly 0 such nodes, another bound for allocations containing exactly 1 such node, and so on. The subsequent comparisons of (2.7) to each of these bounds result in the following theorem:

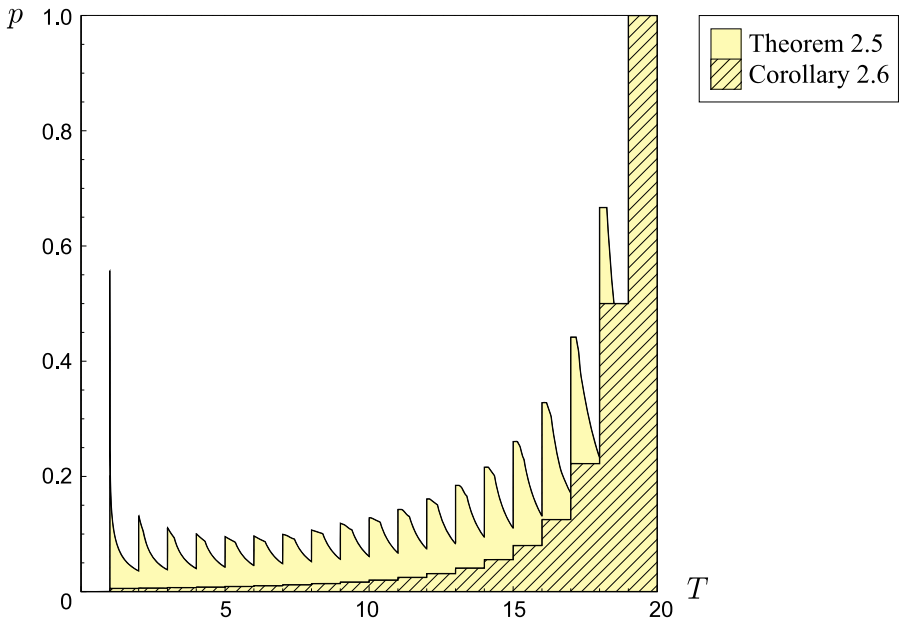


Figure 2.2. Plot of access probability p against budget T, showing regions of (T, p) over which the sufficient conditions of the theorems are satisfied, for $n = 20$. Minimal spreading (uncoded replication) is optimal among all allocations in the colored regions.

Theorem 2.5. *If* $1 < T < n$ *and*

$$1 - (1-p)^{\lfloor T \rfloor - n} + (n-\ell)\left(\frac{p}{1-p}\right) + \sum_{r=2}^{\lceil \frac{n-\ell}{T-\ell} \rceil - 1} \left(1 - \frac{T-\ell}{n-\ell} \cdot r\right) \binom{n-\ell}{r} \left(\frac{p}{1-p}\right)^r \geq 0 \tag{2.8}$$

for all $\ell \in \{0, 1, \ldots, \lfloor T \rfloor - 1\}$*, then* $\bar{\mathbf{x}}\,(n, T, m{=}\lfloor T \rfloor)$*, which corresponds to a minimal spreading of the budget (uncoded replication), is an optimal allocation.*

The following corollary shows that this allocation is indeed optimal for sufficiently small p:

Corollary 2.6. *If* $1 < T < n$ *and* $p \leq \frac{2}{(n - \lfloor T \rfloor)^2}$*, then* $\bar{\mathbf{x}}\,(n, T, m{=}\lfloor T \rfloor)$ *is an optimal allocation.*

Figure 2.2 illustrates these results in the form of a region plot for an instance of n.

2.2.3 Optimal Symmetric Allocation

The optimization problem appears nontrivial even if we were to consider only *symmetric* allocations. Figure 2.3, which compares the performance of different symmetric allocations over different budgets for an instance of (n, p), demonstrates that the value of m corresponding to the optimal

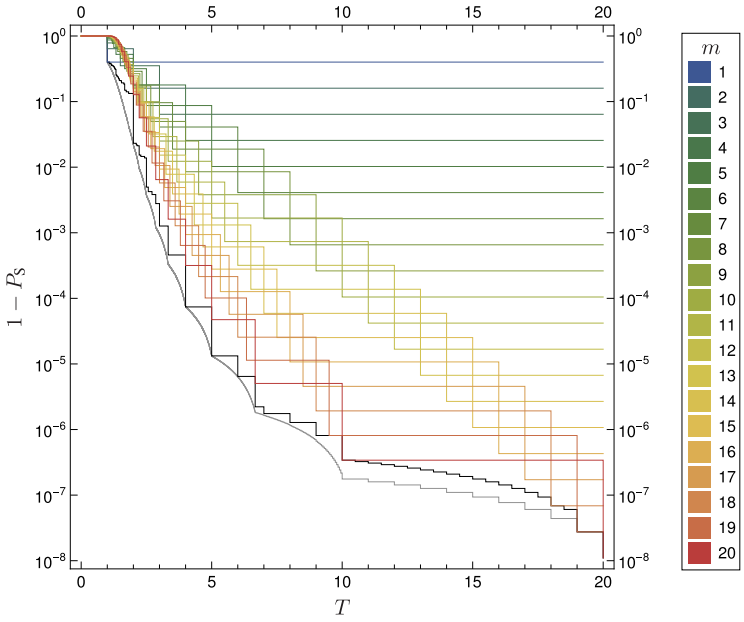


Figure 2.3. Plot of recovery failure probability $1 - P_{\mathrm{S}}$ against budget T for each symmetric allocation $\bar{\mathbf{x}}(n, T, m)$, for $(n, p) = \left(20, \frac{3}{5}\right)$. Parameter m denotes the number of nonempty nodes in the symmetric allocation. The gray and black curves describe lower bounds for the recovery failure probability of an optimal allocation, as given by Lemma 2.3 and (2.23), respectively.

symmetric allocation can change drastically as the budget T varies.

Fortunately, we can eliminate many candidates for the optimal value of m by making the following observation: Recall that the recovery probability of the symmetric allocation $\bar{\mathbf{x}}(n, T, m)$ is given by $P_{\mathrm{S}}(p, T, m) \triangleq \mathbb{P}\left[\mathcal{B}\left(m, p\right) \geq \left\lceil \frac{m}{T} \right\rceil\right]$. For fixed n, p, and T, we have

$$\left\lceil \frac{m}{T} \right\rceil = k \qquad \text{when } m \in \big((k-1)T, kT\big],$$

for $k = 1, 2, \ldots, \left\lfloor \frac{n}{T} \right\rfloor$, and finally,

$$\left\lceil \frac{m}{T} \right\rceil = \left\lfloor \frac{n}{T} \right\rfloor + 1 \qquad \text{when } m \in \left(\left\lfloor \frac{n}{T} \right\rfloor T, n\right].$$

Since $\mathbb{P}\left[\mathcal{B}\left(m, p\right) \geq k\right]$ is nondecreasing in m for constant p and k, it follows that $P_{\mathrm{S}}(p, T, m)$ is maximized within each of these intervals of m when we pick m to be the largest integer in the corresponding interval. Thus, given n, p, and T, we can find an optimal m^* that maximizes

$P_{\mathrm{S}}(p,T,m)$ over all m from among $\left\lceil \frac{n}{T} \right\rceil$ candidates:

$$\left\{ \lfloor T \rfloor, \lfloor 2T \rfloor, \ldots, \left\lfloor \left\lfloor \frac{n}{T} \right\rfloor T \right\rfloor, n \right\}. \tag{2.9}$$

For $m = \lfloor kT \rfloor$, where $k \in \mathbb{Z}^+$, the corresponding probability of successful recovery is given by

$$P_{\mathrm{S}}\left(p,T,m{=}\lfloor kT \rfloor\right) = \mathbb{P}\left[\mathcal{B}\left(\lfloor kT \rfloor, p\right) \geq k\right].$$

The difference between the probabilities of successful recovery for consecutive values of $k \in \mathbb{Z}^+$ can be written as

$$\begin{aligned}
\Delta(p,T,k) &\triangleq P_{\mathrm{S}}\left(p,T,m{=}\lfloor (k+1)T \rfloor\right) - P_{\mathrm{S}}\left(p,T,m{=}\lfloor kT \rfloor\right) \\
&= \mathbb{P}\left[\mathcal{B}\left(\lfloor (k+1)T \rfloor, p\right) \geq k+1\right] - \mathbb{P}\left[\mathcal{B}\left(\lfloor kT \rfloor, p\right) \geq k\right] \\
&= \left(\sum_{i=1}^{\min(\alpha_{k,T}-1,k)} \mathbb{P}\left[\mathcal{B}\left(\lfloor kT \rfloor, p\right) = k-i\right] \cdot \mathbb{P}\left[\mathcal{B}\left(\alpha_{k,T}, p\right) \geq i+1\right] \right) \\
&\qquad - \mathbb{P}\left[\mathcal{B}\left(\lfloor kT \rfloor, p\right) = k\right] \cdot \mathbb{P}\left[\mathcal{B}\left(\alpha_{k,T}, p\right) = 0\right],
\end{aligned}$$

where $\alpha_{k,T} \triangleq \lfloor (k+1)T \rfloor - \lfloor kT \rfloor$. The above expression is obtained by comparing the branches of the probability tree for $\lfloor kT \rfloor$ vs. $\lfloor (k+1)T \rfloor$ independent Bernoulli(p) trials: the first term describes unsuccessful events ("$\mathcal{B}\left(\lfloor kT \rfloor, p\right) < k$") becoming successful ("$\mathcal{B}\left(\lfloor (k+1)T \rfloor, p\right) \geq k+1$") after the additional $\alpha_{k,T}$ trials, while the second term describes successful events ("$\mathcal{B}\left(\lfloor kT \rfloor, p\right) \geq k$") becoming unsuccessful ("$\mathcal{B}\left(\lfloor (k+1)T \rfloor, p\right) < k+1$") after the additional $\alpha_{k,T}$ trials. After further simplification, we arrive at

$$\begin{aligned}
\Delta(p,T,k) = p^k (1-p)^{\lfloor (k+1)T \rfloor - k} &\cdot \\
\left(\left(\sum_{i=1}^{\min\left(\alpha_{k,T}-1,k\right)} \sum_{j=i+1}^{\alpha_{k,T}} \binom{\lfloor kT \rfloor}{k-i} \binom{\alpha_{k,T}}{j} \left(\frac{p}{1-p}\right)^{-i+j} \right) - \binom{\lfloor kT \rfloor}{k} \right).
\end{aligned} \tag{2.10}$$

The following theorem essentially provides a sufficient condition on p and T for $\Delta(p,T,k) \geq 0$ for any $k \in \mathbb{Z}^+$, thereby eliminating all but the two largest candidate values for m^* in (2.9), i.e., $m = \left\lfloor \left\lfloor \frac{n}{T} \right\rfloor T \right\rfloor$ and $m = n$, which correspond to a maximal spreading of the budget over (almost) all

nodes (they are identical when $\frac{n}{T} \in \mathbb{Z}^+$, i.e., $T = n, \frac{n}{2}, \frac{n}{3}, \ldots$):

Theorem 2.7. *If*

$$(1-p)^{\lfloor T \rfloor} + 2\lfloor T \rfloor p(1-p)^{\lfloor T \rfloor - 1} - 1 \leq 0, \tag{2.11}$$

then either $\bar{\mathbf{x}}\left(n, T, m{=}\left\lfloor \left\lfloor \frac{n}{T} \right\rfloor T \right\rfloor\right)$ *or* $\bar{\mathbf{x}}\left(n, T, m{=}n\right)$*, which correspond to a maximal spreading of the budget, is an optimal symmetric allocation.*

The following corollary restates Theorem 2.7 in a slightly weaker but more convenient form:

Corollary 2.8. *If* $p \geq \frac{4}{3\lfloor T \rfloor}$*, then either* $\bar{\mathbf{x}}\left(n, T, m{=}\left\lfloor \left\lfloor \frac{n}{T} \right\rfloor T \right\rfloor\right)$ *or* $\bar{\mathbf{x}}\left(n, T, m{=}n\right)$ *is an optimal symmetric allocation.*

The following lemma mirrors Theorem 2.7 by providing a sufficient condition on p and T for $\Delta(p, T, k) \leq 0$ for any $k \in \mathbb{Z}^+$, thereby eliminating all but the smallest candidate value for m^* in (2.9), i.e., $m = \lfloor T \rfloor$, which corresponds to a minimal spreading of the budget (uncoded replication):

Lemma 2.9. *If* $T > 1$*, and either*

$$T = \frac{1}{p} \in \mathbb{Z}^+ \tag{2.12}$$

or

$$T < \frac{1}{p} \quad \textit{and} \quad p\,(1-p)^{\lceil T \rceil - 1} \leq \frac{1}{T}\left(1 - \frac{1}{T}\right)^{\lceil T \rceil - 1}, \tag{2.13}$$

then $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor T \rfloor\right)$ *is an optimal symmetric allocation.*

The following lemma restates Lemma 2.9 in a slightly weaker but more convenient form:

Lemma 2.10. *If* $p \leq \frac{2}{\lceil T \rceil} - \frac{1}{T}$*, then* $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor T \rfloor\right)$ *is an optimal symmetric allocation.*

The following theorem expands the region covered by Lemma 2.10 by showing that $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor T \rfloor\right)$ remains optimal between the "peaks" in Figure 2.4:

Theorem 2.11. *If* $p \leq \frac{1}{\lceil T \rceil}$*, then* $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor T \rfloor\right)$*, which corresponds to a minimal spreading of the budget (uncoded replication), is an optimal symmetric allocation.*

Figure 2.4 illustrates these results in the form of a region plot. The theorems cover all choices

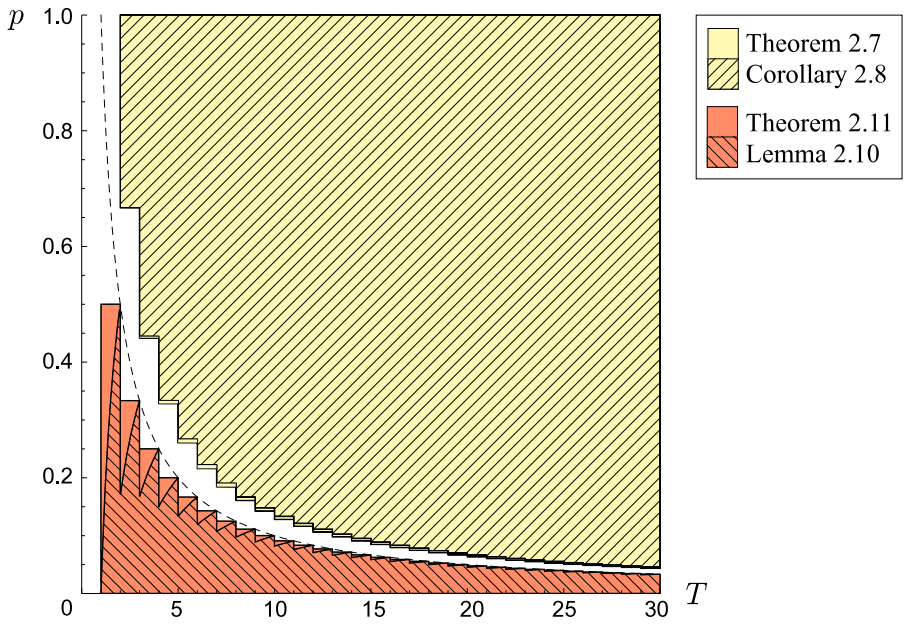


Figure 2.4. Plot of access probability p against budget T, showing regions of (T, p) over which the sufficient conditions of the theorems are satisfied. The black dashed curve marks the points satisfying $p = \frac{1}{T}$. Maximal spreading is optimal among symmetric allocations in the colored regions above the curve, while minimal spreading (uncoded replication) is optimal among symmetric allocations in the colored regions below the curve.

of p and T except for the gap around $p = \frac{1}{T}$, which diminishes with increasing T. Both minimal and maximal spreading of the budget may be suboptimal among symmetric allocations in this gap on either side of the curve $p = \frac{1}{T}$: for example, when $(n, p, T) = \left(10, \frac{9}{25}, \frac{5}{2}\right)$, for which $p < \frac{1}{T}$, the optimal symmetric allocation is $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor 2T \rfloor\right)$; when $(n, p, T) = \left(10, \frac{3}{5}, \frac{12}{5}\right)$, for which $p > \frac{1}{T}$, the optimal symmetric allocation is $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor 3T \rfloor\right)$. In general, for any budget $T \geq 2$, the optimal symmetric allocation changes from minimal spreading to maximal spreading eventually, as the access probability p increases. This transition, which is not necessarily sharp, appears to occur at around $p = \frac{1}{T}$. Interestingly, when $p = \frac{1}{T}$ exactly, we observe numerically that $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor T \rfloor\right)$ is the optimal symmetric allocation for *most* values of T; the optimal symmetric allocation changes continually over the intervals

$$1.5 \leq T < 2 \quad \text{and} \quad 2.5 \leq T \leq 2.8911,$$

while $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor 2T \rfloor\right)$ is optimal for $3.5 \leq T \leq 3.5694$. These findings suggest that it may be difficult to specify an optimal symmetric allocation for values of p and T in the gap; we can, however, restrict our search for an optimal symmetric allocation to the $\left\lceil \frac{n}{T} \right\rceil$ candidates given by (2.9).

2.3 Access to a Random Fixed-Size Subset of Nodes

In the second variation of the storage allocation problem, we consider a data collector that accesses an r-subset of the n nodes selected uniformly at random from the collection of all $\binom{n}{r}$ possible r-subsets, where r is a given constant; successful recovery occurs if and only if the total amount of data stored in the accessed nodes is at least 1. We seek an optimal allocation $(x_1, \ldots, x_n)$ of the budget T that maximizes the probability of successful recovery, for a given choice of n, r, and T. This optimization problem can be expressed as follows:

$\mathbf{\Pi}_2(n, r, T)$:

$$\underset{x_1,\ldots,x_n,P_{\mathrm{S}}}{\text{maximize}} \qquad P_{\mathrm{S}} \tag{2.14}$$

subject to

$$\sum_{\substack{\mathbf{r}\subseteq\{1,\ldots,n\}:\\ |\mathbf{r}|=r}} \frac{1}{\binom{n}{r}} \cdot \mathbb{1}\left[\sum_{i\in\mathbf{r}} x_i \geq 1\right] \geq P_{\mathrm{S}} \tag{2.15}$$

$$\sum_{i=1}^{n} x_i \leq T \tag{2.16}$$

$$x_i \geq 0 \qquad \forall\ i \in \{1, \ldots, n\}. \tag{2.17}$$

The left-hand side of inequality (2.15) is just the recovery probability, expressed as the sum of the probabilities corresponding to the r-subsets $\mathbf{r}$ that allow successful recovery. The objective function (2.14) is therefore equal to the recovery probability since P_{S} is maximized when (2.15) holds with equality. Inequality (2.16) expresses the budget constraint, and inequality (2.17) ensures that a nonnegative amount of data is stored in each node. For the trivial budget $T = 1$, the allocation $(1, 0, \ldots, 0)$ is optimal; for $T \geq \frac{n}{r}$, the allocation $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$, which has the maximal recovery probability of 1, is optimal. Incidentally, computing the recovery probability of a *given* allocation turns out to be #P-complete:

Proposition 2.12. *Computing the recovery probability*

$$\sum_{\substack{\mathbf{r}\subseteq\{1,\ldots,n\}:\\ |\mathbf{r}|=r}} \frac{1}{\binom{n}{r}} \cdot \mathbb{1}\left[\sum_{i\in\mathbf{r}} x_i \geq 1\right]$$

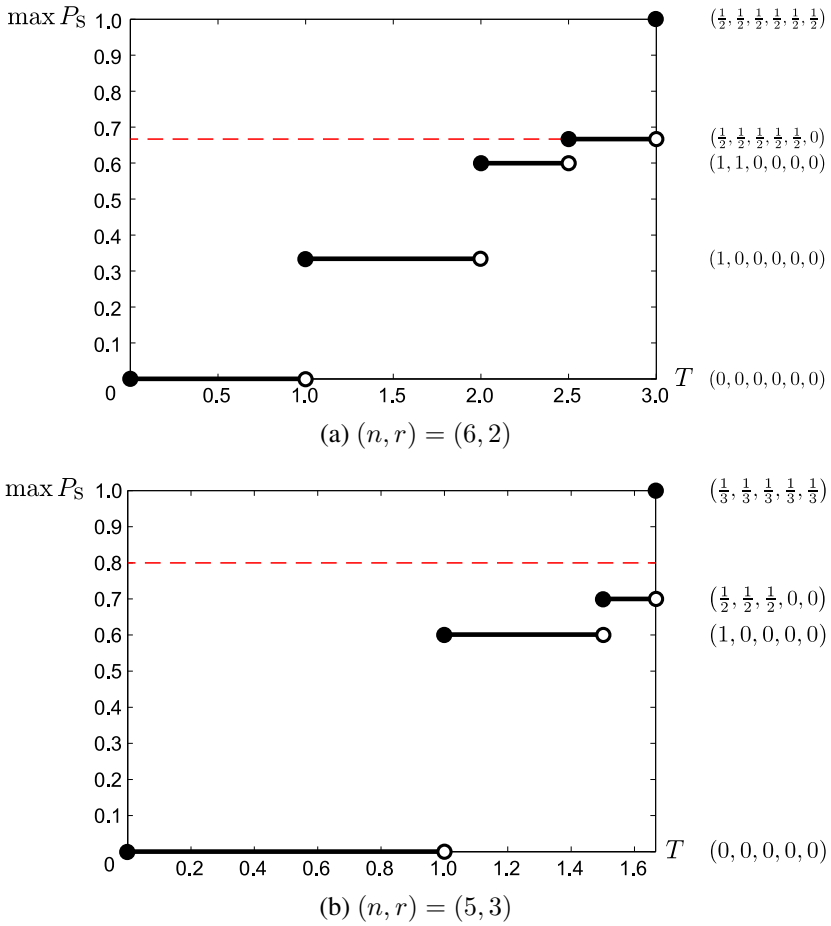


Figure 2.5. Plot of the optimal recovery probability $\max P_S$ against budget T, for (a) $(n,r)=(6,2)$ and (b) $(n,r)=(5,3)$. The optimal allocation corresponding to each value of $\max P_S$ is given on the right-hand side of the plot. In (a), the red dashed line marks the threshold on P_S derived in Theorem 2.15; the allocation $\left(\frac{1}{r},\ldots,\frac{1}{r}\right)$ is optimal for $\mathbf{\Pi}_2'(n,r,P_S)$ if and only if the desired recovery probability P_S exceeds this threshold. In (b), the red dashed line marks the threshold on P_S derived in Theorem 2.16; the allocation $\left(\frac{1}{r},\ldots,\frac{1}{r}\right)$ is optimal for $\mathbf{\Pi}_2'(n,r,P_S)$ if P_S exceeds this threshold.

for a given allocation $(x_1,\ldots,x_n)$ and choice of r is #P-complete.

An alternate way of formulating this problem is to minimize the budget T required to achieve a desired recovery probability P_S:

$\mathbf{\Pi}_2'(n,r,P_S)$:

$$\underset{x_1,\ldots,x_n,T}{\text{minimize}} \qquad T$$

subject to the three constraints (2.15)–(2.17) of $\mathbf{\Pi}_2(n,r,T)$.

Figure 2.5 shows how the optimal recovery probability $\max P_S$ varies with the budget T, for

two instances of (n, r). These plots are obtained by solving $\mathbf{\Pi}_2'(n, r, P_{\mathrm{S}})$ for each possible value of P_{S}. We observe that when the budget T drops below $\frac{n}{r}$, the optimal recovery probability $\max P_{\mathrm{S}}$ is reduced by a significant margin below 1. In other words, if the desired recovery probability P_{S} in $\mathbf{\Pi}_2'(n, r, P_{\mathrm{S}})$ is sufficiently high, then the optimal allocation is $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$, which requires a budget of $T = \frac{n}{r}$. In Section 2.3.1, we examine the optimality of this allocation for the high recovery probability regime.

2.3.1 Regime of High Recovery Probability

Consider the optimization problem $\mathbf{\Pi}_2'(n, r, P_{\mathrm{S}})$. We will demonstrate that the allocation $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$ is optimal when the desired recovery probability P_{S} exceeds a specified threshold expressed in terms of n and r. Our results follow from the observation that successful recovery for certain combinations of r-subsets of nodes can impose a lower bound on the required budget T. For example, given $(n, r) = (4, 2)$, if successful recovery is to occur for $\{1, 2\}$ and $\{3, 4\}$, possibly among other r-subsets of nodes, then we have

$$\sum_{i \in \{1,2\}} x_i \geq 1 \quad \text{and} \quad \sum_{i \in \{3,4\}} x_i \geq 1,$$

which would imply that the minimum budget T must be at least 2, since

$$T \geq \sum_{i=1}^{4} x_i = \sum_{i \in \{1,2\}} x_i + \sum_{i \in \{3,4\}} x_i \geq 2.$$

This observation is generalized by the following lemma:

Lemma 2.13. *Consider a set $S \subseteq \{1, \ldots, n\}$, and c subsets of S given by $\mathbf{r}_j \subseteq S$, $j = 1, \ldots, c$. If*

$$\sum_{i \in \mathbf{r}_j} x_i \geq 1 \quad \forall \ j \in \{1, \ldots, c\}, \tag{2.18}$$

and each element in S appears exactly $b > 0$ times among the c subsets, i.e.,

$$\sum_{j=1}^{c} \mathbb{1}\left[i \in \mathbf{r}_j\right] = b \quad \forall \ i \in S, \tag{2.19}$$

then

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

We begin with the special case of probability-1 recovery, i.e., $P_{\mathrm{S}} = 1$. The resulting optimization problem is just a linear program with all $\binom{n}{r}$ possible r-subset constraints.

Lemma 2.14. *If $P_S = 1$, then $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$ is an optimal allocation.*

When the desired recovery probability P_{S} is less than 1, we can afford to drop *some* of the r-subset constraints from this linear program (recall that the recovery probability of an allocation is just the fraction of these $\binom{n}{r}$ constraints that are satisfied). Our task is to determine how many such constraints can be dropped before the lower bound for T obtained with the help of Lemma 2.13 falls below $\frac{n}{r}$, in which case the allocation $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$ may no longer be optimal. We do this by constructing collections of r-subset constraints that yield the required lower bound of $\frac{n}{r}$ for T, and counting how many r-subset constraints need to be removed from the linear program before no such collection remains. Our answer depends on the divisibility of n by r.

When n is a multiple of r, we are able to state a necessary and sufficient condition on P_{S} for the allocation to be optimal:

Theorem 2.15. *If n is a multiple of r, then $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$ is an optimal allocation if and only if*

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

When n is *not* a multiple of r, we are only able to state a sufficient condition on P_{S} for the allocation to be optimal:

Theorem 2.16. *If n is not a multiple of r, then $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$ is an optimal allocation if*

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

where α and r' are uniquely defined integers satisfying

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

However, if n is a multiple of $(n - r)$, then this sufficient condition becomes necessary too:

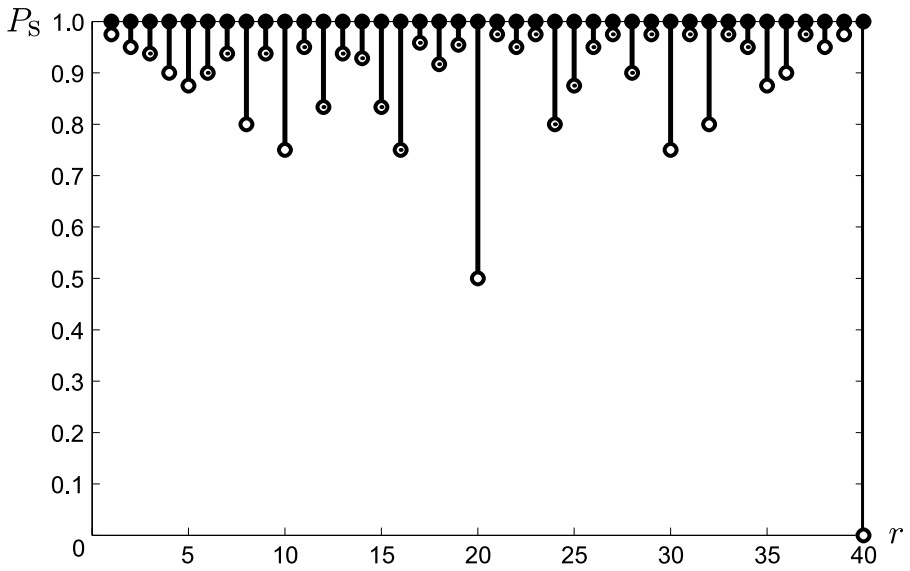


Figure 2.6. Plot of the desired recovery probability P_{S} against the number of nodes accessed r, showing intervals of P_{S} over which the allocation $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$ is optimal for $\mathbf{\Pi}_2'(n, r, P_{\mathrm{S}})$, for $n = 40$. A dotted circle marker denotes an endpoint that may not be tight, i.e., we have not demonstrated that the allocation is suboptimal everywhere outside the interval.

Corollary 2.17. *If n is a multiple of $(n - r)$, then $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$ is an optimal allocation if and only if*

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

Note that Corollary 2.17 allows us to solve $\mathbf{\Pi}_2(n, r, T)$ *completely* when n is a multiple of $(n - r)$: for any $T \in \left[1, \frac{n}{r}\right)$, the allocation $(1, 0, \ldots, 0)$ is optimal since it has a recovery probability of $\frac{\binom{n-1}{r-1}}{\binom{n}{r}} = \frac{r}{n}$, i.e., exactly the threshold in Corollary 2.17; higher recovery probabilities are not achievable unless $T \geq \frac{n}{r}$.

Figure 2.6 illustrates these results for an instance of n.

2.3.2 Upper Bounds for the Optimal Recovery Probability

Consider the optimization problem $\mathbf{\Pi}_2(n, r, T)$. For a given allocation $(x_1, \ldots, x_n)$, let $\mathcal{S}$ be the collection of successful subsets, i.e.,

$$\mathcal{S} \triangleq \left\{ \mathbf{r} \subseteq \{1, \ldots, n\} : |\mathbf{r}| = r, \sum_{i \in \mathbf{r}} x_i \geq 1 \right\},$$

and let S be the number of successful subsets, i.e., $S = |\mathcal{S}|$. The following lemma provides several upper bounds on S, which can be used to bound the optimal recovery probability in both the fixed-size subset access model of Section 2.3 and the independent probabilistic access model of Section 2.2:

Lemma 2.18. *For any feasible allocation $(x_1, \ldots, x_n)$, i.e., such that $\sum_{i=1}^{n} x_i \le T$ and $x_i \ge 0$ for all $i \in \{1, \ldots, n\}$, the number of successful subsets S has the following upper bounds:*

$$S \le \binom{n}{r} - \frac{\gcd(r, r')}{\alpha \gcd(r, r') + r'} \binom{n}{r} \qquad \text{if } T < \frac{n}{r}, \tag{2.20}$$

$$S \le \lfloor T \rfloor \frac{r}{n} \binom{n}{r} \qquad \text{if } T < \left\lfloor \frac{n}{r} \right\rfloor, \tag{2.21}$$

$$S \le \frac{\lfloor \beta T \rfloor}{\beta} \frac{r}{n} \binom{n}{r}, \tag{2.22}$$

where α and r' are uniquely defined integers satisfying $n = \alpha\, r + r'$, $\alpha \in \mathbb{Z}_0^+$, and $r' \in \{r, \ldots, 2r-1\}$, and $\beta \triangleq \frac{\operatorname{lcm}(n,r)}{n}$.

Upper bound (2.20) is a corollary of Theorems 2.15 and 2.16. To obtain upper bounds (2.21) and (2.22), we apply permutation counting arguments similar to Katona's proof of the Erdős-Ko-Rado theorem [38,39]. Picking the tightest of these bounds, and applying the fact that S is an integer that is at most $\binom{n}{r}$, produces the following upper bound for S:

$$S^{\mathsf{UB}}(n, r, T) \triangleq \begin{cases} 0 & \text{if } T < 1, \\ \left\lfloor \min\left(1 - \mathbb{1}\left[T < \frac{n}{r}\right] \cdot \frac{\gcd(r,r')}{\alpha \gcd(r,r') + r'}, \right.\right. & \\ \qquad \left.\left. 1 - \mathbb{1}\left[T < \left\lfloor \frac{n}{r} \right\rfloor\right] \cdot \left(1 - \lfloor T \rfloor \frac{r}{n}\right), \frac{\lfloor \beta T \rfloor}{\beta} \frac{r}{n} \right) \binom{n}{r} \right\rfloor & \text{otherwise.} \end{cases}$$

By combining this bound with the proof technique of Theorem 2.5, we can in turn derive an improved upper bound for the optimal recovery probability in the independent probabilistic access model of Section 2.2 (cf. Lemma 2.3):

$$P_{\mathrm{S}}^{\mathsf{UB}}(n, p, T) \triangleq \max_{\ell \in \{0, 1, \ldots, \lfloor T \rfloor\}} \overbrace{1 - (1-p)^{\ell}}^{\text{(i)}} + \overbrace{(1-p)^{\ell} \sum_{r=2}^{n-\ell} S^{\mathsf{UB}}(n-\ell, r, T-\ell)\, p^r (1-p)^{n-\ell-r}}^{\text{(ii)}}. \tag{2.23}$$

Parameter ℓ denotes the number of individual nodes that store at least a unit amount of data. At least ℓ amount of data is stored in these *complete* nodes, leaving the remaining budget of at most $T - \ell$ to be allocated over the remaining $n - \ell$ *incomplete* nodes. Term (i) gives the probability of successful recovery from accessing at least one complete node, while term (ii) gives an upper bound on the probability of successful recovery from accessing exactly $r \in \{2, \ldots, n - \ell\}$ incomplete nodes. A plot of this bound is shown in Figure 2.3.

2.4 Probabilistic Symmetric Allocations

In the third variation of the storage allocation problem, we consider the case where each of the n nodes is selected by the source independently with probability $\min\left(\frac{\ell T}{n}, 1\right)$ to store $\frac{1}{\ell}$ amount of data, so that the expected total amount of storage used in the resulting *symmetric* allocation is at most $n \cdot \frac{\ell T}{n} \cdot \frac{1}{\ell} = T$, the given budget. The data collector subsequently accesses an r-subset of the n nodes selected uniformly at random from the collection of all $\binom{n}{r}$ possible r-subsets, where r is a given constant; successful recovery occurs if and only if the total amount of data stored in the accessed nodes is at least 1. We seek an optimal probabilistic symmetric allocation of the budget T, specified by the value of parameter ℓ, that maximizes the probability of successful recovery, for a given choice of n, r, and T. Since successful recovery for a particular choice of ℓ occurs if and only if at least $\left\lceil 1 \Big/ \left(\frac{1}{\ell}\right) \right\rceil = \lceil \ell \rceil$ out of the r accessed nodes are nonempty, the corresponding probability of successful recovery can be written as

$$P_{\mathrm{S}}(n, r, T, \ell) \triangleq \mathbb{P}\left[\mathcal{B}\left(r, \min\left(\frac{\ell T}{n}, 1\right)\right) \geq \lceil \ell \rceil\right].$$

This optimization problem can therefore be expressed as follows:

$$\begin{aligned} &\boldsymbol{\Pi}_3(n, r, T): \\ &\quad \underset{\ell}{\text{maximize}} \quad \mathbb{P}\left[\mathcal{B}\left(r, \min\left(\frac{\ell T}{n}, 1\right)\right) \geq \lceil \ell \rceil\right] \\ &\quad \text{subject to} \quad \ell > 0. \end{aligned}$$

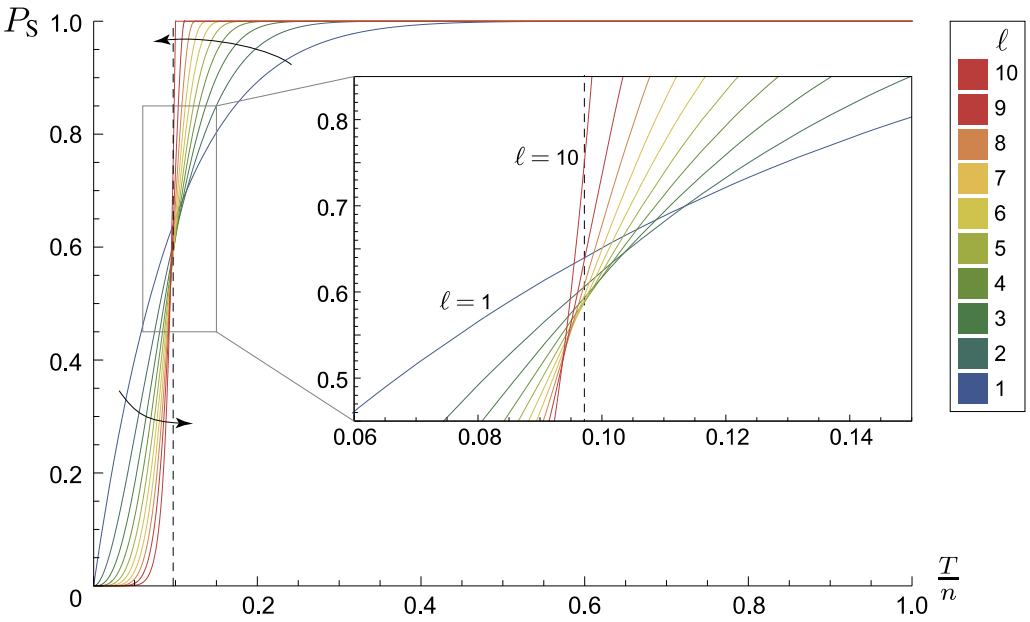


Figure 2.7. Plot of recovery probability P_{S} against budget-per-node $\frac{T}{n}$ for each choice of parameter $\ell \in \{1, 2, \ldots, r\}$, for $r = 10$. Parameter ℓ controls how much the budget is spread in the probabilistic symmetric allocation; specifically, each of the n nodes is selected by the source independently with probability $\min\left(\frac{\ell T}{n}, 1\right)$ to store $\frac{1}{\ell}$ amount of data. Arrows indicate the direction of increasing ℓ. The black dashed line marks the threshold on $\frac{T}{n}$ derived in Theorem 2.20; maximal spreading ($\ell = r$) is optimal for any $\frac{T}{n}$ greater than or equal to this threshold.

For budget $T \geq \frac{n}{r}$, the choice of $\ell = r$, which yields a recovery probability of $\mathbb{P}\left[\mathcal{B}\left(r, 1\right) \geq r\right] = 1$, is optimal.

Observe that the recovery probability $P_{\mathrm{S}}(n, r, T, \ell)$ is zero when $\ell > r$. Furthermore, for fixed n, r, and T, the recovery probability is nondecreasing in ℓ within each of the unit intervals $(0, 1]$, $(1, 2]$, $(2, 3]$, …, since as ℓ increases within each interval, $\lceil \ell \rceil$ remains constant while $\min\left(\frac{\ell T}{n}, 1\right)$ either increases or remains constant at 1. Thus, given n, r, and T, we can find an optimal ℓ^* from among r candidates:

$$\left\{1, 2, \ldots, r\right\}. \tag{2.24}$$

Figure 2.7, which compares the performance of different probabilistic symmetric allocations over different budgets for an instance of r, suggests that there are two distinct phases pertaining to the optimal choice of ℓ: when the budget is below a certain threshold, the choice of $\ell = 1$, which corresponds to a minimal spreading of the budget (uncoded replication), is optimal; when the budget exceeds that same threshold, the choice of $\ell = r$, which corresponds to a maximal spreading of the

budget, becomes optimal. This observation echoes our findings on the allocation and access models of the preceding sections, namely that minimal spreading ($\ell = 1$) is optimal for sufficiently small budgets, while maximal spreading ($\ell = r$) is optimal for sufficiently large budgets. However, we note two important distinctions in contrast to the previous models. First, the recovery probability for a probabilistic symmetric allocation in this model is a *continuous* nondecreasing function of the given budget; there are no "jumps" from one discrete value to the next. Second, our empirical computations suggest that the phase transition from the optimality of minimal spreading to that of maximal spreading in this model is *sharp*; the other intermediate values of $\ell \in \{2, \ldots, r-1\}$ never perform better than both $\ell = 1$ and $\ell = r$ simultaneously.

In Section 2.4.1, we shall demonstrate that the choice of $\ell = r$, which corresponds to a maximal spreading of the budget, is indeed optimal when the given budget T is sufficiently large, or equivalently, when a sufficiently high recovery probability is achievable.

2.4.1 Optimality of Maximal Spreading

Assume that $r \geq 2$. As noted earlier, the choice of $\ell = r$, which corresponds to a maximal spreading of the budget, is optimal for any $T \geq \frac{n}{r}$ because it yields the maximal recovery probability of 1. The following lemma provides an upper bound for the recovery probabilities corresponding to the *other* candidate values for ℓ^* in (2.24) at the critical budget $T = \frac{n}{r}$:

Lemma 2.19. *The probability of successful recovery $P_S(n, r, T, \ell)$ at $T = \frac{n}{r}$ is at most $\frac{3}{4}$ for any $\ell \in \{1, 2, \ldots, r-1\}$.*

Such an upper bound allows us to derive a sufficient condition for the optimality of $\ell = r$, by making use of the fact that the recovery probability $P_{\mathrm{S}}(n, r, T, \ell)$ is a nondecreasing function of the budget T. The following theorem shows that the choice of $\ell = r$ is optimal when the budget T is at least a specified threshold expressed in terms of n and r:

Theorem 2.20. *If*

$$T \geq \frac{n}{r} \left(\frac{3}{4}\right)^{\frac{1}{r}},$$

then the choice of $\ell = r$, which corresponds to a maximal spreading of the budget, is optimal.

The following corollary states an equivalent result in terms of the achievable recovery probability;

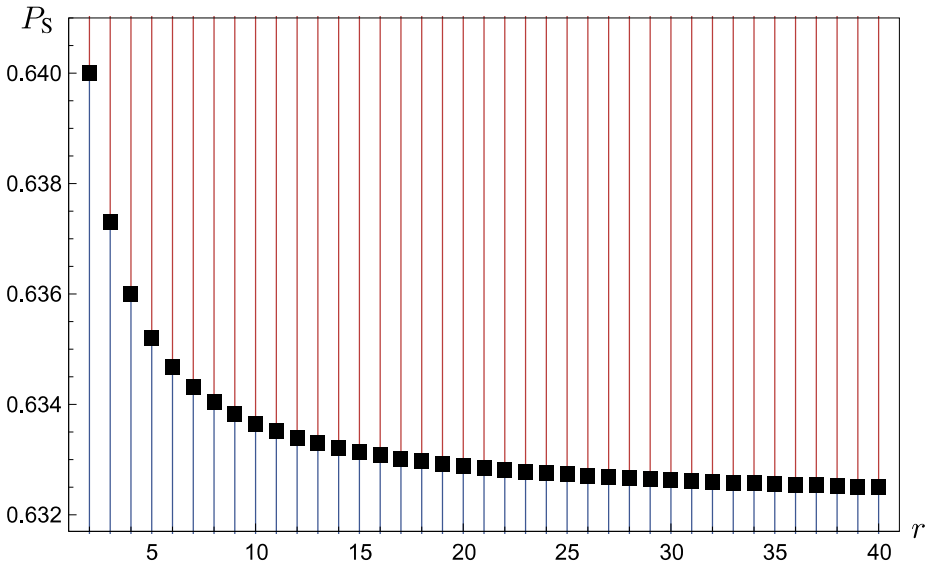


Figure 2.8. Plot of recovery probability P_S against the number of nodes accessed r, indicating the value of P_S at which the optimal choice of parameter ℓ changes from 1 to r, for each given value of r. Specifically, if it is possible to achieve a recovery probability P_S above the square marker, then maximal spreading ($\ell = r$) is optimal; otherwise, minimal spreading or uncoded replication ($\ell = 1$) is optimal. Observe that the critical value of P_S for $r = 10$ (which is approximately 0.633652) corresponds to the intersection point of the curves for $\ell = 1$ and $\ell = 10$ in Figure 2.7.

it demonstrates the optimality of $\ell = r$ in the high recovery probability regime:

Corollary 2.21. *If a probability of successful recovery of at least $\frac{3}{4}$ is achievable for the given n, r, and T, then the choice of $\ell = r$ is optimal.*

Figure 2.8 describes the optimal choice of ℓ for different values of r. We observe that the gap between the threshold of $\frac{3}{4}$ derived in Corollary 2.21 and the actual critical value of P_S indicated in the plot appears to be no more than 0.12.

2.5 Conclusion and Future Work

We examined the problem of allocating a given total storage budget in a distributed storage system for maximum reliability. Three variations of the problem were studied in detail, and we are able to specify the optimal allocation or optimal symmetric allocation for a variety of cases. Although the exact optimal allocation is difficult to find in general, our results suggest a simple heuristic for achieving reliable storage: *when the budget is small, spread it minimally; when the budget is large, spread it maximally.* In other words, coding is unnecessary when the budget is

small, but is beneficial when the budget is large.

The work in this chapter can be extended in several directions. We can impose additional system design constraints on the model; one practical example is the application of a tighter per-node storage constraint $x_i \leq c_i < 1$. The independent probabilistic access model of Section 2.2 can be naturally generalized to handle heterogeneous access, e.g., nonuniform access probabilities p_i for individual nodes (e.g., [40]). It would also be interesting to find reliable allocations for specific codes with desirable encoding or decoding properties, e.g., sparse codes that offer efficient algorithms (e.g., [4–7]). A related problem would be to construct such codes that work well under different allocations. Another set of interesting problems involves the application of richer access models; for instance, we can introduce a network topology to a set of storage nodes and data collectors, and allow each data collector to access only the nodes close to it. More generally, we can assign different priorities to each node for data storage and access, so as to reflect the costs of storing data in the node and communicating with it.

2.6 Proofs of Theorems

2.6.1 Proof of Proposition 2.2

We note that the computational complexity of this problem was well understood in the Berkeley meetings [8] and is by no means a major contribution of this work. We present the detailed proofs here for completeness.

Consider an allocation $(x_1, \ldots, x_n)$ where each x_i is a nonnegative rational number. The problem of computing the recovery probability of this allocation for the special case of $p = \frac{1}{2}$, for which $p^{|\mathbf{r}|}(1-p)^{n-|\mathbf{r}|} = \left(\frac{1}{2}\right)^n$ for *any* subset $\mathbf{r} \subseteq \{1, \ldots, n\}$, is equivalent to the counting version of the following decision problem (which happens to be polynomial-time solvable):

Definition 2.22. Largest Subset Sum (LSS)

Instance: Finite n-vector $(a_1, \ldots, a_n)$ with $a_i \in \mathbb{Z}_0^+$, and file size $d \in \mathbb{Z}^+$, where all a_i and d can be written as decimal numbers of length at most ℓ.

Question: Is there a subset $\mathbf{r} \subseteq \{1, \ldots, n\}$ that satisfies $\sum_{i \in \mathbf{r}} a_i \geq d$?

Note that the allocation and file size have been scaled so that the problem parameters are all integers. We will proceed to show that the counting problem #LSS is #P-complete; this would in

Table 2.3. Construction of a #LSS instance for a given #3SAT instance.

	v_1 v_2 $\cdots$ v_m	C_1	C_2	$\cdots$	C_k
v_1	1	$\mathbb{1}[v_1 \in C_1]$ 0	$\mathbb{1}[v_1 \in C_2]$ 0	$\cdots$	$\mathbb{1}[v_1 \in C_k]$ 0
$\overline{v_1}$	1	$\mathbb{1}[\overline{v_1} \in C_1]$ 0	$\mathbb{1}[\overline{v_1} \in C_2]$ 0	$\cdots$	$\mathbb{1}[\overline{v_1} \in C_k]$ 0
v_2	1	$\mathbb{1}[v_2 \in C_1]$ 0	$\mathbb{1}[v_2 \in C_2]$ 0	$\cdots$	$\mathbb{1}[v_2 \in C_k]$ 0
$\overline{v_2}$	1	$\mathbb{1}[\overline{v_2} \in C_1]$ 0	$\mathbb{1}[\overline{v_2} \in C_2]$ 0	$\cdots$	$\mathbb{1}[\overline{v_2} \in C_k]$ 0
$\vdots$	$\ddots$	$\vdots$ $\vdots$	$\vdots$ $\vdots$	$\vdots$	$\vdots$ $\vdots$
v_m	1	$\mathbb{1}[v_m \in C_1]$ 0	$\mathbb{1}[v_m \in C_2]$ 0	$\cdots$	$\mathbb{1}[v_m \in C_k]$ 0
$\overline{v_m}$	1	$\mathbb{1}[\overline{v_m} \in C_1]$ 0	$\mathbb{1}[\overline{v_m} \in C_2]$ 0	$\cdots$	$\mathbb{1}[\overline{v_m} \in C_k]$ 0
C_1		0 1 1 1 2 1			
C_2			0 1 1 1 2 1		
$\vdots$				$\ddots$	
C_k					0 1 1 1 2 1
d	1 1 $\cdots$ 1	3 1	3 1	$\cdots$	3 1

turn establish the #P-hardness of computing the recovery probability for an arbitrary value of p.

The index set $\mathbf{r}$ can be represented as an n-vector of bits. Using this representation of $\mathbf{r}$ as the certificate, it is easy to see that the binary relation corresponding to #LSS is both polynomially balanced (since the size of each certificate is n), and polynomial-time decidable (since the inequality can be verified in $O(n\ell)$ time for each certificate). It therefore follows that #LSS is in #P.

To show that #LSS is also #P-hard, we describe a polynomial-time Turing reduction of the #P-complete problem #3SAT [41] to #LSS. Our approach is similar to the standard method of reducing 3SAT to Subset Sum (e.g., [42]). Let ϕ be the Boolean formula in the given #3SAT instance; denote its m variables by $v_1, \ldots, v_m$, and k clauses by $C_1, \ldots, C_k$. To count the number of satisfying truth assignments for ϕ, we construct a #LSS instance with the help of Table 2.3, whose entries are 0, 1, 2, or 3 (all blank entries are 0's). The entries of the n-vector for the #LSS instance are given by the first $(2m + 3k)$ rows of the table; the file size d is given by the last row of the table. Each entry a_i, $i \in \{1, \ldots, 2m + 3k\}$, as well as d, is a positive integer with at most $(m + 2k)$ decimal digits. Observe that the set of satisfying truth assignments for ϕ can be put in a one-to-

one correspondence with the collection of subsets $\mathbf{r} \subseteq \{1, \ldots, 2m + 3k\}$ that satisfy $\sum_{i \in \mathbf{r}} a_i = d$; for each $i \in \{1, \ldots, m\}$, we have "v_i" $\in \mathbf{r}$ if and only if $v_i = \texttt{TRUE}$, and "$\overline{v_i}$" $\in \mathbf{r}$ if and only if $v_i = \texttt{FALSE}$. Therefore, if $f\big((a_1, \ldots, a_n), d\big)$ is a subroutine for computing #LSS, then the number of satisfying truth assignments can be computed by calling f twice: first with d taking the value as prescribed above, and second with d taking the prescribed value *plus one*. The difference between the outputs from the two subroutine calls is equal to the number of distinct subsets $\mathbf{r}$ that satisfy $\sum_{i \in \mathbf{r}} a_i = d$, which is equal to the number of satisfying truth assignments for ϕ. Finally, we note that this is indeed a polynomial-time Turing reduction since the table can be populated in $O\left(m^2 k^2\right)$ simple steps, and the subroutine f is called exactly twice. ■

2.6.2 Proof of Lemma 2.3

Consider a feasible allocation $(x_1, \ldots, x_n)$; we have $\sum_{i=1}^{n} x_i \leq T$, where $x_i \geq 0$, $i = 1, \ldots, n$. Let S_r denote the number of r-subsets of $\{x_1, \ldots, x_n\}$ that have a sum of at least 1, where $r \in \{1, \ldots, n\}$. By conditioning on the number of nodes accessed by the data collector, the probability of successful recovery for this allocation can be written as

$$
\begin{aligned}
&\mathbb{P}\left[\text{successful recovery}\right] \\
&= \sum_{r=0}^{n} \mathbb{P}\left[\text{successful recovery} \,\middle|\, \text{exactly } r \text{ nodes were accessed}\right] \cdot \mathbb{P}\left[\text{exactly } r \text{ nodes were accessed}\right] \\
&= \sum_{r=1}^{n} \frac{S_r}{\binom{n}{r}} \cdot \mathbb{P}\left[\mathcal{B}\left(n, p\right) = r\right] \\
&= \sum_{r=1}^{n} S_r \; p^r (1-p)^{n-r}. \qquad (2.25)
\end{aligned}
$$

We proceed to find an upper bound for S_r. For a given r, we can write S_r inequalities of the form

$$x'_1 + \cdots + x'_r \geq 1.$$

Summing up these S_r inequalities produces an inequality of the form

$$a_1 x_1 + \cdots + a_n x_n \geq S_r.$$

Since each x_i belongs to exactly $\binom{n-1}{r-1}$ distinct r-subsets of $\{x_1, \ldots, x_n\}$, it follows that $0 \leq a_i \leq \binom{n-1}{r-1}$, $i = 1, \ldots, n$. Therefore,

$$S_r \leq a_1 x_1 + \cdots + a_n x_n \leq \binom{n-1}{r-1} \sum_{i=1}^{n} x_i \leq \binom{n-1}{r-1} T.$$

Since S_r is an integer that is at most $\binom{n}{r}$, which is the total number of r-subsets, we have

$$S_r \leq \left\lfloor \min\left(\binom{n-1}{r-1} T, \binom{n}{r} \right) \right\rfloor = \left\lfloor \min\left(\frac{rT}{n}, 1 \right) \binom{n}{r} \right\rfloor.$$

Substituting this bound into (2.25) completes the proof. ■

2.6.3 Proof of Theorem 2.4

The suboptimality gap for the symmetric allocation $\bar{\mathbf{x}}\,(n, T, m{=}n)$ is at most the difference between its recovery probability (2.5) and the upper bound (2.6) from Lemma 2.3 for the optimal recovery probability. This difference is given by

$$\begin{aligned}
&\left(\sum_{r=0}^{n} \left\lfloor \min\left(\frac{rT}{n}, 1 \right) \binom{n}{r} \right\rfloor p^r (1-p)^{n-r} \right) - \mathbb{P}\left[\mathcal{B}\,(n,p) \geq \left\lceil \frac{n}{T} \right\rceil \right] \\
&\leq \left(\sum_{r=0}^{n} \min\left(\frac{rT}{n}, 1 \right) \binom{n}{r} p^r (1-p)^{n-r} \right) - \left(\sum_{r=\lceil \frac{n}{T} \rceil}^{n} \binom{n}{r} p^r (1-p)^{n-r} \right) \\
&= \sum_{r=1}^{\lceil \frac{n}{T} \rceil - 1} \frac{rT}{n} \binom{n}{r} p^r (1-p)^{n-r} \\
&= T \sum_{r=1}^{\lceil \frac{n}{T} \rceil - 1} \binom{n-1}{r-1} p^r (1-p)^{n-r} \\
&= pT \sum_{r=1}^{\lceil \frac{n}{T} \rceil - 1} \binom{n-1}{r-1} p^{r-1} (1-p)^{(n-1)-(r-1)} \\
&= pT \sum_{\ell=0}^{\lceil \frac{n}{T} \rceil - 2} \binom{n-1}{\ell} p^{\ell} (1-p)^{(n-1)-\ell} \\
&= pT\, \mathbb{P}\left[\mathcal{B}\,(n-1,p) \leq \left\lceil \frac{n}{T} \right\rceil - 2 \right] \triangleq \delta(n,p,T),
\end{aligned}$$

as required. Assuming now that $p > \frac{1}{T}$, we have

$$\delta(n,p,T) \leq pT\,\mathbb{P}\left[\mathcal{B}\left(n-1,p\right) \leq \frac{n-1}{T}\right] \tag{2.26}$$

$$= pT\,\mathbb{P}\left[\mathcal{B}\left(n-1,p\right) \leq \frac{1}{pT}(n-1)p\right]$$

$$\leq pT\,\exp\left(-\frac{(n-1)p}{2}\left(1-\frac{1}{pT}\right)^2\right). \tag{2.27}$$

Inequality (2.26) follows from the fact that

$$\left\lceil \frac{n}{T} \right\rceil - 2 \;<\; \frac{n}{T} + 1 - 2 \;<\; \frac{n}{T} - \frac{1}{T}.$$

Inequality (2.27) follows from the observation that $\frac{1}{pT} \in (0,1)$, and the subsequent application of the Chernoff bound for deviation below the mean of the binomial distribution (e.g., [43]). For fixed p and T, this upper bound approaches zero as n goes to infinity. ■

2.6.4 Proof of Theorem 2.5

We compare the recovery probability of $\bar{\mathbf{x}}\left(n,T,m{=}\lfloor T\rfloor\right)$ to an upper bound for the recovery probabilities of all other allocations.

Suppose that $1 < T < n$. Recall from (2.7) that the probability of successful recovery for $\bar{\mathbf{x}}\left(n,T,m{=}\lfloor T\rfloor\right)$ is given by

$$P_1(p,T) \triangleq 1-(1-p)^{\lfloor T\rfloor}.$$

Consider a feasible allocation $(x_1,\ldots,x_n)$; we have $\sum_{i=1}^n x_i \leq T$, where $x_i \geq 0$, $i = 1,\ldots,n$. Let ℓ be the number of individual nodes in this allocation that store at least a unit amount of data; for brevity, we refer to these nodes as being *complete*. It follows from the budget constraint that the number of complete nodes $\ell \in \{0,1,\ldots,\lfloor T\rfloor\}$. When $\ell = \lfloor T\rfloor$, the allocation has a recovery probability identical to $P_1(p,T)$. Now, assuming that $\ell \in \{0,1,\ldots,\lfloor T\rfloor - 1\}$, successful recovery can occur in two ways:

1) when the accessed subset contains one or more complete nodes, which occurs with probability $1-(1-p)^{\ell}$; or

2) when the accessed subset contains no complete nodes but has a sum of at least 1.

In the second case, the accessed subset would consist of two or more *incomplete* nodes. Using the argument in the proof of Lemma 2.3, we can show that there are at most

$$\min\left(\binom{n-\ell-1}{r-1}(T-\ell), \binom{n-\ell}{r}\right)$$

r-subsets of incomplete nodes whose sum is at least 1, since the total amount of data stored over the $n-\ell$ incomplete nodes is at most $T-\ell$. It follows then that the recovery probability for a feasible allocation with exactly $\ell \in \{0, 1, \ldots, \lfloor T \rfloor - 1\}$ complete nodes is at most

$$P_2(n, p, T, \ell) \triangleq 1 - (1-p)^\ell + (1-p)^\ell \cdot \sum_{r=2}^{n-\ell} \min\left(\frac{T-\ell}{n-\ell} \cdot r, 1\right) \binom{n-\ell}{r} p^r (1-p)^{n-\ell-r}.$$

Thus,

$$P_1(p, T) \geq P_2(n, p, T, \ell)$$

for all $\ell \in \{0, 1, \ldots, \lfloor T \rfloor - 1\}$ is a sufficient condition for $\bar{\mathbf{x}}\,(n, T, m{=}\lfloor T \rfloor)$ to be an optimal allocation. After further simplification of this inequality, we arrive at inequality (2.8) as required. ■

2.6.5 Proof of Corollary 2.6

Suppose that $1 < T < n$. We will show that the sufficient condition of Theorem 2.5 is satisfied for any $p \leq \frac{2}{(n-\lfloor T \rfloor)^2}$. Note that when $n - \lfloor T \rfloor = 1$, or equivalently $T \in [n-1, n)$, we have to show that $\bar{\mathbf{x}}\,(n, T, m{=}\lfloor T \rfloor)$ is an optimal allocation for *any* p, i.e., in the interval $(0, 1)$.

First, observe that the summation term in inequality (2.8) is always nonnegative, i.e.,

$$\sum_{r=2}^{\left\lceil \frac{n-\ell}{T-\ell} \right\rceil - 1} \left(1 - \frac{T-\ell}{n-\ell} \cdot r\right) \binom{n-\ell}{r} \left(\frac{p}{1-p}\right)^r \geq 0,$$

since for any $r \in \left\{2, \ldots, \left\lceil \frac{n-\ell}{T-\ell} \right\rceil - 1\right\}$ and $\ell \in \{0, 1, \ldots, \lfloor T \rfloor - 1\}$, we have

$$r \leq \left\lceil \frac{n-\ell}{T-\ell} \right\rceil - 1 \iff r < \frac{n-\ell}{T-\ell} \iff 1 - \frac{T-\ell}{n-\ell} \cdot r > 0.$$

Therefore, a simpler but weaker sufficient condition for $\bar{\mathbf{x}}\,(n, T, m{=}\lfloor T \rfloor)$ to be an optimal allocation

is

$$\begin{aligned}&1-(1-p)^{\lfloor T\rfloor-n}+(n-(\lfloor T\rfloor-1))\left(\frac{p}{1-p}\right)\geq 0\\ \Longleftrightarrow&1+(n-\lfloor T\rfloor)\,p-(1-p)^{1-(n-\lfloor T\rfloor)}\geq 0,\end{aligned}$$

which is an inequality in only two variables p and $s \triangleq n-\lfloor T\rfloor$, where $s\in\{1,\ldots,n-1\}$. When $s=1$, or equivalently $T\in[n-1,n)$, this inequality is satisfied for any $p\in(0,1)$, as required. Defining the function

$$f(s,p)\triangleq 1+s\,p-(1-p)^{1-s},$$

it suffices to show that $f(s,p)\geq 0$ for any $s\in\mathbb{Z}^+$, $s\geq 2$, and $p\in\left(0,\frac{2}{s^2}\right]$. We do this by demonstrating that for any $s\in\mathbb{Z}^+$, $s\geq 2$, the function $f(s,p)$ is concave in p on the interval $p\in\left(0,\frac{2}{s^2}\right]$, and is nonnegative at both endpoints, i.e., $f(s,p{=}0)\geq 0$ and $f\left(s,p{=}\frac{2}{s^2}\right)\geq 0$.

The second-order partial derivative of $f(s,p)$ wrt p is given by

$$\frac{\partial^2}{\partial p^2}f(s,p)=-s(s-1)(1-p)^{-1-s}.$$

Since $\frac{\partial^2}{\partial p^2}f(s,p)<0$ for any $s\in\mathbb{Z}^+$, $s\geq 2$, and $p\in\left(0,\frac{2}{s^2}\right]$, it follows that the function $f(s,p)$ is concave in p on the interval $p\in\left(0,\frac{2}{s^2}\right]$ for any $s\in\mathbb{Z}^+$, $s\geq 2$.

Suppose that $s\in\mathbb{Z}^+$, $s\geq 2$. Clearly, $f(s,p{=}0)=0$. To show that $f\left(s,p{=}\frac{2}{s^2}\right)\geq 0$, we define the function

$$g(s)\triangleq\ln\left(1+\frac{2}{s}\right)+(s-1)\ln\left(1-\frac{2}{s^2}\right),$$

and show that $g(s)\geq 0$ for any $s\in\mathbb{Z}^+$, $s\geq 2$. Direct evaluation of the function gives us $g(s{=}2)=0$, and $g(s{=}3)=\ln\frac{5}{3}-2\ln\frac{9}{7}>0$. For $s\geq 4$, we consider the derivatives of $g(s)$:

$$\begin{aligned}g'(s)&=\frac{1}{s}+\frac{1}{s+2}-\frac{2(s-2)}{s^2-2}+\ln\left(1-\frac{2}{s^2}\right),\\ g''(s)&=\frac{8\left(s^3-s^2-6s-2\right)}{s^2(s+2)^2\left(s^2-2\right)^2}.\end{aligned}$$

Since $g''(s)\geq 0$ for any $s\geq 4$, and $\lim_{s\to\infty}g'(s)=0$, it follows that $g'(s)\leq 0$ for any $s\geq 4$. Now, since $g'(s)\leq 0$ for any $s\geq 4$, and $\lim_{s\to\infty}g(s)=0$, it follows that $g(s)\geq 0$ for any $s\geq 4$.

Therefore, for any $s \in \mathbb{Z}^+$, $s \geq 2$, we have

$$\begin{aligned} &\ln\left(1+\frac{2}{s}\right)+(s-1)\ln\left(1-\frac{2}{s^2}\right)=g(s)\geq 0 \\ \Longleftrightarrow &1+\frac{2}{s}\geq\left(1-\frac{2}{s^2}\right)^{1-s} \\ \Longleftrightarrow &f\left(s,p{=}\frac{2}{s^2}\right)\geq 0, \end{aligned}$$

as required. ■

2.6.6 Proof of Theorem 2.7

We will show that if condition (2.11) is satisfied, then $\Delta(p,T,k)\geq 0$ for any $k\in\mathbb{Z}^+$. First, we note that

$$\begin{aligned} \frac{\binom{\lfloor kT\rfloor}{k-1}}{\binom{\lfloor kT\rfloor}{k}} &= \frac{k}{\lfloor kT\rfloor-k+1} \\ &= \frac{k}{\lfloor k(\lfloor T\rfloor+\tau)\rfloor-k+1}, \text{ where } \tau\triangleq T-\lfloor T\rfloor\in[0,1) \\ &= \frac{k}{k\lfloor T\rfloor+\lfloor k\tau\rfloor-k+1} \\ &\geq \frac{k}{k\lfloor T\rfloor} \qquad (2.28) \\ &= \frac{1}{\lfloor T\rfloor}. \qquad (2.29) \end{aligned}$$

Inequality (2.28) follows from the fact that

$$\lfloor k\tau\rfloor\leq k\tau<k \iff \lfloor k\tau\rfloor\leq k-1 \iff \lfloor k\tau\rfloor-k+1\leq 0.$$

Now, if condition (2.11) is satisfied, then we necessarily have $T\geq 2$; otherwise, $T\in[1,2)$ would imply that $\lfloor T\rfloor=1$, which produces $(1-p)^{\lfloor T\rfloor}+2\lfloor T\rfloor p(1-p)^{\lfloor T\rfloor-1}-1=p>0$, contradicting our assumption. It follows that

$$\begin{aligned} &(1-p)^{\lfloor T\rfloor}+2\lfloor T\rfloor p(1-p)^{\lfloor T\rfloor-1}-1\leq 0 \\ \Longleftrightarrow\ &\mathbb{P}\left[\mathcal{B}\left(\lfloor T\rfloor,p\right)=0\right]+2\mathbb{P}\left[\mathcal{B}\left(\lfloor T\rfloor,p\right)=1\right]-1\leq 0 \end{aligned}$$

$$\begin{aligned}
&\iff \mathbb{P}\left[\mathcal{B}\left(\lfloor T\rfloor, p\right) \geq 2\right] \geq \mathbb{P}\left[\mathcal{B}\left(\lfloor T\rfloor, p\right) = 1\right] \\
&\iff \sum_{j=2}^{\lfloor T\rfloor} \binom{\lfloor T\rfloor}{j} p^j (1-p)^{\lfloor T\rfloor - j} \geq \lfloor T\rfloor p (1-p)^{\lfloor T\rfloor - 1} \\
&\iff \sum_{j=2}^{\lfloor T\rfloor} \frac{1}{\lfloor T\rfloor} \binom{\lfloor T\rfloor}{j} \left(\frac{p}{1-p}\right)^{j-1} \geq 1 \qquad (2.30) \\
&\implies \sum_{j=2}^{\lceil T\rceil} \frac{1}{\lfloor T\rfloor} \binom{\lceil T\rceil}{j} \left(\frac{p}{1-p}\right)^{j-1} \geq 1. \qquad (2.31)
\end{aligned}$$

Observe that $\alpha_{k,T} \triangleq \lfloor (k+1)T\rfloor - \lfloor kT\rfloor \in \{\lfloor T\rfloor, \lceil T\rceil\}$, because $\alpha_{k,T} \in \big(T-1, T+1\big)$ and there are only two integers $\lfloor T\rfloor$ and $\lceil T\rceil$, which are possibly nondistinct, in this interval. It follows from (2.30) and (2.31) that

$$\sum_{j=2}^{\alpha_{k,T}} \frac{1}{\lfloor T\rfloor} \binom{\alpha_{k,T}}{j} \left(\frac{p}{1-p}\right)^{j-1} \geq 1. \qquad (2.32)$$

Therefore, we have

$$\begin{aligned}
&\sum_{i=1}^{\min(\alpha_{k,T}-1,k)} \sum_{j=i+1}^{\alpha_{k,T}} \frac{\binom{\lfloor kT\rfloor}{k-i}}{\binom{\lfloor kT\rfloor}{k}} \binom{\alpha_{k,T}}{j} \left(\frac{p}{1-p}\right)^{-i+j} \\
&\geq \sum_{i=1}^{1} \sum_{j=i+1}^{\alpha_{k,T}} \frac{\binom{\lfloor kT\rfloor}{k-i}}{\binom{\lfloor kT\rfloor}{k}} \binom{\alpha_{k,T}}{j} \left(\frac{p}{1-p}\right)^{-i+j} \qquad (2.33) \\
&= \sum_{j=2}^{\alpha_{k,T}} \frac{\binom{\lfloor kT\rfloor}{k-1}}{\binom{\lfloor kT\rfloor}{k}} \binom{\alpha_{k,T}}{j} \left(\frac{p}{1-p}\right)^{j-1} \\
&\geq \sum_{j=2}^{\alpha_{k,T}} \frac{1}{\lfloor T\rfloor} \binom{\alpha_{k,T}}{j} \left(\frac{p}{1-p}\right)^{j-1}, \text{ from (2.29)} \\
&\geq 1, \text{ from (2.32).}
\end{aligned}$$

Inequality (2.33) follows from the fact that

$$\min(\alpha_{k,T}-1, k) \;\geq\; \min(2-1, 1) \;=\; 1.$$

Consequently,

$$\sum_{i=1}^{\min(\alpha_{k,T}-1,k)} \sum_{j=i+1}^{\alpha_{k,T}} \binom{\lfloor kT \rfloor}{k-i} \binom{\alpha_{k,T}}{j} \left(\frac{p}{1-p}\right)^{-i+j} \geq \binom{\lfloor kT \rfloor}{k}$$

$$\iff \Delta(p,T,k) \geq 0, \text{ from (2.10).}$$

It follows that

$$P_{\mathrm{S}}\left(p,T,m{=}\lfloor T \rfloor\right) \leq P_{\mathrm{S}}\left(p,T,m{=}\lfloor 2T \rfloor\right) \leq \cdots \leq P_{\mathrm{S}}\left(p,T,m{=}\left\lfloor \left\lfloor \tfrac{n}{T} \right\rfloor T \right\rfloor\right),$$

and so we conclude that an optimal m^* is given by either $m = \left\lfloor \left\lfloor \frac{n}{T} \right\rfloor T \right\rfloor$ or $m = n$. ■

2.6.7 Proof of Corollary 2.8

If $p \geq \frac{4}{3\lfloor T \rfloor}$, then we necessarily have $T \geq 2$; otherwise, $T \in [1,2)$ would imply that $\lfloor T \rfloor = 1$, which produces $p \geq \frac{4}{3\lfloor T \rfloor} = \frac{4}{3}$, contradicting the definition of p. We will show that condition (2.11) of Theorem 2.7 is satisfied for any $T \geq 2$ and $p \geq \frac{4}{3\lfloor T \rfloor}$. To do this, we define the function

$$f(p,T) \triangleq (1-p)^{\lfloor T \rfloor} + 2\lfloor T \rfloor p(1-p)^{\lfloor T \rfloor - 1} - 1,$$

and show that $f(p,T) \leq f\left(p{=}\frac{4}{3\lfloor T \rfloor}, T\right) \leq 0$ for any $T \geq 2$ and $p \geq \frac{4}{3\lfloor T \rfloor}$.

The partial derivative of $f(p,T)$ wrt p is given by

$$\frac{\partial}{\partial p} f(p,T) = \lfloor T \rfloor (1-p)^{\lfloor T \rfloor - 2} \left(1 + p - 2\lfloor T \rfloor p\right).$$

Observe that $f(p,T)$ is decreasing wrt p for any $T \geq 2$ and $p \geq \frac{4}{3\lfloor T \rfloor}$, since

$$p \geq \frac{4}{3\lfloor T \rfloor} = \frac{1}{\frac{3}{4}\lfloor T \rfloor} > \frac{1}{2\lfloor T \rfloor - 1}$$

$$\implies 2\lfloor T \rfloor p - p > 1 \iff 1 + p - 2\lfloor T \rfloor p < 0 \iff \tfrac{\partial}{\partial p} f(p,T) < 0.$$

Now, consider the function

$$g(T) \triangleq f\left(p{=}\frac{4}{3\lfloor T\rfloor}, T\right) = \left(1-\frac{4}{3\lfloor T\rfloor}\right)^{\lfloor T\rfloor-1}\left(\frac{11}{3}-\frac{4}{3\lfloor T\rfloor}\right)-1.$$

We will proceed to show that $g(T)\leq 0$ for any $T\geq 2$. For $T\in[2,3)$, we have $\lfloor T\rfloor=2$ and $g(T)=0$. To show that $g(T)\leq 0$ for any $T\geq 3$, we consider the function

$$h(T) \triangleq (T-1)\ln\left(1-\frac{4}{3T}\right)+\ln\left(\frac{11}{3}-\frac{4}{3T}\right),$$

which has the derivatives

$$\begin{aligned} h'(T) &= \frac{1}{3T-4}+\frac{11}{11T-4}+\ln\left(1-\frac{4}{3T}\right), \\ h''(T) &= \frac{16\left(11T^2-24T-16\right)}{T\left(33T^2-56T+16\right)^2}. \end{aligned}$$

Since $h''(T)>0$ for any $T\geq 3$, and $\lim_{T\to\infty}h'(T)=0$, it follows that $h'(T)\leq 0$ for any $T\geq 3$. Now, since $h'(T)\leq 0$ for any $T\geq 3$, and $h(T{=}3)=\ln\frac{29}{9}-2\ln\frac{9}{5}<0$, it follows that $h(T)<0$ for any $T\geq 3$. Thus, for any $T\geq 3$, we have

$$\begin{aligned} &(\lfloor T\rfloor{-}1)\ln\left(1-\frac{4}{3\lfloor T\rfloor}\right)+\ln\left(\frac{11}{3}-\frac{4}{3\lfloor T\rfloor}\right)=h(\lfloor T\rfloor)<0 \\ &\iff \ln\left\{\left(1-\frac{4}{3\lfloor T\rfloor}\right)^{\lfloor T\rfloor-1}\left(\frac{11}{3}-\frac{4}{3\lfloor T\rfloor}\right)\right\}<0 \\ &\iff \left(1-\frac{4}{3\lfloor T\rfloor}\right)^{\lfloor T\rfloor-1}\left(\frac{11}{3}-\frac{4}{3\lfloor T\rfloor}\right)<1 \iff g(T)<0. \end{aligned}$$

Combining these results, we obtain

$$f(p,T)\leq f\left(p{=}\frac{4}{3\lfloor T\rfloor}, T\right)=g(T)\leq 0$$

for any $T\geq 2$ and $p\geq\frac{4}{3\lfloor T\rfloor}$, as required. ■

2.6.8 Proof of Lemma 2.9

Suppose that $T > 1$. We will show that if condition (2.12) or condition (2.13) is satisfied, then $\Delta(p, T, k) \leq 0$ for any $k \in \mathbb{Z}^+$. First, we note that for any $i \in \{1, \ldots, k\}$,

$$\begin{aligned}
\frac{\binom{\lfloor kT \rfloor}{k-i}}{\binom{\lfloor kT \rfloor}{k}} &= \frac{\overbrace{(k)(k-1)\cdots(k-i+1)}^{i \text{ terms}}}{\underbrace{(\lfloor kT \rfloor - k + i)\cdots(\lfloor kT \rfloor - k + 2)(\lfloor kT \rfloor - k + 1)}_{i \text{ terms}}} \\
&\leq \left(\frac{k}{\lfloor kT \rfloor - k + 1}\right)^i \\
&\leq \left(\frac{k}{kT - 1 - k + 1}\right)^i \\
&= \left(\frac{1}{T-1}\right)^i. \qquad (2.34)
\end{aligned}$$

Now, if condition (2.12) is satisfied, then

$$\begin{aligned}
&\sum_{i=1}^{\lceil T \rceil - 1} \sum_{j=i+1}^{\lceil T \rceil} \left(\frac{1}{T-1}\right)^i \binom{\lceil T \rceil}{j} \left(\frac{p}{1-p}\right)^{-i+j} \\
&= \sum_{i=1}^{T-1} \sum_{j=i+1}^{T} \left(\frac{1}{T-1}\right)^i \binom{T}{j} \left(\frac{\frac{1}{T}}{1-\frac{1}{T}}\right)^{-i+j} \\
&= \sum_{i=1}^{T-1} \sum_{j=i+1}^{T} \binom{T}{j} \left(\frac{1}{T-1}\right)^j \\
&= \sum_{\ell=2}^{T} (\ell - 1) \binom{T}{\ell} \left(\frac{1}{T-1}\right)^\ell = 1.
\end{aligned}$$

On the other hand, if condition (2.13) is satisfied, then

$$\begin{aligned}
&\sum_{i=1}^{\lceil T \rceil - 1} \sum_{j=i+1}^{\lceil T \rceil} \left(\frac{1}{T-1}\right)^i \binom{\lceil T \rceil}{j} \left(\frac{p}{1-p}\right)^{-i+j} \\
&= \sum_{i=1}^{\lceil T \rceil - 1} \sum_{j=i+1}^{\lceil T \rceil} \binom{\lceil T \rceil}{j} \left(\frac{1-p}{p(T-1)}\right)^i \left(\frac{p}{1-p}\right)^j \\
&= \sum_{\ell=2}^{\lceil T \rceil} \left(\sum_{r=1}^{\ell-1} \left(\frac{1-p}{p(T-1)}\right)^r\right) \binom{\lceil T \rceil}{\ell} \left(\frac{p}{1-p}\right)^\ell \\
&= 1 - \frac{T\left(\frac{1}{T}\left(1-\frac{1}{T}\right)^{\lceil T \rceil - 1} - p(1-p)^{\lceil T \rceil - 1}\right)}{(1-pT)\left(1-\frac{1}{T}\right)^{\lceil T \rceil - 1}(1-p)^{\lceil T \rceil - 1}} \leq 1.
\end{aligned}$$

Thus, if either condition is satisfied, we have

$$\sum_{i=1}^{\lceil T\rceil-1}\sum_{j=i+1}^{\lceil T\rceil}\left(\frac{1}{T-1}\right)^{i}\binom{\lceil T\rceil}{j}\left(\frac{p}{1-p}\right)^{-i+j}\leq 1 \tag{2.35}$$

$$\Longrightarrow \sum_{i=1}^{\lfloor T\rfloor-1}\sum_{j=i+1}^{\lfloor T\rfloor}\left(\frac{1}{T-1}\right)^{i}\binom{\lfloor T\rfloor}{j}\left(\frac{p}{1-p}\right)^{-i+j}\leq 1. \tag{2.36}$$

As in the proof of Theorem 2.7, we note that $\alpha_{k,T}\triangleq\lfloor(k+1)T\rfloor-\lfloor kT\rfloor\in\{\lfloor T\rfloor,\lceil T\rceil\}$. It follows from (2.35) and (2.36) that

$$\sum_{i=1}^{\alpha_{k,T}-1}\sum_{j=i+1}^{\alpha_{k,T}}\left(\frac{1}{T-1}\right)^{i}\binom{\alpha_{k,T}}{j}\left(\frac{p}{1-p}\right)^{-i+j}\leq 1. \tag{2.37}$$

Therefore, we have

$$\begin{aligned}
&\sum_{i=1}^{\min(\alpha_{k,T}-1,k)}\sum_{j=i+1}^{\alpha_{k,T}}\frac{\binom{\lfloor kT\rfloor}{k-i}}{\binom{\lfloor kT\rfloor}{k}}\binom{\alpha_{k,T}}{j}\left(\frac{p}{1-p}\right)^{-i+j}\\
&\leq \sum_{i=1}^{\min(\alpha_{k,T}-1,k)}\sum_{j=i+1}^{\alpha_{k,T}}\left(\frac{1}{T-1}\right)^{i}\binom{\alpha_{k,T}}{j}\left(\frac{p}{1-p}\right)^{-i+j},\ \text{from (2.34)}\\
&\leq \sum_{i=1}^{\alpha_{k,T}-1}\sum_{j=i+1}^{\alpha_{k,T}}\left(\frac{1}{T-1}\right)^{i}\binom{\alpha_{k,T}}{j}\left(\frac{p}{1-p}\right)^{-i+j}\\
&\leq 1,\ \text{from (2.37).}
\end{aligned}$$

Consequently,

$$\begin{aligned}
&\sum_{i=1}^{\min(\alpha_{k,T}-1,k)}\sum_{j=i+1}^{\alpha_{k,T}}\binom{\lfloor kT\rfloor}{k-i}\binom{\alpha_{k,T}}{j}\left(\frac{p}{1-p}\right)^{-i+j}\leq\binom{\lfloor kT\rfloor}{k}\\
&\Longleftrightarrow \Delta(p,T,k)\leq 0,\ \text{from (2.10).}
\end{aligned}$$

It follows that

$$P_{\mathrm{S}}\left(p,T,m{=}\lfloor T\rfloor\right)\geq P_{\mathrm{S}}\left(p,T,m{=}\lfloor 2T\rfloor\right)\geq P_{\mathrm{S}}\left(p,T,m{=}\lfloor 3T\rfloor\right)\geq\cdots,$$

and since

$$P_{\mathrm{S}}\left(p,T,m{=}n\right)\begin{cases}=P_{\mathrm{S}}\left(p,T,m{=}\left\lfloor\left\lfloor\frac{n}{T}\right\rfloor T\right\rfloor\right) & \text{if } \frac{n}{T}\in\mathbb{Z}^{+},\\ \leq P_{\mathrm{S}}\left(p,T,m{=}\left\lfloor\left(\left\lfloor\frac{n}{T}\right\rfloor+1\right)T\right\rfloor\right) & \text{otherwise,}\end{cases}$$

we conclude that an optimal m^* is given by $m=\lfloor T\rfloor$. ■

2.6.9 Proof of Lemma 2.10

Since $\bar{\mathbf{x}}\left(n,T,m{=}\lfloor T\rfloor\right)$ is indeed optimal for *any* p when $T=1$, we need only consider the case of $T>1$. We will show that either condition (2.12) or condition (2.13) of Lemma 2.9 is satisfied for any $T>1$ and $p\leq\frac{2}{\lceil T\rceil}-\frac{1}{T}$. We do this in two steps: First, we define the function

$$f(p,T)\triangleq\frac{p(1-p)^{\lceil T\rceil-1}}{\frac{1}{T}\left(1-\frac{1}{T}\right)^{\lceil T\rceil-1}}-1,$$

and show that $f(p,T)\leq f\left(p{=}\frac{2}{\lceil T\rceil}{-}\frac{1}{T},T\right)\leq 0$ for any $T>1$ and $p\leq\frac{2}{\lceil T\rceil}-\frac{1}{T}$. Second, we apply the appropriate condition from Lemma 2.9 for each pair of T and p.

The partial derivative of $f(p,T)$ wrt p is given by

$$\frac{\partial}{\partial p}f(p,T)=\frac{\left(1-p\lceil T\rceil\right)(1-p)^{\lceil T\rceil-2}}{\frac{1}{T}\left(1-\frac{1}{T}\right)^{\lceil T\rceil-1}}.$$

Observe that $f(p,T)$ is nondecreasing wrt p for any $T>1$ and $p\leq\frac{2}{\lceil T\rceil}-\frac{1}{T}$, since

$$\begin{aligned}p&\leq\frac{2}{\lceil T\rceil}-\frac{1}{T}\leq\frac{2}{\lceil T\rceil}-\frac{1}{\lceil T\rceil}=\frac{1}{\lceil T\rceil}\\ \Longrightarrow p\lceil T\rceil&\leq 1\Longleftrightarrow 1-p\lceil T\rceil\geq 0\Longleftrightarrow\frac{\partial}{\partial p}f(p,T)\geq 0.\end{aligned}$$

Now, consider the function

$$g(T)\triangleq f\left(p{=}\frac{2}{\lceil T\rceil}{-}\frac{1}{T},T\right)=\frac{\left(\frac{2}{\lceil T\rceil}-\frac{1}{T}\right)\left(1-\frac{2}{\lceil T\rceil}+\frac{1}{T}\right)^{\lceil T\rceil-1}}{\frac{1}{T}\left(1-\frac{1}{T}\right)^{\lceil T\rceil-1}}-1.$$

We will proceed to show that $g(T)\leq 0$ for any $T>1$ by reparameterizing $g(T)$ as $h(c,\tau)$, where

$c \triangleq \lceil T \rceil$ and $\tau \triangleq \lceil T \rceil - T$:

$$h(c,\tau) \triangleq g\left(T{=}c{-}\tau\right) = \frac{\left(\frac{2}{c} - \frac{1}{c-\tau}\right)\left(1 - \frac{2}{c} + \frac{1}{c-\tau}\right)^{c-1}}{\frac{1}{c-\tau}\left(1 - \frac{1}{c-\tau}\right)^{c-1}} - 1.$$

The partial derivative of $h(c,\tau)$ wrt τ is given by

$$\frac{\partial}{\partial \tau} h(c,\tau) = -\frac{2\tau^2(c-2)\left(1 - \frac{2}{c} + \frac{1}{c-\tau}\right)^{c}}{\left(c(c-1-\tau)+2\tau\right)^2\left(1 - \frac{1}{c-\tau}\right)^{c}}.$$

Since $\frac{\partial}{\partial \tau} h(c,\tau) \leq 0$ for any $c \in \mathbb{Z}^+$, $c \geq 2$, and $\tau \in [0,1)$, it follows that for any $T > 1$, we have

$$\begin{aligned} g(T) &= h\left(c{=}\lceil T \rceil, \tau{=}\lceil T \rceil {-} T\right) \\ &\leq h\left(c{=}\lceil T \rceil, \tau{=}0\right) \\ &= \frac{\left(\frac{2}{\lceil T \rceil} - \frac{1}{\lceil T \rceil}\right)\left(1 - \frac{2}{\lceil T \rceil} + \frac{1}{\lceil T \rceil}\right)^{\lceil T \rceil - 1}}{\frac{1}{\lceil T \rceil}\left(1 - \frac{1}{\lceil T \rceil}\right)^{\lceil T \rceil - 1}} - 1 = 0. \end{aligned}$$

Combining these results, we obtain

$$f(p,T) \leq f\left(p{=}\frac{2}{\lceil T \rceil}{-}\frac{1}{T}, T\right) = g(T) \leq 0$$

for any $T > 1$ and $p \leq \frac{2}{\lceil T \rceil} - \frac{1}{T}$, which implies

$$p\left(1-p\right)^{\lceil T \rceil - 1} \leq \frac{1}{T}\left(1 - \frac{1}{T}\right)^{\lceil T \rceil - 1}.$$

Finally, we apply the appropriate condition from Lemma 2.9 for each pair of T and p. For $T \in \mathbb{Z}^+$, $T > 1$, we have $\frac{2}{\lceil T \rceil} - \frac{1}{T} = \frac{1}{T}$: we use condition (2.12) for $p = \frac{1}{T}$, and condition (2.13) for $p < \frac{1}{T}$. For $T \notin \mathbb{Z}^+$, $T > 1$, we have $\frac{2}{\lceil T \rceil} - \frac{1}{T} < \frac{1}{T}$: we use condition (2.13) for $p < \frac{1}{T}$. ■

2.6.10 Proof of Theorem 2.11

Since $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor T \rfloor\right)$ is indeed optimal for *any* p when $T = 1$, we need only consider the case of $T > 1$. We will show that $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor T \rfloor\right)$ is an optimal symmetric allocation for any $T > 1$

and $p \leq \frac{1}{\lceil T \rceil}$. We do this by considering subintervals of T over which $\lceil T \rceil$ is constant.

Let T be confined to the unit interval $(c, c+1]$, where $c \in \mathbb{Z}^+$. According to Lemma 2.10, $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor T \rfloor\right)$ is optimal for any $p \in \left(0, \frac{2}{c+1} - \frac{1}{T}\right]$ and $T \in (c, c+1]$, or equivalently, for any

$$p \in \left(0, \frac{1}{c+1}\right] \quad \text{and} \quad T \in \left[\frac{1}{\frac{2}{c+1} - p}, c+1\right] \cap (c, c+1].$$

This is just the area below a "peak" in Figure 2.4, expressed in terms of different independent variables. For each $p \in \left(0, \frac{1}{c+1}\right)$, we can always find a T_0 such that

$$T_0 \in \left[\frac{1}{\frac{2}{c+1} - p}, c+1\right) \cap (c, c+1).$$

For example, we can pick $T_0 = c + 1 - \delta$, where

$$\delta \triangleq \frac{1}{2}\left(c + 1 - \max\left(c, \frac{1}{\frac{2}{c+1} - p}\right)\right) \in (0, 1).$$

Now, we make the crucial observation that if $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor T \rfloor\right)$ is an optimal symmetric allocation for $T = T_0$, then $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor T \rfloor\right)$ is also an optimal symmetric allocation for any $T \in \left[\lfloor T_0 \rfloor, T_0\right]$. This claim can be proven by contradiction: the recovery probability for $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor T \rfloor\right)$ is given by

$$P_{\mathrm{S}}\left(p, T, m{=}\lfloor T \rfloor\right) = \mathbb{P}\left[\mathcal{B}\left(\lfloor T \rfloor, p\right) \geq 1\right]$$

which remains constant for all $T \in \left[\lfloor T_0 \rfloor, T_0\right]$, and a symmetric allocation that performs strictly better than $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor T \rfloor\right)$ for some $T \in \left[\lfloor T_0 \rfloor, T_0\right]$ would therefore also outperform $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor T \rfloor\right)$ for $T = T_0$. Since $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor T \rfloor\right)$ is indeed optimal for our choice of T_0, it follows then that $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor T \rfloor\right)$ is also optimal for any

$$p \in \left(0, \frac{1}{c+1}\right) \quad \text{and} \quad T \in (c, c+1].$$

By applying this result for each $c \in \mathbb{Z}^+$, we reach the conclusion that $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor T \rfloor\right)$ is an optimal symmetric allocation for any $T > 1$ and $p < \frac{1}{\lceil T \rceil}$.

Finally, to extend the optimality of $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor T \rfloor\right)$ to $p = \frac{1}{\lceil T \rceil}$, we note that the recovery

probability $P_{\mathrm{S}}(p,T,m) \triangleq \mathbb{P}\left[\mathcal{B}\left(m,p\right) \geq \left\lceil \frac{m}{T} \right\rceil\right]$ is a polynomial in p and is therefore continuous at $p = \frac{1}{\lceil T \rceil}$. Since $\bar{\mathbf{x}}\left(n,T,m{=}\lfloor T \rfloor\right)$ is optimal as $p \to \frac{1}{\lceil T \rceil}^{-}$, it remains optimal at $p = \frac{1}{\lceil T \rceil}$. ■

2.6.11 Proof of Proposition 2.12

Consider an allocation $(x_1, \ldots, x_n)$ where each x_i is a nonnegative rational number. The problem of computing the recovery probability for this allocation and a given subset size r is equivalent to the counting version of the following decision problem (which happens to be polynomial-time solvable):

Definition 2.23. Largest r-Subset Sum (LRSS)

Instance: Finite n-vector $(a_1, \ldots, a_n)$ with $a_i \in \mathbb{Z}_0^+$, file size $d \in \mathbb{Z}^+$, and subset size $r \in \mathbb{Z}^+$, where all a_i and d can be written as decimal numbers of length at most ℓ.

Question: Is there an r-subset $\mathbf{r} \subseteq \{1, \ldots, n\}$ that satisfies $\sum_{i \in \mathbf{r}} a_i \geq d$?

Note that the allocation and file size have been scaled so that the problem parameters are all integers. To show that the counting problem #LRSS is #P-complete, we essentially apply the proof of Proposition 2.2, substituting #LSS with #LRSS, and stipulating that the subset size $r = m + k$ in the Turing reduction. ■

2.6.12 Proof of Lemma 2.13

Summing up the c inequalities of (2.18) produces

$$\sum_{j=1}^{c} \sum_{i \in \mathbf{r}_j} x_i \geq c.$$

The terms on the left-hand side can be regrouped to obtain

$$\sum_{i \in S} \sum_{j=1}^{c} \mathbb{1}\left[i \in \mathbf{r}_j\right] x_i \geq c.$$

Substituting (2.19) into the above inequality yields

$$\sum_{i \in S} b\, x_i \geq c,$$

as required. ■

2.6.13 Proof of Lemma 2.14

Let $\mathcal{R}$ be the collection of all $\binom{n}{r}$ possible r-subsets of $\{1, \ldots, n\}$. If $P_{\text{S}} = 1$, then any feasible allocation must satisfy

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

Observe that each element in $\{1, \ldots, n\}$ appears the same number of times among the r-subsets in $\mathcal{R}$. Specifically, the number of r-subsets that contain element $i \in \{1, \ldots, n\}$ is just the number of ways of choosing the other $(r-1)$ elements of the r-subset from the remaining $(n-1)$ elements of $\{1, \ldots, n\}$, i.e.,

$$\sum_{\mathbf{r} \in \mathcal{R}} \mathbb{1}\left[i \in \mathbf{r}\right] = \binom{n-1}{r-1} \quad \forall \ i \in \{1, \ldots, n\}.$$

Applying Lemma 2.13 with $S = \{1, \ldots, n\}$, $c = \binom{n}{r}$, and $b = \binom{n-1}{r-1}$ therefore produces

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

for any feasible allocation. Now, $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$ is a feasible allocation since it has a recovery probability of exactly 1; because it uses the minimum possible total amount of storage $\frac{n}{r}$, this allocation is also optimal. ■

2.6.14 Proof of Theorem 2.15

Suppose that n is a multiple of r; let positive integer α be defined such that $n = \alpha r$.

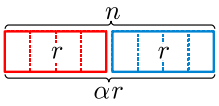
We will first prove that $P_{\text{S}} > 1 - \frac{r}{n}$ is a sufficient condition for the optimality of $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$ by showing that if the constraint

$$\sum_{i \in \mathbf{r}} x_i \geq 1 \tag{2.38}$$

is satisfied for more than $\left(1 - \frac{r}{n}\right)\binom{n}{r}$ distinct r-subsets $\mathbf{r} \subseteq \{1, \ldots, n\}$, then the allocation $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$ minimizes the required budget T. Our approach is motivated by the observation of Lemma 2.13. We begin by constructing a collection of r-subsets such that if constraint (2.38) is

Let $(n,r)=(8,4)$.

Writing $n=\alpha r$ gives $\alpha=2$.



An example of an ordered partition is
$Q=\big(\{1,2,3,4\},\{5,6,7,8\}\big)$.

Its corresponding collection of r-subsets is
$\mathcal{R}_Q=\big\{\{1,2,3,4\},\{5,6,7,8\}\big\}$.

Figure 2.9. Example for the construction of the ordered partition Q and its corresponding collection of r-subsets $\mathcal{R}_Q$, in the proof of Theorem 2.15 (when n is a multiple of r).

satisfied for the r-subsets in this collection, then $\sum_{i=1}^{n} x_i \geq \frac{n}{r}$. We then demonstrate that such a collection of r-subsets can be found among *any* collection of more than $\left(1-\frac{r}{n}\right)\binom{n}{r}$ distinct r-subsets.

Let

$$Q \triangleq (\mathbf{v}_1, \ldots, \mathbf{v}_\alpha)$$

be an ordered partition of $\{1,\ldots,n\}$ that comprises α parts, where $|\mathbf{v}_j| = r$, $j = 1,\ldots,\alpha$. For a given ordered partition Q, we specify a collection of α distinct r-subsets

$$\mathcal{R}_Q \triangleq \{\mathbf{r}_1, \ldots, \mathbf{r}_\alpha\},$$
$$\text{where } \mathbf{r}_j \triangleq \mathbf{v}_j, \quad j = 1,\ldots,\alpha.$$

Figure 2.9 provides an example of how Q and $\mathcal{R}_Q$ are constructed. Let A be the total number of possible ordered partitions Q. By counting the number of ways of picking $\mathbf{v}_j$, we have

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

Let B be the number of ordered partitions Q for which $\mathbf{r} \in \mathcal{R}_Q$, for a given r-subset $\mathbf{r} \subseteq \{1,\ldots,n\}$. By counting the number of ways of picking $\mathbf{v}_j$, subject to the requirement that $\mathbf{r} \in \mathcal{R}_Q$, we have

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

We claim that for any given ordered partition Q, if

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

then $\sum_{i=1}^{n} x_i \geq \frac{n}{r}$. To see this, observe that each element $i \in \{1, \ldots, n\}$ appears in exactly one of the α r-subsets of $\mathcal{R}_Q$, i.e.,

$$\sum_{\mathbf{r} \in \mathcal{R}_Q} \mathbb{1}\left[i \in \mathbf{r}\right] = 1 \quad \forall \ i \in \{1, \ldots, n\}.$$

Applying Lemma 2.13 with $S = \{1, \ldots, n\}$, $c = \alpha$, and $b = 1$ therefore produces $\sum_{i=1}^{n} x_i \geq \frac{\alpha}{1} = \frac{n}{r}$.

Let $\mathcal{R}$ be the collection of all $\binom{n}{r}$ possible r-subsets of $\{1, \ldots, n\}$. Observe that all A collections $\mathcal{R}_Q$ can be found in $\mathcal{R}$, i.e.,

$$\mathcal{R}_{Q_1} \subseteq \mathcal{R}, \quad \mathcal{R}_{Q_2} \subseteq \mathcal{R}, \quad \ldots, \quad \mathcal{R}_{Q_A} \subseteq \mathcal{R}.$$

With each removal of an r-subset from $\mathcal{R}$, we reduce the number of collections $\mathcal{R}_Q$ that can be found among the remaining r-subsets 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 is at least $\left\lceil \frac{A}{B} \right\rceil$, where

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

Thus, if fewer than $\frac{A}{B} = \frac{r}{n}\binom{n}{r}$ r-subsets are removed from $\mathcal{R}$, then at least one collection $\mathcal{R}_Q$ would remain; equivalently, some collection $\mathcal{R}_Q$ can be found among *any* collection of more than $\left(1 - \frac{r}{n}\right)\binom{n}{r}$ distinct r-subsets.

We have therefore shown that if $P_{\mathrm{S}} > 1 - \frac{r}{n}$, then any feasible allocation must satisfy $\sum_{i=1}^{n} x_i \geq \frac{n}{r}$. Now, $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$ is a feasible allocation since it has a recovery probability of exactly 1; because it uses the minimum possible total amount of storage $\frac{n}{r}$, this allocation is also optimal.

We proceed to prove that $P_{\mathrm{S}} > 1 - \frac{r}{n}$ is also a necessary condition for the optimality of $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$ by demonstrating that this allocation is suboptimal for any $P_{\mathrm{S}} \leq 1 - \frac{r}{n}$.

For $r < n$, the allocation $\left(0, \frac{1}{r}, \ldots, \frac{1}{r}\right)$ has a recovery probability of $\binom{n-1}{r} \big/ \binom{n}{r} = 1 - \frac{r}{n}$ and is therefore a feasible allocation for any $P_{\mathrm{S}} \leq 1 - \frac{r}{n}$. Since this allocation uses a smaller total amount of storage $\frac{n-1}{r} < \frac{n}{r}$, it is a strictly better allocation than $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$ for any $P_{\mathrm{S}} \leq 1 - \frac{r}{n}$.

For the trivial case $r = n$, we have $1 - \frac{r}{n} = 0$. The empty allocation $(0, \ldots, 0)$ is clearly optimal for any $P_{\mathrm{S}} \leq 0$. ■

2.6.15 Proof of Theorem 2.16

Suppose that n is not a multiple of r; let integers α and r' be as defined in the theorem. For brevity, we additionally define positive integers d, m, and m' such that

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

We can therefore write $n = (\alpha\, m + m')d$.

We will prove that

$$P_{\mathrm{S}} > 1 - \frac{d}{\alpha\, d + m'\, d} = 1 - \frac{1}{\alpha + m'}$$

is a sufficient condition for the optimality of $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$ by showing that if the constraint

$$\sum_{i \in \mathbf{r}} x_i \geq 1$$

is satisfied for more than $\left(1 - \frac{1}{\alpha + m'}\right)\binom{n}{r}$ distinct r-subsets $\mathbf{r} \subseteq \{1, \ldots, n\}$, then the allocation $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$ minimizes the required budget T. We apply the proof technique of Theorem 2.15, but modify the construction of the ordered partition Q and its corresponding collection of r-subsets $\mathcal{R}_Q$ to take into account the indivisibility of n by r.

For the moment, we will proceed with the assumption that $\alpha \geq 1$. Let

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

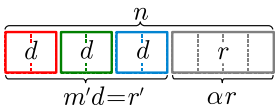
be an ordered partition of $\{1, \ldots, n\}$ that comprises $(m' + \alpha)$ parts, where

$$|\mathbf{u}_j| = d, \qquad\qquad j = 1, \ldots, m',$$

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

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

Figure 2.10. Example for the construction of the ordered partition Q and its corresponding collection of r-subsets $\mathcal{R}_Q$, in the proof of Theorem 2.16 (when n is not a multiple of r).

$$|\mathbf{v}_j| = r = m\,d, \qquad\qquad j = 1,\ldots,\alpha.$$

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

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

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

$$\text{and } \mathbf{u}_j \triangleq \mathbf{u}_{j-m'} \text{ if } j > m'.$$

Figure 2.10 provides an example of how Q and $\mathcal{R}_Q$ are constructed. Let A be the total number of possible ordered partitions Q. By counting the number of ways of picking $\mathbf{u}_j$ and $\mathbf{v}_j$, we have

$$A = \underbrace{\binom{(\alpha m+m')d}{d}\binom{(\alpha m+m'-1)d}{d}\cdots\binom{(\alpha m+1)d}{d}}_{m' \text{ terms}} \cdot \underbrace{\binom{\alpha md}{md}\binom{(\alpha-1)md}{md}\cdots\binom{md}{md}}_{\alpha \text{ terms}}$$

$$= \frac{\big((\alpha m+m')d\big)!}{(d!)^{m'}\big((md)!\big)^{\alpha}}.$$

Let B be the number of ordered partitions Q for which $\mathbf{r}\in\mathcal{R}_Q$, for a given r-subset $\mathbf{r}\subseteq\{1,\ldots,n\}$. By counting the number of ways of picking $\mathbf{u}_j$ and $\mathbf{v}_j$, subject to the requirement that $\mathbf{r}\in\mathcal{R}_Q$, we

have

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

We claim that for any given ordered partition Q, if

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

then $\sum_{i=1}^{n} x_i \geq \frac{n}{r}$. To see this, consider the partition of $\{1, \ldots, n\}$ formed by sets U and V, where

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

Correspondingly, we partition $\mathcal{R}_Q$ into two collections of r-subsets $\mathcal{R}_Q^U$ and $\mathcal{R}_Q^V$, where

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

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 $\mathbf{r}_j, \mathbf{r}_{j-1}, \ldots, \mathbf{r}_{j-(m-1)}$, where $\mathbf{r}_\ell \triangleq \mathbf{r}_{\ell+m'}$ if $\ell < 1$), i.e.,

$$\sum_{\mathbf{r} \in \mathcal{R}_Q^U} \mathbb{1}\left[i \in \mathbf{r}\right] = m \quad \forall \ i \in U.$$

Applying Lemma 2.13 with $S = U$, $c = m'$, and $b = m$ therefore produces $\sum_{i \in U} x_i \geq \frac{m'}{m} = \frac{r'}{r}$. Likewise, observe that each element $i \in V$ appears in exactly one of the α r-subsets of $\mathcal{R}_Q^V$, i.e.,

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

Applying Lemma 2.13 with $S = V$, $c = \alpha$, and $b = 1$ therefore produces $\sum_{i \in V} x_i \geq \alpha$. Combining the sums of U and V yields

$$\sum_{i=1}^{n} x_i = \sum_{i \in U} x_i + \sum_{i \in V} x_i \geq \frac{r'}{r} + \alpha = \frac{n}{r}.$$

Let $\mathcal{R}$ be the collection of all $\binom{n}{r}$ possible r-subsets of $\{1, \ldots, n\}$. As demonstrated in the proof of Theorem 2.15, if fewer than $\frac{A}{B}$ r-subsets are removed from $\mathcal{R}$, then at least one collection $\mathcal{R}_Q$ can be found among the remaining r-subsets. In this case, we have

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

Thus, some collection $\mathcal{R}_Q$ can be found among *any* collection of more than $\left(1 - \frac{1}{\alpha + m'}\right) \binom{n}{r}$ distinct r-subsets.

We have therefore shown that if $P_{\text{S}} > 1 - \frac{1}{\alpha + m'}$, then any feasible allocation must satisfy $\sum_{i=1}^{n} x_i \geq \frac{n}{r}$. Now, $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$ is a feasible allocation since it has a recovery probability of exactly 1; because it uses the minimum possible total amount of storage $\frac{n}{r}$, this allocation is also optimal.

Applying the preceding argument to the degenerate case of $\alpha = 0$ produces $\frac{A}{B} = \frac{1}{m'} \binom{n}{r}$, which is consistent with the above expression. ■

2.6.16 Proof of Corollary 2.17

Suppose that n is a multiple of $(n-r)$; let integer $\beta \geq 2$ be defined such that $n = \beta(n-r)$ $\iff n = \frac{\beta}{\beta-1}r$.

If $\beta = 2$, then $n = 2r$, i.e., n is a multiple of r. According to Theorem 2.15, $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$ is an optimal allocation if and only if

$$P_{\mathrm{S}} > 1 - \frac{r}{n} = 1 - \frac{r}{2r} = \frac{1}{2} = \frac{r}{n},$$

as required.

If $\beta \geq 3$, then n is not a multiple of r. We can write $n = \alpha\, r + r'$, where $\alpha = 0$ and $r' = n \in \{r+1, \ldots, 2r-1\}$. According to Theorem 2.16, $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$ is an optimal allocation if

$$P_{\mathrm{S}} > 1 - \frac{\gcd(r, r')}{\alpha \gcd(r, r') + r'} = 1 - \frac{\gcd(r, n)}{n} = 1 - \frac{n-r}{n} = \frac{r}{n}.$$

To show that $P_{\mathrm{S}} > \frac{r}{n}$ is also a necessary condition for the optimality of $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$, we demonstrate that this allocation is suboptimal for any $P_{\mathrm{S}} \leq \frac{r}{n}$. The allocation $(1, 0, \ldots, 0)$ has a recovery probability of $\binom{n-1}{r-1}/\binom{n}{r} = \frac{r}{n}$ and is therefore a feasible allocation for any $P_{\mathrm{S}} \leq \frac{r}{n}$. Since this allocation uses a smaller total amount of storage $1 < \frac{n}{r}$, it is a strictly better allocation than $\left(\frac{1}{r}, \ldots, \frac{1}{r}\right)$ for any $P_{\mathrm{S}} \leq \frac{r}{n}$. ■

2.6.17 Proof of Lemma 2.18

Proof of Upper Bound (2.21)*:* Consider a given choice of (n, r, T), with $T < \left\lfloor \frac{n}{r} \right\rfloor$, and a feasible allocation $(x_1, \ldots, x_n)$. Since the bound is vacuous when $r = n$, we shall assume that $1 \leq r \leq n-1$. Let $\mathbf{v} = (v_1, \ldots, v_n)$ be a permutation of the index set $\{1, \ldots, n\}$. Arrange the entries of $\mathbf{v}$ sequentially on a circle so that entry v_i is at position i. For each $k \in \{1, \ldots, n\}$, let $\mathbf{r}_k^{\mathbf{v}}$ be the interval of r entries on the circle that begins at position k, i.e., $\mathbf{r}_k^{\mathbf{v}} \triangleq \{v_k, \ldots, v_{k+r-1}\}$, where $v_i \triangleq v_{i-n}$ if $i > n$. Let $\mathcal{R}_{\mathbf{v}}$ be the collection of all n such subsets for a given $\mathbf{v}$, i.e., $\mathcal{R}_{\mathbf{v}} \triangleq \{\mathbf{r}_1^{\mathbf{v}}, \ldots, \mathbf{r}_n^{\mathbf{v}}\}$; note that these subsets are distinct.

We now show by contradiction that for any $\mathbf{v}$, there can be at most $r\lfloor T \rfloor$ successful subsets in

the collection $\mathcal{R}_{\mathbf{v}}$, i.e.,

$$|\mathcal{R}_{\mathbf{v}} \cap \mathcal{S}| \leq r\lfloor T \rfloor. \tag{2.39}$$

First, we note that the condition $T < \left\lfloor \frac{n}{r} \right\rfloor$ can be expressed equivalently as $r(\lfloor T \rfloor + 1) \leq n$ because

$$T < \left\lfloor \frac{n}{r} \right\rfloor \iff \lfloor T \rfloor < \left\lfloor \frac{n}{r} \right\rfloor \iff \lfloor T \rfloor \leq \left\lfloor \frac{n}{r} \right\rfloor - 1 \iff \lfloor T \rfloor \leq \frac{n}{r} - 1 \iff r(\lfloor T \rfloor + 1) \leq n.$$

Suppose that $|\mathcal{R}_{\mathbf{v}} \cap \mathcal{S}| \geq r\lfloor T \rfloor + 1$; for brevity, define $\mathcal{A} \triangleq \mathcal{R}_{\mathbf{v}} \cap \mathcal{S}$. Let $\mathcal{A}'$ and $\mathcal{B}$ be arbitrarily chosen collections such that

$$\mathcal{A}' \subseteq \mathcal{A}, \quad |\mathcal{A}'| = r\lfloor T \rfloor + 1, \quad \mathcal{B} \subseteq \mathcal{R}_{\mathbf{v}} \backslash \mathcal{A}', \quad |\mathcal{B}| = r - 1.$$

Note that $\mathcal{A}'$ and $\mathcal{B}$ are disjoint subcollections of $\mathcal{R}_{\mathbf{v}}$, and they can always be chosen because

$$|\mathcal{A}'| + |\mathcal{B}| = r(\lfloor T \rfloor + 1) \leq n = |\mathcal{R}_{\mathbf{v}}|.$$

We proceed to partition $\mathcal{A}' \cup \mathcal{B}$ into r parts, each containing exactly $\lfloor T \rfloor + 1$ subsets, in the following manner: First, arrange the subsets $\mathbf{r}_k^{\mathbf{v}} \in \mathcal{A}' \cup \mathcal{B}$ in ascending order of their indices k, and relabel them sequentially as $\widetilde{\mathbf{r}}_1, \ldots, \widetilde{\mathbf{r}}_{r\lfloor T \rfloor + r}$. Next, assign to each part $\ell \in \{1, \ldots, r\}$ the $\lfloor T \rfloor + 1$ subsets

$$\widetilde{\mathbf{r}}_\ell, \widetilde{\mathbf{r}}_{r+\ell}, \widetilde{\mathbf{r}}_{2r+\ell}, \ldots, \widetilde{\mathbf{r}}_{\lfloor T \rfloor r + \ell}.$$

We make the crucial observation that the subsets in each part ℓ are disjoint because there is a gap of exactly $r - 1$ entries $\widetilde{\mathbf{r}}_j$ between consecutive pairs of entries, including between the last entry $\widetilde{\mathbf{r}}_{\lfloor T \rfloor r + \ell}$ and the first entry $\widetilde{\mathbf{r}}_\ell$. Now, to recover the subsets in $\mathcal{A}'$ from $\mathcal{A}' \cup \mathcal{B}$, we remove the $r - 1$ subsets in $\mathcal{B}$. By the pigeonhole principle, at least one out of the r parts must have $\lfloor T \rfloor + 1$ subsets remaining after the removal. It follows that $\mathcal{A}'$, and therefore its superset $\mathcal{A}$, contains at least $\lfloor T \rfloor + 1$ disjoint subsets. Since $\mathcal{A}$ is the collection of successful subsets in $\mathcal{R}_{\mathbf{v}}$, we have

$$\sum_{i=1}^{n} x_i = \sum_{j=1}^{n} x_{v_j} \geq \lfloor T \rfloor + 1 > T,$$

which contradicts the budget constraint.

We now express the sum $\sum_{\mathbf{v}} |\mathcal{R}_{\mathbf{v}} \cap \mathcal{S}|$ in two ways. First, applying inequality (2.39) to each of the $n!$ choices of $\mathbf{v}$ produces

$$\sum_{\mathbf{v}} |\mathcal{R}_{\mathbf{v}} \cap \mathcal{S}| \leq n!\, r\lfloor T \rfloor. \tag{2.40}$$

Second, the sum can be written in terms of S in the following manner. For a fixed choice of successful subset $\mathbf{r} \in \mathcal{S}$ and index $k \in \{1, \ldots, n\}$, we have $\mathbf{r} = \mathbf{r}_k^{\mathbf{v}}$ for $r!(n-r)!$ choices of $\mathbf{v}$; there are $r!$ ways of arranging the elements of $\mathbf{r}$ on the corresponding interval in $\mathbf{v}$, and $(n-r)!$ ways of picking the remaining $n-r$ entries in $\mathbf{v}$. Therefore, summing over all $\mathbf{r} \in \mathcal{S}$ and $k \in \{1, \ldots, n\}$ yields

$$\sum_{\mathbf{v}} |\mathcal{R}_{\mathbf{v}} \cap \mathcal{S}| = r!(n-r)!\, S\, n. \tag{2.41}$$

Finally, substituting (2.41) into (2.40) produces

$$S \leq \frac{n!\, r\lfloor T \rfloor}{r!(n-r)!\, n} = \lfloor T \rfloor \frac{r}{n} \binom{n}{r},$$

as required.

Proof of Upper Bound (2.22)*:* Consider a given choice of (n, r, T), and a feasible allocation $(x_1, \ldots, x_n)$. Let $\mathbf{v} = (v_1, \ldots, v_n)$ be a permutation of the index set $\{1, \ldots, n\}$, and let $\widehat{\mathbf{v}}$ be the concatenation of β copies of $\mathbf{v}$. For each $k \in \{1, \ldots, \frac{\beta n}{r}\}$, let $\mathbf{r}_k^{\mathbf{v}}$ be the interval of r entries in $\widehat{\mathbf{v}}$ that begins at position $(k-1)r+1$, i.e., $\mathbf{r}_k^{\mathbf{v}} \triangleq \{v_{(k-1)r+1}, \ldots, v_{(k-1)r+r}\}$, where $v_i \triangleq v_{i-n}$ if $i > n$. Let $\mathcal{R}_{\mathbf{v}}$ be the collection of all $\frac{\beta n}{r}$ such subsets for a given $\mathbf{v}$, i.e., $\mathcal{R}_{\mathbf{v}} \triangleq \{\mathbf{r}_1^{\mathbf{v}}, \ldots, \mathbf{r}_{\beta n/r}^{\mathbf{v}}\}$. Note that the subsets in $\mathcal{R}_{\mathbf{v}}$ are distinct; otherwise, the length of $\widehat{\mathbf{v}}$ would exceed $\mathrm{lcm}(n, r)$. Observe that there can be at most $\lfloor \beta T \rfloor$ successful subsets in the collection $\mathcal{R}_{\mathbf{v}}$; otherwise, if there are $\lfloor \beta T \rfloor + 1$ or more successful subsets, then

$$\beta \sum_{i=1}^{n} x_i = \sum_{j=1}^{\beta n} x_{v_j} \geq \lfloor \beta T \rfloor + 1 > \beta T,$$

which contradicts the budget constraint. Thus, for any $\mathbf{v}$, we have

$$|\mathcal{R}_{\mathbf{v}} \cap \mathcal{S}| \leq \lfloor \beta T \rfloor. \tag{2.42}$$

We now express the sum $\sum_{\mathbf{v}} |\mathcal{R}_{\mathbf{v}} \cap \mathcal{S}|$ in two ways. First, applying inequality (2.42) to each of the $n!$ choices of $\mathbf{v}$ produces

$$\sum_{\mathbf{v}} |\mathcal{R}_{\mathbf{v}} \cap \mathcal{S}| \leq n! \, \lfloor \beta T \rfloor. \tag{2.43}$$

Second, the sum can be written in terms of S in the following manner. For a fixed choice of successful subset $\mathbf{r} \in \mathcal{S}$ and index $k \in \{1, \ldots, \frac{\beta n}{r}\}$, we have $\mathbf{r} = \mathbf{r}_k^{\mathbf{v}}$ for $r!(n-r)!$ choices of $\mathbf{v}$; there are $r!$ ways of arranging the elements of $\mathbf{r}$ on the corresponding interval in $\mathbf{v}$, and $(n-r)!$ ways of picking the remaining $n-r$ entries in $\mathbf{v}$. Therefore, summing over all $\mathbf{r} \in \mathcal{S}$ and $k \in \{1, \ldots, \frac{\beta n}{r}\}$ yields

$$\sum_{\mathbf{v}} |\mathcal{R}_{\mathbf{v}} \cap \mathcal{S}| = r!(n-r)! \, S \, \frac{\beta n}{r}. \tag{2.44}$$

Finally, substituting (2.44) into (2.43) produces

$$S \leq \frac{n! \, \lfloor \beta T \rfloor}{r!(n-r)! \frac{\beta n}{r}} = \frac{\lfloor \beta T \rfloor}{\beta} \frac{r}{n} \binom{n}{r},$$

as required. ■

2.6.18 Proof of Lemma 2.19

At $T = \frac{n}{r}$, the recovery probability corresponding to a particular choice of $\ell \in \{1, 2, \ldots, r-1\}$ is given by

$$P_{\mathrm{S}}\left(n, r, T{=}\frac{n}{r}, \ell\right) = \mathbb{P}\left[\mathcal{B}\left(r, \frac{\ell}{r}\right) \geq \ell\right].$$

We will prove that the above expression is at most $\frac{3}{4}$ for any $\ell \in \{1, 2, \ldots, r-1\}$ and $r \geq 2$ by showing that

$$\mathbb{P}\left[\mathcal{B}\left(a+b, \frac{a}{a+b}\right) \geq a\right] \leq \frac{3}{4}$$

for any positive integers a and b. To do this, we consider the following three exhaustive cases separately:

Case 1: Suppose that $a \geq 18$ and $b \geq 3$. We will first derive an upper bound for $\mathbb{P}\left[\mathcal{B}\left(a+b, \frac{a}{a+b}\right) \geq a\right]$ by finding separate bounds for $\mathbb{P}\left[\mathcal{B}\left(a+b, \frac{a}{a+b}\right) = a\right]$ and $\mathbb{P}\left[\mathcal{B}\left(a+b, \frac{a}{a+b}\right) \geq a+1\right]$; we then proceed to show that this upper bound is smaller than $\frac{3}{4}$ for any $a \geq 18$ and $b \geq 3$.

For any positive integers a and b, we have

$$\begin{aligned}\mathbb{P}\left[\mathcal{B}\left(a+b, \frac{a}{a+b}\right) = a\right] &= \binom{a+b}{a}\left(\frac{a}{a+b}\right)^a\left(\frac{b}{a+b}\right)^b \\ &< \frac{\mathrm{e}^{\frac{1}{12(a+b)}}}{\sqrt{2\pi}}\sqrt{\frac{a+b}{ab}}. \end{aligned} \tag{2.45}$$

Inequality (2.45) follows from the application of the following bound for the binomial coefficient:

$$\binom{a+b}{a} < \frac{\mathrm{e}^{\frac{1}{12(a+b)}}}{\sqrt{2\pi}}\frac{(a+b)^{a+b+\frac{1}{2}}}{a^{a+\frac{1}{2}}\,b^{b+\frac{1}{2}}},$$

which is derived from the following Stirling-based bounds for the factorial (e.g., [44]):

$$\sqrt{2\pi k}\left(\frac{k}{\mathrm{e}}\right)^k < k! < \sqrt{2\pi k}\left(\frac{k}{\mathrm{e}}\right)^k \mathrm{e}^{\frac{1}{12k}}, \qquad k \geq 1.$$

For any positive integers a and b, we have

$$\mathbb{P}\left[\mathcal{B}\left(a+b, \frac{a}{a+b}\right) \geq a+1\right] \leq \frac{1}{2}, \tag{2.46}$$

which follows from the definition of the median: The mean of the binomial random variable $\mathcal{B}\left(a+b, \frac{a}{a+b}\right)$ is $(a+b)\cdot\frac{a}{a+b} = a$; since the mean is an integer, the median coincides with the mean [45]. Therefore, according to the definition of the median, we have

$$\mathbb{P}\left[\mathcal{B}\left(a+b, \frac{a}{a+b}\right) \leq a\right] \geq \frac{1}{2},$$

which leads to inequality (2.46).

Combining bounds (2.45) and (2.46) produces

$$\mathbb{P}\left[\mathcal{B}\left(a+b,\frac{a}{a+b}\right)\geq a\right] < \frac{\mathrm{e}^{\frac{1}{12(a+b)}}}{\sqrt{2\pi}}\sqrt{\frac{a+b}{ab}}+\frac{1}{2} \triangleq f(a,b)$$

for any positive integers a and b. Now, the upper bound $f(a,b)$ is a decreasing function of both a and b since $f(a,b)$ is a symmetric function and the partial derivative

$$\frac{\partial}{\partial a}f(a,b) = -\frac{6b^2+6ab+a}{12a(a+b)^2}\,\frac{\mathrm{e}^{\frac{1}{12(a+b)}}}{\sqrt{2\pi}}\sqrt{\frac{a+b}{ab}}$$

is negative for any $a \geq 1$ and $b \geq 1$. Thus, for any $a \geq 18$ and $b \geq 3$, we have

$$f(a,b) \leq f(a{=}18, b{=}3) = \frac{\mathrm{e}^{\frac{1}{252}}}{6}\sqrt{\frac{7}{\pi}}+\frac{1}{2} \approx 0.749773 < \frac{3}{4},$$

which implies that $\mathbb{P}\left[\mathcal{B}\left(a+b,\frac{a}{a+b}\right)\geq a\right] < \frac{3}{4}$ for any positive integers $a \geq 18$ and $b \geq 3$.

Case 2: Suppose that $b \in \{1,2\}$. We will show that

$$\mathbb{P}\left[\mathcal{B}\left(a+1,\frac{a}{a+1}\right)\geq a\right] \leq \frac{3}{4} \quad \text{and} \quad \mathbb{P}\left[\mathcal{B}\left(a+2,\frac{a}{a+2}\right)\geq a\right] < \frac{3}{4}$$

for any positive integer a. The left-hand side of each inequality can be expanded and simplified to obtain the following:

$$\begin{aligned}
\mathbb{P}\left[\mathcal{B}\left(a+1,\frac{a}{a+1}\right)\geq a\right] &= \frac{a^a(2a+1)}{(a+1)^{a+1}} \triangleq f_1(a),\\
\mathbb{P}\left[\mathcal{B}\left(a+2,\frac{a}{a+2}\right)\geq a\right] &= \frac{a^a(5a^2+10a+4)}{(a+2)^{a+2}} \triangleq f_2(a).
\end{aligned}$$

The first derivatives of $f_1(a)$ and $f_2(a)$, which are given by

$$\begin{aligned}
f_1'(a) &= \frac{a^a}{(a+1)^{a+1}}\left\{2-(2a+1)\ln\left(\frac{a+1}{a}\right)\right\},\\
f_2'(a) &= \frac{a^a}{(a+2)^{a+2}}\left\{(10a+10)-(5a^2+10a+4)\ln\left(\frac{a+2}{a}\right)\right\},
\end{aligned}$$

can be shown to be negative for any $a \geq 1$. Since $f_1(a{=}1) = \frac{3}{4}$, $f_2(a{=}1) = \frac{19}{27} < \frac{3}{4}$, and both $f_1(a)$ and $f_2(a)$ are decreasing functions of a for any $a \geq 1$, it follows that $f_1(a) \leq \frac{3}{4}$ and $f_2(a) < \frac{3}{4}$ for

any positive integer a, as required.

Case 3: Suppose that $a \in \{1, 2, \ldots, 17\}$. We will describe our approach for $a = 1$ and $a = 2$; the proofs for the other 15 cases are similar, and can be verified with the help of a computer. We will show that

$$\mathbb{P}\left[\mathcal{B}\left(b+1, \frac{1}{b+1}\right) \geq 1\right] \leq \frac{3}{4} \quad \text{and} \quad \mathbb{P}\left[\mathcal{B}\left(b+2, \frac{2}{b+2}\right) \geq 2\right] < \frac{3}{4}$$

for any positive integer b. The left-hand side of each inequality can be expanded and simplified to obtain the following:

$$\mathbb{P}\left[\mathcal{B}\left(b+1, \frac{1}{b+1}\right) \geq 1\right] = 1 - \frac{b^{b+1}}{(b+1)^{b+1}} \triangleq g_1(b),$$
$$\mathbb{P}\left[\mathcal{B}\left(b+2, \frac{2}{b+2}\right) \geq 2\right] = 1 - \frac{b^{b+1}(3b+4)}{(b+2)^{b+2}} \triangleq g_2(b).$$

The first derivatives of $g_1(b)$ and $g_2(b)$, which are given by

$$g_1'(b) = \frac{b^b}{(b+1)^{b+1}} \left\{ b \ln\left(\frac{b+1}{b}\right) - 1 \right\},$$
$$g_2'(b) = \frac{b^b}{(b+2)^{b+2}} \left\{ (3b^2+4b) \ln\left(\frac{b+2}{b}\right) - (6b+4) \right\},$$

can be shown to be negative for any $b \geq 1$. Since $g_1(b{=}1) = \frac{3}{4}$, $g_2(b{=}1) = \frac{20}{27} < \frac{3}{4}$, and both $g_1(b)$ and $g_2(b)$ are decreasing functions of b for any $b \geq 1$, it follows that $g_1(b) \leq \frac{3}{4}$ and $g_2(b) < \frac{3}{4}$ for any positive integer b, as required. ■

2.6.19 Proof of Theorem 2.20

We have already established that the choice of $\ell = r$ is optimal for any $T \geq \frac{n}{r}$; it therefore suffices to show that $\ell = r$ is also optimal for any $T \in \left[\frac{n}{r}\left(\frac{3}{4}\right)^{\frac{1}{r}}, \frac{n}{r}\right)$.

The recovery probability corresponding to any $\ell \in \{1, 2, \ldots, r\}$ is given by

$$P_{\mathrm{S}}(n, r, T, \ell) = \mathbb{P}\left[\mathcal{B}\left(r, \min\left(\frac{\ell T}{n}, 1\right)\right) \geq \ell\right],$$

which is a nondecreasing function of T since $\min\left(\frac{\ell T}{n}, 1\right)$ either increases or remains constant at 1

as T increases. More precisely, $P_{\mathrm{S}}(n, r, T, \ell)$ is an increasing function of T on the interval $\left(0, \frac{n}{\ell}\right)$; for higher values of T, the function saturates at 1. We can verify this claim by checking that the partial derivative

$$\frac{\partial}{\partial p}\mathbb{P}\left[\mathcal{B}\left(r, p\right) \geq \ell\right] = \ell\binom{r}{\ell}p^{\ell-1}(1-p)^{r-\ell}$$

is positive for any $p \in (0, 1)$.

Now, the recovery probability corresponding to the choice of $\ell = r$ at $T = \frac{n}{r}\left(\frac{3}{4}\right)^{\frac{1}{r}}$ is given by

$$P_{\mathrm{S}}\left(n, r, T{=}\frac{n}{r}\left(\frac{3}{4}\right)^{\frac{1}{r}}, \ell{=}r\right) = \mathbb{P}\left[\mathcal{B}\left(r, \left(\frac{3}{4}\right)^{\frac{1}{r}}\right) \geq r\right] = \frac{3}{4}.$$

Since $P_{\mathrm{S}}(n, r, T, \ell)$ is a nondecreasing function of T, we have

$$P_{\mathrm{S}}(n, r, T, \ell{=}r) \geq \frac{3}{4} \quad \text{for any } T \geq \frac{n}{r}\left(\frac{3}{4}\right)^{\frac{1}{r}}.$$

On the other hand, for any $\ell \in \{1, 2, \ldots, r-1\}$, we have

$$P_{\mathrm{S}}(n, r, T, \ell) \leq \frac{3}{4} \quad \text{for any } T \leq \frac{n}{r},$$

from the upper bound of Lemma 2.19. It therefore follows that the choice of $\ell = r$ is optimal for any $T \in \left[\frac{n}{r}\left(\frac{3}{4}\right)^{\frac{1}{r}}, \frac{n}{r}\right)$, as required. ■

2.6.20 Proof of Corollary 2.21

Theorem 2.20 already demonstrates that the choice of $\ell = r$ is optimal for any $T \geq \frac{n}{r}\left(\frac{3}{4}\right)^{\frac{1}{r}}$; we will proceed to show that a recovery probability of at least $\frac{3}{4}$ is *not* achievable for any $T < \frac{n}{r}\left(\frac{3}{4}\right)^{\frac{1}{r}}$.

Recall from the proof of Theorem 2.20 that the recovery probability $P_{\mathrm{S}}(n, r, T, \ell)$ corresponding to any $\ell \in \{1, 2, \ldots, r\}$ is an increasing function of T on the interval $\left(0, \frac{n}{\ell}\right)$. Thus, for the choice of $\ell = r$, the function $P_{\mathrm{S}}(n, r, T, \ell{=}r)$ increases wrt T on the subinterval $\left(0, \frac{n}{r}\left(\frac{3}{4}\right)^{\frac{1}{r}}\right] \subset \left(0, \frac{n}{r}\right)$; since $P_{\mathrm{S}}\left(n, r, T{=}\frac{n}{r}\left(\frac{3}{4}\right)^{\frac{1}{r}}, \ell{=}r\right) = \frac{3}{4}$, it follows that

$$P_{\mathrm{S}}(n, r, T, \ell{=}r) < \frac{3}{4} \quad \text{for any } T < \frac{n}{r}\left(\frac{3}{4}\right)^{\frac{1}{r}}.$$

On the other hand, for any $\ell \in \{1, 2, \ldots, r-1\}$, the function $P_{\mathrm{S}}(n, r, T, \ell)$ increases wrt T on the subinterval $\left(0, \frac{n}{r}\right] \subset \left(0, \frac{n}{\ell}\right)$; since $P_{\mathrm{S}}\left(n, r, T{=}\frac{n}{r}, \ell\right) \leq \frac{3}{4}$ according to Lemma 2.19, it follows that

$$P_{\mathrm{S}}(n, r, T, \ell) < \frac{3}{4} \quad \text{for any } T < \frac{n}{r}.$$

Hence, the optimal recovery probability for any $T < \frac{n}{r}\left(\frac{3}{4}\right)^{\frac{1}{r}}$ is strictly less than $\frac{3}{4}$. ■

2.7 Acknowledgment

The author and his coauthors A. G. Dimakis and T. Ho would like to thank Brighten Godfrey and Robert Kleinberg for introducing the problem to them and for sharing their insights. They also thank Dimitris Achlioptas, Shaowei Lin, Liang Ze Wong, and Daniel Chen for the helpful discussions.

Chapter 3

Distributed Storage Allocations for Optimal Delay

3.1 Introduction

Consider a network of n mobile storage nodes. A source node creates a single data object of unit size (without loss of generality), and disseminates an encoded representation of it to other nodes for storage, subject to a given total storage budget T. Let x_i be the amount of coded data eventually stored in node $i \in \{1, \ldots, n\}$ at the end of the data dissemination process. Any amount of data may be stored in each node, as long as the total amount of storage used over all nodes is at most the given budget T, that is, $\sum_{i=1}^{n} x_i \leq T$.

At some time after the completion of the data dissemination process, a data collector node begins to recover the original data object by contacting other nodes and accessing the data stored in them. We make the simplifying assumption that the stored data is instantaneously transmitted on contact; this approximates the case where there is sufficient bandwidth and time for data transmission during each contact. This data recovery process continues until the data object can be recovered from the cumulatively accessed data. Let random variable D denote the recovery delay incurred by the data collector, defined as the earliest time at which successful recovery can occur, measured from the beginning of the data recovery process. Figure 3.1 depicts the information flows in such a network.

By using an appropriate code for the data dissemination process and eventual storage, successful

The material in this chapter was presented in part in [46].

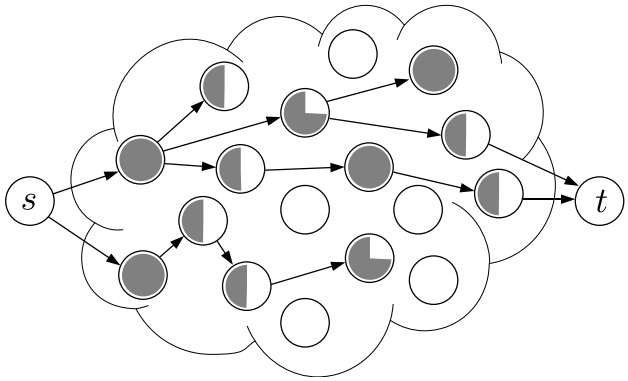


Figure 3.1. Information flows originating at the source s, some of which finally arrive at the data collector t. Different amounts of coded data may eventually be stored in each storage node, subject to the given total storage budget T.

recovery can be achieved when the total amount of data accessed by the data collector is at least the size of the original data object. This can be accomplished with random linear codes [28, 29] or a suitable MDS code, for example. Thus, if $\mathbf{r}_d \subseteq \{1, \ldots, n\}$ is the set of all nodes contacted by the data collector by time d, then the recovery delay D can be written as

$$D \triangleq \min \left\{ d : \sum_{i \in \mathbf{r}_d} x_i \geq 1 \right\}.$$

Our goal is to find a storage allocation $(x_1, \ldots, x_n)$ that produces the optimal recovery delay, subject to the given budget constraint. Specifically, we shall examine the following two objectives involving the recovery delay D:

1) maximization of the probability of successful recovery by a given deadline d, or **recovery probability** $\mathbb{P}[D \leq d]$; and

2) minimization of the **expected recovery delay** $\mathbb{E}[D]$.

By solving for the optimal allocation, we will also be able to determine whether coding is beneficial for recovery delay. For example, uncoded replication would suffice if each nonempty node is to store the data object in its entirety (i.e., $x_i \geq 1$ for all $i \in S$, and $x_i = 0$ for all $i \notin S$, where S is some subset of $\{1, \ldots, n\}$); the data collector would not need to combine data accessed from different nodes in order to recover the data object.

The nodes of the network are assumed to move around and contact each other according to an exogenous random process; they are unable to change their trajectories in response to the data

dissemination or recovery processes. (The recovery delay could be improved significantly if nodes were otherwise allowed to act on oracular knowledge about future contact opportunities [47], for example.)

Most work on delay-tolerant networking traditionally assume that the data object is intended for immediate consumption; both the data dissemination and recovery processes would therefore begin at the same time, and the recovery delay would be measured from the beginning of the data dissemination process. In contrast, our model more accurately reflects the characteristics of longer-term storage where the data object can be consumed long after its creation. Nonetheless, our model can still be a good approximation for short-term storage especially when the data dissemination process occurs very rapidly, as in the case of binary Spray-and-Wait [13] where the number of nodes disseminating or spraying data grows exponentially over time.

We also note that in most of the literature involving distributed storage, either the data object is assumed to be replicated in its entirety (see, for e.g., [13]), or, if coding is used, every node is assumed to store the same amount of coded data (see, for e.g., [3–7]). Allocations of a storage budget with nodes possibly storing different amounts of data are not usually considered.

3.1.1 Our Contribution

This chapter attempts to address the gaps in our understanding of how the choice of storage allocation can affect the recovery delay performance. We formulate a simple analytical model of the problem and show that the maximization of the **recovery probability** $\mathbb{P}[D \leq d]$ can be expressed in terms of the reliability maximization problem introduced in [8]. It turns out that the simple strategies of spreading the budget minimally (i.e., uncoded replication) and spreading the budget maximally over all n nodes (i.e., setting $x_i = \frac{T}{n}$ for all i) may both be suboptimal; in fact, the optimal allocation may not even be symmetric (we say that an allocation is *symmetric* when all nonzero x_i are equal). Applying our results from Section 2.2.3, we can show that minimal spreading is optimal among symmetric allocations when the deadline d is sufficiently small, while maximal spreading is optimal among symmetric allocations when the deadline d is sufficiently large.

For the minimization of the **expected recovery delay** $\mathbb{E}[D]$, we are able to characterize the optimal *symmetric* allocation completely: minimal spreading (i.e., uncoded replication) turns out to be optimal whenever the budget T is an integer; otherwise, the amount of spreading in the optimal

symmetric allocation increases with the fractional part of T.

Interestingly, our analytical results demonstrate that the optimal symmetric allocation for the two objectives can be quite different. In particular, when the budget T is an integer, we observe a phase transition in the optimal symmetric allocation as the deadline d increases, for the maximization of **recovery probability** $\mathbb{P}[D \leq d]$; however, minimal spreading (i.e., uncoded replication) alone turns out to be optimal for the minimization of **expected recovery delay** $\mathbb{E}[D]$.

We proceed to apply our theoretical insights to the design of a simple data dissemination and storage protocol for a mobile delay-tolerant network. Our protocol generalizes Spray-and-Wait [13] by allowing the use of variable-size coded packets. Using network simulations, we compare the performance of different symmetric allocations under various circumstances. These simulations allow us to capture the transient dynamics of the data dissemination process that were simplified in the analytical model. Our main result shows that a maximal spreading of the budget is optimal in the *high recovery probability regime*. Specifically, maximal spreading can lead to a significant reduction in the wait time required to attain a desired recovery probability. We also evaluate the protocol against a real-world data set consisting of the mobility traces of taxi cabs operating in a city. Besides validating the predictions made in our theoretical analysis, these simulations also reveal several interesting properties of the allocations under different circumstances.

3.1.2 Other Related Work

Jain *et al.* [11] and Wang *et al.* [48] evaluated the delay performance of symmetric allocations experimentally in the context of routing in a delay-tolerant network. Our results complement and generalize several aspects of their work.

We present a theoretical analysis of the problem in Section 3.2, and undertake a simulation study in Section 3.3. Proofs of theorems are deferred to Section 3.5.

3.2 Theoretical Analysis

We adopt the following notation throughout this chapter:

n total number of storage nodes, $n \geq 2$

λ contact rate between any given pair of nodes, $\lambda > 0$

x_i amount of data stored in node $i \in \{1, \ldots, n\}$, $x_i \geq 0$

T total storage budget, $1 \leq T \leq n$

D random variable denoting recovery delay

The indicator function is denoted by $\mathbb{1}[G]$, which equals 1 if statement G is true, and 0 otherwise. We use $\mathcal{B}(n, p)$ to denote the binomial random variable with n trials and success probability p. An allocation $(x_1, \ldots, x_n)$ is said to be *symmetric* when all nonzero x_i are equal; for brevity, let $\bar{\mathbf{x}}(n, T, m)$ denote the symmetric allocation for n nodes that uses a total storage of T and contains exactly $m \in \{1, \ldots, n\}$ nonempty nodes, that is,

$$\bar{\mathbf{x}}(n, T, m) \triangleq \Bigg(\underbrace{\frac{T}{m}, \ldots, \frac{T}{m}}_{m \text{ terms}}, \underbrace{0, \ldots, 0}_{(n-m) \text{ terms}} \Bigg).$$

The number of contacts between any given pair of nodes in the network is assumed to follow a Poisson distribution with rate parameter λ; the time between contacts is therefore described by an exponential distribution with mean $\frac{1}{\lambda}$. Let $W_1, \ldots, W_n$ be i.i.d. random variables denoting the times at which the data collector first contacts node $1, \ldots, n$, respectively, where $W_i \sim \text{Exponential}(\lambda)$.

3.2.1 Maximization of Recovery Probability $\mathbb{P}[D \leq d]$

Let the given recovery deadline be $d > 0$, and let the subset of nodes contacted by the data collector by time d be $\mathbf{r} \subseteq \{1, \ldots, n\}$. Successful recovery occurs by time d if and only if the total amount of data stored in the subset $\mathbf{r}$ of nodes is at least 1. In other words, the recovery delay D is at most d if and only if $\sum_{i \in \mathbf{r}} x_i \geq 1$. Since the data collector contacts each node by time d independently with constant probability $p_{\lambda,d}$, given by

$$p_{\lambda,d} \triangleq \mathbb{P}[W \leq d] = F_W(d) = 1 - \mathrm{e}^{-\lambda d},$$

it follows that the probability of contacting exactly a subset $\mathbf{r}$ of nodes by time d is $p_{\lambda,d}^{|\mathbf{r}|}(1 - p_{\lambda,d})^{n-|\mathbf{r}|}$. The recovery probability $\mathbb{P}[D \leq d]$ can therefore be obtained by summing

over all possible subsets $\mathbf{r}$ that allow successful recovery:

$$\mathbb{P}\left[D \le d\right] = \sum_{\substack{\mathbf{r}\subseteq\{1,\ldots,n\}:\\ |\mathbf{r}|\ge 1}} p_{\lambda,d}^{|\mathbf{r}|}(1-p_{\lambda,d})^{n-|\mathbf{r}|} \cdot \mathbb{1}\left[\sum_{i\in\mathbf{r}} x_i \ge 1\right]. \tag{3.1}$$

We seek an optimal allocation $(x_1,\ldots,x_n)$ of the budget T (that is, subject to $\sum_{i=1}^n x_i \le T$, where $x_i \ge 0$ for all i) that maximizes $\mathbb{P}\left[D \le d\right]$, for a given choice of n, λ, d, and T.

This problem matches the reliability maximization problem of Section 2.2 with $p_{\lambda,d}$ as the access probability; we recall that the optimal allocation may be nonsymmetric and can be difficult to find. However, if we restrict the optimization to only *symmetric* allocations, then we can specify the solution for a wide range of parameter values of $p_{\lambda,d}$ and T. Specifically, if λ or d is sufficiently small, e.g., $p_{\lambda,d} \le \frac{1}{\lceil T\rceil}$, then $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor T\rfloor\right)$, which corresponds to a minimal spreading of the budget (i.e., uncoded replication), is an optimal symmetric allocation. On the other hand, if λ or d is sufficiently large, e.g., $p_{\lambda,d} \ge \frac{4}{3\lfloor T\rfloor}$, then either $\bar{\mathbf{x}}\left(n, T, m{=}\left\lfloor \left\lfloor \frac{n}{T}\right\rfloor T\right\rfloor\right)$ or $\bar{\mathbf{x}}\left(n, T, m{=}n\right)$, which correspond to a maximal spreading of the budget, is an optimal symmetric allocation.

3.2.2 Minimization of Expected Recovery Delay $\mathbb{E}\left[D\right]$

Rewriting (3.1) in terms of the underlying random variables gives us the following c.d.f. for the recovery delay D:

$$F_D(t) = \sum_{\substack{\mathbf{r}\subseteq\{1,\ldots,n\}:\\ |\mathbf{r}|\ge 1}} \big(F_W(t)\big)^{|\mathbf{r}|}\big(1-F_W(t)\big)^{n-|\mathbf{r}|} \cdot \mathbb{1}\left[\sum_{i\in\mathbf{r}} x_i \ge 1\right].$$

Differentiating $F_D(t)$ wrt t produces the p.d.f.

$$f_D(t) = \sum_{\substack{\mathbf{r}\subseteq\{1,\ldots,n\}:\\ |\mathbf{r}|\ge 1}} \big(F_W(t)\big)^{|\mathbf{r}|-1}\big(1-F_W(t)\big)^{n-|\mathbf{r}|-1}\big(|\mathbf{r}| - n\,F_W(t)\big) f_W(t) \cdot \mathbb{1}\left[\sum_{i\in\mathbf{r}} x_i \ge 1\right].$$

Therefore, assuming $\sum_{i=1}^n x_i \ge 1$ which is necessary for successful recovery, we can compute the expected recovery delay as follows:

$$\mathbb{E}\left[D\right] = \int_0^\infty t\, f_D(t)\, dt$$

$$= \sum_{\substack{\mathbf{r}\subseteq\{1,\ldots,n\}:\\ |\mathbf{r}|\geq 1}} \left(\int_0^\infty t\big(F_W(t)\big)^{|\mathbf{r}|-1}\big(1-F_W(t)\big)^{n-|\mathbf{r}|-1}\big(|\mathbf{r}|-n\,F_W(t)\big) f_W(t)\,dt \right) \cdot \mathbb{1}\left[\sum_{i\in\mathbf{r}} x_i \geq 1\right]$$

$$= \frac{1}{\lambda}\left(H_n - \sum_{\substack{\mathbf{r}\subseteq\{1,\ldots,n\}:\\ 1\leq|\mathbf{r}|\leq n-1}} \frac{1}{(n-|\mathbf{r}|)\binom{n}{|\mathbf{r}|}} \cdot \mathbb{1}\left[\sum_{i\in\mathbf{r}} x_i \geq 1\right]\right), \tag{3.2}$$

where $H_n \triangleq \sum_{i=1}^n \frac{1}{i}$ is the n^{th} harmonic number. We seek an optimal allocation $(x_1,\ldots,x_n)$ of the budget T (that is, subject to $\sum_{i=1}^n x_i \leq T$, where $x_i \geq 0$ for all i) that minimizes $\mathbb{E}[D]$, for a given choice of n, λ, and T. Note that the optimal allocation is independent of λ for the minimization of $\mathbb{E}[D]$ but not for the maximization of $\mathbb{P}[D \leq d]$.

The optimal value of $\mathbb{E}[D]$ can be bounded as follows:

Lemma 3.1. *The expected recovery delay $\mathbb{E}[D]$ of an optimal allocation is at least*

$$\frac{1}{\lambda}\left(H_n - \sum_{r=1}^{n-1} \frac{\min\left(\frac{rT}{n},1\right)}{n-r}\right).$$

We make the following conjecture about the optimal allocation, based on our numerical observations:

Conjecture 3.2. *A* symmetric *optimal allocation always exists for any n, λ, and T.*

As a simplification, we now proceed to restrict the optimization to only *symmetric* allocations (which are easier to describe and implement, and appear to perform well). For the symmetric allocation $\bar{\mathbf{x}}(n,T,m)$, successful recovery occurs by a given deadline d if and only if $\left\lceil 1/\left(\frac{T}{m}\right)\right\rceil = \left\lceil \frac{m}{T}\right\rceil$ or more nonempty nodes are contacted by the data collector by time d, out of a total of m nonempty nodes. It follows that the resulting recovery probability is given by $\mathbb{P}[D\leq d] = \mathbb{P}\left[\mathcal{B}(m,p_{\lambda,d}) \geq \left\lceil \frac{m}{T}\right\rceil\right]$. We therefore obtain the following c.d.f. and p.d.f. for the recovery delay D:

$$F_D(t) = \sum_{r=\left\lceil \frac{m}{T}\right\rceil}^{m} \binom{m}{r}\big(F_W(t)\big)^r\big(1-F_W(t)\big)^{m-r},$$

$$f_D(t) = \binom{m}{\left\lceil \frac{m}{T}\right\rceil}\left\lceil\frac{m}{T}\right\rceil\big(F_W(t)\big)^{\left\lceil \frac{m}{T}\right\rceil-1}\big(1-F_W(t)\big)^{m-\left\lceil \frac{m}{T}\right\rceil} f_W(t).$$

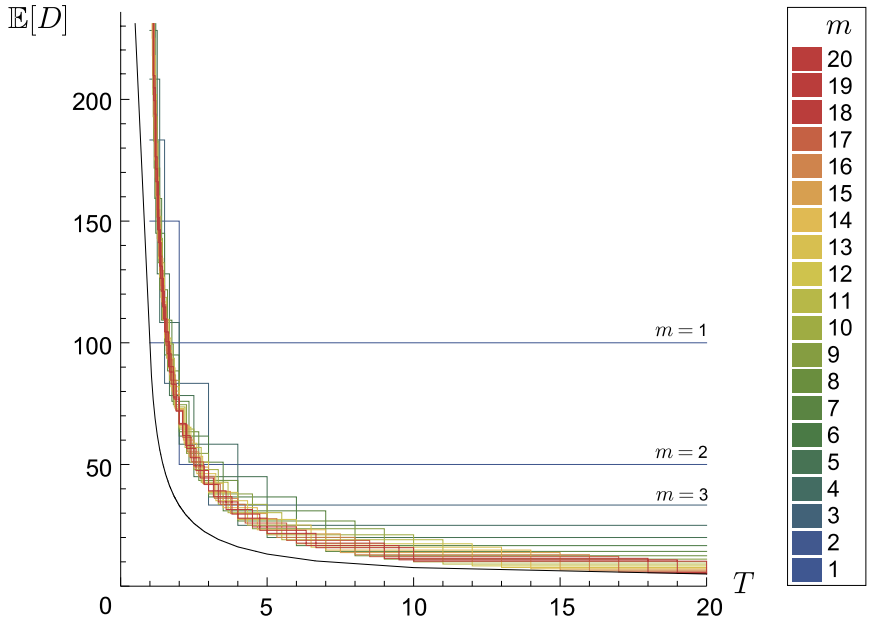


Figure 3.2. Plot of expected recovery delay $\mathbb{E}[D]$ against budget T for each symmetric allocation $\bar{\mathbf{x}}(n, T, m)$, for $(n, \lambda) = \left(20, \frac{1}{100}\right)$. Parameter m denotes the number of nonempty nodes in the symmetric allocation. The black curve gives a lower bound for the expected recovery delay of an optimal allocation, as derived in Lemma 3.1.

Thus, we can compute the expected recovery delay as follows:

$$\mathbb{E}[D] = \int_0^\infty t\, f_D(t)\, dt = \frac{1}{\lambda} \sum_{i=1}^{\lceil \frac{m}{T} \rceil} \frac{1}{m - \lceil \frac{m}{T} \rceil + i} \triangleq E_D(\lambda, T, m).$$

Figure 3.2 compares the performance of different symmetric allocations over different budgets T, for an instance of n and λ; the value of m corresponding to the optimal symmetric allocation appears to change in a nontrivial manner as we vary the budget T. Fortunately, we can eliminate many candidates for the optimal value of m by making the following observation (a similar observation was made for the maximization of the recovery probability in Section 2.2.3): For fixed n, λ, and T, we have

$$\left\lceil \frac{m}{T} \right\rceil = k \qquad \text{when } m \in \big((k-1)T, kT\big],$$

for $k = 1, 2, \ldots, \left\lfloor \frac{n}{T} \right\rfloor$, and finally,

$$\left\lceil \frac{m}{T} \right\rceil = \left\lfloor \frac{n}{T} \right\rfloor + 1 \text{ when } m \in \left(\left\lfloor \frac{n}{T} \right\rfloor T, n \right].$$

Since $\frac{1}{\lambda}\sum_{i=1}^{k}\frac{1}{m-k+i}$ is decreasing in m for constant λ and k, it follows that $E_D(\lambda,T,m)$ is minimized over each of these intervals of m when we pick m to be the largest integer in the corresponding interval. Thus, given n, λ, and T, we can find an optimal m^* that minimizes $E_D(\lambda,T,m)$ over all m from among $\lceil \frac{n}{T} \rceil$ candidates:

$$\left\{ \lfloor T \rfloor, \lfloor 2T \rfloor, \ldots, \left\lfloor \left\lfloor \frac{n}{T} \right\rfloor T \right\rfloor, n \right\}. \tag{3.3}$$

Note that when $m = \lfloor kT \rfloor$, $k \in \mathbb{Z}^+$, the expected recovery delay simplifies to the following expression:

$$E_D\left(\lambda, T, m{=}\lfloor kT \rfloor\right) = \frac{1}{\lambda}\sum_{i=1}^{k}\frac{1}{\lfloor kT \rfloor - k + i}.$$

By further eliminating suboptimal candidate values for m^* using suitable bounds for the harmonic number, we are able to completely characterize the optimal symmetric allocation for any n, λ, and T:

Theorem 3.3. *Suppose $T = a + 1 - \frac{1}{\ell}$, where $a \in \mathbb{Z}^+$, $\ell \geq 1$. If $\lfloor \ell \rfloor \leq \left\lfloor \frac{n}{T} \right\rfloor$, then*

$$\bar{\mathbf{x}}\left(n, T, m{=}\lfloor \lfloor \ell \rfloor T \rfloor\right)$$

is an optimal symmetric allocation; if $\lfloor \ell \rfloor > \left\lfloor \frac{n}{T} \right\rfloor$, then

$$\textit{either } \bar{\mathbf{x}}\left(n, T, m{=}\left\lfloor \left\lfloor \tfrac{n}{T} \right\rfloor T \right\rfloor\right) \textit{ or } \bar{\mathbf{x}}\left(n, T, m{=}n\right)$$

is an optimal symmetric allocation.

If the budget T is an integer (i.e., $\ell = 1$), then $\lfloor \ell \rfloor \leq \left\lfloor \frac{n}{T} \right\rfloor$ is always true, and so $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor T \rfloor\right)$, which corresponds to a minimal spreading of the budget (i.e., uncoded replication), is an optimal symmetric allocation. However, if the budget T is not an integer (i.e., $\ell > 1$), then the amount of spreading in the optimal symmetric allocation increases with the fractional part of T, up to a point at which either $\bar{\mathbf{x}}\left(n, T, m{=}\left\lfloor \left\lfloor \frac{n}{T} \right\rfloor T \right\rfloor\right)$ or $\bar{\mathbf{x}}\left(n, T, m{=}n\right)$, which correspond to a maximal spreading of the budget, becomes optimal. Minimal spreading (i.e., uncoded replication) therefore performs well over the whole range of budgets T, being optimal among symmetric

allocations whenever T is an integer (its suboptimality at noninteger $T = T_0$ can be bounded by the step difference in $E_D(\lambda, T, m{=}\lfloor T \rfloor)$ between $T = T_0$ and $T = \lceil T_0 \rceil$, since $E_D(\lambda, T, m)$ is a nonincreasing function of T).

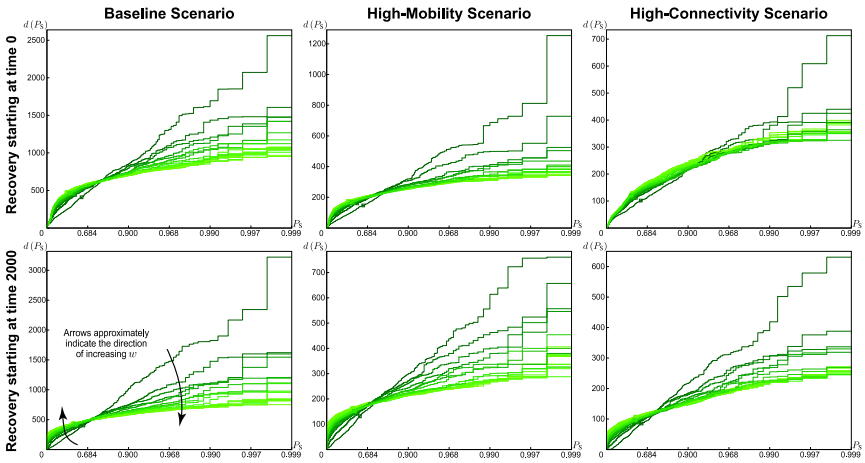
In summary, we note that the optimal symmetric allocation for the two objectives can be quite different. In particular, when the budget T is an integer, we observe a phase transition from a regime where minimal spreading is optimal to a regime where maximal spreading is optimal, as the deadline d increases, for the maximization of **recovery probability** $\mathbb{P}[D \leq d]$; however, with the averaging over both regimes, minimal spreading (i.e., uncoded replication) alone turns out to be optimal for the minimization of **expected recovery delay** $\mathbb{E}[D]$.

3.3 Simulation Study

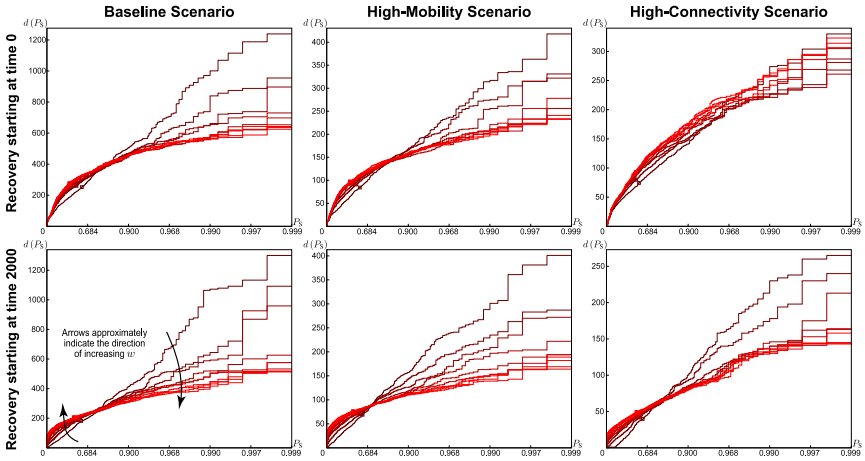
We apply our theoretical insights to the design of a simple data dissemination and storage protocol for a mobile delay-tolerant network. Our protocol extends Spray-and-Wait [13] by allowing nodes to store *coded* packets that are each $\frac{1}{w}$ the size of the original data object, where parameter w is a positive integer; successful recovery occurs when the data collector accesses at least w such packets. Different *symmetric* allocations of the given total storage budget T can be realized by choosing different values of w; the original protocol, which uses uncoded replication, corresponds to $w = 1$.

3.3.1 Protocol Description

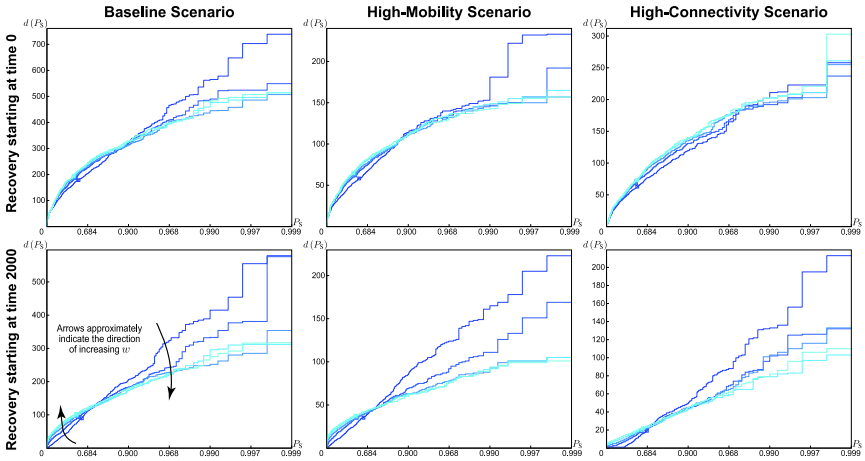
The source node begins with a total storage budget of T times the size of the original data object, which translates to wT coded packets, each $\frac{1}{w}$ the size of the original data object. Whenever a node with more than one packet contacts another node without any packets, the former gives *half its packets* to the latter. The actual amount of data stored or transmitted by a node never exceeds the size of the original data object (or w packets) since the excess packets can always be generated on demand (using random linear coding, for example). To reduce the total transmission cost incurred, a node can also directly transmit *one packet* to each node it meets when it has w or fewer packets left; otherwise, these last few packets would be transmitted multiple times by different nodes. The dissemination process is completed when no node has more than one packet.



(a) Budget $T = 5$

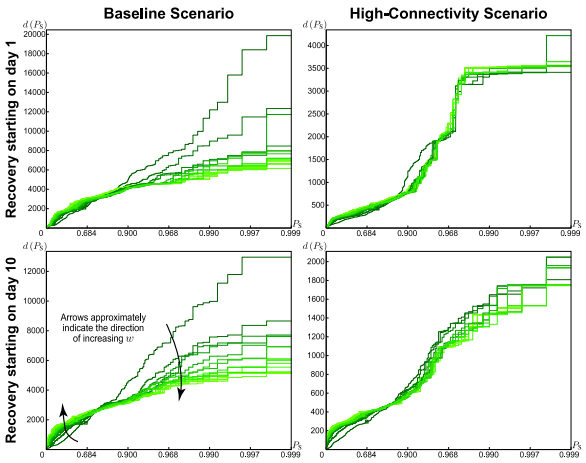


(b) Budget $T = 10$

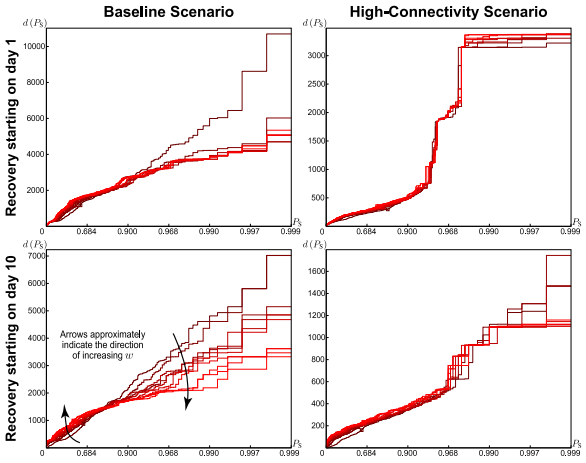


(c) Budget $T = 20$

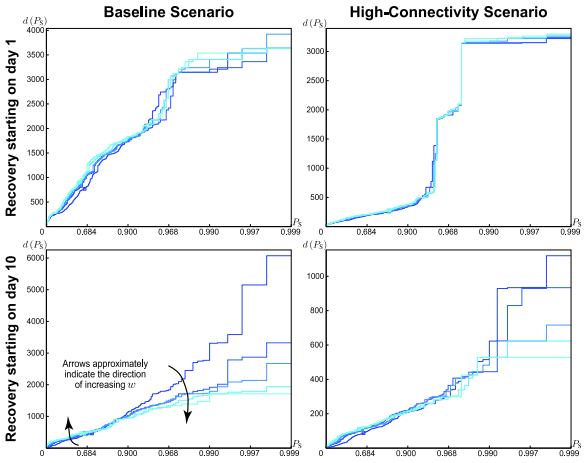
Figure 3.3. Plots of required wait time $d(P_{\mathrm{S}})$ against desired recovery probability P_{S} for the simulations using a **random waypoint** mobility model, for budgets $T = 5, 10, 20$. Each colored line represents a specific choice of parameter $w \in \left\{1, \ldots, \frac{n}{T}\right\}$, with $w = 1$ (darkest) corresponding to a minimal spreading of the budget (i.e., uncoded replication), and $w = \frac{n}{T}$ (lightest) corresponding to a maximal spreading of the budget. The mean recovery delay corresponding to each line is indicated by a square marker.



(a) Budget $T = 5$



(b) Budget $T = 10$



(c) Budget $T = 20$

Figure 3.4. Plots of required wait time in minutes $d(P_S)$ against desired recovery probability P_S for the simulations using **mobility traces**, for budgets $T = 5, 10, 20$. Each colored line represents a specific choice of parameter $w \in \left\{1, \ldots, \frac{n}{T}\right\}$, with $w = 1$ (darkest) corresponding to a minimal spreading of the budget (i.e., uncoded replication), and $w = \frac{n}{T}$ (lightest) corresponding to a maximal spreading of the budget. The mean recovery delay corresponding to each line is indicated by a square marker.

3.3.2 Network Model and Simulation Setup

We implemented a discrete-time simulation of $n = 100$ wireless mobile nodes in a 1000×1000 grid. A random waypoint mobility model is assumed where at each time step, each node moves a random distance $L \sim$ Uniform[5,10] towards a selected destination; on arrival, the node selects a random point on the grid as its next destination. Each node has a communication range of 20, and the bandwidth of each point-to-point link is large enough to support the transmission of w packets in one time step. At each time step, a maximal number of transmissions are randomly scheduled such that each node can transmit to or receive from at most one other node in range, and exactly one node may transmit in the range of a node receiving a transmission. In addition to this **baseline scenario**, we also considered the following two scenarios:

1) a **high-mobility scenario**, where the distance traveled by each node is increased to $L \sim$ Uniform[25,50]; and

2) a **high-connectivity scenario**, where the communication range is increased to 80.

We measured the recovery delay incurred by the data collector for two cases:

1) when the data recovery process begins at **time 0**, i.e., at the beginning of the data dissemination process; and

2) when the data recovery process begins at **time 2000**, i.e., when the data dissemination process is already underway or completed. (This is a more appropriate performance metric for longer-term storage.)

We ran the simulation 500 times for each choice of budget $T \in \{5,10,20\}$ and parameter $w \in \left\{1, 2, \ldots, \frac{n}{T}\right\}$ under each scenario, with a random pair of nodes appointed as the source and data collector for each run.

3.3.3 Simulation Results

Figure 3.3 shows how the required wait time $d(P_{\mathrm{S}})$, given by

$$d(P_{\mathrm{S}}) \triangleq \min\{d : \mathbb{P}[D \leq d] \geq P_{\mathrm{S}}\},$$

varies with the desired recovery probability P_{S} for each choice of parameter w; these plots essentially describe how much time must elapse before a desired percentage of data collectors are able to recover the data object. The **recovery probability** performance of the protocol (which can be inferred by flipping the axes) is mostly consistent with our analysis in Section 3.2.1; specifically, the phase transition in the optimal symmetric allocation is clearly discernible in most of the plots. The **expected recovery delay** performance is also mostly consistent with our analysis in Section 3.2.2, with minimal spreading of the budget ($w = 1$) being optimal in most of the plots.

The plots for the high-mobility scenario appear to be vertically scaled versions of the plots for the baseline scenario. This is not surprising because an increase in node mobility approximately translates to a speeding up of time. The effect of increasing node connectivity, on the other hand, seems less straightforward: the phase transition in the optimal symmetric allocation is evident for recovery starting at time 2000 but not for recovery starting at time 0. This discrepancy suggests that the data dissemination process is somewhat impeded by the increased connectivity, possibly due to greater interference.

We observe that in the *high recovery probability regime*, maximal spreading of the budget ($w = \frac{n}{T}$) can lead to a significant reduction in the required wait time. For example, given a budget of $T = 10$ and a desired recovery probability of $P_{\mathrm{S}} = 0.99$, choosing maximal spreading ($w = 10$) instead of minimal spreading or uncoded replication ($w = 1$) can yield a reduction of 40% to 60% in the required wait time for the baseline and high-mobility scenarios.

We also observe that the recovery start time appears to have a limited impact on how the different allocations perform relative to each other; the most noticeable effect of starting recovery at time 0 is the reduced spread in performance across different choices of parameter w, especially in the low recovery probability regime. This can be explained by the similarity of the different allocations during the data dissemination process: in the beginning, the different choices of parameter w would see the same allocation of the budget over the nodes because only a few nodes have been reached by the source directly or indirectly through relays; the different allocations are eventually realized only after a sufficient amount of time has passed.

3.3.4 Evaluation on Mobility Traces

To gain a better understanding of how our protocol might perform in a real-world setting, we evaluated it on a CRAWDAD data set comprising mobility traces of taxi cabs in San Francisco [49]. The traces of 100 randomly selected cabs with GPS coordinate readings over the span of an 18-day period were used. The GPS readings were sampled at approximately 60-second intervals; because reading times were not synchronized across cabs, we estimated the position of a cab at any given time using linear interpolation. For better accuracy, we assumed that a cab became inactive whenever the time between consecutive readings exceeded 2 minutes. As in the preceding simulations, we considered different scenarios and data recovery start times. Two scenarios were considered here:

1) a **baseline scenario**, where the communication range of each cab is 20 m; and

2) a **high-connectivity scenario**, where the communication range is increased to 80 m.

We measured the recovery delay incurred by the data collector for two cases:

1) when the data recovery process begins on **day 1**; and

2) when the data recovery process begins on **day 10**, i.e., half-way through the 18-day period.

We ran the simulation 500 times for each choice of budget $T \in \{5,10,20\}$ and parameter $w \in \left\{1, 2, \ldots, \frac{n}{T}\right\}$ under each scenario, with a random pair of cabs appointed as the source and data collector for each run.

Figure 3.4 shows how the required wait time $d(P_{\mathrm{S}})$ varies with the desired recovery probability P_{S} for each choice of parameter w. Compared to the plots of Figure 3.3 for the random waypoint simulations, these plots exhibit distinct "jumps" in the wait times, which can be attributed to the reduced mobility of the cabs at night. Despite these nonideal conditions, many of the observations made for the previous simulations are still applicable here. For instance, the phase transition in the optimal symmetric allocation is discernible in most of the plots for the baseline scenario. Also, starting recovery on day 1 has the effect of reducing the spread in performance across different choices of parameter w, especially in the low recovery probability regime.

Once again, we observe that in the *high recovery probability regime*, maximal spreading of the budget ($w = \frac{n}{T}$) can lead to a significant reduction in the required wait time. For example, given

a budget of $T = 10$ and a desired recovery probability of $P_{\mathrm{S}} = 0.99$, choosing maximal spreading ($w = 10$) instead of minimal spreading or uncoded replication ($w = 1$) can yield a reduction of 30% to 50% in the required wait time for the baseline scenario.

3.4 Conclusion and Future Work

We examined the recovery delay performance of different distributed storage allocations for a network of mobile storage nodes. Our theoretical analysis and simulation study show that the choice of objective function (i.e., recovery probability vs. expected recovery delay) can lead to very different optimal symmetric allocations, and that picking the right allocation for the given circumstances can make a significant difference in performance.

The work in this chapter can be extended in several directions. The simple contact model assumed here can be generalized to the case where a variable amount of data is transmitted during each contact between nodes. Another natural generalization is to allow nonuniform contact rates λ_i between the data collector and individual nodes.

3.5 Proofs of Theorems

3.5.1 Proof of Lemma 3.1

Consider a feasible allocation $(x_1, \ldots, x_n)$; we have $\sum_{i=1}^{n} x_i \leq T$, where $x_i \geq 0$, $i = 1, \ldots, n$. Let S_r denote the number of r-subsets of $\{x_1, \ldots, x_n\}$ that have a sum of at least 1, where $r \in \{1, \ldots, n\}$. Recall from the proof of Lemma 2.3 that S_r can be bounded as follows:

$$S_r \leq \min\left(\binom{n-1}{r-1}T, \binom{n}{r}\right).$$

We can now rewrite (3.2) in terms of S_r by enumerating subsets according to size:

$$\begin{aligned}\mathbb{E}[D] &= \frac{1}{\lambda}\left(H_n - \sum_{r=1}^{n-1} S_r \cdot \frac{1}{(n-r)\binom{n}{r}}\right) \\ &\geq \frac{1}{\lambda}\left(H_n - \sum_{r=1}^{n-1} \frac{\min\left(\binom{n-1}{r-1}T, \binom{n}{r}\right)}{(n-r)\binom{n}{r}}\right)\end{aligned}$$

$$= \frac{1}{\lambda}\left(H_n - \sum_{r=1}^{n-1} \frac{\min\left(\frac{rT}{n}, 1\right)}{n-r}\right).$$

■

3.5.2 Proof of Theorem 3.3

Suppose $T = a+1-\frac{1}{\ell}$, where $a \in \mathbb{Z}^+, \ell \geq 1$. Since $kT = (a+1)k - \frac{k}{\ell}$, the expected recovery delay for the symmetric allocation $\bar{\mathbf{x}}\,(n, T, m{=}\lfloor kT \rfloor)$, where $k \in \mathbb{Z}^+$, can be written as

$$E_D\,(\lambda, T, m{=}\lfloor kT \rfloor) = \frac{1}{\lambda}\sum_{i=1}^{k} \frac{1}{(a+1)k - \left\lceil \frac{k}{\ell} \right\rceil - k + i} = \frac{1}{\lambda}\sum_{i=1}^{k} \frac{1}{ak - \left\lceil \frac{k}{\ell} \right\rceil + i}.$$

Observe that $\left\lceil \frac{k}{\ell} \right\rceil = v$ when $k \in \big((v-1)\ell, v\ell\big]$, for $v = 1, 2, \ldots$. To compare $E_D\,(\lambda, T, m{=}\lfloor kT \rfloor)$ within each of these intervals of k, we introduce Lemma 3.4:

Lemma 3.4. *For $a, v, k \in \mathbb{Z}^+$, $k \geq \frac{v}{a}$, the function*

$$f(a, v, k) \triangleq \sum_{i=1}^{k} \frac{1}{ak - v + i} = H_{ak-v+k} - H_{ak-v}$$

decreases with k.

Proof of Lemma 3.4: Let $\Delta(a, v, k)$ denote the difference in the function value between consecutive values of k, that is,

$$\begin{aligned}
&\Delta(a, v, k) \triangleq f(a, v, k) - f(a, v, k+1) \\
&= (H_{ak-v+k} - H_{ak-v}) - (H_{ak-v+k+a+1} - H_{ak-v+a}) \\
&= (H_{ak-v+a} - H_{ak-v}) - (H_{ak-v+k+a+1} - H_{ak-v+k}) \\
&= \left(\sum_{i=1}^{a} \frac{1}{ak - v + i} - \frac{1}{ak - v + k + i}\right) - \frac{1}{ak - v + k + a + 1} \\
&= \left(\sum_{i=1}^{a} \frac{k}{(ak - v + i)(ak - v + k + i)}\right) - \frac{1}{ak - v + k + a + 1}.
\end{aligned}$$

We will proceed to show that $\Delta(a,v,k) > 0$ for any $a,v,k \in \mathbb{Z}^+$, $k \geq \frac{v}{a}$. First, we find a lower bound for the summation term using a geometrical argument. Consider the function

$$g(t) \triangleq \frac{k}{(ak-v+t)(ak-v+k+t)},$$

which has the second derivative

$$g''(t) = \frac{2}{(ak-v+t)^3} - \frac{2}{(ak-v+k+t)^3}.$$

For any $a,v,k \in \mathbb{Z}^+$, $k \geq \frac{v}{a}$, the function $g(t)$ is positive, decreasing with t, and convex (since $g''(t) > 0$), on the interval $t \in (0,\infty)$. We therefore have the lower bound

$$\sum_{i=1}^{a} \frac{k}{(ak-v+i)(ak-v+k+i)} > \int_1^{a+1} g(t)\,dt + \frac{g(1)-g(a+1)}{2},$$

which implies that

$$\begin{aligned}\Delta(a,v,k) > \ln\left(\frac{(ak-v+a+1)(ak-v+k+1)}{(ak-v+k+a+1)(ak-v+1)}\right) + \frac{k}{2(ak-v+1)(ak-v+k+1)} \\ - \frac{k}{2(ak-v+a+1)(ak-v+k+a+1)} - \frac{1}{ak-v+k+a+1} \;\triangleq\; h(a,v,k).\end{aligned}$$

Now, it suffices to show that $h(a,v,k) \geq 0$ for any $a,v,k \in \mathbb{Z}^+$, $k \geq \frac{v}{a}$. This is indeed the case since

$$\lim_{k\to\infty} h(a,v,k) = 0,$$

and the partial derivative $\frac{\partial}{\partial k}h(a,v,k)$, which is given by

$$\begin{aligned}\frac{a}{2}\left(\frac{2(ak-v+a+1)+1}{(ak-v+a+1)^2} - \frac{2(ak-v+1)+1}{(ak-v+1)^2}\right) \\ + \frac{a+1}{2}\left(\frac{2(ak-v+k+1)+1}{(ak-v+k+1)^2} - \frac{2(ak-v+k+a+1)-1}{(ak-v+k+a+1)^2}\right),\end{aligned}$$

can be shown to be negative. ■

It follows from Lemma 3.4 that for each $v \in \mathbb{Z}^+$, the expected recovery delay

$E_D\left(\lambda, T, m{=}\lfloor kT \rfloor\right)$ decreases as k takes larger values in the interval $\big((v-1)\ell, v\ell\big]$, that is,

$$\begin{aligned} & E_D\left(\lambda, T, m{=}\Big\lfloor\big(\lfloor(v-1)\ell\rfloor+1\big)T\Big\rfloor\right) \\ > \; & E_D\left(\lambda, T, m{=}\Big\lfloor\big(\lfloor(v-1)\ell\rfloor+2\big)T\Big\rfloor\right) \\ > \; & \cdots \\ > \; & E_D\left(\lambda, T, m{=}\lfloor\lfloor v\ell\rfloor T\rfloor\right). \end{aligned}$$

We will proceed to show that

$$E_D\left(\lambda, T, m{=}\lfloor\lfloor v\ell\rfloor T\rfloor\right) \geq E_D\left(\lambda, T, m{=}\lfloor\lfloor \ell\rfloor T\rfloor\right)$$

for all $v \in \mathbb{Z}^+$. This is equivalent to showing that

$$\sum_{i=1}^{\lfloor v\ell\rfloor} \frac{1}{a\lfloor v\ell\rfloor - v + i} \geq \sum_{i=1}^{\lfloor \ell\rfloor} \frac{1}{a\lfloor \ell\rfloor - 1 + i}$$

for any $\ell \geq 1$, $a, v \in \mathbb{Z}^+$. According to Lemma 3.4, we have

$$\sum_{i=1}^{\lfloor v\ell\rfloor} \frac{1}{a\lfloor v\ell\rfloor - v + i} \geq \sum_{i=1}^{v\lfloor\ell\rfloor+v-1} \frac{1}{a\left(v\lfloor\ell\rfloor + v - 1\right) - v + i},$$

since we can substitute ℓ with $\lfloor\ell\rfloor + \tau$, where $\tau \in [0, 1)$, which yields

$$\lfloor v\ell\rfloor = \lfloor v\lfloor\ell\rfloor + v\tau\rfloor = v\lfloor\ell\rfloor + \lfloor v\tau\rfloor \leq v\lfloor\ell\rfloor + v - 1.$$

Defining the function

$$f(a, \ell, v) \triangleq \sum_{i=1}^{v\ell+v-1} \frac{1}{a\left(v\ell + v - 1\right) - v + i} = H_{\big((a+1)(\ell+1)-1\big)v-(a+1)} - H_{\big(a(\ell+1)-1\big)v-a},$$

it therefore suffices to show that

$$f(a, \ell, v) \geq f(a, \ell, v{=}1) \tag{3.4}$$

for any $a, \ell, v \in \mathbb{Z}^+$.

To obtain lower and upper bounds for $f(a,\ell,v)$, we apply the following bounds [50] for the harmonic number H_n, $n \geq 1$:

$$\underbrace{\ln\left(n+\frac{1}{2}\right)+\gamma+\frac{1}{24(n+1)^2}}_{\triangleq H_{\text{LB}}(n)} < H_n < \underbrace{\ln\left(n+\frac{1}{2}\right)+\gamma+\frac{1}{24n^2}}_{\triangleq H_{\text{UB}}(n)},$$

where γ is the Euler-Mascheroni constant. This produces the lower bound

$$f_{\text{LB}}(a,\ell,v) \triangleq H_{\text{LB}}\left(\big((a+1)(\ell+1)-1\big)v-(a+1)\big)\right) - H_{\text{UB}}\left(\big(a(\ell+1)-1\big)v-a\big)\right),$$

and the upper bound

$$f_{\text{UB}}(a,\ell,v) \triangleq H_{\text{UB}}\left(\big((a+1)(\ell+1)-1\big)v-(a+1)\big)\right) - H_{\text{LB}}\left(\big(a(\ell+1)-1\big)v-a\big)\right),$$

for $\big(a(\ell+1)-1\big)v-a \geq 1$. The lower bound $f_{\text{LB}}(a,\ell,v)$ is an increasing function of v for any $a \geq 1$, $\ell \geq 1$, $v \geq 2$, since the partial derivative $\frac{\partial}{\partial v} f_{\text{LB}}(a,\ell,v)$, which is given by

$$\frac{2(\ell-1)}{\big(2\big((a+1)(\ell+1)-1\big)v-2(a+1)+1\big)\big(2\big(a(\ell+1)-1\big)v-2a+1\big)} + \frac{a(\ell+1)-1}{12\left(\big(a(\ell+1)-1\big)v-a\right)^3} - \frac{(a+1)(\ell+1)-1}{12\left(\big((a+1)(\ell+1)-1\big)v-a\right)^3},$$

can be shown to be positive. We therefore have

$$f(a,\ell,v) \geq f_{\text{LB}}(a,\ell,v) \geq f_{\text{LB}}(a,\ell,v{=}2)$$

for any $v \geq 2$, $a,\ell,v \in \mathbb{Z}^+$. We now proceed to demonstrate that $f_{\text{LB}}(a,\ell,v{=}2) \geq f(a,\ell,v{=}1)$.

For the case $\ell = 1$, consider the function

$$g(a) \triangleq f_{\text{LB}}(a,\ell{=}1,v{=}2) - f(a,\ell{=}1,v{=}1) = \ln\left(\frac{2a+1}{2a-1}\right) - \frac{81a^4-71a^2+16}{a(9a^2-4)^2}.$$

It suffices to show that $g(a) \geq 0$ for any $a \geq 1$, which is indeed the case since

$$\lim_{a\to\infty} g(a) = 0,$$

and the derivative

$$g'(a) = -\frac{621a^6 - 961a^4 + 436a^2 - 64}{a^2(4a^2-1)(9a^2-4)^3}$$

is negative.

For the case $\ell \geq 2$, we consider the function

$$h(a,\ell) \triangleq f_{\text{LB}}(a,\ell,v{=}2) - f_{\text{UB}}(a,\ell,v{=}1),$$

which can be shown to be nonnegative for any $a \geq 1$, $\ell \geq 2$. It follows that

$$f_{\text{LB}}(a,\ell,v{=}2) \geq f_{\text{UB}}(a,\ell,v{=}1) \geq f(a,\ell,v{=}1)$$

for any $\ell \geq 2$, $a, \ell \in \mathbb{Z}^+$.

Combining these results, we obtain

$$f(a,\ell,v) \geq f_{\text{LB}}(a,\ell,v) \geq f_{\text{LB}}(a,\ell,v{=}2) \geq f(a,\ell,v{=}1)$$

for any $v \geq 2$, $a, \ell, v \in \mathbb{Z}^+$, which gives us inequality (3.4) as required. Consequently, we have

$$E_D\left(\lambda, T, m{=}\lfloor kT \rfloor\right) \geq E_D\left(\lambda, T, m{=}\lfloor \lfloor \ell \rfloor T \rfloor\right)$$

for any $k \in \mathbb{Z}^+$. Since

$$E_D\left(\lambda, T, m{=}n\right) \begin{cases} = E_D\left(\lambda, T, m{=}\left\lfloor \left\lfloor \frac{n}{T} \right\rfloor T \right\rfloor\right) & \text{if } \frac{n}{T} \in \mathbb{Z}^+, \\ \geq E_D\left(\lambda, T, m{=}\left\lfloor \left(\left\lfloor \frac{n}{T} \right\rfloor + 1\right) T \right\rfloor\right) & \text{otherwise,} \end{cases}$$

we also have

$$E_D\left(\lambda, T, m{=}n\right) \geq E_D\left(\lambda, T, m{=}\lfloor \lfloor \ell \rfloor T \rfloor\right).$$

Therefore, if $\lfloor \ell \rfloor \leq \left\lfloor \frac{n}{T} \right\rfloor$, then $\bar{\mathbf{x}}\left(n, T, m{=}\lfloor \lfloor \ell \rfloor T \rfloor\right)$ is an optimal symmetric allocation. On the other hand, if $\lfloor \ell \rfloor > \left\lfloor \frac{n}{T} \right\rfloor$, then we can eliminate all but the two largest candidate values for m^* in (3.3), since

$$E_D\left(\lambda, T, m{=}\lfloor T \rfloor\right) > E_D\left(\lambda, T, m{=}\lfloor 2T \rfloor\right) > \cdots > E_D\left(\lambda, T, m{=}\left\lfloor \left\lfloor \frac{n}{T} \right\rfloor T \right\rfloor\right)$$

by Lemma 3.4. ■

Chapter 4

Coding for Real-Time Streaming under Packet Erasures

4.1 Introduction

We consider a real-time streaming system where messages created at regular time intervals at a source are encoded for transmission to a receiver over a packet erasure link; the receiver must subsequently decode each message within a given delay from its creation time.

Three erasure models are studied in this chapter. The first is a window-based erasure model in which all erasure patterns containing a limited number of erasures in each specifically defined window are admissible. We consider two variations of this model; one based on the coding window and the other on a sliding window. The second is a bursty erasure model in which all erasure patterns containing erasure bursts of a limited length are admissible. The third is an i.i.d. erasure model in which each transmitted packet is erased independently with the same probability. For the first and second erasure models, the objective is to find a code that achieves the maximum message size, among all codes that allow all messages to be decoded by their respective decoding deadlines under all admissible erasure patterns. For the third erasure model, the objective is to find a code that achieves the maximum decoding probability for a given message size.

Our Contribution: We show that a time-invariant intrasession code is asymptotically optimal over all codes (time-varying and time-invariant, intersession and intrasession) as the number of

The material in this chapter was presented in part in [51].

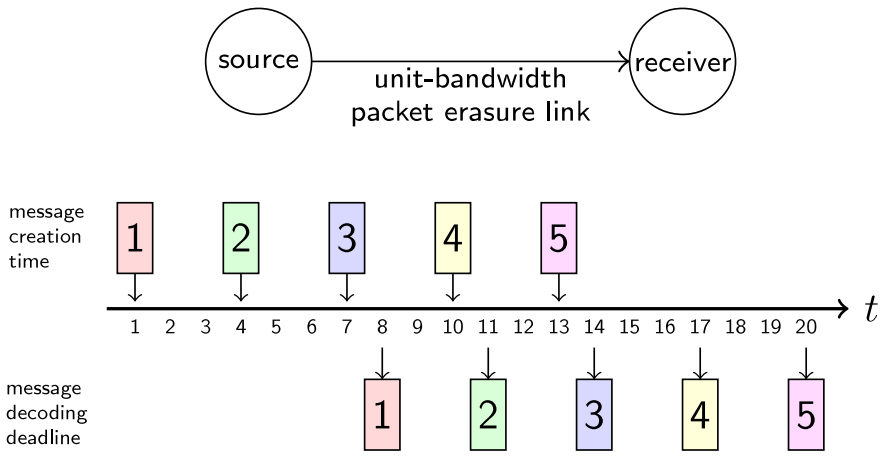


Figure 4.1. Real-time streaming system for $(c, d) = (3, 8)$. Each of the messages $\{1, \ldots, 5\}$ is assigned a unique color. Messages are created at regular intervals of c time steps at the source, and must be decoded within a delay of d time steps from their respective creation times at the receiver. At each time step t, the source transmits a single data packet of normalized unit size over the packet erasure link.

messages goes to infinity, for both the coding window and sliding window variations of the window-based erasure model, and for the bursty erasure model when the maximum erasure burst length is sufficiently short or long. Intrasession coding is attractive due to its relative simplicity (it allows coding within the same message but not across different messages), but it is not known in general when intrasession coding is sufficient or when intersession coding is necessary. For the bursty erasure model, we also show that diagonally interleaved codes derived from specific systematic block codes are asymptotically optimal over all codes in certain cases.

For the i.i.d. erasure model, we derive an upper bound on the decoding probability for any time-invariant code, and show that the gap between this bound and the performance of a family of time-invariant intrasession codes is small when the message size and packet erasure probability are small. In a simulation study, these codes performed well against a family of random time-invariant convolutional codes under a number of scenarios.

Related Work: Martinian *et al.* [52, 53] and Badr *et al.* [54] provide constructions of streaming codes that minimize the decoding delay for certain types of bursty erasure models. Tree codes or anytime codes, for which the decoding failure probability decays exponentially with delay, are considered in [15–17]. Convolutional codes similar to those in our simulation study for the i.i.d. erasure model were also examined by Polyanskiy [14] with the expected decoding delay as the performance metric. Tekin *et al.* [55] considered erasure correction coding for a non-real-time streaming system

where all messages are initially present at the source.

The systems literature on real-time streaming deals mainly with the transmission of media content (i.e., video and audio) over the Internet, with the user-perceived quality of the received stream as the performance metric. In practice, the encoding of the raw media content, packetization of the coded data (possibly with interleaving) for transmission, and application of forward error correction (FEC) codes are usually performed by different components of the system separately (e.g., [56,57]). FEC codes (e.g., exclusive-or parity [58], Reed-Solomon [59]), if used, are typically applied to blocks of packets to generate separate parity or repair packets (e.g., [60, 61]). Furthermore, the decoding delay requirement is not explicitly considered during the coding process. The patent of Rasmussen *et al.* [62] describes a system in which a live stream of data is divided into segments, each of which is encoded into one or more transmission blocks using an FEC code (e.g., LT [63], Reed-Solomon); these blocks are optionally subdivided and interleaved in a variety of ways before being transmitted over one or more channels. A similar streaming system is also considered in the patent of Luby *et al.* [64], which describes computationally efficient methods for decoding FEC-encoded blocks to achieve low latency.

We begin with a formal definition of the problem in Section 4.2, followed by a description of our code constructions in Section 4.3. In Sections 4.4, 4.5, and 4.6, we examine the three erasure models in detail and state our main results. Proofs of theorems are deferred to Section 4.8.

4.2 Problem Definition

Consider a discrete-time data streaming system comprising a source and a receiver, with a directed unit-bandwidth packet erasure link from the source to the receiver. Independent messages of uniform size $s > 0$ are created at regular intervals of $c \in \mathbb{Z}^+$ time steps at the source. At each time step $t \in \mathbb{Z}^+$, the source transmits a single data packet of normalized unit size over the packet erasure link; either the entire packet is received instantaneously by the receiver at time step t, or the entire packet is erased and never received. The receiver must subsequently decode each message within a delay of $d \in \mathbb{Z}^+$ time steps from its creation time. Figure 4.1 depicts this real-time streaming system for an instance of (c, d).

More precisely, let random variable M_k denote message k; the random variables $\{M_k\}$ are

independent, and $H(M_k) = s$ for each $k \in \mathbb{Z}^+$. To simplify our definition of the encoding functions, we shall further assume that $M_1, M_2, \ldots$ are identically distributed, and nonpositive messages $M_0, M_{-1}, \ldots$ are zeros.

Each message $k \in \mathbb{Z}^+$ is created at time step $(k-1)c+1$, and is to be decoded by time step $(k-1)c+d$. Let W_k be the *coding window* for message k, which we define as the interval of d time steps between its creation time and decoding deadline, i.e.,

$$W_k \triangleq \{(k-1)c+1, \ldots, (k-1)c+d\}.$$

We shall assume that $d > c$ so as to avoid the degenerate case of nonoverlapping coding windows for which it is sufficient to code individual messages separately.

The unit-size packet transmitted at each time step $t \in \mathbb{Z}^+$ must be a function of messages created at time step t or earlier. Let random variable X_t denote the packet transmitted at time step t; we have $H(X_t) \leq 1$ for each $t \in \mathbb{Z}^+$. For brevity, we define $X[A] \triangleq (X_t)_{t \in A}$.

Because we are dealing with hard message decoding deadlines and fixed-size messages and packets, it is reasonable to adopt a zero-error notion of decodability. Specifically, a given message k is considered to be decodable from the packets received at time steps $t \in A$ if and only if

$$H\big(M_k \,\big|\, X[A]\big) = 0.$$

Consider the first n messages $\{1, \ldots, n\}$, and the union of their (overlapping) coding windows T_n given by

$$T_n \triangleq W_1 \cup \cdots \cup W_n = \{1, \ldots, (n-1)c+d\}.$$

An *erasure pattern* $E \subseteq T_n$ specifies a set of erased packet transmissions over the link; the packets transmitted at time steps $t \in E$ are erased, while those transmitted at time steps $t \in T_n \backslash E$ are received. An *erasure model* essentially describes a distribution of erasure patterns.

For a given pair of positive integers a and b, we define the *offset quotient* $q_{a,b}$ and *offset remainder* $r_{a,b}$ to be the unique integers satisfying the following three conditions:

$$a = q_{a,b}\, b + r_{a,b}, \qquad q_{a,b} \in \mathbb{Z}_0^+, \qquad r_{a,b} \in \{1, \ldots, b\},$$

where $\mathbb{Z}_0^+$ denotes the set of nonnegative integers, i.e., $\mathbb{Z}^+ \cup \{0\}$. Note that this definition departs from the usual definition of quotient and remainder in that $r_{a,b}$ can be equal to b but not zero.

4.3 Code Constructions

We analyze the performance of two types of time-invariant codes in this chapter: *symmetric intrasession codes* for the window-based, bursty, and i.i.d. erasure models, and *diagonally interleaved codes* for the bursty erasure model. For ease of reference, we present their constructions and some general properties here.

The usual definition of a time-invariant code applies in the case of $c = 1$, where every packet is generated by applying a common encoding function to some recent interval of messages. For larger values of c, the notion of time-invariance can be generalized as follows:

Definition 4.1 (Time-Invariant Code)**.** A code is time-invariant if there exist causal and deterministic encoding functions $f_1, \ldots, f_c$ and a finite encoder memory size $m_{\mathsf{E}} \in \mathbb{Z}^+$ such that the packet transmitted at each time step $(k-1)c + i$, where $k \in \mathbb{Z}^+$, $i \in \{1, \ldots, c\}$, is given by the function f_i applied to the m_{E} most recent messages, i.e.,

$$X_{(k-1)c+i} = f_i\big(\underbrace{M_k, M_{k-1}, \ldots, M_{k-m_{\mathsf{E}}+1}}_{m_{\mathsf{E}} \text{ most recent messages}}\big).$$

4.3.1 Symmetric Intrasession Codes

In an *intrasession* code, coding is allowed within the same message but not across different messages. To describe such a code, we first specify how the link bandwidth or data packet space at each time step is allocated among the different messages. Each unit-size packet is essentially divided into multiple subpackets or blocks of possibly different sizes, each encoding a different message. We assume that an appropriate code (e.g., a maximum distance separable (MDS) code or a random linear code) is subsequently applied to this allocation so that each message is decodable whenever the total amount of received data that encodes that message, or the total size of the corresponding blocks, is at least the message size s.

The blocks that encode a given message k are confined to the packets transmitted in the corresponding coding window W_k; they cannot be created before the message creation time, and are

useless after the message decoding deadline. Thus, to decode each message, the decoder needs to access only the packets received at the most recent d time steps. The decoder memory requirements for intrasession codes are therefore modest compared to an intersession code requiring older packets or previous messages for decoding.

In a *time-invariant* intrasession code, the encoding functions $f_1, \ldots, f_c$ determine the sizes of the blocks that encode the m_{E} most recent messages in each interval of c packets or time steps. For each $i \in \{1, \ldots, m_{\mathsf{E}}c\}$, let $x_i \geq 0$ be the size of the block that encodes message $k - q_{i,c}$ at time step $(k-1)c + r_{i,c}$. Therefore, the size of the block that encodes message k at time step $(k-1)c + i$ is x_i if $i \in \{1, \ldots, m_{\mathsf{E}}c\}$, and zero otherwise. Because of the unit packet size constraint, we require that the sum of block sizes at each of the c time steps is at most one, i.e.,

$$\sum_{\substack{i \in \{1,\ldots,m_{\mathsf{E}}c\}: \\ r_{i,c} = j}} x_i \leq 1 \quad \forall\, j \in \{1, \ldots, c\}.$$

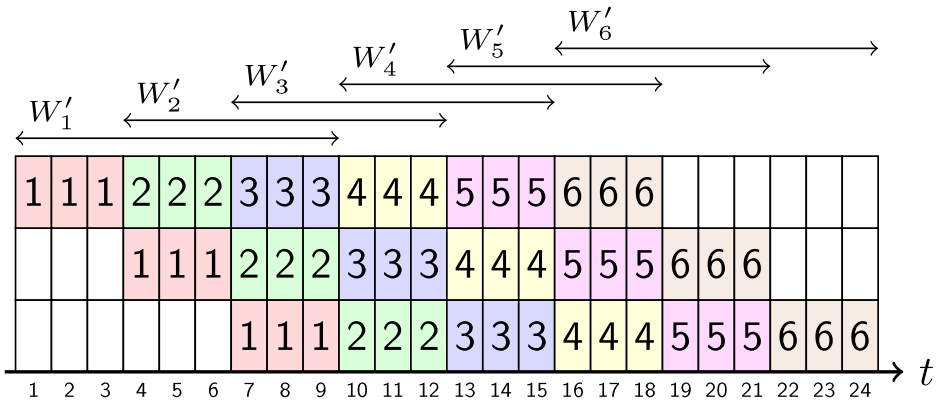
Motivated by the symmetric allocation strategy of Section 2.2, we introduce the family of *symmetric intrasession codes*, which are time-invariant intrasession codes with a symmetric allocation of packet space. For each symmetric code, we define a *spreading parameter* $m \in \{c, \ldots, d'\}$, where $d' \triangleq \min(d, m_{\mathsf{E}}c)$. (We would expect that $d' = d$ for most real-time streaming systems because the decoding deadline constraint is typically stricter than the encoder memory size limit, i.e., $d \leq m_{\mathsf{E}}c$.) Let $W'_k \subseteq W_k$ be the *effective coding window* for message k, which we define as the interval of m time steps beginning at its creation time, i.e.,

$$W'_k \triangleq \{(k-1)c + 1, \ldots, (k-1)c + m\}.$$

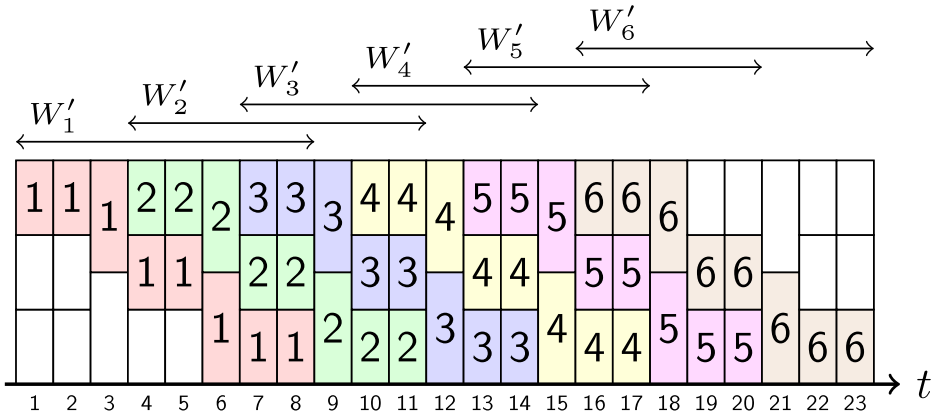
Let A_t be the set of *active messages* at time step t, which we define as the messages whose effective coding windows contain time step t, i.e.,

$$A_t \triangleq \{k \in \mathbb{Z} : t \in W'_k\}.$$

(Note that nonpositive messages are included as dummy messages.) For each symmetric code, the unit packet space at each time step is divided evenly among the active messages at that time



(a) $(c, m) = (3, 9)$



(b) $(c, m) = (3, 8)$

Figure 4.2. Allocation of the unit packet space at each time step t among messages $\{1, \ldots, 6\}$ in the symmetric intrasession code with spreading parameter m, for (a) $(c, m) = (3, 9)$ and (b) $(c, m) = (3, 8)$. Each message is assigned a unique color. In (a), because m is a multiple of c, we have $q_{m,c} + 1 = 3$ active messages at each time step. In (b), because m is not a multiple of c, we have either $q_{m,c} = 2$ or $q_{m,c} + 1 = 3$ active messages at each time step.

step. Thus, the number of blocks allocated to each message $k \in \mathbb{Z}^+$ is given by the spreading parameter m, and the size of the block that encodes each active message $k \in A_t$ at each time step $t \in \mathbb{Z}^+$ is given by $\frac{1}{|A_t|}$. Figure 4.2 illustrates this allocation of the unit packet space at each time step, for two instances of (c, m).

4.3.1.1 Active Messages at Each Time Step

For a given choice of (c, m), the set of active messages at each time step $t \in \mathbb{Z}^+$ can be stated explicitly as follows:

$$A_t = \{k \in \mathbb{Z} : t \in W_k'\}$$

$$
\begin{aligned}
&= \{k \in \mathbb{Z} : (k-1)c + 1 \le t \le (k-1)c + m\} \\
&= \left\{ k \in \mathbb{Z} : \frac{t-m}{c} + 1 \le k \le \frac{t-1}{c} + 1 \right\} \\
&= \left\{ \left\lceil \frac{t-m}{c} + 1 \right\rceil, \ldots, \left\lfloor \frac{t-1}{c} + 1 \right\rfloor \right\}.
\end{aligned}
$$

Expressing this in terms of $q_{m,c}$, $r_{m,c}$, $q_{t,c}$, $r_{t,c}$ yields

$$
A_t = \left\{ q_{t,c} + 1 - q_{m,c} + \left\lceil \frac{r_{t,c} - r_{m,c}}{c} \right\rceil, \ldots, q_{t,c} + 1 \right\}.
$$

It follows that the *number* of active messages $|A_t|$ varies over time depending on the value of $r_{t,c}$; specifically, two cases are possible:

Case 1: If $r_{t,c} \le r_{m,c}$, then

$$
-1 < \frac{1-c}{c} \le \frac{r_{t,c} - r_{m,c}}{c} \le 0,
$$

which implies that $\left\lceil \frac{r_{t,c} - r_{m,c}}{c} \right\rceil = 0$, and

$$
A_t = \{q_{t,c} + 1 - q_{m,c}, \ldots, q_{t,c} + 1\}.
$$

Therefore, there are $q_{m,c} + 1$ active messages at time step t, each of which is allocated a block of size $\frac{1}{q_{m,c}+1}$.

Case 2: If $r_{t,c} > r_{m,c}$, then

$$
0 < \frac{r_{t,c} - r_{m,c}}{c} \le \frac{c-1}{c} < 1,
$$

which implies that $\left\lceil \frac{r_{t,c} - r_{m,c}}{c} \right\rceil = 1$, and

$$
A_t = \{q_{t,c} + 1 - (q_{m,c} - 1), \ldots, q_{t,c} + 1\}.
$$

Therefore, there are $q_{m,c}$ active messages at time step t, each of which is allocated a block of size $\frac{1}{q_{m,c}}$.

Note that when m is a multiple of c, we have $r_{t,c} \le r_{m,c} = c$ for any t, which implies that there

are $q_{m,c}+1$ active messages at every time step, and all blocks are of size $\frac{1}{q_{m,c}+1} = \frac{c}{m}$.

4.3.1.2 Block Sizes for Each Message

As a consequence of the number of active messages, message k is allocated either a small block of size $\frac{1}{q_{m,c}+1}$ or a big block of size $\frac{1}{q_{m,c}}$ at each time step $t \in W'_k$; no blocks are allocated to message k at all other time steps $t \notin W'_k$. Writing each time step $t \in W'_k$ as

$$t = (k-1)c + i = \underbrace{(k-1+q_{i,c})}_{q_{t,c}}\, c + \underbrace{r_{i,c}}_{r_{t,c}},$$

where $i \in \{1, \ldots, m\}$, we observe that the size of the block that encodes message k at time step $(k-1)c+i$, which has been defined as x_i, depends on the value of $r_{i,c}$; specifically, two cases are possible:

Case 1: If $r_{i,c} \leq r_{m,c}$, then $x_i = \frac{1}{q_{m,c}+1}$. Since $i \in \{1, \ldots, m\}$, this condition corresponds to the case where $q_{i,c} \in \{0, \ldots, q_{m,c}\}$ and $r_{i,c} \in \{1, \ldots, r_{m,c}\}$. Therefore, message k is allocated a small block of size $\frac{1}{q_{m,c}+1}$ per time step for a total of $(q_{m,c}+1)r_{m,c}$ time steps in the effective coding window W'_k.

Case 2: If $r_{i,c} > r_{m,c}$, then $x_i = \frac{1}{q_{m,c}}$. Since $i \in \{1, \ldots, m\}$, this condition corresponds to the case where $q_{i,c} \in \{0, \ldots, q_{m,c}-1\}$ and $r_{i,c} \in \{r_{m,c}+1, \ldots, c\}$. Therefore, message k is allocated a big block of size $\frac{1}{q_{m,c}}$ per time step for a total of $q_{m,c}(c - r_{m,c})$ time steps in the effective coding window W'_k.

4.3.1.3 Achievability

We use the following lemma to determine the message sizes achievable by the symmetric code with spreading parameter $m = d$ (for which $W'_k = W_k$), for the window-based and bursty erasure models:

Lemma 4.2 (Achievability)**.** *Consider the symmetric intrasession code (Section 4.3.1) with spreading parameter $m = d$ for a given choice of (c, d). If message size s satisfies the inequality*

$$s \leq \sum_{j=1}^{\ell} y_j,$$

where $\mathbf{y} = (y_1, \ldots, y_d)$ *is defined as*

$$\mathbf{y} \triangleq \Bigg(\overbrace{\underbrace{\frac{1}{q_{d,c}+1}, \ldots, \frac{1}{q_{d,c}+1}}_{(q_{d,c}+1)r_{d,c} \text{ entries}}, \underbrace{\frac{1}{q_{d,c}}, \ldots, \frac{1}{q_{d,c}}}_{q_{d,c}(c-r_{d,c}) \text{ entries}}}^{d \text{ entries}} \Bigg),$$

then each message $k \in \mathbb{Z}^+$ *is decodable from any* ℓ *packets transmitted in its coding window* W_k.

Note that the maximum message size s that can be supported by this code is given by $\sum_{j=1}^{d} y_j = c$, which corresponds to the choice of $\ell = d$.

4.3.1.4 Partitioning of Coding Windows

We use the following lemma to select worst-case erasure patterns with cut-set bounds that match the message sizes achievable by the symmetric code with spreading parameter $m = d$, for the window-based and bursty erasure models:

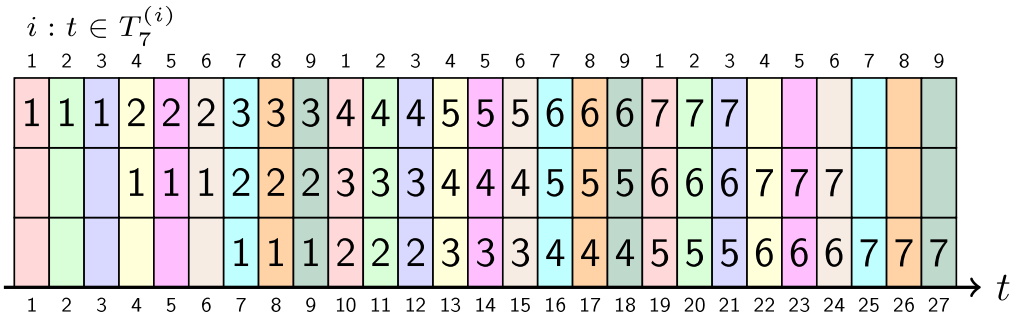
Lemma 4.3 (Partitioning of Coding Windows)**.** *Consider the symmetric intrasession code (Section 4.3.1) with spreading parameter* $m = d$ *for a given choice of* (c, d)*. Consider the first* n *messages* $\{1, \ldots, n\}$*, and the union of their (overlapping) coding windows* T_n*. The set of time steps* T_n *can be partitioned into* d *sets* $T_n^{(1)}, \ldots, T_n^{(d)}$*, given by*

$$T_n^{(i)} \triangleq \begin{cases} \Big\{ \big(j(q_{d,c}+1) + q_{i,c}\big)c + r_{i,c} \in T_n : j \in \mathbb{Z}_0^+ \Big\} & \text{if } r_{i,c} \leq r_{d,c}, \\ \Big\{ \big(j\, q_{d,c} + q_{i,c}\big)c + r_{i,c} \in T_n : j \in \mathbb{Z}_0^+ \Big\} & \text{if } r_{i,c} > r_{d,c}, \end{cases}$$

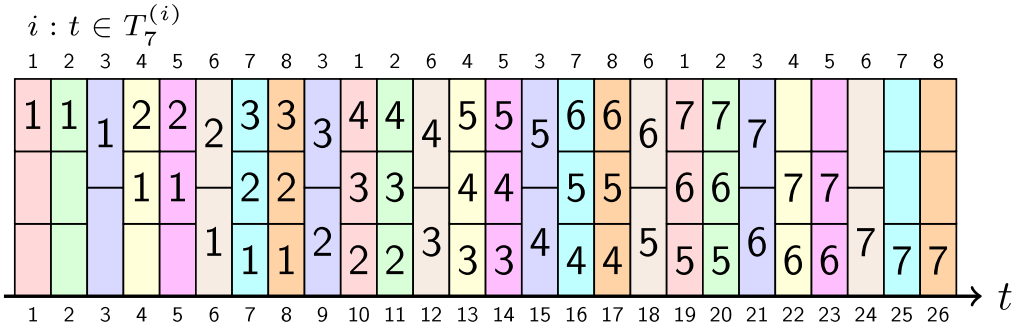
with the following properties:

P1) Over the packets transmitted at time steps $T_n^{(i)}$*, each message* $k \in \{1, \ldots, n\}$ *is allocated exactly one block; this block is contained within the coding window* W_k*, and has a size of* $\frac{1}{q_{d,c}+1}$ *if* $r_{i,c} \leq r_{d,c}$*, and* $\frac{1}{q_{d,c}}$ *if* $r_{i,c} > r_{d,c}$.

P2) The total size of the packets transmitted at time steps $T_n^{(i)}$*, i.e.,* $\big|T_n^{(i)}\big|$*, has the following upper*



(a) $(c, d) = (3, 9)$



(b) $(c, d) = (3, 8)$

Figure 4.3. Partitioning of the set of time steps T_n into the d sets $T_n^{(1)}, \ldots, T_n^{(d)}$, and the allocation of the unit packet space at each time step t among messages $\{1, \ldots, 7\}$, in the symmetric intrasession code with spreading parameter $m = d$, for (a) $(c, d) = (3, 9)$ and (b) $(c, d) = (3, 8)$. Each set $T_n^{(i)}$ is assigned a unique color. The number i at the top of each time step t indicates the set $T_n^{(i)}$ to which t belongs.

bound:

$$\left|T_n^{(i)}\right| < \begin{cases} \dfrac{n}{q_{d,c}+1} + 2 & \text{if } r_{i,c} \leq r_{d,c}, \\ \dfrac{n}{q_{d,c}} + 2 & \text{if } r_{i,c} > r_{d,c}. \end{cases}$$

Figure 4.3 shows how the set of time steps T_n is partitioned into the d sets $T_n^{(1)}, \ldots, T_n^{(d)}$, for two instances of (c, d).

4.3.2 Diagonally Interleaved Codes

Consider a systematic block code $\mathcal{C}$ that encodes a given vector of $d - \alpha$ information symbols $\mathbf{a} = (a[1], \ldots, a[d-\alpha])$ as a codeword vector of d symbols $(a[1], \ldots, a[d-\alpha], b[1], \ldots, b[\alpha])$, where each symbol has a normalized size of $\frac{1}{d}$. For each $i \in \{1, \ldots, \alpha\}$, we define an encoding

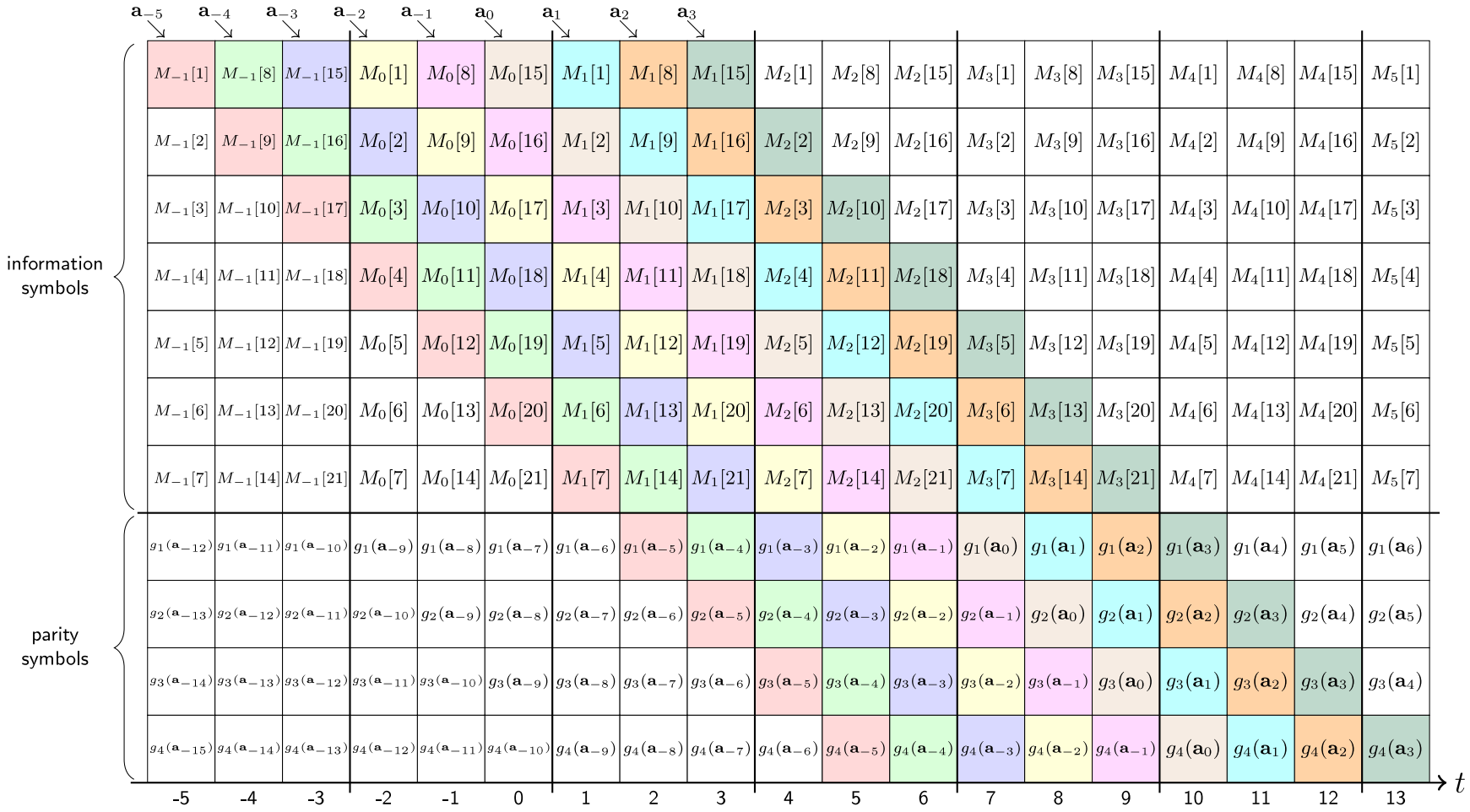


Figure 4.4. Construction of the diagonally interleaved code, for $(c, d, \alpha) = (3, 11, 4)$. Rows $\{1, \ldots, d - \alpha\}$ of the table are populated by information symbols, while rows $\{d - \alpha + 1, \ldots, d\}$ are populated by parity symbols. The d symbols on each diagonal spanning across d consecutive time steps constitute one codeword produced by the component systematic block code $\mathcal{C}$.

function g_i so that the parity symbol $b[i]$ is given by $b[i] = g_i(\mathbf{a})$.

For a given choice of (c, d, α), we can derive a time-invariant *diagonally interleaved code* for a message size of $s = \frac{d-\alpha}{d}c$ by interleaving codeword symbols produced by the component systematic block code $\mathcal{C}$ in a diagonal pattern.

First, to facilitate code construction, we represent the derived code by a table of symbols, with each cell in the table assigned one symbol of size $\frac{1}{d}$. Figure 4.4 illustrates our construction for an instance of (c, d, α). Let $x_t[i]$ denote the symbol in column $t \in \mathbb{Z}$ and row $i \in \{1, \ldots, d\}$. The unit-size packet transmitted at each time step t is composed of the d symbols $x_t[1], \ldots, x_t[d]$ in column t of the table. Rows $\{1, \ldots, d-\alpha\}$ of the table are populated by information symbols, while rows $\{d-\alpha+1, \ldots, d\}$ are populated by parity symbols.

Next, we divide each message k into $(d-\alpha)c$ submessages or information symbols denoted by $M_k[1], \ldots, M_k[(d-\alpha)c]$, with each symbol having a size of $\frac{s}{(d-\alpha)c} = \frac{1}{d}$. The information symbols corresponding to each message k are assigned evenly to the columns representing the first c time steps in coding window W_k, so that

$$x_t[i] = M_{q_{t,c}+1}[(r_{t,c}-1)(d-\alpha)+i]$$

for each $i \in \{1, \ldots, d-\alpha\}$. To obtain the parity symbols for column t, we apply the component systematic block code $\mathcal{C}$ to the information symbols on each diagonal, so that

$$x_t[d-\alpha+i] = g_i\left(\left(x_{t-i-(d-\alpha)+\ell}[\ell]\right)_{\ell=1}^{d-\alpha}\right)$$

for each $i \in \{1, \ldots, \alpha\}$. Thus, the d symbols on each diagonal spanning across d consecutive time steps in the derived code constitute one codeword produced by $\mathcal{C}$. Note that the information symbols for nonexistent messages (i.e., nonpositive messages and messages after the actual final message) are assumed to be zeros so that all codeword symbols are well defined.

4.4 Window-Based Erasure Model

For the first erasure model, all erasure patterns containing a limited number of erasures in each specifically defined window are admissible. We consider two variations of this model; one based on

the coding window W_k in Section 4.4.1, and the other on a sliding window of h time steps, where $h \geq d$, in Section 4.4.2.

4.4.1 Coding Window Erasure Model

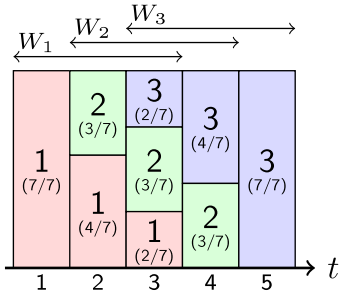
Consider the first n messages $\{1, \ldots, n\}$, and the union of their (overlapping) coding windows T_n. Let $\mathcal{E}_n^{\mathsf{CW}}$ be the set of erasure patterns that have at most z erased time steps in each coding window W_k, i.e.,

$$\mathcal{E}_n^{\mathsf{CW}} \triangleq \big\{E \subseteq T_n : |E \cap W_k| \leq z \; \forall \; k \in \{1, \ldots, n\}\big\}.$$

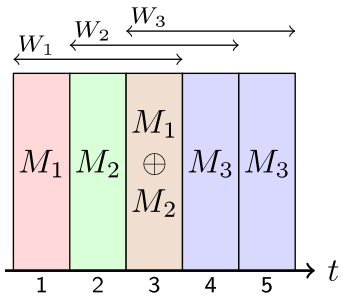
The objective is to construct a code that allows all n messages $\{1, \ldots, n\}$ to be decoded by their respective decoding deadlines under any erasure pattern $E \in \mathcal{E}_n^{\mathsf{CW}}$. Let s_n^{CW} be the maximum message size that can be achieved by such a code, for a given choice of (n, c, d, z).

The following example demonstrates that over a finite time horizon (i.e., when the number of messages n is finite), intrasession coding can be strictly suboptimal:

Example 4.4 (Finite Time Horizon)**.** Suppose that $(n, c, d, z) = (3, 1, 3, 1)$. The maximum message size that can be achieved by an intrasession code is $s = \frac{6}{7}$; one such optimal intrasession code, which can be found by solving a linear program, is as follows (the size of each block is indicated in parentheses):



The following intersession code achieves a strictly larger message size of $s = 1$ (M_k denotes message k):



Using a simple cut-set bound argument, we can show that this is also the maximum achievable message size, i.e., $s_n^{\mathsf{CW}} = 1$.

However, it turns out that the symmetric intrasession code (Section 4.3.1) with spreading parameter $m = d$ is *asymptotically* optimal over all codes; the gap between the maximum achievable message size s_n^{CW} and the message size achieved by this code vanishes as the number of messages n goes to infinity:

Theorem 4.5. *Consider the coding window erasure model for a given choice of* (c, d, z)*. The symmetric intrasession code (Section 4.3.1) with spreading parameter* $m = d$ *is asymptotically optimal over all codes in the following sense: it achieves a message size of*

$$\sum_{j=1}^{d-z} y_j,$$

which is equal to the asymptotic maximum achievable message size $\lim_{n\to\infty} s_n^{\mathsf{CW}}$*, where* $\mathbf{y} = (y_1, \ldots, y_d)$ *is as defined in Lemma 4.2.*

The achievability claim of this theorem is a consequence of Lemma 4.2; to prove the converse claim, we consider a cut-set bound corresponding to a specific *worst-case* erasure pattern in which exactly z erasures occur in every coding window. This erasure pattern is chosen with the help of Lemma 4.3; specifically, the erased time steps are chosen to coincide with the bigger blocks allocated to each message in the symmetric code.

4.4.2 Sliding Window Erasure Model

Consider the first n messages $\{1, \ldots, n\}$, and the union of their (overlapping) coding windows T_n. Let sliding window L_t denote the interval of h time steps beginning at time step t, where $h \geq d$,

i.e.,

$$L_t \triangleq \{t, \ldots, t+h-1\}.$$

Let $\mathcal{E}_n^{\mathsf{SW}}$ be the set of erasure patterns that have at most z erased time steps in each sliding window L_t, i.e.,

$$\mathcal{E}_n^{\mathsf{SW}} \triangleq \big\{E \subseteq T_n : |E \cap L_t| \leq z \; \forall \; t \in \{1, \ldots, (n-1)c + d - h + 1\}\big\}.$$

The objective is to construct a code that allows all n messages $\{1, \ldots, n\}$ to be decoded by their respective decoding deadlines under any erasure pattern $E \in \mathcal{E}_n^{\mathsf{SW}}$. Let s_n^{SW} be the maximum message size that can be achieved by such a code, for a given choice of (n, c, d, h, z).

We note that if $E \in \mathcal{E}_n^{\mathsf{SW}}$, then $E \in \mathcal{E}_n^{\mathsf{CW}}$; therefore, $\mathcal{E}_n^{\mathsf{SW}} \subseteq \mathcal{E}_n^{\mathsf{CW}}$, which implies that $s_n^{\mathsf{SW}} \geq s_n^{\mathsf{CW}}$. For the special case of $(c, h) = (1, d)$, each sliding window is also a coding window, and so this sliding window erasure model reduces to the coding window erasure model of Section 4.4.1, i.e., $\mathcal{E}_n^{\mathsf{SW}} = \mathcal{E}_n^{\mathsf{CW}}$. Over a finite time horizon, intrasession coding can also be suboptimal for this erasure model; the illustrating example from Section 4.4.1 applies here as well.

Surprisingly, the symmetric code with spreading parameter $m = d$ also turns out to be asymptotically optimal over all codes here; the omission of erasure patterns in $\mathcal{E}_n^{\mathsf{SW}}$ compared to $\mathcal{E}_n^{\mathsf{CW}}$ has not led to an increase in the maximum achievable message size (cf. Theorem 4.5):

Theorem 4.6. *Consider the sliding window erasure model for a given choice of (c, d, h, z). The symmetric intrasession code (Section 4.3.1) with spreading parameter $m = d$ is asymptotically optimal over all codes in the following sense: it achieves a message size of*

$$\sum_{j=1}^{d-z} y_j,$$

which is equal to the asymptotic maximum achievable message size $\lim_{n\to\infty} s_n^{\mathsf{SW}}$.

Proving the converse claim of this theorem requires a different approach from that of Theorem 4.5. It may not be possible to find a single admissible erasure pattern that provides a cut-set bound matching the symmetric code; instead, we need to combine different erasure patterns for different messages. To pick these erasure patterns, we first choose a specific *base erasure pattern* E' (which may not be admissible in general) with the help of Lemma 4.3. We then derive admissible

erasure patterns from E' by taking its intersection with each coding window, i.e., $(E' \cap W_k) \in \mathcal{E}_n^{\mathsf{SW}}$. These derived erasure patterns are used in the inductive computation of an upper bound for the conditional entropy

$$H\Big(X[W_n \backslash E'] \,\Big|\, M_1^n, X_1^{(n-1)c}\Big).$$

Intuitively, this conditional entropy term expresses how much space is left in the unerased data packets of the coding window for message n, after encoding the first n messages, and conditioned on the previous time steps. The nonnegativity of the conditional entropy leads us to a bound for s_n^{SW} that matches the message size achieved by the symmetric code in the limit $n \to \infty$.

4.5 Bursty Erasure Model

For the second erasure model, all erasure patterns containing erasure bursts of a limited length are admissible. Consider the first n messages $\{1, \ldots, n\}$, and the union of their (overlapping) coding windows T_n. Let $\mathcal{E}_n^{\mathsf{B}}$ be the set of erasure patterns in which each erasure burst is an interval of at most z erased time steps, and consecutive erasure bursts are separated by a guard interval or gap of at least $d - z$ unerased time steps, i.e.,

$$\begin{aligned}\mathcal{E}_n^{\mathsf{B}} \triangleq \Big\{ E \subseteq T_n :\ & (t \notin E \ \wedge\ t{+}1 \in E) \Rightarrow |E \cap \{t{+}1, \ldots, t{+}z{+}1\}| \leq z, \\ & (t \in E \ \wedge\ t{+}1 \notin E) \Rightarrow |E \cap \{t{+}1, \ldots, t{+}d{-}z\}| = 0 \Big\}.\end{aligned}$$

The objective is to construct a code that allows all n messages $\{1, \ldots, n\}$ to be decoded by their respective decoding deadlines under any erasure pattern $E \in \mathcal{E}_n^{\mathsf{B}}$. Let s_n^{B} be the maximum message size that can be achieved by such a code, for a given choice of (n, c, d, z).

This model can be seen as an instance of a more general class of bursty erasure models where the maximum erasure burst length and the minimum guard interval length can be arbitrarily specified. In a similar bursty erasure model considered by Martinian *et al.* [52, 53] and Badr *et al.* [54], the maximum erasure burst length (given by B) is z, while the minimum guard interval length (given by T) is $d - 1$. For the same choice of (d, z), our model captures a larger set of erasure patterns and is therefore stricter (the respective cut-set bounds reflect this comparison).

In Section 4.5.1, we show that the symmetric intrasession code with spreading parameter $m = d$,

which is asymptotically optimal for the window-based erasure model of Section 4.4, is also asymptotically optimal here in a number of cases. In Section 4.5.2, we show that diagonally interleaved codes derived from specific systematic block codes are asymptotically optimal in several other cases.

4.5.1 Optimality of Symmetric Intrasession Codes

Using the proof technique of Theorem 4.6, we can show that the symmetric intrasession code (Section 4.3.1) with spreading parameter $m = d$ is also asymptotically optimal over all codes here when d is a multiple of c, or when the maximum erasure burst length z is sufficiently short or long:

Theorem 4.7. *Consider the bursty erasure model for a given choice of* (c, d, z) *satisfying* any *of the following three conditions:*

1) d *is a multiple of* c*;*

2) d *is not a multiple of* c*, and* $z \leq c - r_{d,c}$*; or*

3) d *is not a multiple of* c*, and* $z \geq d - r_{d,c}$*.*

The symmetric intrasession code (Section 4.3.1) with spreading parameter $m = d$ *is asymptotically optimal over all codes in the following sense: it achieves a message size of*

$$\sum_{j=1}^{d-z} y_j,$$

which is equal to the asymptotic maximum achievable message size $\lim_{n\to\infty} s_n^{\mathsf{B}}$.

When the maximum erasure burst length z takes on intermediate values, intersession coding may become necessary. Asymptotically optimal codes for a variety of these cases are presented in Section 4.5.2.

4.5.2 Optimality of Diagonally Interleaved Codes

Diagonally interleaved codes (Section 4.3.2) that are derived from systematic block codes $\mathcal{C}$ with certain properties turn out to be asymptotically optimal in several cases. These sufficient code properties are given by the following lemma:

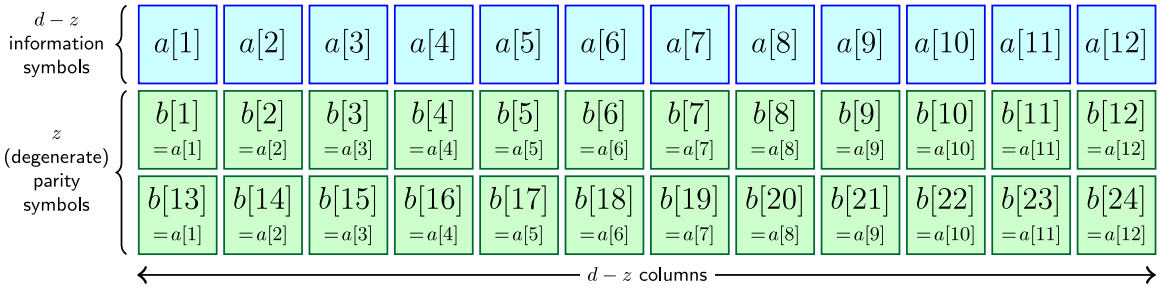
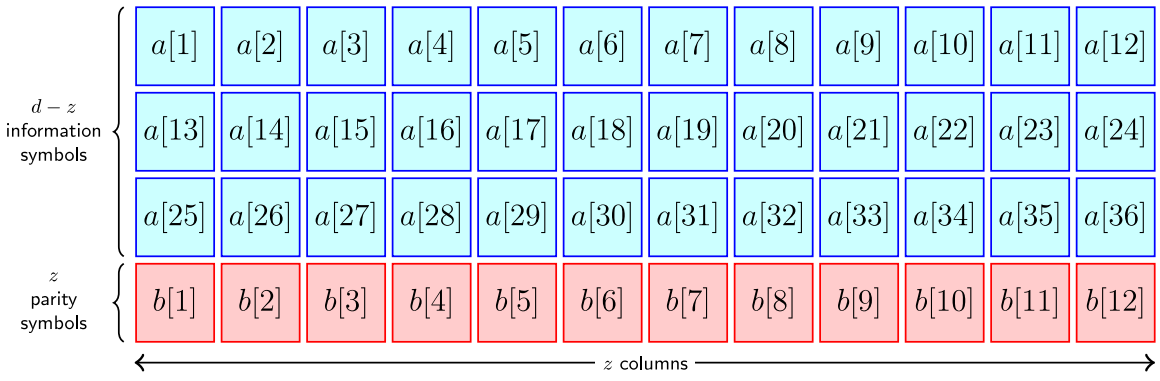
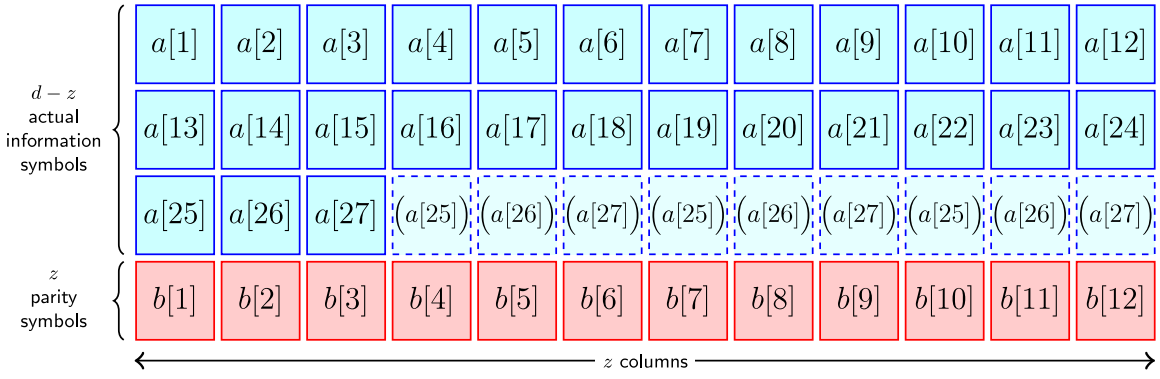


Figure 4.5. The d symbols of the codeword vector produced by the systematic block code $\mathcal{C}$ of Theorem 4.9, for $(c, d, z) = (5, 36, 24)$. For each $i \in \{1, \ldots, d - z\}$, all the (degenerate) parity symbols below the information symbol $a[i]$ in column i of the table have a value of $a[i]$.



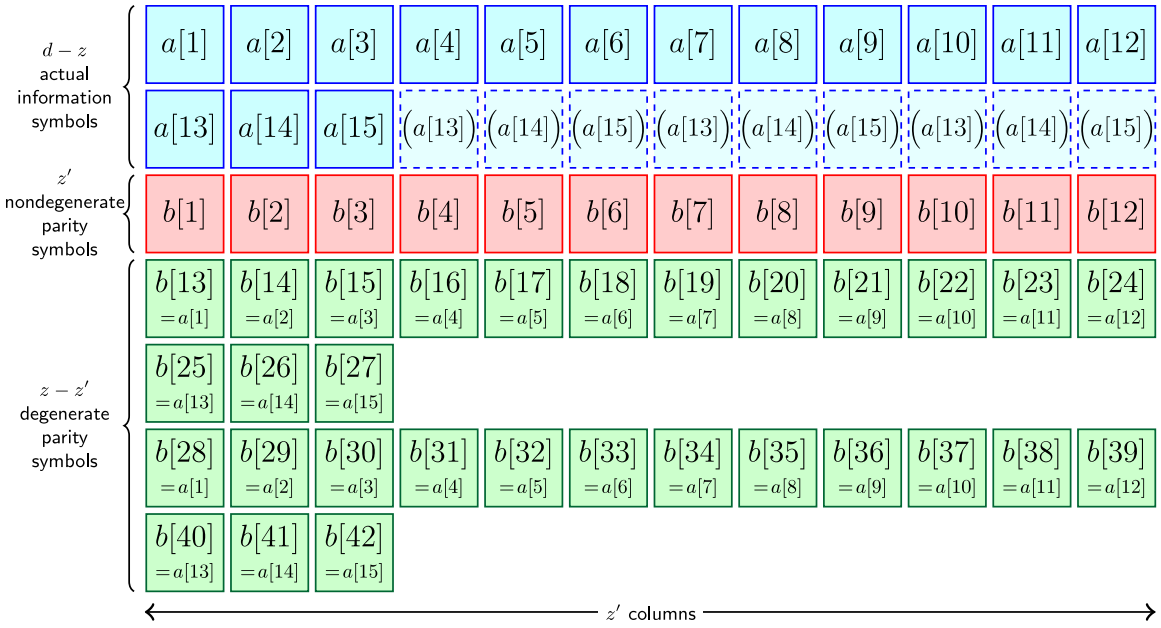
(a) $(c, d, z) = (5, 48, 12)$



(b) $(c, d, z) = (5, 39, 12)$

Figure 4.6. The d symbols of the codeword vector produced by the systematic block code $\mathcal{C}$ of Theorem 4.10, for (a) $(c, d, z) = (5, 48, 12)$ and (b) $(c, d, z) = (5, 39, 12)$. In (a), because $r' = z$, there are no virtual information symbols. In (b), because $r' < z$, we have virtual information symbols on the second last row (in parentheses). For each $i \in \{1, \ldots, z\}$, the value of the parity symbol $b[i]$ is given by the bit-wise modulo-2 sum (i.e., exclusive-or) of the actual and virtual information symbols above it in column i of the table.

(a) $(c, d, z) = (5, 84, 60)$



(b) $(c, d, z) = (5, 57, 42)$

Figure 4.7. The d symbols of the codeword vector produced by the systematic block code $\mathcal{C}$ of Theorem 4.12, for (a) $(c, d, z) = (5, 84, 60)$ and (b) $(c, d, z) = (5, 57, 42)$. In (a), because $r' = z'$, there are no virtual information symbols. In (b), because $r' < z'$, we have virtual information symbols on the $\left(\frac{d-z-r'}{z'} + 1\right)$th row (in parentheses). For each $i \in \{1, \ldots, z'\}$, the value of the nondegenerate parity symbol $b[i]$ is given by the bit-wise modulo-2 sum (i.e., exclusive-or) of the actual and virtual information symbols above it in column i of the table.

Lemma 4.8. *Consider the diagonally interleaved code (Section 4.3.2) for a given choice of* $(c, d, \alpha{=}z)$ *satisfying* $c \leq z \leq d - c$. *Suppose that the* d *symbols of the codeword vector* $(a[1], \ldots, a[d{-}z], b[1], \ldots, b[z])$ *produced by the component systematic block code* $\mathcal{C}$ *are transmitted sequentially across an erasure link, one symbol per time step, over the time interval* $L \triangleq \{1, \ldots, d\}$. *For each* $j \in \{1, \ldots, d\}$, *let* $E_j^{\mathsf{z}} \subseteq L$ *be the erasure pattern that contains a single wrap-around erasure burst of exactly* z *erased time steps (which may wrap around the last and first time steps in the interval) with the* j*th time step in the interval as the "leading" erasure, i.e.,*

$$E_j^{\mathsf{z}} \triangleq \big\{ r_{j+\ell,d} : \ell \in \{0, \ldots, z-1\} \big\}.$$

Let $\mathcal{E}^{\mathsf{z}}$ *be the set of all such erasure patterns, i.e.,*

$$\mathcal{E}^{\mathsf{z}} \triangleq \big\{ E_1^{\mathsf{z}}, \ldots, E_d^{\mathsf{z}} \big\}.$$

If the systematic block code $\mathcal{C}$ *satisfies both of the following symbol decoding requirements, then the diagonally interleaved code derived from* $\mathcal{C}$ *achieves a message size of* $\frac{d-z}{d}c$ *for the bursty erasure model:*

- *D1) For each* $i \in \{1, \ldots, c\}$, *the information symbol* $a[i]$ *is decodable by the* $(d - c + i)$*th time step in interval* L *under any erasure pattern* $E^{\mathsf{z}} \in \mathcal{E}^{\mathsf{z}}$.
- *D2) The information symbols* $a[c{+}1], \ldots, a[d{-}z]$ *are decodable by the last time step in interval* L *under any erasure pattern* $E^{\mathsf{z}} \in \mathcal{E}^{\mathsf{z}}$.

The condition $c \leq z \leq d - c$ is actually implied by the symbol decoding requirements: the first information symbol $a[1]$ would otherwise be undecodable by its decoding deadline under erasure pattern E_1^{z} because by that time step, no parity symbols would have been transmitted if $c > z$, and no symbols would have been received if $z > d - c$. Note that the use of a systematic MDS code as the component systematic block code $\mathcal{C}$ may not be sufficient here because of the additional decoding deadlines imposed on individual symbols.

The following theorem shows that a degenerate diagonally interleaved code that uses only intrasession coding is asymptotically optimal over all codes for the specified parameter conditions:

Theorem 4.9. *Consider the bursty erasure model for a given choice of* (c, d, z) *satisfying* all *of the following three conditions:*

1) d *is not a multiple of* c*;*

2) $c \leq z \leq d - c$*; and*

3) d *is a multiple of* $d - z$*.*

Let $\mathcal{C}$ *be a systematic block code that encodes a given vector of* $d - z$ *information symbols* $\mathbf{a} = (a[1], \ldots, a[d-z])$ *as a codeword vector of* d *symbols* $(a[1], \ldots, a[d-z], b[1], \ldots, b[z])$*, where each symbol has a normalized size of* $\frac{1}{d}$*, and the parity symbol* $b[i]$ *is given by*

$$b[i] = g_i(\mathbf{a}) \triangleq a[r_{i,d-z}]$$

for each $i \in \{1, \ldots, z\}$*. The diagonally interleaved code (Section 4.3.2) derived from* $\mathcal{C}$ *is asymptotically optimal over all codes in the following sense: it achieves a message size of* $\frac{d-z}{d}c$*, which is equal to the asymptotic maximum achievable message size* $\lim_{n\to\infty} s_n^{\mathsf{B}}$*.*

The systematic block code $\mathcal{C}$ of Theorem 4.9 is illustrated in Figure 4.5 for an instance of (c, d, z). Note that all the parity symbols in $\mathcal{C}$ are degenerate in the sense that they are just uncoded copies of information symbols.

The following two theorems describe diagonally interleaved codes that are asymptotically optimal over all codes for the specified parameter conditions:

Theorem 4.10. *Consider the bursty erasure model for a given choice of* (c, d, z) *satisfying* all *of the following five conditions:*

1) d *is not a multiple of* c*;*

2) $c \leq z \leq d - c$*;*

3) d *is not a multiple of* $d - z$*;*

4) $z < d - z$*; and*

5) z *is a multiple of* r'*, where*

$$r' \triangleq r_{d-z,z} \in \{1, \ldots, z\}.$$

Let $\mathcal{C}$ be a systematic block code that encodes a given vector of $d-z$ information symbols $\mathbf{a} = (a[1], \ldots, a[d-z])$ as a codeword vector of d symbols $(a[1], \ldots, a[d-z], b[1], \ldots, b[z])$, where each symbol has a normalized size of $\frac{1}{d}$, and the parity symbol $b[i]$ is given by

$$b[i] = g_i(\mathbf{a}) \triangleq \left(\bigoplus_{k=1}^{\frac{d-z-r'}{z}} a[(k-1)z+i] \right) \oplus a[d-z-r'+r_{i,r'}]$$

for each $i \in \{1, \ldots, z\}$. The diagonally interleaved code (Section 4.3.2) derived from $\mathcal{C}$ is asymptotically optimal over all codes in the following sense: it achieves a message size of $\frac{d-z}{d}c$, which is equal to the asymptotic maximum achievable message size $\lim_{n\to\infty} s_n^{\mathsf{B}}$.

The systematic block code $\mathcal{C}$ of Theorem 4.10 is illustrated in Figure 4.6 for two instances of (c, d, z). The following example demonstrates that in this case, intrasession coding can be strictly suboptimal:

Example 4.11 (Suboptimality of Intrasession Coding)**.** Suppose that $(c, d, z) = (2, 5, 2)$. For $n = 9$, the maximum message size that can be achieved by an intrasession code has an upper bound of $s < 1.193$; such a bound can be found by solving a linear program for a subset of erasure patterns in $\mathcal{E}_n^{\mathsf{B}}$ (namely, those with alternating intervals of z erased time steps and $d - z$ unerased time steps). The same upper bound also holds for $n > 9$ because any message size that can be achieved for a larger number of messages can also be achieved for a smaller number of messages (we simply apply the same code and ignore the additional messages and packets). On the other hand, the diagonally interleaved code derived from the systematic block code $\mathcal{C}$ of Theorem 4.10 achieves a strictly larger message size of $s = \frac{6}{5} = 1.2$.

Theorem 4.12. *Consider the bursty erasure model for a given choice of (c, d, z) satisfying* all *of the following five conditions:*

1) d is not a multiple of c;

2) $c \leq z \leq d - c$;

3) d is not a multiple of $d - z$;

4) $z > d - z$; and

5) z' is a multiple of r', where

$$z' \triangleq r_{z,d-z} \in \{1, \ldots, d-z-1\},$$
$$r' \triangleq r_{d-z,z'} \in \{1, \ldots, z'\}.$$

Let $\mathcal{C}$ be a systematic block code that encodes a given vector of $d-z$ information symbols $\mathbf{a} = (a[1], \ldots, a[d-z])$ as a codeword vector of d symbols $(a[1], \ldots, a[d-z], b[1], \ldots, b[z])$, where each symbol has a normalized size of $\frac{1}{d}$, and the parity symbol $b[i]$ is given by

$$b[i] = g_i(\mathbf{a}) \triangleq \begin{cases} \left(\bigoplus_{k=1}^{\frac{d-z-r'}{z'}} a[(k-1)z'+i] \right) \oplus a[d-z-r'+r_{i,r'}] & \text{if } i \in \{1, \ldots, z'\}, \\ a[r_{i-z',d-z}] & \text{if } i \in \{z'+1, \ldots, z\} \end{cases}$$

for each $i \in \{1, \ldots, z\}$. The diagonally interleaved code (Section 4.3.2) derived from $\mathcal{C}$ is asymptotically optimal over all codes in the following sense: it achieves a message size of $\frac{d-z}{d}c$, which is equal to the asymptotic maximum achievable message size $\lim_{n\to\infty} s_n^{\mathsf{B}}$.

The systematic block code $\mathcal{C}$ of Theorem 4.12 is illustrated in Figure 4.7 for two instances of (c, d, z). Note that there are two types of parity symbols in $\mathcal{C}$: $b[1], \ldots, b[z']$ are nondegenerate parity symbols, while $b[z'+1], \ldots, b[z]$ are degenerate parity symbols which are just uncoded copies of information symbols. The following example demonstrates that in this case, intrasession coding can be strictly suboptimal:

Example 4.13 (Suboptimality of Intrasession Coding, cf. Example 4.11)**.** Suppose that $(c, d, z) = (3, 8, 5)$. For $n \geq 10$, the maximum message size that can be achieved by an intrasession code has an upper bound of $s < 1.118$. On the other hand, the diagonally interleaved code derived from the systematic block code $\mathcal{C}$ of Theorem 4.12 achieves a strictly larger message size of $s = \frac{9}{8} = 1.125$.

4.6 IID Erasure Model

For the third erasure model, each packet transmitted over the link is erased independently with the same probability p_{e}. For brevity, let S_k denote the success event "message k is decodable by its

decoding deadline, i.e., time step $(k-1)c+d$", and let $\overline{S_k}$ denote the complementary failure event. We restrict our attention to time-invariant codes here in the interest of practicality.

Consider the i.i.d. erasure model for a given choice of $(c, d, p_{\mathsf{e}}, s)$. We shall adopt the *decoding probability* $\mathbb{P}[S_k]$, i.e., the probability that a given message k is decodable by its decoding deadline, as the primary performance metric. The decoder memory size is assumed to be unbounded so that the decoder has access to all received packets. Let the random subset $U_k \subseteq T_k$ be the unerased time steps that are no later than the decoding deadline for message k; the received packets that can be used by the decoder for decoding message k are therefore given by $X[U_k]$. Consequently, the decoding probability $\mathbb{P}[S_k]$, where $k \in \mathbb{Z}^+$, can be expressed in terms of U_k as follows:

$$\begin{aligned}\mathbb{P}[S_k] &= \mathbb{P}\left[H\big(M_k \,\big|\, X[U_k]\big) = 0\right] \\ &= \sum_{U_k \subseteq T_k} \mathbb{1}\left[H\big(M_k \,\big|\, X[U_k]\big) = 0\right] \cdot (1-p_{\mathsf{e}})^{|U_k|}(p_{\mathsf{e}})^{|T_k|-|U_k|}. \end{aligned} \tag{4.1}$$

By combining the proof techniques of Lemma 4.16 and Lemma 2.3, we can derive an upper bound on the decoding probability $\mathbb{P}[S_k]$ for any time-invariant code:

Theorem 4.14. *Consider the i.i.d. erasure model for a given choice of $(c, d, p_{\mathsf{e}}, s)$. For any time-invariant code with encoder memory size m_{E}, the probability that a given message $k \geq m_{\mathsf{E}}$ is decodable by its decoding deadline is upper-bounded as follows:*

$$\mathbb{P}[S_k] \leq \sum_{z=0}^{d} \left\lfloor \min\left(\frac{(d-z)c}{d\,s}, 1\right) \binom{d}{z} \right\rfloor (1-p_{\mathsf{e}})^{d-z}(p_{\mathsf{e}})^{z}. \tag{4.2}$$

Note that the decoding probability $\mathbb{P}[S_k]$ for the early messages $k < m_{\mathsf{E}}$ can potentially be higher than that for the subsequent messages $k \geq m_{\mathsf{E}}$ because the decoder already knows the non-positive messages (which are assumed to be zeros).

For real-time streaming applications that are sensitive to bursts of decoding failures, it may be useful to adopt the *burstiness of undecodable messages* as a secondary performance metric. One way of measuring this burstiness is to compute the conditional probability $\mathbb{P}\left[\,\overline{S_{k+1}} \,\middle|\, \overline{S_k}\,\right]$, i.e., the conditional probability that the next message is undecodable by its decoding deadline given that the current message is undecodable by its decoding deadline. The higher this conditional probability is, the more likely it is to remain "stuck" in a burst of undecodable messages.

In Section 4.6.1, we discuss the problem of finding an optimal time-invariant intrasession code, and evaluate the performance of symmetric intrasession codes. In Section 4.6.2, we conduct a simulation study to compare symmetric intrasession codes against a family of random time-invariant convolutional codes.

4.6.1 Performance of Symmetric Intrasession Codes

For any time-invariant intrasession code (Section 4.3.1), the decoding probability $\mathbb{P}[S_k]$, where $k \in \mathbb{Z}^+$, can be written in terms of the block sizes or allocation $(x_1, \ldots, x_{m_{\mathsf{E}}c})$ as

$$\mathbb{P}[S_k] = \sum_{U \subseteq \{1,\ldots,d'\}} \mathbb{1}\left[\sum_{i \in U} x_i \geq s\right] \cdot (1-p_{\mathsf{e}})^{|U|} (p_{\mathsf{e}})^{d'-|U|},$$

where $d' \triangleq \min(d, m_{\mathsf{E}}c)$. Since $\mathbb{P}[S_k]$ is constant wrt $k \in \mathbb{Z}^+$, we can drop the index k and consider $\mathbb{P}[S] \triangleq \mathbb{P}[S_1]$ instead.

For a given choice of $(c, d, p_{\mathsf{e}}, s, m_{\mathsf{E}})$, our objective is to find a time-invariant intrasession code, as specified by the allocation $(x_1, \ldots, x_{m_{\mathsf{E}}c})$, that maximizes the decoding probability $\mathbb{P}[S]$. This optimization problem can be expressed explicitly as follows:

$$\begin{aligned}
&\mathbf{\Pi}(c, d, p_{\mathsf{e}}, s, m_{\mathsf{E}}) : \\
&\underset{x_1,\ldots,x_{m_{\mathsf{E}}c}}{\text{maximize}} \sum_{U \subseteq \{1,\ldots,d'\}} \mathbb{1}\left[\sum_{i \in U} x_i \geq s\right] \cdot (1-p_{\mathsf{e}})^{|U|} (p_{\mathsf{e}})^{d'-|U|} \\
&\text{subject to} \\
&\qquad \sum_{\substack{i \in \{1,\ldots,m_{\mathsf{E}}c\}: \\ r_{i,c}=j}} x_i \leq 1 \qquad \forall\, j \in \{1, \ldots, c\}, \\
&\qquad\qquad x_i \geq 0 \qquad \forall\, i \in \{1, \ldots, m_{\mathsf{E}}c\}, \\
&\text{where} \\
&\qquad\qquad d' = \min(d, m_{\mathsf{E}}c).
\end{aligned}$$

For the special case of $c = 1$, this problem reduces to the independent probabilistic access variation of the distributed storage allocation problem (Section 2.2), with number of nodes $n = d'$, access probability $p = 1 - p_{\mathsf{e}}$, and budget $T = 1/s$. As demonstrated in Sections 2.1.1 and 2.2, the

optimal allocation can have nonintuitive structure and can be difficult to find in general. Dividing the unit packet size evenly among m out of the m_{E} most recent messages, where $m \in \{1, \ldots, m_{\mathsf{E}}\}$, may not necessarily produce an optimal allocation. For example, given $c = 1$, $d \geq m_{\mathsf{E}} = 4$, $p_{\mathsf{e}} < \frac{1}{2}$, and $\frac{1}{3} < s \leq \frac{2}{5}$, the allocation $\left(\frac{2}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}\right)$, which is optimal, achieves a strictly higher decoding probability than $(1, 0, 0, 0)$, $\left(\frac{1}{2}, \frac{1}{2}, 0, 0\right)$, $\left(\frac{1}{3}, \frac{1}{3}, \frac{1}{3}, 0\right)$, and $\left(\frac{1}{4}, \frac{1}{4}, \frac{1}{4}, \frac{1}{4}\right)$.

Given the difficulty of finding the optimal time-invariant intrasession code, we shall restrict our attention to the family of symmetric intrasession codes (Section 4.3.1).

4.6.1.1 Decoding Probability

Consider the decodability of a given message $k \in \mathbb{Z}^+$ for the symmetric code with spreading parameter m. Suppose that V_{S} small blocks and V_{B} big blocks that encode message k are received by the decoder; V_{S} and V_{B} are independent binomial random variables with the following distributions:

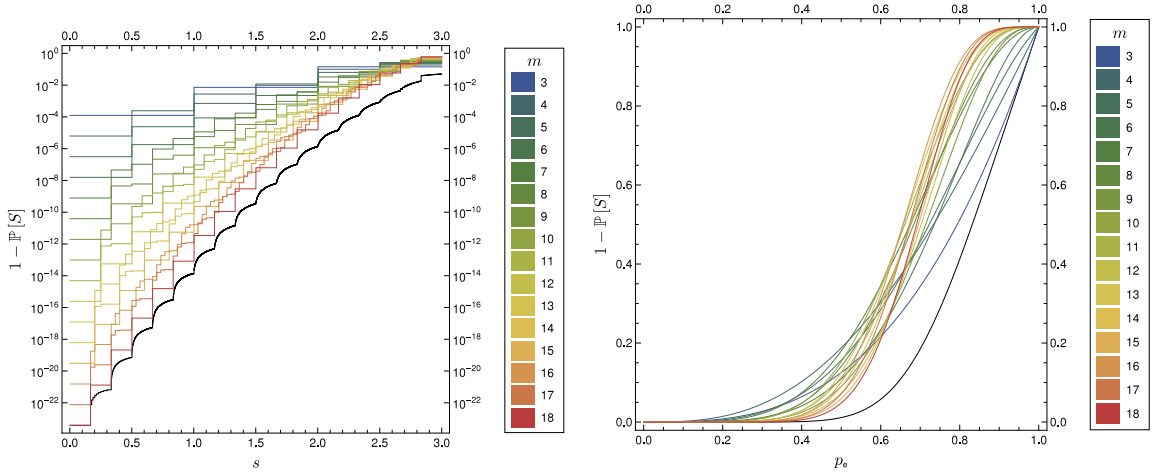
$$V_{\mathsf{S}} \sim \text{Binomial}\big((q_{m,c} + 1)r_{m,c}, 1 - p_{\mathsf{e}}\big),$$
$$V_{\mathsf{B}} \sim \text{Binomial}\big(q_{m,c}(c - r_{m,c}), 1 - p_{\mathsf{e}}\big).$$

The decoding probability $\mathbb{P}[S]$ can therefore be expressed in terms of these random variables as follows:

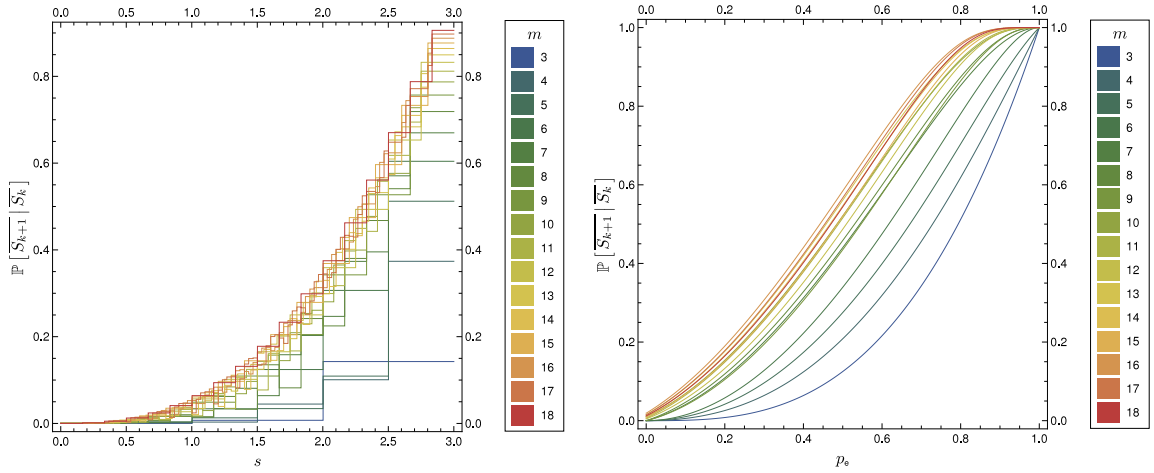
$$\mathbb{P}[S] = \begin{cases} \mathbb{P}\left[\dfrac{1}{q_{m,c}+1} V_{\mathsf{S}} \geq s\right] & \text{if } r_{m,c} = c, \\ \mathbb{P}\left[\dfrac{1}{q_{m,c}+1} V_{\mathsf{S}} + \dfrac{1}{q_{m,c}} V_{\mathsf{B}} \geq s\right] & \text{otherwise} \end{cases}$$

$$= \begin{cases} \mathbb{P}\left[V_{\mathsf{S}} \geq \lceil s(q_{m,c}+1) \rceil\right] & \text{if } r_{m,c} = c, \\ \displaystyle\sum_{v_{\mathsf{S}}} \mathbb{P}\left[V_{\mathsf{S}} = v_{\mathsf{S}}\right] \cdot \mathbb{P}\left[V_{\mathsf{B}} \geq \left\lceil \left(s - \frac{v_{\mathsf{S}}}{q_{m,c}+1}\right) q_{m,c} \right\rceil\right] & \text{otherwise.} \end{cases}$$

Figures 4.8a and 4.8b show how the family of symmetric codes perform in terms of the decoding probability $\mathbb{P}[S]$, for an instance of (c, d, m_{E}). These plots and other empirical observations suggest that maximal spreading (i.e., $m = d'$) performs well, i.e., achieves a relatively high $\mathbb{P}[S]$, when the message size s and the packet erasure probability p_{e} are small, while minimal spreading (i.e.,



(a) $p_e = 0.05$ (b) $s = 1$



(c) $p_e = 0.05$ (d) $s = 1$

Figure 4.8. Plots of the decoding failure probability $1 - \mathbb{P}[S]$ and the burstiness of undecodable messages as measured by the conditional probability $\mathbb{P}\left[\overline{S_{k+1}} \,\middle|\, \overline{S_k}\right]$, where $k \in \mathbb{Z}^+$, against message size s and packet erasure probability p_e, for the family of symmetric intrasession codes, for $(c, d, m_E) = (3, 18, 6)$. In (a) and (c), we set $p_e = 0.05$. In (b) and (d), we set $s = 1$. Spreading parameter $m \in \{c, \ldots, d'\}$, where $d' = \min(d, m_E c) = 18$, gives the size of the effective coding window for each code. The black curve in (a) and (b) describes a lower bound on the decoding failure probability for any time-invariant code, as given by Theorem 4.14.

$m = c$) performs well when s and p_{e} are large (this echoes the analytical findings of Section 2.2.3). Furthermore, although this family of codes may not always contain an optimal time-invariant intrasession code, we can find good codes with decoding probabilities close to the upper bound of Theorem 4.14 among them when s and p_{e} are small.

4.6.1.2 Burstiness of Undecodable Messages

Consider the decodability of a given pair of consecutive messages k and $k+1$, where $k \in \mathbb{Z}^+$, for the symmetric code with spreading parameter m. The $2m$ blocks that encode the pair of messages are spread over the $m + c$ time steps in the union of the two effective coding windows, i.e.,

$$\{(k-1)c + 1, \ldots, kc + m\}.$$

These time steps can be partitioned into the following three intervals:

1) $\{(k-1)c + 1, \ldots, (k-1)c + c\}$, in which c blocks that encode message k and zero blocks that encode message $k+1$ are transmitted;

2) $\{kc+1, \ldots, (k-1)c+m\}$, in which $m - c$ blocks that encode message k and $m - c$ blocks that encode message $k+1$ are transmitted; and

3) $\{(k-1)c + m + 1, \ldots, kc + m\}$, in which zero blocks that encode message k and c blocks that encode message $k+1$ are transmitted.

Suppose that $V_{\mathsf{S}}^{(1)}$ small blocks and $V_{\mathsf{B}}^{(1)}$ big blocks that encode message k are received by the decoder in the first interval, $V_{\mathsf{S}}^{(2)}$ small blocks and $V_{\mathsf{B}}^{(2)}$ big blocks that encode message k are received by the decoder in the second interval (the same numbers of blocks that encode message $k+1$ are also received in the same interval), and $V_{\mathsf{S}}^{(3)}$ small blocks and $V_{\mathsf{B}}^{(3)}$ big blocks that encode message $k+1$ are received by the decoder in the third interval; $V_{\mathsf{S}}^{(1)}$, $V_{\mathsf{B}}^{(1)}$, $V_{\mathsf{S}}^{(2)}$, $V_{\mathsf{B}}^{(2)}$, $V_{\mathsf{S}}^{(3)}$, and $V_{\mathsf{B}}^{(3)}$ are independent binomial random variables with the following distributions:

$$V_{\mathsf{S}}^{(1)} \sim \text{Binomial}\big(r_{m,c}, 1 - p_{\mathsf{e}}\big),$$
$$V_{\mathsf{B}}^{(1)} \sim \text{Binomial}\big(c - r_{m,c}, 1 - p_{\mathsf{e}}\big),$$
$$V_{\mathsf{S}}^{(2)} \sim \text{Binomial}\big(q_{m,c} r_{m,c}, 1 - p_{\mathsf{e}}\big),$$

$$V_{\mathsf{B}}^{(2)} \sim \text{Binomial}\big((q_{m,c}-1)(c-r_{m,c}), 1-p_{\mathsf{e}}\big),$$
$$V_{\mathsf{S}}^{(3)} \sim \text{Binomial}\big(r_{m,c}, 1-p_{\mathsf{e}}\big),$$
$$V_{\mathsf{B}}^{(3)} \sim \text{Binomial}\big(c-r_{m,c}, 1-p_{\mathsf{e}}\big).$$

The conditional probability $\mathbb{P}\left[\,\overline{S_{k+1}}\,\middle|\,\overline{S_k}\,\right]$ can therefore be expressed in terms of these random variables as follows:

$$\mathbb{P}\left[\,\overline{S_{k+1}}\,\middle|\,\overline{S_k}\,\right] = \begin{cases} \mathbb{P}\left[\dfrac{1}{q_{m,c}+1}\Big(V_{\mathsf{S}}^{(2)}+V_{\mathsf{S}}^{(3)}\Big) < s \;\middle|\; \dfrac{1}{q_{m,c}+1}\Big(V_{\mathsf{S}}^{(1)}+V_{\mathsf{S}}^{(2)}\Big) < s\right] & \text{if } r_{m,c}=c, \\ \mathbb{P}\left[\begin{array}{l} \dfrac{1}{q_{m,c}+1}\Big(V_{\mathsf{S}}^{(2)}+V_{\mathsf{S}}^{(3)}\Big) + \dfrac{1}{q_{m,c}}\Big(V_{\mathsf{B}}^{(2)}+V_{\mathsf{B}}^{(3)}\Big) < s \;\Big|\; \\ \qquad \dfrac{1}{q_{m,c}+1}\Big(V_{\mathsf{S}}^{(1)}+V_{\mathsf{S}}^{(2)}\Big) + \dfrac{1}{q_{m,c}}\Big(V_{\mathsf{B}}^{(1)}+V_{\mathsf{B}}^{(2)}\Big) < s \end{array}\right] & \text{otherwise.} \end{cases}$$

Figures 4.8c and 4.8d show how the family of symmetric codes perform in terms of the burstiness of undecodable messages as measured by the conditional probability $\mathbb{P}\left[\,\overline{S_{k+1}}\,\middle|\,\overline{S_k}\,\right]$, for an instance of (c, d, m_{E}). These plots and other empirical observations suggest that over a wide range of message sizes s and packet erasure probabilities p_{e}, minimal spreading (i.e., $m = c$) performs well, i.e., achieves a relatively low $\mathbb{P}\left[\,\overline{S_{k+1}}\,\middle|\,\overline{S_k}\,\right]$, while maximal spreading (i.e., $m = d'$) performs poorly. This agrees with the intuition that for a pair of consecutive messages, a greater overlap in their effective coding windows would tend to increase the correlation between their decodabilities. In the case of minimal spreading (i.e., $m = c$), the decodability of a message is independent of the decodability of other messages because the effective coding windows do not overlap at all.

4.6.1.3 Trade-off between Performance Metrics

Our results show that for the family of symmetric codes, a trade-off exists between the decoding probability $\mathbb{P}[S]$ and the burstiness of undecodable messages as measured by the conditional probability $\mathbb{P}\left[\,\overline{S_{k+1}}\,\middle|\,\overline{S_k}\,\right]$ when the message size s and packet erasure probability p_{e} are small (this is a regime of interest because it supports a high decoding probability). Specifically, although maximal spreading (i.e., $m = d'$) achieves a high decoding probability, it also exhibits a higher burstiness of undecodable messages. Thus, a symmetric code with a suboptimal decoding probability but lower

burstiness may be preferred for an application that is sensitive to bursty undecodable messages.

4.6.2 Simulation Study: Symmetric Intrasession Codes vs. Random Time-Invariant Convolutional Codes

In this section, we compare the family of symmetric intrasession codes (Section 4.3.1) against a family of *random time-invariant convolutional codes*, for the special case of unit-size messages, i.e., $s = 1$. These random convolutional codes are constructed as follows: for each code, we specify a finite field GF(q) and a *mixing parameter* $r \in \{1, \ldots, m_{\mathsf{E}}\}$, and set each encoding function f_i, $i \in \{1, \ldots, c\}$, to a random linear combination of the most recent r messages, i.e.,

$$f_i\big(M_k, M_{k-1}, \ldots, M_{k-m_{\mathsf{E}}+1}\big) \triangleq \sum_{j=1}^{r} \omega_j^{(i)} M_{k-j+1},$$

where each coefficient $\omega_j^{(i)}$, $i \in \{1, \ldots, c\}$, $j \in \{1, \ldots, r\}$, is independently selected uniformly at random from the set $\{0, 1, \ldots, q-1\}$. To determine the decodability of each message in our simulation, the decoder attempts to solve for the unknown message at the corresponding decoding deadline by applying Gauss-Jordan elimination to the matrix of coefficients collected from all unerased packets received up to that time step.

Polyanskiy [14] analyzed similar convolutional codes over GF(2) for $(c, m_{\mathsf{E}}) = (2, 3)$ and $(2, 4)$, adopting the expected message decoding delay as the performance metric; combinatorial code properties were defined and used by the author to find good codes with low latency. Similar linear time-invariant binary codes were also examined by Sukhavasi [17] for the binary erasure channel operating at the bit level; the author presented a random construction that is anytime reliable (i.e., the message decoding failure probability decays exponentially with delay) with high probability.

4.6.2.1 Simulation Setup

Setting $(c, d, s, m_{\mathsf{E}}) = (2, 5, 1, 8)$, we evaluated the performance of the two families of codes

- **symmetric intrasession codes**, with spreading parameter $m \in \{c, \ldots, d'\} = \{2, 3, 4, 5\}$; and
- **random time-invariant convolutional codes**, over finite fields of size $q \in \{2, 16, 256\}$ and with mixing parameter $r \in \{2, 4, 8\}$,

under three erasure scenarios

- **low-erasure scenario**, with $p_{\mathsf{e}} = 0.1$;
- **medium-erasure scenario**, with $p_{\mathsf{e}} = 0.3$; and
- **high-erasure scenario**, with $p_{\mathsf{e}} = 0.5$.

We considered two time horizons

- **long time horizon**, with the receiver attempting to decode $10\,000$ consecutive messages; and
- **short time horizon**, with the receiver attempting to decode 20 consecutive messages,

and two joining times for the receiver

- at the **beginning** of the stream, with message 1 as the first message to be decoded; and
- **midway** through the stream, with a high-numbered message as the first message to be decoded.

A total of 100 and 500 random convolutional codes were generated for the long and short time horizons, respectively, for each choice of (q, r). A total of 100 and 500 random erasure patterns of the appropriate length were generated for the long and short time horizons, respectively, for each erasure scenario. We computed the following two performance metrics for each simulation run:

- **decoding probability**, given by the fraction

$$\frac{\text{number of messages that are decodable by their decoding deadlines}}{\text{total number of messages } n};\text{ and}$$

- **burstiness of undecodable messages** as measured by the conditional probability that the next message is undecodable by its decoding deadline given that the current message is undecodable by its decoding deadline, given by the fraction

$$\frac{\begin{array}{c}\text{number of message pairs } (k, k{+}1), \text{ where } k{\in}\{1, \ldots, n{-}1\}, \text{ for which} \\ \text{both messages } k \text{ and } k{+}1 \text{ are undecodable by their decoding deadlines}\end{array}}{\begin{array}{c}\text{number of message pairs } (k, k{+}1), \text{ where } k{\in}\{1, \ldots, n{-}1\}, \text{ for which} \\ \text{message } k \text{ is undecodable by its decoding deadline}\end{array}}.$$

4.6.2.2 Simulation Results and Discussion

Figures 4.9, 4.10, and 4.11 summarize the simulation results obtained under each of the three erasure scenarios. Results for the random convolutional codes over GF(2) have been omitted from the plots because of their poor performance.

Optimal Symmetric Codes. We observe a phase transition in the optimal symmetric code in terms of the decoding probability: near-maximal spreading (i.e., $m = 4$) performed best for both the low-erasure and medium-erasure scenarios, whereas minimal spreading (i.e., $m = 2$) performed best for the high-erasure scenario. Also, for all three erasure scenarios, the smaller the amount of spreading (as measured by m), the better the performance in terms of the burstiness of undecodable messages. These findings are consistent with the analytical observations of Section 4.6.1.

Effect of Finite Field Size q on Random Convolutional Codes. For the long time horizon, we observe that increasing the finite field size q improved the decoding probability, especially when the packet erasure probability p_{e} and mixing parameter r are small. A larger finite field size increases the likelihood that a received packet is innovative (i.e., it is not a linear combination of previously received packets); however, this advantage is diluted if too few packets are received in the first place, or if each packet already combines many messages (in both cases, each received packet is already likely to be innovative).

Effect of Mixing Parameter r on Random Convolutional Codes. For the long time horizon, we observe that increasing the mixing parameter r produced a mixed effect on the decoding probability: it was improved for the low-erasure scenario, but was worsened for the high-erasure scenario; for the medium-erasure scenario, the decoding probability was initially worsened but subsequently improved. Mixing more messages in each packet makes the packet useful for the decoding of more messages, but more packets are also required to decode a given message. The former positive effect is dominant for the low-erasure scenario, while the latter negative effect is dominant for the high-erasure scenario.

Effect of Receiver Joining Time. For the short time horizon, we observe that the receiver joining time had a negligible effect on the performance of symmetric codes. The robustness of symmetric codes can be explained by the fact that the decodability of each message depends only on the packets received within a very recent and short effective coding window of m time steps. The performance of the random convolutional codes, on the other hand, was sensitive to the receiver joining time. In

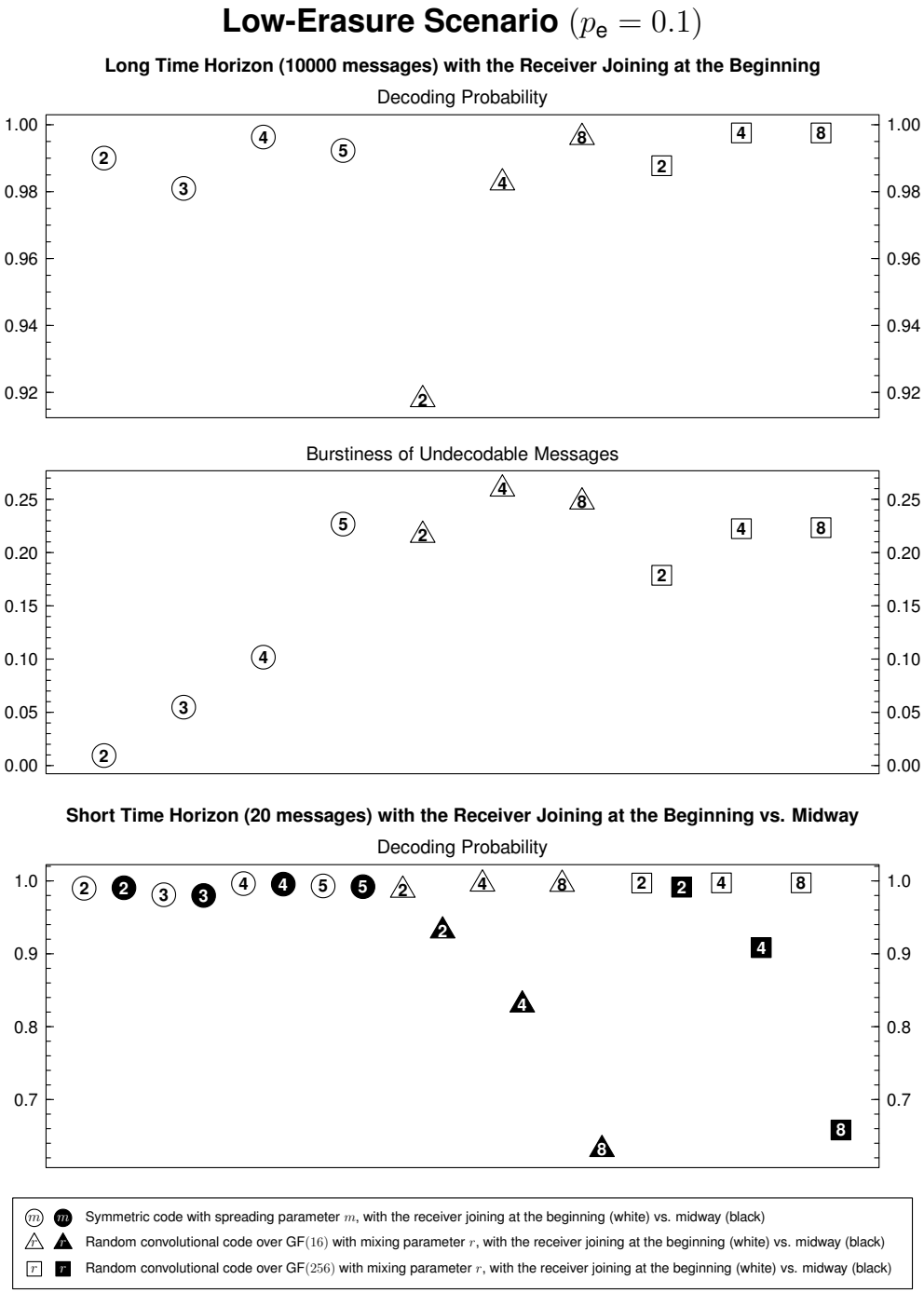


Figure 4.9. Simulation results for the **low-erasure scenario** ($p_e = 0.1$), with $(c, d, s, m_E) = (2, 5, 1, 8)$. Each data point indicates the mean value taken over all randomly generated codes and erasure patterns.

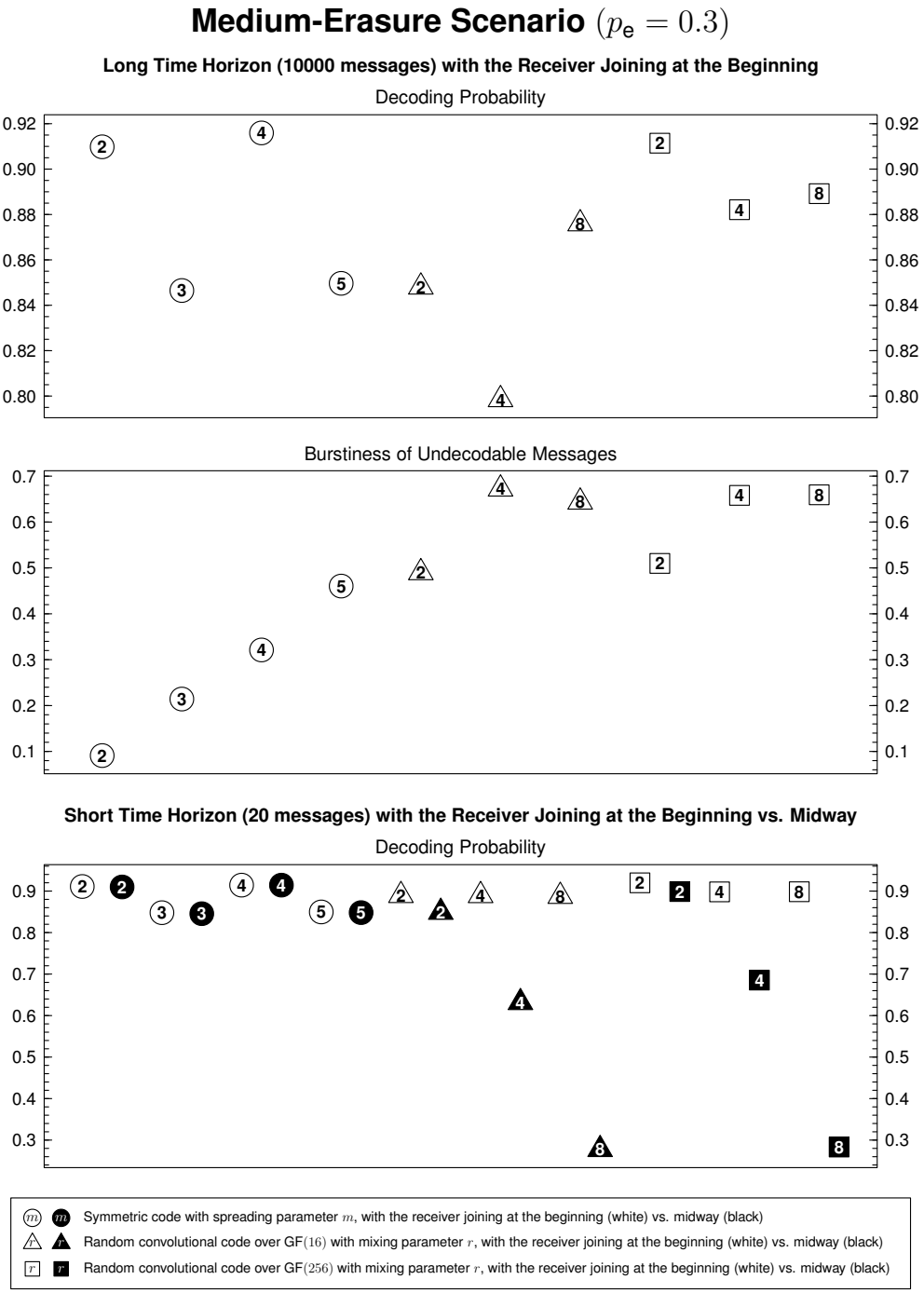


Figure 4.10. Simulation results for the **medium-erasure scenario** ($p_{\mathsf{e}} = 0.3$), with $(c, d, s, m_{\mathsf{E}}) = (2, 5, 1, 8)$. Each data point indicates the mean value taken over all randomly generated codes and erasure patterns.

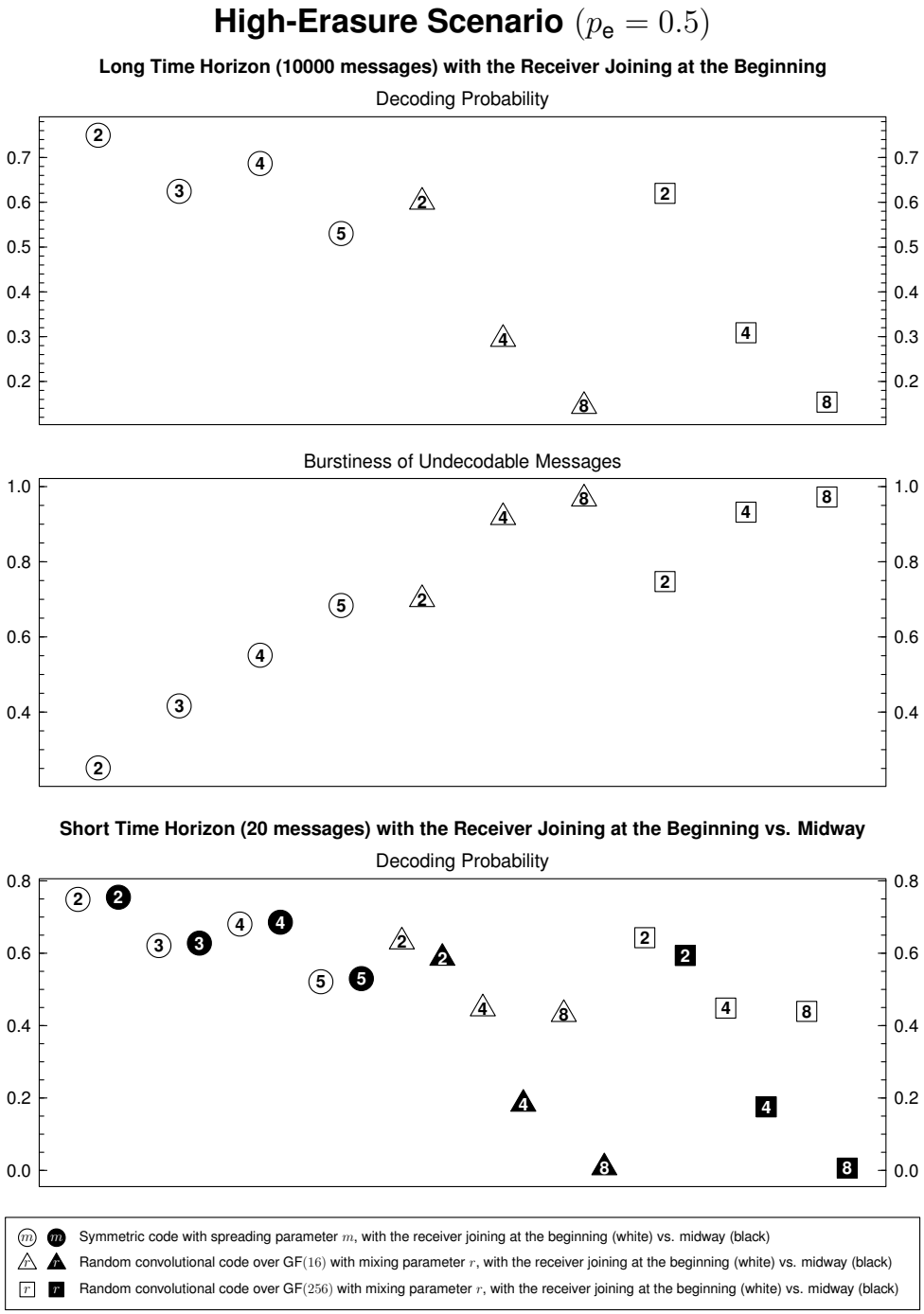


Figure 4.11. Simulation results for the **high-erasure scenario** ($p_e = 0.5$), with $(c, d, s, m_E) = (2, 5, 1, 8)$. Each data point indicates the mean value taken over all randomly generated codes and erasure patterns.

particular, the receiver joining the stream midway achieved a significantly worse decoding probability than the receiver joining the stream at the beginning; furthermore, the gap in their performance increased with the mixing parameter r. These effects can be explained by the *a priori* knowledge of previous messages transmitted before joining the stream, which can be beneficial in decoding subsequent messages: a receiver joining at the beginning would have perfect knowledge of the previous (nonpositive) messages (which are assumed to be zeros), while a receiver joining midway would not have the benefit of this knowledge; furthermore, the larger the value of r, the more difficult decoding is for the latter receiver because each packet is a combination of more unknown messages.

Symmetric Codes vs. Random Convolutional Codes. For the long time horizon, the highest decoding probability was achieved by a symmetric code in two out of the three erasure scenarios; for the low-erasure scenario, the optimal symmetric code performed only slightly worse than the optimal random convolutional code. Also, for each of the three erasure scenarios, the symmetric code with the highest decoding probability achieved a significantly better burstiness of undecodable messages than all the random convolutional codes. For the short time horizon with the receiver joining at the beginning vs. midway, symmetric codes also performed more robustly than the random convolutional codes.

In summary, while the family of symmetric codes may be suboptimal in certain cases, we can find, for a wide range of scenarios, symmetric codes that perform well in terms of both the decoding probability and the burstiness of undecodable messages.

The strengths and weaknesses of the two families of codes suggest that it may be beneficial to consider a hybrid code construction. One approach is to allocate the packet space between a symmetric code and a random convolutional code, and adjust the allocation depending on the operating regime. Another approach is to divide each message into smaller submessage blocks, and have each packet comprise multiple coded blocks that are random linear combinations of carefully chosen submessage blocks.

4.7 Conclusion and Future Work

We considered a real-time streaming problem for a packet erasure link, where each message must be decoded within a given delay from its creation time.

We showed that the symmetric intrasession code with spreading parameter $m = d$ is asymptotically optimal over all codes for both the coding window and sliding window variations of the window-based erasure model, and for the bursty erasure model when the maximum erasure burst length is sufficiently short or long. We also showed that diagonally interleaved codes derived from specific systematic block codes are asymptotically optimal over all codes for the bursty erasure model in several other cases.

For the i.i.d. erasure model, we derived an upper bound on the decoding probability for any time-invariant code. We also analyzed the performance of symmetric intrasession codes, and observed a phase transition in their relative performance in terms of the decoding probability: maximal spreading performs well when the message size s and packet erasure probability p_{e} are small, while minimal spreading performs well when s and p_{e} are large. In terms of the burstiness of undecodable messages, minimal spreading performs consistently well over a wide range of scenarios. Thus, a trade-off between the two performance metrics exists for this family of codes when s and p_{e} are small; this is also the regime in which maximal spreading achieves a decoding probability close to the derived upper bound. In a simulation study, these symmetric codes performed well against a family of random time-invariant convolutional codes under a number of scenarios.

The work in this chapter can be extended in several directions. While optimal real-time streaming codes have been constructed for both variations of the window-based erasure model, such codes have yet to be found for the bursty erasure model in a number of cases, e.g., when $c - r_{d,c} < z < c$, or $d - c < z < d - r_{d,c}$, or when only the first four conditions of Theorems 4.10 and 4.12 are satisfied. The i.i.d. erasure model also offers many interesting problems for future work. In an effort to find the optimal code, it may be useful to consider hybrid code constructions that capture the strengths of both the symmetric intrasession codes and the random time-invariant convolutional codes that were examined in the simulation study.

4.8 Proofs of Theorems

4.8.1 Proof of Lemma 4.2

Observe that $\mathbf{y}$ is simply a vector containing the block sizes for each message, i.e., $\{x_i\}_{i=1}^{d}$, sorted in ascending order. Since

$$\sum_{i\in U} x_i \geq \sum_{j=1}^{|U|} y_j \quad \forall\, U \subseteq \{1,\ldots,d\},$$

it follows that over any ℓ packets transmitted in the coding window W_k, the total size of the blocks allocated to message k is at least $\sum_{j=1}^{\ell} y_j$. Therefore, assuming that an appropriate code is applied to the allocation, message k is always decodable from any ℓ packets transmitted in W_k as long as the message size s does not exceed $\sum_{j=1}^{\ell} y_j$. ■

4.8.2 Proof of Lemma 4.3

The stated partition can be constructed by assigning each time step $t \in T_n$ to the set $T_n^{(q_{i,c}c+r_{i,c})}$, where

$$r_{i,c} = r_{t,c},$$
$$q_{i,c} = \begin{cases} q_{t,c} - \left\lfloor \frac{q_{t,c}}{q_{d,c}+1} \right\rfloor (q_{d,c}+1) & \text{if } r_{t,c} \leq r_{d,c}, \\ q_{t,c} - \left\lfloor \frac{q_{t,c}}{q_{d,c}} \right\rfloor q_{d,c} & \text{if } r_{t,c} > r_{d,c}. \end{cases}$$

Note that index $q_{i,c}\,c + r_{i,c} \in \{1,\ldots,d\}$ since $q_{i,c} \in \{0,\ldots,q_{d,c}\}$ when $r_{i,c} \in \{1,\ldots,r_{d,c}\}$, and $q_{i,c} \in \{0,\ldots,q_{d,c}-1\}$ when $r_{i,c} \in \{r_{d,c}+1,\ldots,c\}$. To prove the required code properties, we consider two separate cases:

Case 1: Consider the set $T_n^{(i)}$ for a choice of i satisfying $r_{i,c} \leq r_{d,c}$. Since each time step $t \in T_n^{(i)}$ can be expressed as

$$t = \underbrace{\big(j(q_{d,c}+1)+q_{i,c}\big)}_{q_{t,c}}\, c + \underbrace{r_{i,c}}_{r_{t,c}} \triangleq t_j, \quad \text{where } j \in \mathbb{Z}_0^+,$$

it follows from the code construction that the set of active messages at each time step contains $q_{d,c}+1$ messages, and is given by

$$A_{t_j} = \Big\{ \underbrace{j(q_{d,c}+1)+q_{i,c}}_{q_{t,c}} +1-q_{d,c}, \ldots, \underbrace{j(q_{d,c}+1)+q_{i,c}}_{q_{t,c}} +1 \Big\}.$$

The smallest time step in $T_n^{(i)}$ corresponds to the choice of $j=0$, which produces $t_0 = q_{i,c}\, c + r_{i,c} = i$ and the set of active messages

$$A_{t_0} = \{q_{i,c}+1-q_{d,c}, \ldots, q_{i,c}+1\}.$$

Note that A_{t_0} contains message 1 since $q_{i,c} \in \{0, \ldots, q_{d,c}\}$, which implies that

$$q_{i,c}+1-q_{d,c} \leq 1 \leq q_{i,c}+1.$$

At the other extreme, let the largest time step in $T_n^{(i)}$ correspond to the choice of $j=j'$; we therefore have

$$t_{j'} \leq (n-1)c + d < t_{j'+1}, \tag{4.3}$$

and the final set of active messages

$$A_{t_{j'}} = \{j'(q_{d,c}+1)+q_{i,c}+1-q_{d,c}, \ldots, j'(q_{d,c}+1)+q_{i,c}+1\}.$$

From the first inequality of (4.3), we obtain

$$\begin{aligned}
&\big(j'(q_{d,c}+1)+q_{i,c}\big)c + r_{i,c} \leq (n-1+q_{d,c})c + r_{d,c} \\
&\implies n \geq \left\lceil \frac{\big(j'(q_{d,c}+1)+q_{i,c}+1-q_{d,c}\big)c + r_{i,c} - r_{d,c}}{c} \right\rceil \\
&\qquad = j'(q_{d,c}+1)+q_{i,c}+1-q_{d,c} + \left\lceil \frac{r_{i,c}-r_{d,c}}{c} \right\rceil \\
&\qquad = j'(q_{d,c}+1)+q_{i,c}+1-q_{d,c},
\end{aligned} \tag{4.4}$$

where the final step follows from the fact that $1 \le r_{i,c} \le r_{d,c} \le c$, which implies that

$$-1 < \frac{1-c}{c} \le \frac{r_{i,c} - r_{d,c}}{c} \le 0 \quad \Longrightarrow \quad \left\lceil \frac{r_{i,c} - r_{d,c}}{c} \right\rceil = 0.$$

From the second inequality of (4.3), we obtain

$$\begin{aligned}
(n-1+q_{d,c})c + r_{d,c} &\le \big((j'+1)(q_{d,c}+1) + q_{i,c}\big)c + r_{i,c} - 1 \\
\Longrightarrow n &\le \left\lfloor \frac{\big((j'+1)(q_{d,c}+1) + q_{i,c} + 1 - q_{d,c}\big)c + r_{i,c} - r_{d,c} - 1}{c} \right\rfloor \\
&= (j'+1)(q_{d,c}+1) + q_{i,c} + 1 - q_{d,c} + \left\lfloor \frac{r_{i,c} - r_{d,c} - 1}{c} \right\rfloor \\
&= j'(q_{d,c}+1) + q_{i,c} + 1, \qquad (4.5)
\end{aligned}$$

where the final step follows from the fact that $1 \le r_{i,c} \le r_{d,c} \le c$, which implies that

$$-1 = \frac{1-c-1}{c} \le \frac{r_{i,c} - r_{d,c} - 1}{c} \le -\frac{1}{c} < 0 \quad \Longrightarrow \quad \left\lfloor \frac{r_{i,c} - r_{d,c} - 1}{c} \right\rfloor = -1.$$

By combining inequalities (4.4) and (4.5), we arrive at

$$j'(q_{d,c}+1) + q_{i,c} + 1 - q_{d,c} \le n \le j'(q_{d,c}+1) + q_{i,c} + 1,$$

which enables us to infer that $A_{t_{j'}}$ contains message n.

For any pair of consecutive time steps $t_j, t_{j+1} \in T_n^{(i)}$, where

$$\begin{aligned}
t_j &= \big(j(q_{d,c}+1) + q_{i,c}\big)c + r_{i,c}, \\
t_{j+1} &= \big((j+1)(q_{d,c}+1) + q_{i,c}\big)c + r_{i,c},
\end{aligned}$$

we observe that the smallest message in $A_{t_{j+1}}$ is exactly one larger than the largest message in A_{t_j}, i.e.,

$$\begin{aligned}
&(j+1)(q_{d,c}+1) + q_{i,c} + 1 - q_{d,c} \\
&\quad = j(q_{d,c}+1) + q_{i,c} + 1 - q_{d,c} + q_{d,c} + 1
\end{aligned}$$

$$= \big(j(q_{d,c}+1) + q_{i,c} + 1\big) + 1.$$

Thus, there are no overlapping or omitted messages among the sets of active messages corresponding to $T_n^{(i)}$. Property P1 therefore follows.

Case 2: Consider the set $T_n^{(i)}$ for a choice of i satisfying $r_{i,c} > r_{d,c}$. Since each time step $t \in T_n^{(i)}$ can be expressed as

$$t = \underbrace{(j\, q_{d,c} + q_{i,c})}_{q_{t,c}}\, c + \underbrace{r_{i,c}}_{r_{t,c}} \triangleq t_j, \quad \text{where } j \in \mathbb{Z}_0^+,$$

it follows from the code construction that the set of active messages at each time step contains $q_{d,c}$ messages, and is given by

$$A_{t_j} = \Big\{ \underbrace{j\, q_{d,c} + q_{i,c}}_{q_{t,c}} + 1 - (q_{d,c}-1), \ldots, \underbrace{j\, q_{d,c} + q_{i,c}}_{q_{t,c}} + 1 \Big\}.$$

The smallest time step in $T_n^{(i)}$ corresponds to the choice of $j = 0$, which produces $t_0 = q_{i,c}\, c + r_{i,c} = i$ and the set of active messages

$$A_{t_0} = \{q_{i,c} + 1 - (q_{d,c}-1), \ldots, q_{i,c} + 1\}.$$

Note that A_{t_0} contains message 1 since $q_{i,c} \in \{0, \ldots, q_{d,c} - 1\}$, and therefore

$$q_{i,c} + 1 - (q_{d,c} - 1) \leq 1 \leq q_{i,c} + 1.$$

At the other extreme, let the largest time step in $T_n^{(i)}$ correspond to the choice of $j = j'$; we therefore have

$$t_{j'} \leq (n-1)c + d < t_{j'+1}, \tag{4.6}$$

and the final set of active messages

$$A_{t_{j'}} = \{j'\, q_{d,c} + q_{i,c} + 1 - (q_{d,c}-1), \ldots, j'\, q_{d,c} + q_{i,c} + 1\}.$$

From the first inequality of (4.6), we obtain

$$\begin{aligned}
&(j'\,q_{d,c}+q_{i,c})c+r_{i,c}\le(n-1+q_{d,c})c+r_{d,c}\\
&\implies n\ge\left\lceil\frac{(j'\,q_{d,c}+q_{i,c}+1-q_{d,c})c+r_{i,c}-r_{d,c}}{c}\right\rceil\\
&\qquad=j'\,q_{d,c}+q_{i,c}+1-q_{d,c}+\left\lceil\frac{r_{i,c}-r_{d,c}}{c}\right\rceil\\
&\qquad=j'\,q_{d,c}+q_{i,c}+1-(q_{d,c}-1),
\end{aligned}\tag{4.7}$$

where the final step follows from the fact that $1\le r_{d,c}<r_{i,c}\le c$, which implies that

$$0<\frac{r_{i,c}-r_{d,c}}{c}\le\frac{c-1}{c}<1\quad\implies\quad\left\lceil\frac{r_{i,c}-r_{d,c}}{c}\right\rceil=1.$$

From the second inequality of (4.6), we obtain

$$\begin{aligned}
&(n-1+q_{d,c})c+r_{d,c}\le\big((j'+1)q_{d,c}+q_{i,c}\big)c+r_{i,c}-1\\
&\implies n\le\left\lfloor\frac{\big((j'+1)q_{d,c}+q_{i,c}+1-q_{d,c}\big)c+r_{i,c}-r_{d,c}-1}{c}\right\rfloor\\
&\qquad=(j'+1)q_{d,c}+q_{i,c}+1-q_{d,c}+\left\lfloor\frac{r_{i,c}-r_{d,c}-1}{c}\right\rfloor\\
&\qquad=j'\,q_{d,c}+q_{i,c}+1,
\end{aligned}\tag{4.8}$$

where the final step follows from the fact that $1\le r_{d,c}<r_{i,c}\le c$, which implies that

$$0=\frac{1-1}{c}\le\frac{r_{i,c}-r_{d,c}-1}{c}\le\frac{c-1-1}{c}<1\quad\implies\quad\left\lfloor\frac{r_{i,c}-r_{d,c}-1}{c}\right\rfloor=0.$$

By combining inequalities (4.7) and (4.8), we arrive at

$$j'\,q_{d,c}+q_{i,c}+1-(q_{d,c}-1)\le n\le j'\,q_{d,c}+q_{i,c}+1,$$

which enables us to infer that $A_{t_{j'}}$ contains message n.

For any pair of consecutive time steps $t_j,t_{j+1}\in T_n^{(i)}$, where

$$t_j=\big(j\,q_{d,c}+q_{i,c}\big)c+r_{i,c},$$

$$t_{j+1} = \big((j+1)\, q_{d,c} + q_{i,c}\big)c + r_{i,c},$$

we observe that the smallest message in $A_{t_{j+1}}$ is exactly one larger than the largest message in A_{t_j}, i.e.,

$$\begin{aligned}
&(j+1)q_{d,c} + q_{i,c} + 1 - (q_{d,c} - 1) \\
&\quad = j\, q_{d,c} + q_{i,c} + 1 - (q_{d,c} - 1) + q_{d,c} \\
&\quad = \big(j\, q_{d,c} + q_{i,c} + 1\big) + 1.
\end{aligned}$$

Thus, there are no overlapping or omitted messages among the sets of active messages corresponding to $T_n^{(i)}$. Property P1 therefore follows.

For both Case 1 and Case 2, the total size of the packets transmitted at time steps $T_n^{(i)}$, i.e., $\big|T_n^{(i)}\big|$, can be computed by taking the total size of the blocks allocated to the n messages, and adding the unused packet space at the smallest time step (which is allocated to nonpositive dummy messages) and at the largest time step (which is allocated to messages larger than n); this produces the required upper bound of Property P2. ■

4.8.3 Proof of Theorem 4.5

Consider the symmetric code with spreading parameter $m = d$. Observe that under each erasure pattern $E \in \mathcal{E}_n^{\mathsf{CW}}$, at least $d - z$ unerased packets are received in each coding window W_k, because there are at most z erased time steps in each coding window W_k. According to Lemma 4.2, if message size s satisfies the inequality

$$s \leq \sum_{j=1}^{d-z} y_j,$$

then each message $k \in \{1, \ldots, n\}$ is decodable from any $d - z$ packets transmitted in its coding window W_k. Therefore, it follows that the code achieves a message size of $\sum_{j=1}^{d-z} y_j$, by allowing all n messages $\{1, \ldots, n\}$ to be decoded by their respective decoding deadlines under any erasure pattern $E \in \mathcal{E}_n^{\mathsf{CW}}$.

To demonstrate the asymptotic optimality of the code, we will show that this message size

matches the maximum achievable message size s_n^{CW} in the limit, i.e.,

$$\lim_{n\to\infty} s_n^{\mathsf{CW}} = \sum_{j=1}^{d-z} y_j. \tag{4.9}$$

To obtain an upper bound for s_n^{CW}, we consider the cut-set bound corresponding to a specific erasure pattern E' from $\mathcal{E}_n^{\mathsf{CW}}$. Let $\{1,\ldots,d\}$ be partitioned into two sets $V^{(1)}$ and $V^{(2)}$, where

$$V^{(1)} \triangleq \big\{i \in \{1,\ldots,d\} : r_{i,c} \leq r_{d,c}\big\},$$
$$V^{(2)} \triangleq \big\{i \in \{1,\ldots,d\} : r_{i,c} > r_{d,c}\big\}.$$

Let $\mathbf{v} = (v_1,\ldots,v_d)$ be defined as $\mathbf{v} \triangleq \big(\mathbf{v}^{(1)} \mid \mathbf{v}^{(2)}\big)$, where $\mathbf{v}^{(1)}$ is the vector containing the $(q_{d,c}+1)r_{d,c}$ elements of $V^{(1)}$ sorted in ascending order, and $\mathbf{v}^{(2)}$ is the vector containing the $q_{d,c}(c-r_{d,c})$ elements of $V^{(2)}$ sorted in ascending order. Define the erasure pattern $E' \subseteq T_n$ as follows:

$$E' \triangleq \bigcup_{j=d-z+1}^{d} T_n^{(v_j)},$$

where $T_n^{(i)}$ is as defined in Lemma 4.3. The erased time steps in E' have been chosen to coincide with the bigger blocks allocated to each message in the symmetric code. To show that E' is an admissible erasure pattern, we introduce the following lemma:

Lemma 4.15. *If $A \subseteq \{1,\ldots,d\}$, then*

$$\left|\left(\bigcup_{i\in A} T_n^{(i)}\right) \cap W_k\right| = |A| \quad \forall\, k \in \{1,\ldots,n\}, \tag{4.10}$$

where $T_n^{(i)}$ is as defined in Lemma 4.3.

Proof of Lemma 4.15: For each $k \in \{1,\ldots,n\}$, the symmetric code with spreading parameter $m = d$ allocates a block to message k at each time step in its coding window W_k. Thus, it follows from Property P1 of Lemma 4.3 that for each $i \in \{1,\ldots,d\}$, we have

$$\big|T_n^{(i)} \cap W_k\big| = 1 \quad \forall\, k \in \{1,\ldots,n\}.$$

Equation (4.10) therefore follows from the fact that $T_n^{(1)}, \ldots, T_n^{(d)}$ are disjoint sets. ■

Applying Lemma 4.15 with $A = \{v_j\}_{j=d-z+1}^{d}$ produces

$$\left|E' \cap W_k\right| = z \quad \forall\, k \in \{1, \ldots, n\},$$

and thus E' is an admissible erasure pattern, i.e., $E' \in \mathcal{E}_n^{\mathsf{CW}}$.

Now, consider a code that achieves the maximum message size s_n^{CW}. Such a code must allow all n messages $\{1, \ldots, n\}$ to be decoded under the specific erasure pattern E'. We therefore have the following cut-set bound for s_n^{CW}:

$$n\, s_n^{\mathsf{CW}} \leq \left|T_n \backslash E'\right| \iff s_n^{\mathsf{CW}} \leq \frac{1}{n}\left|T_n \backslash E'\right| = \frac{1}{n}\sum_{j=1}^{d-z} \left|T_n^{(v_j)}\right|.$$

Applying the upper bounds in Property P2 of Lemma 4.3, and writing the resulting expression in terms of y_j produces

$$s_n^{\mathsf{CW}} \leq \frac{1}{n}\sum_{j=1}^{d-z} \left|T_n^{(v_j)}\right| \leq \frac{1}{n}\sum_{j=1}^{d-z}(n\, y_j + 2).$$

Since a message size of $\sum_{j=1}^{d-z} y_j$ is known to be achievable (by the symmetric code), we have the following upper and lower bounds for s_n^{CW}:

$$\sum_{j=1}^{d-z} y_j \leq s_n^{\mathsf{CW}} \leq \frac{1}{n}\sum_{j=1}^{d-z}(n\, y_j + 2).$$

These turn out to be matching bounds in the limit as $n \to \infty$:

$$\sum_{j=1}^{d-z} y_j \leq \lim_{n\to\infty} s_n^{\mathsf{CW}} \leq \lim_{n\to\infty} \frac{1}{n}\sum_{j=1}^{d-z}(n\, y_j + 2) = \sum_{j=1}^{d-z} y_j.$$

We therefore have (4.9) as required. ■

4.8.4 Proof of Theorem 4.6

Consider the symmetric code with spreading parameter $m = d$. Observe that under each erasure pattern $E \in \mathcal{E}_n^{\mathsf{SW}}$, at least $d - z$ unerased packets are received in each coding window W_k, because there are at most z erased time steps in each sliding window L_t (which is an interval of at least d

time steps), which implies that there are at most z erased time steps in each coding window W_k (which is an interval of exactly d time steps). According to Lemma 4.2, if message size s satisfies the inequality

$$s \leq \sum_{j=1}^{d-z} y_j,$$

then each message $k \in \{1, \ldots, n\}$ is decodable from any $d - z$ packets transmitted in its coding window W_k. Therefore, it follows that the code achieves a message size of $\sum_{j=1}^{d-z} y_j$, by allowing all n messages $\{1, \ldots, n\}$ to be decoded by their respective decoding deadlines under any erasure pattern $E \in \mathcal{E}_n^{\mathsf{SW}}$.

To demonstrate the asymptotic optimality of the code, we will show that this message size matches the maximum achievable message size s_n^{SW} in the limit, i.e.,

$$\lim_{n\to\infty} s_n^{\mathsf{SW}} = \sum_{j=1}^{d-z} y_j. \tag{4.11}$$

Consider a specific *base erasure pattern* $E' \subseteq T_n$ given by

$$E' \triangleq \bigcup_{j=d-z+1}^{d} T_n^{(v_j)},$$

where $T_n^{(i)}$ is as defined in Lemma 4.3, and $\mathbf{v} = (v_1, \ldots, v_d)$ is as defined in the proof of Theorem 4.5. The erased time steps in E' have been chosen to coincide with the bigger blocks allocated to each message in the symmetric code. From E', we derive the erasure patterns $E'_1, \ldots, E'_n$ given by

$$E'_k \triangleq E' \cap W_k = \bigcup_{j=d-z+1}^{d} \left(T_n^{(v_j)} \cap W_k\right).$$

Applying Lemma 4.15 with $A = \{v_j\}_{j=d-z+1}^{d}$ produces

$$\left|E'_k\right| = \left|E' \cap W_k\right| = z \quad \forall\, k \in \{1, \ldots, n\},$$

which implies that

$$|E'_k \cap L_t| \leq z \quad \forall\, t \in \{1, \ldots, (n-1)c + d - h + 1\}$$

for each $k \in \{1, \ldots, n\}$. Thus, E'_k is an admissible erasure pattern, i.e., $E'_k \in \mathcal{E}_n^{\mathsf{SW}}$, for each $k \in \{1, \ldots, n\}$.

To obtain an upper bound for s_n^{SW}, we introduce the following lemma:

Lemma 4.16. *Suppose that a code achieves a message size of s under a given set of erasure patterns $\mathcal{E}$ for a given choice of (n, c, d). If $E \subseteq T_n$ is such that $E \cap W_k$ is an admissible erasure pattern, i.e., $(E \cap W_k) \in \mathcal{E}$, for each $k \in \{1, \ldots, n\}$, then for each $k \in \{1, \ldots, n\}$,*

$$H\Big(X[W_k \backslash E] \,\Big|\, M_1^k, X_1^{(k-1)c}\Big) \leq \big|T_k \backslash E\big| - k\,s. \tag{4.12}$$

Proof of Lemma 4.16: First, we show that for any $k \in \{1, \ldots, n\}$,

$$H\Big(X[W_k \backslash E] \,\Big|\, M_1^k, X_1^{(k-1)c}\Big) = H\Big(X[W_k \backslash E] \,\Big|\, M_1^{k-1}, X_1^{(k-1)c}\Big) - s. \tag{4.13}$$

To do this, we consider the conditional mutual information

$$\begin{aligned}
&I\Big(X[W_k \backslash E]\,;\, M_k \,\Big|\, M_1^{k-1}, X_1^{(k-1)c}\Big) \\
&= H\Big(X[W_k \backslash E] \,\Big|\, M_1^{k-1}, X_1^{(k-1)c}\Big) - H\Big(X[W_k \backslash E] \,\Big|\, M_1^{k}, X_1^{(k-1)c}\Big) \\
&= H\Big(M_k \,\Big|\, M_1^{k-1}, X_1^{(k-1)c}\Big) - H\Big(M_k \,\Big|\, M_1^{k-1}, X[\{1, \ldots, (k-1)c\} \cup (W_k \backslash E)]\Big).
\end{aligned}$$

Rearranging terms produces

$$\begin{aligned}
&H\Big(X[W_k \backslash E] \,\Big|\, M_1^k, X_1^{(k-1)c}\Big) \\
&= H\Big(X[W_k \backslash E] \,\Big|\, M_1^{k-1}, X_1^{(k-1)c}\Big) - H\Big(M_k \,\Big|\, M_1^{k-1}, X_1^{(k-1)c}\Big) \\
&\quad + H\Big(M_k \,\Big|\, M_1^{k-1}, X[\{1, \ldots, (k-1)c\} \cup (W_k \backslash E)]\Big).
\end{aligned} \tag{4.14}$$

Since messages are independent and message k is created at time step $(k-1)c + 1$, we have

$$H\Big(M_k \,\Big|\, M_1^{k-1}, X_1^{(k-1)c}\Big) = H\big(M_k\big) = s. \tag{4.15}$$

Furthermore, since $E \cap W_k$ is an admissible erasure pattern, message k must be decodable from the packets transmitted at time steps $T_k \backslash (E \cap W_k) = (T_k \backslash W_k) \cup (W_k \backslash E) =$

$\{1,\ldots,(k-1)c\} \cup (W_k \backslash E)$, and so

$$H\Big(M_k \,\Big|\, M_1^{k-1}, X[\{1,\ldots,(k-1)c\} \cup (W_k \backslash E)]\Big) = 0. \tag{4.16}$$

Substituting (4.15) and (4.16) into (4.14) yields (4.13), as required.

We now proceed to prove by induction that inequality (4.12) holds for any $k \in \{1,\ldots,n\}$.

(Base case) Consider the case of $k = 1$. According to (4.13), we have

$$H\big(X[W_1 \backslash E] \,\big|\, M_1\big) = H\big(X[W_1 \backslash E]\big) - s \leq \big|W_1 \backslash E\big| - s = \big|T_1 \backslash E\big| - s,$$

as required, where the inequality follows from the fact that $H(X_t) \leq 1$ for any t because of the unit packet size.

(Inductive step) Suppose that (4.12) holds for some $k \in \{1,\ldots,n-1\}$. According to (4.13), we have

$$\begin{aligned}
&H\Big(X[W_{k+1} \backslash E] \,\Big|\, M_1^{k+1}, X_1^{kc}\Big) \\
&= H\Big(X[W_{k+1} \backslash E] \,\Big|\, M_1^{k}, X_1^{kc}\Big) - s \\
&\overset{\text{(a)}}{\leq} H\Big(X\big[(W_k \backslash E) \cup (W_{k+1} \backslash E)\big] \,\Big|\, M_1^{k}, X_1^{kc}\Big) - s \\
&\overset{\text{(b)}}{\leq} H\Big(X\big[(W_k \backslash E) \cup (W_{k+1} \backslash E)\big] \,\Big|\, M_1^{k}, X_1^{(k-1)c}\Big) - s \\
&\overset{\text{(c)}}{\leq} H\Big(X[W_k \backslash E] \,\Big|\, M_1^{k}, X_1^{(k-1)c}\Big) + H\Big(X\big[(W_{k+1} \backslash E) \backslash (W_k \backslash E)\big]\Big) - s \\
&\overset{\text{(d)}}{\leq} \big|T_k \backslash E\big| - k\,s + \big|(W_{k+1} \backslash E) \backslash (W_k \backslash E)\big| - s \\
&\overset{\text{(e)}}{=} \big|T_{k+1} \backslash E\big| - (k+1)s,
\end{aligned}$$

as required, where

(a) follows from the addition of random variables $X[W_k \backslash E]$ in the entropy term;

(b) follows from the removal of conditioned random variables $X_{(k-1)c+1}^{kc}$ in the entropy term;

(c) follows from the chain rule for joint entropy, and the removal of conditioned random variables $X[W_k \backslash E]$, M_1^k, and $X_1^{(k-1)c}$ in the second entropy term;

(d) follows from the inductive hypothesis, and the fact that $H(X_t) \leq 1$ for any t because of the unit packet size;

(e) follows from the fact that

$$\begin{aligned}
&\big|T_k \backslash E\big| + \big|(W_{k+1} \backslash E) \big\backslash (W_k \backslash E)\big| \\
&= \big|T_k \backslash E\big| + \big|(W_{k+1} \backslash W_k) \big\backslash E\big| \\
&= \big|T_k \backslash E\big| + \big|(T_{k+1} \backslash T_k) \big\backslash E\big| = \big|T_{k+1} \backslash E\big|.
\end{aligned}$$

■

Applying Lemma 4.16 with $\mathcal{E} = \mathcal{E}_n^{\mathsf{SW}}$ and $E = E'$ to an optimal code that achieves a message size of s_n^{SW} produces

$$H\Big(X[W_k \backslash E'] \,\Big|\, M_1^k, X_1^{(k-1)c}\Big) \leq \big|T_k \backslash E'\big| - k\, s_n^{\mathsf{SW}}$$

for any $k \in \{1, \ldots, n\}$. Since the conditional entropy term is nonnegative, it follows that for the choice of $k = n$, we have

$$\big|T_n \backslash E'\big| - n\, s_n^{\mathsf{SW}} \geq 0 \iff s_n^{\mathsf{SW}} \leq \frac{1}{n}\big|T_n \backslash E'\big| = \frac{1}{n} \sum_{j=1}^{d-z} \big|T_n^{(v_j)}\big|.$$

The rest of the proof leading to the attainment of (4.11) is the same as that of Theorem 4.5, with s_n^{CW} replaced by s_n^{SW}. ■

4.8.5 Proof of Theorem 4.7

Consider the symmetric code with spreading parameter $m = d$. Observe that under each erasure pattern $E \in \mathcal{E}_n^{\mathsf{B}}$, at least $d - z$ unerased packets are received in each coding window W_k, because there are at most z erased time steps in each coding window W_k: if W_k intersects with zero erasure bursts, then it contains zero erased time steps; if W_k intersects with exactly one erasure burst, then it contains at most z erased time steps (i.e., the maximum length of an erasure burst); if W_k intersects with two or more erasure bursts, then it contains a gap of at least $d - z$ unerased time steps between consecutive erasure bursts, and therefore contains at most z erased time steps. According

to Lemma 4.2, if message size s satisfies the inequality

$$s \leq \sum_{j=1}^{d-z} y_j,$$

then each message $k \in \{1, \ldots, n\}$ is decodable from any $d - z$ packets transmitted in its coding window W_k. Therefore, it follows that the code achieves a message size of $\sum_{j=1}^{d-z} y_j$, by allowing all n messages $\{1, \ldots, n\}$ to be decoded by their respective decoding deadlines under any erasure pattern $E \in \mathcal{E}_n^{\mathsf{B}}$.

To demonstrate the asymptotic optimality of the code, we will show that this message size matches the maximum achievable message size s_n^{B} in the limit, i.e.,

$$\lim_{n\to\infty} s_n^{\mathsf{B}} = \sum_{j=1}^{d-z} y_j, \tag{4.17}$$

for the following three cases:

Case 1: Suppose that d is a multiple of c. In this case, the message size achieved by the symmetric code simplifies to

$$\sum_{j=1}^{d-z} y_j = \frac{d-z}{q_{d,c}+1} = \frac{d-z}{d}c.$$

To obtain an upper bound for s_n^{B}, we consider the cut-set bound corresponding to a specific periodic erasure pattern $E' \subseteq T_n$ given by

$$E' \triangleq \big\{ j\,d + i \in T_n : j \in \mathbb{Z}_0^+, i \in \{1, \ldots, z\} \big\}.$$

Since E' comprises alternating intervals of z erased time steps and $d - z$ unerased time steps, it is an admissible erasure pattern, i.e., $E' \in \mathcal{E}_n^{\mathsf{B}}$.

Now, consider a code that achieves the maximum message size s_n^{B}. Such a code must allow all n messages $\{1, \ldots, n\}$ to be decoded under the specific erasure pattern E'. We therefore have the following cut-set bound for s_n^{B}:

$$n\, s_n^{\mathsf{B}} \leq \big|T_n \backslash E'\big| \leq \left(\frac{(n-1)c + d}{d} + 1 \right) (d - z)$$

$$\implies s_n^{\mathsf{B}} \leq \frac{1}{n}\frac{(n-1)c+2d}{d}(d-z) = \frac{d-z}{d}\left(c+\frac{2d-c}{n}\right).$$

Since a message size of $\frac{d-z}{d}c$ is known to be achievable (by the symmetric code), we have the following upper and lower bounds for s_n^{B}:

$$\frac{d-z}{d}c \leq s_n^{\mathsf{B}} \leq \frac{d-z}{d}\left(c+\frac{2d-c}{n}\right).$$

These turn out to be matching bounds in the limit as $n \to \infty$:

$$\frac{d-z}{d}c \leq \lim_{n\to\infty} s_n^{\mathsf{B}} \leq \lim_{n\to\infty} \frac{d-z}{d}\left(c+\frac{2d-c}{n}\right) = \frac{d-z}{d}c.$$

We therefore have (4.17) as required.

Case 2: Suppose that d is not a multiple of c, and $z \leq c - r_{d,c}$. In this case, the message size achieved by the symmetric code simplifies to

$$\sum_{j=1}^{d-z} y_j = c - \sum_{j=d-z+1}^{d} y_j = c - \frac{z}{q_{d,c}}.$$

Consider a specific *base erasure pattern* $E' \subseteq T_n$ given by

$$E' \triangleq \bigcup_{j=d-z+1}^{d} T_n^{(v_j)},$$

where $T_n^{(i)}$ is as defined in Lemma 4.3, and $\mathbf{v} = (v_1, \ldots, v_d)$ is as defined in the proof of Theorem 4.5. The erased time steps in E' have been chosen to coincide with the bigger blocks allocated to each message in the symmetric code. In this case, E' simplifies to

$$\begin{aligned} E' &= \bigcup_{r_{i,c}=c-z+1}^{c} T_n^{((q_{d,c}-1)c+r_{i,c})} \\ &= \Big\{\big((j+1)q_{d,c}-1\big)c + r_{i,c} \in T_n : j \in \mathbb{Z}_0^+, r_{i,c} \in \{c-z+1, \ldots, c\}\Big\}, \end{aligned}$$

which follows from the definition of $T_n^{(i)}$ and the fact that $r_{i,c} > r_{d,c}$ when $r_{i,c} \in \{c-z+1, \ldots, c\}$. Observe that E' comprises alternating intervals of z erased time

steps and $q_{d,c}\, c - z$ unerased time steps, with each interval of erased time steps corresponding to a specific choice of $j \in \mathbb{Z}_0^+$. Since each erased time step $t \in E'$ can be expressed as

$$t = \underbrace{\big((j+1)q_{d,c} - 1\big)}_{q_{t,c}}\, c + \underbrace{r_{i,c}}_{r_{t,c}},$$

it follows that the set of active messages at time step t is given by

$$A_t = \Big\{ \underbrace{(j+1)q_{d,c}}_{q_{t,c}+1} - (q_{d,c} - 1), \ldots, \underbrace{(j+1)q_{d,c}}_{q_{t,c}+1} \Big\}.$$

Therefore, the set of active messages A_t is the same at every time step t in a given interval of z erased time steps (corresponding to a specific j).

From E', we derive the erasure patterns $E'_1, \ldots, E'_n$ given by

$$E'_k \triangleq E' \cap W_k = \bigcup_{j=d-z+1}^{d} \Big(T_n^{(v_j)} \cap W_k \Big).$$

Applying Lemma 4.15 with $A = \{v_j\}_{j=d-z+1}^{d}$ produces

$$\big|E'_k\big| = \big|E' \cap W_k\big| = z \quad \forall\, k \in \{1, \ldots, n\}.$$

Let $t' \in E'_k$ be one of the z erased time steps in W_k under erasure pattern E'_k. As previously established, t' belongs to an interval of z erased time steps in E' that have the same set of active messages $A_{t'}$ (which contains message k). It follows that this interval of z erased time steps is also in E'_k, and must therefore constitute E'_k itself. Thus, E'_k is an admissible erasure pattern, i.e., $E'_k \in \mathcal{E}_n^{\mathsf{B}}$, for each $k \in \{1, \ldots, n\}$, because it comprises a single erasure burst of z time steps.

Applying Lemma 4.16 with $\mathcal{E} = \mathcal{E}_n^{\mathsf{B}}$ and $E = E'$ to an optimal code that achieves a message size of s_n^{B} produces

$$H\Big(X[W_k \backslash E'] \,\Big|\, M_1^k, X_1^{(k-1)c} \Big) \leq \big|T_k \backslash E'\big| - k\, s_n^{\mathsf{B}}$$

for any $k \in \{1, \ldots, n\}$. Since the conditional entropy term is nonnegative, it follows that for the

choice of $k = n$, we have

$$\big|T_n \backslash E'\big| - n\, s_n^{\mathsf{B}} \geq 0 \iff s_n^{\mathsf{B}} \leq \frac{1}{n}\big|T_n \backslash E'\big| = \frac{1}{n}\sum_{j=1}^{d-z} \big|T_n^{(v_j)}\big|.$$

The rest of the proof leading to the attainment of (4.17) is the same as that of Theorem 4.5, with s_n^{CW} replaced by s_n^{B}.

Case 3: Suppose that d is not a multiple of c, and $z \geq d - r_{d,c} = q_{d,c}\, c$. In this case, the message size achieved by the symmetric code simplifies to

$$\sum_{j=1}^{d-z} y_j = \frac{d-z}{q_{d,c}+1}.$$

Consider a specific *base erasure pattern* $E' \subseteq T_n$ given by

$$E' \triangleq \bigcup_{j=d-z+1}^{d} T_n^{(v_j)},$$

where $T_n^{(i)}$ is as defined in Lemma 4.3, and $\mathbf{v} = (v_1, \ldots, v_d)$ is as defined in the proof of Theorem 4.5. The erased time steps in E' have been chosen to coincide with the bigger blocks allocated to each message in the symmetric code. In this case, E' simplifies to

$$\begin{aligned} E' &= T_n \Bigg\backslash \Bigg(\bigcup_{r_{i,c}=1}^{d-z} T_n^{(r_{i,c})} \Bigg) \\ &= T_n \Big\backslash \Big\{ \big(j(q_{d,c}+1)\big)c + r_{i,c} \in T_n : j \in \mathbb{Z}_0^+, r_{i,c} \in \{1, \ldots, d-z\} \Big\}, \end{aligned}$$

which follows from the definition of $T_n^{(i)}$ and the fact that $r_{i,c} \leq r_{d,c}$ when $r_{i,c} \in \{1, \ldots, d-z\}$. Observe that E' comprises alternating intervals of $d-z$ unerased time steps and $(q_{d,c}+1)c - (d-z) = c - r_{d,c} + z$ erased time steps, with each interval of unerased time steps corresponding to a specific choice of $j \in \mathbb{Z}_0^+$. Since each unerased time step $t \in T_n \backslash E'$ can be expressed as

$$t = \underbrace{\big(j(q_{d,c}+1)\big)}_{q_{t,c}}\, c + \underbrace{r_{i,c}}_{r_{t,c}},$$

it follows that the set of active messages at time step t is given by

$$A_t = \Big\{ \underbrace{j(q_{d,c}+1)}_{q_{t,c}} + 1 - q_{d,c}, \ldots, \underbrace{j(q_{d,c}+1)}_{q_{t,c}} + 1 \Big\}.$$

Therefore, the set of active messages A_t is the same at every time step t in a given interval of $d-z$ unerased time steps (corresponding to a specific j).

From E', we derive the erasure patterns $E'_1, \ldots, E'_n$ given by

$$E'_k \triangleq E' \cap W_k = \bigcup_{j=d-z+1}^{d} \Big(T_n^{(v_j)} \cap W_k \Big).$$

Applying Lemma 4.15 with $A = \{v_j\}_{j=d-z+1}^{d}$ produces

$$\big|E'_k\big| = \big|E' \cap W_k\big| = z \quad \forall\, k \in \{1, \ldots, n\}.$$

Let $t' \in W_k \backslash E'_k$ be one of the $d-z$ unerased time steps in W_k under erasure pattern E'_k. As previously established, t' belongs to an interval of $d-z$ unerased time steps in $T_n \backslash E'$ that have the same set of active messages $A_{t'}$ (which contains message k). It follows that this interval of $d-z$ unerased time steps is also in $W_k \backslash E'_k$, and must therefore constitute $W_k \backslash E'_k$ itself. Thus, E'_k is an admissible erasure pattern, i.e., $E'_k \in \mathcal{E}_n^{\mathsf{B}}$, for each $k \in \{1, \ldots, n\}$, because it comprises either a single erasure burst of z time steps, or two erasure bursts with a combined length of z time steps separated by a gap of $d-z$ unerased time steps.

Applying Lemma 4.16 with $\mathcal{E} = \mathcal{E}_n^{\mathsf{B}}$ and $E = E'$ to an optimal code that achieves a message size of s_n^{B} produces

$$H\Big(X[W_k \backslash E'] \,\Big|\, M_1^k, X_1^{(k-1)c} \Big) \leq \big|T_k \backslash E'\big| - k\, s_n^{\mathsf{B}}$$

for any $k \in \{1, \ldots, n\}$. Since the conditional entropy term is nonnegative, it follows that for the choice of $k = n$, we have

$$\big|T_n \backslash E'\big| - n\, s_n^{\mathsf{B}} \geq 0 \iff s_n^{\mathsf{B}} \leq \frac{1}{n} \big|T_n \backslash E'\big| = \frac{1}{n} \sum_{j=1}^{d-z} \big|T_n^{(v_j)}\big|.$$

The rest of the proof leading to the attainment of (4.17) is the same as that of Theorem 4.5, with s_n^{CW}

replaced by s_n^{B}. ■

4.8.6 Proof of Lemma 4.8

Suppose that the component systematic block code $\mathcal{C}$ satisfies the symbol decoding requirements given by D1 and D2. We will show that the diagonally interleaved code derived from $\mathcal{C}$ achieves a message size of $\frac{d-z}{d}c$ for the bursty erasure model, by allowing each information symbol $M_k[i]$ to be decoded by its respective message decoding deadline under any erasure pattern $E \in \mathcal{E}_n^{\mathsf{B}}$.

Let $\mathcal{M}_k$ be the set of information symbols corresponding to message k, i.e.,

$$\begin{aligned}\mathcal{M}_k &\triangleq \big\{M_k[i] : i \in \{1,\ldots,(d-z)c\}\big\} \\ &= \big\{x_t[i] : t \in \{(k-1)c+1,\ldots,kc\}, i \in \{1,\ldots,d-z\}\big\}.\end{aligned}$$

Let $\mathcal{M}$ be the set of information symbols corresponding to all n messages $\{1,\ldots,n\}$, i.e., $\mathcal{M} \triangleq \bigcup_{k=1}^{n} \mathcal{M}_k$. Recall that each diagonal of d symbols in the derived code is a codeword produced by $\mathcal{C}$. Let $L(x_t[i])$ denote the interval of d consecutive time steps across which the codeword containing symbol $x_t[i]$ spans, i.e.,

$$L(x_t[i]) \triangleq \{t-i+1,\ldots,t-i+d\}.$$

Each information symbol in $\mathcal{M}$ can be mapped to such an interval; the earliest such interval corresponds to $M_1[d-z] = x_1[d-z]$ and is given by

$$L(x_1[d-z]) = \{2-(d-z),\ldots,1+d-(d-z)\},$$

while the latest such interval corresponds to $M_n[(d-z)(c-1)+1] = x_{nc}[1]$ and is given by

$$L(x_{nc}[1]) = \{nc,\ldots,nc+d-1\}.$$

Consider the set of information symbols $\mathcal{M}_k$ corresponding to a given message $k \in \{1,\ldots,n\}$. We will show that each information symbol in $\mathcal{M}_k$ is decodable by time step $(k-1)c+d$, which is the decoding deadline for message k, under any erasure pattern $E \in \mathcal{E}_n^{\mathsf{B}}$. We do this by consid-

ering the codewords that contain one or more information symbols from $\mathcal{M}_k$. There are a total of $d - z + c - 1$ such codewords, corresponding to the intervals

$$L(x_{(k-1)c+1}[d-z]), \ldots, L(x_{(k-1)c+1}[1]), \ldots, L(x_{(k-1)c+c}[1]).$$

We consider two cases separately, depending on whether the entire codeword interval occurs by the message decoding deadline:

Case 1: Consider the codeword corresponding to the interval $L(x_{(k-1)c+1}[i])$, where $i \in \{1, \ldots, d-z\}$. For brevity, we define

$$L_i \triangleq L(x_{(k-1)c+1}[i]) = \{(k-1)c + 1 - i + 1, \ldots, (k-1)c + 1 - i + d\}.$$

Observe that the entire interval L_i occurs by the message decoding deadline since $(k-1)c + 1 - i + d \leq (k-1)c + d$. Let $\mathcal{E}_i^{\mathsf{z}}$ be the set of erasure patterns from $\mathcal{E}^{\mathsf{z}}$ that have been time-shifted to align with L_i, i.e.,

$$\mathcal{E}_i^{\mathsf{z}} \triangleq \left\{\{(k-1)c + 1 - i + t : t \in E^{\mathsf{z}}\} : E^{\mathsf{z}} \in \mathcal{E}^{\mathsf{z}}\right\}.$$

Under each erasure pattern $E \in \mathcal{E}_n^{\mathsf{B}}$, the interval L_i intersects with either

1) zero erasure bursts, in which case L_i contains zero erased time steps; or

2) exactly one erasure burst, in which case L_i contains at most z erased time steps (i.e., the maximum length of an erasure burst), all in one contiguous subinterval; or

3) two or more erasure bursts, in which case L_i contains a gap of at least $d - z$ unerased time steps between consecutive erasure bursts.

In each of these three cases, there exists some erasure pattern $E^{\mathsf{z}} \in \mathcal{E}_i^{\mathsf{z}}$ that is a superset of the erased time steps in L_i, i.e., $(E \cap L_i) \subseteq E^{\mathsf{z}}$. Since the symbol decoding deadlines of D1 and D2 are satisfied under erasure pattern E^{z}, they must also be satisfied under erasure pattern $E \cap L_i$. It follows that all information symbols in the codeword are decodable by the last time step in interval L_i, and therefore by the message decoding deadline. Note that nonpositive time steps are always unerased

under the erasure patterns in $\mathcal{E}_n^{\mathsf{B}}$; their corresponding codeword symbols are therefore always known (recall that information symbols corresponding to nonpositive messages are assumed to be zeros).

Case 2: Consider the codeword corresponding to the interval $L(x_{(k-1)c+i}[1])$, where $i \in \{2, \ldots, c\}$. For brevity, we define

$$L_i \triangleq L(x_{(k-1)c+i}[1]) = \{(k-1)c + i - 1 + 1, \ldots, (k-1)c + i - 1 + d\}.$$

Observe that one or more time steps at the end of the interval L_i occur after the message decoding deadline since $(k-1)c + i - 1 + d > (k-1)c + d$. The first $\min(c + 1 - i, d - z)$ information symbols in the codeword correspond to message k; subsequent information symbols in the codeword (if any) correspond to later messages. Let $\mathcal{E}_i^{\mathsf{Z}}$ be the set of erasure patterns from $\mathcal{E}^{\mathsf{Z}}$ that have been time-shifted to align with L_i, i.e.,

$$\mathcal{E}_i^{\mathsf{Z}} \triangleq \big\{\{(k-1)c + i - 1 + t : t \in E^{\mathsf{Z}}\} : E^{\mathsf{Z}} \in \mathcal{E}^{\mathsf{Z}}\big\}.$$

As in Case 1, under each erasure pattern $E \in \mathcal{E}_n^{\mathsf{B}}$, there exists some erasure pattern $E^{\mathsf{Z}} \in \mathcal{E}_i^{\mathsf{Z}}$ that is a superset of the erased time steps in L_i, i.e., $(E \cap L_i) \subseteq E^{\mathsf{Z}}$. Since the symbol decoding deadlines of D1 and D2 are satisfied under erasure pattern E^{Z}, they must also be satisfied under erasure pattern $E \cap L_i$. In particular, since

$$\min(c + 1 - i, d - z) \leq c + 1 - i < c,$$

it follows from D1 that the first $\min(c + 1 - i, d - z)$ information symbols in the codeword, which correspond to message k, are decodable by the $\big(d - c + \min(c + 1 - i, d - z)\big)$th time step in interval L_i, which is time step

$$\begin{aligned}
&(k-1)c + i - 1 + d - c + \min(c + 1 - i, d - z) \\
&\leq (k-1)c + i - 1 + d - c + c + 1 - i \\
&= (k-1)c + d,
\end{aligned}$$

and therefore by the message decoding deadline. Note that although time steps after $(n-1)c + d$

(which is the final time step in T_n) are always unerased under the erasure patterns in $\mathcal{E}_n^{\mathsf{B}}$, their corresponding codeword symbols are never used for decoding because all the information symbols in $\mathcal{M}$ have to be decoded by the final message decoding deadline, which is time step $(n-1)c+d$.

■

4.8.7 Proof of Theorem 4.9

We will apply Lemma 4.8 to show that the diagonally interleaved code derived from the stated systematic block code $\mathcal{C}$ achieves a message size of $\frac{d-z}{d}c$ for the specified bursty erasure model. To demonstrate the asymptotic optimality of the code, we will show that this message size matches the maximum achievable message size s_n^{B} in the limit, i.e.,

$$\lim_{n\to\infty} s_n^{\mathsf{B}} = \frac{d-z}{d}c. \tag{4.18}$$

To facilitate our description of the decoding procedure for $\mathcal{C}$, we arrange the d symbols of the codeword vector produced by $\mathcal{C}$ sequentially across $d-z$ columns, with all the information symbols on the first row, as shown in Figure 4.5. Note that each column $i \in \{1,\ldots,d-z\}$ of the table contains exactly $\frac{z}{d-z} \geq 1$ parity symbols. For each $i \in \{1,\ldots,d-z\}$, all the (degenerate) parity symbols below the information symbol $a[i]$ in column i of the table have a value of $a[i]$.

Suppose that the d symbols of the codeword vector are transmitted sequentially across an erasure link, one symbol per time step, over the time interval $L \triangleq \{1,\ldots,d\}$. Under each erasure pattern $E^{\mathsf{z}} \in \mathcal{E}^{\mathsf{z}}$ (as defined in Lemma 4.8), exactly one symbol in each column of the table is unerased. Because the degenerate parity symbols take on the values of information symbols in a periodic manner, all the information symbols $a[1],\ldots,a[d-z]$ can be recovered using the $d-z$ unerased symbols. In particular, for each $i \in \{1,\ldots,d-z\}$, the information symbol $a[i]$ can be recovered by time step $d-(d-z)+i$. Since $d-z \geq c$, it follows that the symbol decoding requirements given by D1 and D2 in Lemma 4.8 are satisfied by $\mathcal{C}$. Therefore, according to Lemma 4.8, the derived code achieves a message size of $\frac{d-z}{d}c$.

To obtain an upper bound for s_n^{B}, we consider the cut-set bound corresponding to a specific

periodic erasure pattern $E' \subseteq T_n$ given by

$$E' \triangleq \big\{ j\,d + i \in T_n : j \in \mathbb{Z}_0^+, i \in \{1, \ldots, z\} \big\}.$$

Since E' comprises alternating intervals of z erased time steps and $d - z$ unerased time steps, it is an admissible erasure pattern, i.e., $E' \in \mathcal{E}_n^{\mathsf{B}}$. Now, consider a code that achieves the maximum message size s_n^{B}. Such a code must allow all n messages $\{1, \ldots, n\}$ to be decoded under the specific erasure pattern E'. We therefore have the following cut-set bound for s_n^{B}:

$$\begin{aligned} n\,s_n^{\mathsf{B}} &\leq \big|T_n \backslash E'\big| \leq \left(\frac{(n-1)c+d}{d} + 1\right)(d-z) \\ \implies s_n^{\mathsf{B}} &\leq \frac{1}{n}\frac{(n-1)c+2d}{d}(d-z) = \frac{d-z}{d}\left(c + \frac{2d-c}{n}\right). \end{aligned}$$

Since a message size of $\frac{d-z}{d}c$ is known to be achievable (by the derived code), we have the following upper and lower bounds for s_n^{B}:

$$\frac{d-z}{d}c \leq s_n^{\mathsf{B}} \leq \frac{d-z}{d}\left(c + \frac{2d-c}{n}\right).$$

These turn out to be matching bounds in the limit as $n \to \infty$:

$$\frac{d-z}{d}c \leq \lim_{n\to\infty} s_n^{\mathsf{B}} \leq \lim_{n\to\infty} \frac{d-z}{d}\left(c + \frac{2d-c}{n}\right) = \frac{d-z}{d}c.$$

We therefore have (4.18) as required. ■

4.8.8 Proof of Theorem 4.10

Our proof technique expands that of Theorem 4.9. First, we arrange the d symbols of the codeword vector produced by the stated systematic block code $\mathcal{C}$ sequentially across z columns, with r' information symbols on the second last row, and all the parity symbols on a separate last row, as shown in Figure 4.6. For the case of $r' < z$, we repeat the r' information symbols on the second last row, i.e., $a[d-z-r'+1], \ldots, a[d-z]$, across the row; these repeated *virtual* information symbols are parenthesized to distinguish them from the original *actual* information symbols of the codeword vector. Note that each column $i \in \{1, \ldots, z\}$ of the table contains exactly $\frac{d-z-r'}{z} + 1 \geq 2$ actual

and virtual information symbols. For each $i \in \{1, \ldots, z\}$, the value of the parity symbol $b[i]$ is given by the bit-wise modulo-2 sum (i.e., exclusive-or) of the actual and virtual information symbols above it in column i of the table.

Suppose that the d symbols of the codeword vector are transmitted sequentially across an erasure link, one symbol per time step, over the time interval $L \triangleq \{1, \ldots, d\}$. To show that the symbol decoding requirements given by D1 and D2 in Lemma 4.8 are satisfied by $\mathcal{C}$, we consider the following four exhaustive cases separately:

Case 1: Consider the case of $r' = z$, for which there are no virtual information symbols. Under each erasure pattern $E^{\mathsf{z}} \in \mathcal{E}^{\mathsf{z}}$ (as defined in Lemma 4.8), exactly one symbol in each column of the table is erased. For each $i \in \{1, \ldots, z\}$, if the parity symbol $b[i]$ is erased, then all the information symbols in column i, which include $a[i]$, are unerased. On the other hand, if $b[i]$ is unerased, then 1) exactly one information symbol in column i is erased; and 2) this information symbol can be recovered by time step $d - z + i$ using the unerased parity symbol $b[i]$ and the unerased information symbols in the column.

Case 2.1: Consider the case of $r' < z$, with the erasure pattern E^{z}_j, where

$$j \in \{1, \ldots, d - 2z\} \cup \{d - z + 1, \ldots, d\}.$$

Recall that index j gives the time step of the "leading" erasure in the burst, which is of length z. Under erasure pattern E^{z}_j, the information symbol $a[d-z]$ and the parity symbol $b[1]$ are not simultaneously erased.

For each $i \in \{1, \ldots, r'\}$, if the parity symbol $b[i]$ is erased, then all the information symbols in column i of the table, which include $a[i]$, are unerased. On the other hand, if $b[i]$ is unerased, then 1) exactly one information symbol in the column is erased; and 2) this information symbol can be recovered by time step $d - z + i$ using the unerased parity symbol $b[i]$ and the unerased information symbols in the column. It follows that all the information symbols on the second last row, i.e., $a[d-z-r'+1], \ldots, a[d-z]$, can be recovered by time step $d - z + r'$.

For each $i \in \{r' + 1, \ldots, z\}$, if the parity symbol $b[i]$ is erased, then all the actual information symbols in column i of the table, which include $a[i]$, are unerased. On the other hand, if $b[i]$ is unerased, then 1) exactly one actual information symbol in the column is erased; and 2) this in-

formation symbol can be recovered by time step $d - z + i$ using the unerased parity symbol $b[i]$, the unerased actual information symbols in the column, and the recovered virtual information symbol $a[d-z-r'+r_{i,r'}]$.

Case 2.2: Consider the case of $r' < z$, with the erasure pattern E_j^z, where

$$j \in \{d - 2z + 1, \ldots, d - z - r'\}.$$

Under erasure pattern E_j^z, 1) the information symbols $a[1], \ldots, a[d-2z]$, which include $a[1], \ldots, a[r']$, are unerased; 2) the information symbols on the second last row, i.e., $a[d-z-r'+1], \ldots, a[d-z]$, are erased; and 3) the parity symbols $b[z-r'+1], \ldots, b[z]$ are unerased.

For each $i \in \{1, \ldots, z - r'\}$, if the information symbol $a[d-2z+i]$ is erased, then 1) the parity symbols $b[i], \ldots, b[z]$ are unerased; 2) the information symbol $a[d-z-r'+r_{i,r'}]$ can therefore be recovered by time step $d-z+i$ using the unerased parity symbol $b[i]$ and the unerased and recovered information symbols in the column; and 3) the information symbol $a[d-2z+i]$ can subsequently be recovered by time step $d - z + r' + i$ using the unerased parity symbol $b[r'+i]$, the unerased actual information symbols in the column, and the recovered virtual information symbol $a[d-z-r'+r_{i,r'}]$. It follows that for each $i \in \{1, \ldots, z\}$, the information symbols $a[1], \ldots, a[d-2z-r'+i]$, which include $a[i]$, can be recovered by time step $d - z + i$.

For each $i \in \{1, \ldots, r'\}$, the information symbol $a[d-z-r'+i]$ can be recovered by time step $d - r' + i$ using the unerased parity symbol $b[z-r'+i]$ and the unerased and recovered information symbols in the column.

Case 2.3: Consider the case of $r' < z$, with the erasure pattern E_j^z, where

$$j \in \{d - z - r' + 1, \ldots, d - z\}.$$

Under erasure pattern E_j^z, the information symbols $a[1], \ldots, a[d-z-r']$, which include $a[1], \ldots, a[z]$, are unerased.

For each $i \in \{1, \ldots, r'\}$, if the information symbol $a[d-z-r'+i]$ is erased, then 1) the parity symbols $b[z-r'+i], \ldots, b[z]$ are unerased; and 2) the information symbol $a[d-z-r'+i]$ can therefore be recovered by time step $d - r' + i$ using the unerased parity symbol $b[z-r'+i]$ and the unerased information symbols in the column.

Hence, under any erasure pattern $E^{\mathsf{z}} \in \mathcal{E}^{\mathsf{z}}$, all the information symbols $a[1], \ldots, a[d-z]$ are decodable by the last time step in interval L; in particular, the information symbol $a[i]$ is decodable by the $(d-z+i)$th time step in interval L, for each $i \in \{1, \ldots, z\}$. Since $z \geq c$, it follows that the symbol decoding requirements given by D1 and D2 in Lemma 4.8 are satisfied by $\mathcal{C}$. Therefore, according to Lemma 4.8, the derived code achieves a message size of $\frac{d-z}{d}c$.

The rest of the proof leading to the attainment of (4.18) is the same as that of Theorem 4.9. ■

4.8.9 Proof of Theorem 4.12

Our proof technique expands that of Theorem 4.10. First, we arrange the d symbols of the codeword vector produced by the stated systematic block code $\mathcal{C}$ sequentially across z' columns, with r' information symbols on the $\left(\frac{d-z-r'}{z'}+1\right)$th row, and all the nondegenerate parity symbols $b[1], \ldots, b[z']$ on a separate row, followed by the degenerate parity symbols, as shown in Figure 4.7. For the case of $r' < z'$, we repeat the r' information symbols on the $\left(\frac{d-z-r'}{z'}+1\right)$th row, i.e., $a[d-z-r'+1], \ldots, a[d-z]$, across the row; these repeated *virtual* information symbols are parenthesized to distinguish them from the original *actual* information symbols of the codeword vector. Note that each column $i \in \{1, \ldots, z'\}$ of the table contains exactly $\frac{d-z-r'}{z'}+1 \geq 2$ actual and virtual information symbols. For each $i \in \{1, \ldots, z'\}$, the value of the nondegenerate parity symbol $b[i]$ is given by the bit-wise modulo-2 sum (i.e., exclusive-or) of the actual and virtual information symbols above it in column i of the table.

Suppose that the d symbols of the codeword vector are transmitted sequentially across an erasure link, one symbol per time step, over the time interval $L \triangleq \{1, \ldots, d\}$. To show that the symbol decoding requirements given by D1 and D2 in Lemma 4.8 are satisfied by $\mathcal{C}$, we consider the following six exhaustive cases separately:

Case 1: Consider the case of $r' = z'$, for which there are no virtual information symbols. Under each erasure pattern $E^{\mathsf{z}} \in \mathcal{E}^{\mathsf{z}}$ (as defined in Lemma 4.8), exactly $\frac{d-z}{z'}$ symbols in each column of the table are unerased. Since the $d-z$ unerased symbols in the codeword vector are consecutive (possibly wrapping around symbols $b[z]$ and $a[1]$), it follows that the $\frac{d-z}{z'}$ unerased symbols in each column of the table are on consecutive rows (possibly wrapping around the last and first rows).

For each $i \in \{1, \ldots, z'\}$, let S_i be the set of indices corresponding to the erased information symbols in column i of the table. If the nondegenerate parity symbol $b[i]$ is erased, then because

the degenerate parity symbols take on the values of information symbols in a periodic manner, each information symbol $a[k]$, where $k \in S_i$, can be recovered by time step $d-(d-z)+k$ using a matching unerased degenerate parity symbol with a value of $a[k]$. On the other hand, if $b[i]$ is unerased, then 1) let $\sigma_i \triangleq \max\{k : k \in S_i\}$; 2) each information symbol $a[k]$, where $k \in S_i \backslash \{\sigma_i\}$, can be recovered by time step $d-z+z'+k$ using the unerased degenerate parity symbol $b[z'+k]$, which has a value of $a[k]$; and 3) the remaining information symbol $a[\sigma_i]$ can be recovered by time step $d-z+\sigma_i$ using the unerased nondegenerate parity symbol $b[i]$ and the unerased and recovered information symbols in the column.

Case 2.1: Consider the case of $r' < z'$, with the erasure pattern E_j^z, where

$$j \in \{1, \ldots, d-z\} \cup \{d-z+z'+d-z+1, \ldots, d\}.$$

Under erasure pattern E_j^z, all the nondegenerate parity symbols $b[1], \ldots, b[z']$ are erased. Each of the $d-z$ unerased symbols is therefore either an information symbol or a degenerate parity symbol (which is a copy of an information symbol). Because the degenerate parity symbols take on the values of information symbols in a periodic manner, all the information symbols $a[1], \ldots, a[d-z]$ can be recovered using the unerased symbols. In particular, for each $i \in \{1, \ldots, d-z\}$, the information symbol $a[i]$ can be recovered by time step $d-(d-z)+i$.

Case 2.2: Consider the case of $r' < z'$, with the erasure pattern E_j^z, where

$$j \in \{d-z+1, \ldots, d-z+d-z-r'\}.$$

Under erasure pattern E_j^z, all the information symbols on the $\left(\frac{d-z-r'}{z'}+1\right)$th row, i.e., $a[d-z-r'+1], \ldots, a[d-z]$, are unerased.

For each $i \in \{1, \ldots, z'\}$, let S_i be the set of indices corresponding to the erased information symbols in column i of the table. If $|S_i| = 0$, then all information symbols in the column are unerased. If $|S_i| \geq 1$, then 1) let $\sigma_i \triangleq \max\{k : k \in S_i\}$; 2) each information symbol $a[k]$, where $k \in S_i \backslash \{\sigma_i\}$, can be recovered by time step $d-z+z'+k$ using the unerased degenerate parity symbol $b[z'+k]$, which has a value of $a[k]$; and 3) the remaining information symbol $a[\sigma_i]$ can be recovered by time step $d-z+\sigma_i$ using the unerased nondegenerate parity symbol $b[i]$ and the unerased and recovered information symbols in the column.

Case 2.3: Consider the case of $r' < z'$, with the erasure pattern E_j^{z}, where

$$j \in \{d-z+d-z-r'+1, \ldots, d-z+d-z\}.$$

Under erasure pattern E_j^{z}, 1) the information symbols $a[1], \ldots, a[d-z-r']$ are erased; 2) all the nondegenerate parity symbols $b[1], \ldots, b[z']$ are unerased; and 3) the degenerate parity symbols $b[z'+1], \ldots, b[d-z-r']$ are unerased.

For each $i \in \{1, \ldots, d-z-r'-z'\}$, the information symbol $a[i]$ can be recovered by time step $d-z+z'+i$ using the unerased degenerate parity symbol $b[z'+i]$, which has a value of $a[i]$.

For each $i \in \{1, \ldots, r'\}$, if the degenerate parity symbol $b[d-z-r'+i]$, which has a value of $a[d-z-r'-z'+i]$, is unerased, then 1) the information symbol $a[d-z-r'-z'+i]$ can be recovered by time step $d-z+d-z-r'+i$ using it; and 2) the information symbol $a[d-z-r'+i]$ can subsequently be recovered by time step $d-z+d-z-r'+i$ using the unerased nondegenerate parity symbol $b[i]$ and the recovered information symbols in the column. On the other hand, if $b[d-z-r'+i]$ is erased, then 1) the information symbols $a[d-z-r'+i], \ldots, a[d-z]$ are unerased; and 2) the information symbol $a[d-z-r'-z'+i]$ can therefore be recovered by time step $d-z+d-z-r'-z'+i$ using the unerased nondegenerate parity symbol $b[i]$ and the unerased and recovered information symbols in the column. It follows that all the information symbols on the $\left(\frac{d-z-r'}{z'}+1\right)$th row, i.e., $a[d-z-r'+1], \ldots, a[d-z]$, can be recovered by time step $d-z+d-z$.

For each $i \in \{1, \ldots, z'-r'\}$, the information symbol $a[d-z-z'+i]$ can be recovered by time step $d-z+d-z$ using the unerased nondegenerate parity symbol $b[r'+i]$ and the recovered information symbols in the column.

Case 2.4: Consider the case of $r' < z'$, with the erasure pattern E_j^{z}, where

$$j \in \{d-z+d-z+1, \ldots, d-z+d-z+z'-r'\}.$$

Under erasure pattern E_j^{z}, 1) all the information symbols $a[1], \ldots, a[d-z]$ are erased; 2) the nondegenerate parity symbols $b[z'-r'+1], \ldots, b[z']$ are unerased; and 3) the degenerate parity symbols $b[z'+1], \ldots, b[d-z]$ are unerased.

For each $i \in \{1, \ldots, d-z-z'\}$, the information symbol $a[i]$ can be recovered by time step $d-z+z'+i$ using the unerased degenerate parity symbol $b[z'+i]$, which has a value of

$a[i]$.

For each $i \in \{1, \ldots, z' - r'\}$, if the degenerate parity symbol $b[d-z+i]$, which has a value of $a[d-z-z'+i]$, is unerased, then the information symbol $a[d-z-z'+i]$ can be recovered by time step $d - z + d - z + i$ using it. On the other hand, if $b[d-z+i]$ is erased, then 1) the nondegenerate parity symbols $b[i], \ldots, b[z']$ are unerased; 2) the information symbol $a[d-z-r'+r_{i,r'}]$ can therefore be recovered by time step $d - z + d - z - r' + i$ using the unerased nondegenerate parity symbol $b[i]$ and the recovered information symbols in the column; and 3) the information symbol $a[d-z-z'+i]$ can subsequently be recovered by time step $d - z + d - z - r' + i$ using the unerased parity symbol $b[r'+i]$ and the recovered information symbols in the column.

For each $i \in \{1, \ldots, r'\}$, the information symbol $a[d-z-r'+i]$ can be recovered by time step $d - z + z' + d - z - 2r' + i$ using the unerased nondegenerate parity symbol $b[z'-r'+i]$ and the recovered information symbols in the column.

Case 2.5: Consider the case of $r' < z'$, with the erasure pattern E^{z}_j, where

$$j \in \{d - z + z' + d - z - r' + 1, \ldots, d - z + z' + d - z\}.$$

Under erasure pattern E^{z}_j, 1) all the information symbols $a[1], \ldots, a[d-z]$ are erased; and 2) the degenerate parity symbols $b[z'+1], \ldots, b[z'+d-z-r']$ are unerased.

For each $i \in \{1, \ldots, d - z - r'\}$, the information symbol $a[i]$ can be recovered by time step $d - z + z' + i$ using the unerased degenerate parity symbol $b[z'+i]$, which has a value of $a[i]$.

For each $i \in \{1, \ldots, r'\}$, if the degenerate parity symbol $b[z'+d-z-r'+i]$, which has a value of $a[d-z-r'+i]$, is unerased, then the information symbol $a[d-z-r'+i]$ can be recovered by time step $d - z + z' + d - z - r' + i$ using it. On the other hand, if $b[z'+d-z-r'+i]$ is erased, then 1) the nondegenerate parity symbols $b[z'-r'+i], \ldots, b[z']$ are unerased; and 2) the information symbol $a[d-z-r'+i]$ can therefore be recovered by time step $d - z + z' + d - z - 2r' + i$ using the unerased nondegenerate parity symbol $b[z'-r'+i]$ and the recovered information symbols in the column.

Hence, under any erasure pattern $E^{\mathsf{z}} \in \mathcal{E}^{\mathsf{z}}$, the information symbol $a[i]$ is decodable by the $\big(d - (d - z) + i\big)$th time step in interval L, for each $i \in \{1, \ldots, d - z\}$. Since $d - z \geq c$, it fol-

lows that the symbol decoding requirements given by D1 and D2 in Lemma 4.8 are satisfied by $\mathcal{C}$. Therefore, according to Lemma 4.8, the derived code achieves a message size of $\frac{d-z}{d}c$.

The rest of the proof leading to the attainment of (4.18) is the same as that of Theorem 4.9. ■

4.8.10 Proof of Theorem 4.14

By partitioning the set of unerased time steps $U_k \subseteq T_k$ into two sets $U_k^{(1)} \subseteq T_k \backslash W_k$ (i.e., unerased time steps before the coding window W_k) and $U_k^{(2)} \subseteq W_k$ (i.e., unerased time steps in the coding window W_k), we can rewrite (4.1) as follows:

$$\mathbb{P}\left[S_k\right] = \sum_{U_k^{(1)} \subseteq T_k \backslash W_k} \sum_{z=0}^{d} \sum_{\substack{U_k^{(2)} \subseteq W_k: \\ |U_k^{(2)}| = d-z}} \mathbb{1}\left[H\big(M_k \,\big|\, X[U_k^{(1)}], X[U_k^{(2)}]\big) = 0\right] \cdot (1-p_{\mathsf{e}})^{|U_k^{(1)}|+d-z} (p_{\mathsf{e}})^{|T_k|-d-|U_k^{(1)}|+z}. \tag{4.19}$$

Observe that the conditional entropy term appearing in (4.19) can be lower-bounded as follows:

$$\begin{aligned} & H\big(M_k \,\big|\, X[U_k^{(1)}], X[U_k^{(2)}]\big) \\ & \overset{\text{(a)}}{\geq} H\big(M_k \,\big|\, M_1^{k-1}, X[T_k \backslash W_k], X[U_k^{(2)}]\big) \\ & \overset{\text{(b)}}{=} H\big(M_k \,\big|\, M_1^{k-1}, X[U_k^{(2)}]\big) \\ & \overset{\text{(c)}}{=} H\big(M_k \,\big|\, M_{k-m_{\mathsf{E}}+1}^{k-1}, X[U_k^{(2)}]\big), \end{aligned} \tag{4.20}$$

where

(a) follows from the addition of conditioned random variables M_1^{k-1}, $X[(T_k \backslash W_k) \backslash U_k^{(1)}]$;

(b) follows from the fact that packets $X[T_k \backslash W_k]$ are functions of messages M_1^{k-1};

(c) follows from the fact that messages are independent, and packets $X[U_k^{(2)}]$ are independent of messages $M_1^{k-m_{\mathsf{E}}}$ (we can show this explicitly by considering the conditional mutual information

$$\begin{aligned} & I\big(M_k; M_1^{k-m_{\mathsf{E}}} \,\big|\, M_{k-m_{\mathsf{E}}+1}^{k-1}, X[U_k^{(2)}]\big) \\ & = H\big(M_k \,\big|\, M_{k-m_{\mathsf{E}}+1}^{k-1}, X[U_k^{(2)}]\big) - H\big(M_k \,\big|\, M_1^{k-1}, X[U_k^{(2)}]\big) \end{aligned}$$

$$= H\big(M_1^{k-m_{\mathsf{E}}} \,\big|\, M_{k-m_{\mathsf{E}}+1}^{k-1}, X[U_k^{(2)}]\big) - H\big(M_1^{k-m_{\mathsf{E}}} \,\big|\, M_{k-m_{\mathsf{E}}+1}^{k}, X[U_k^{(2)}]\big),$$

where both conditional entropy terms on the third line are equal to $H\big(M_1^{k-m_{\mathsf{E}}}\big)$, which implies that both conditional entropy terms on the second line are equal).

As a consequence of (4.20), we have

$$\mathbb{1}\left[H\big(M_k \,\big|\, X[U_k^{(1)}], X[U_k^{(2)}]\big) = 0\right] \le \mathbb{1}\left[H\big(M_k \,\big|\, M_{k-m_{\mathsf{E}}+1}^{k-1}, X[U_k^{(2)}]\big) = 0\right],$$

and therefore (4.19) can be upper-bounded as follows:

$$\begin{aligned}
\mathbb{P}\left[S_k\right] &\le \sum_{U_k^{(1)} \subseteq T_k \setminus W_k} \sum_{z=0}^{d} \sum_{\substack{U_k^{(2)} \subseteq W_k: \\ |U_k^{(2)}| = d-z}} \mathbb{1}\left[H\big(M_k \,\big|\, M_{k-m_{\mathsf{E}}+1}^{k-1}, X[U_k^{(2)}]\big) = 0\right] \cdot (1-p_{\mathsf{e}})^{|U_k^{(1)}|+d-z} (p_{\mathsf{e}})^{|T_k|-d-|U_k^{(1)}|+z} \\
&\overset{\text{(a)}}{=} \sum_{z=0}^{d} \sum_{\substack{U_k^{(2)} \subseteq W_k: \\ |U_k^{(2)}| = d-z}} \mathbb{1}\left[H\big(M_k \,\big|\, M_{k-m_{\mathsf{E}}+1}^{k-1}, X[U_k^{(2)}]\big) = 0\right] \cdot (1-p_{\mathsf{e}})^{d-z} (p_{\mathsf{e}})^{z} \\
&= \sum_{z=0}^{d} \alpha_k(z) \cdot (1-p_{\mathsf{e}})^{d-z} (p_{\mathsf{e}})^{z}, \qquad (4.21)
\end{aligned}$$

where

$$\alpha_k(z) \triangleq \sum_{\substack{U_k^{(2)} \subseteq W_k: \\ |U_k^{(2)}| = d-z}} \mathbb{1}\left[H\big(M_k \,\big|\, M_{k-m_{\mathsf{E}}+1}^{k-1}, X[U_k^{(2)}]\big) = 0\right],$$

and (a) follows from a reordering of the sums, and the removal of the factor

$$\sum_{U_k^{(1)} \subseteq T_k \setminus W_k} (1-p_{\mathsf{e}})^{|U_k^{(1)}|} \, (p_{\mathsf{e}})^{|T_k|-d-|U_k^{(1)}|} = 1.$$

Consider a fixed choice of subset $U \subseteq \{1, \ldots, d\}$. Suppose that $U_k^{(2)} \subseteq W_k$ is the appropriately time-shifted version of U, i.e.,

$$U_k^{(2)} = \big\{(k-1)c + i : i \in U\big\}.$$

According to the definition of time-invariant codes, the packets $X[U_k^{(2)}]$ can consequently be written in terms of U as

$$X[U_k^{(2)}] = \Big(f_{r_{i,c}}\big(M_{k+q_{i,c}}, \ldots, M_{k+q_{i,c}-m_{\mathsf{E}}+1}\big)\Big)_{i\in U}.$$

The conditional entropy term in the definition of $\alpha_k(z)$ can therefore be written in terms of the message random variables as

$$H\big(M_k \,\big|\, M_{k-m_{\mathsf{E}}+1}^{k-1}, X[U_k^{(2)}]\big) = H\Big(M_k\Big|M_{k-m_{\mathsf{E}}+1}^{k-1}, \Big(f_{r_{i,c}}\big(M_{k+q_{i,c}}, \ldots, M_{k+q_{i,c}-m_{\mathsf{E}}+1}\big)\Big)_{i\in U}\Big).$$

Since the joint probability distribution of the random variables in this expression is the same for any $k \geq m_{\mathsf{E}}$, it follows that this conditional entropy term is constant wrt $k \geq m_{\mathsf{E}}$. Defining $\alpha(z) \triangleq \alpha_{m_{\mathsf{E}}}(z)$, we therefore have

$$\alpha(z) = \alpha_{m_{\mathsf{E}}}(z) = \alpha_{m_{\mathsf{E}}+1}(z) = \alpha_{m_{\mathsf{E}}+2}(z) = \cdots \tag{4.22}$$

for any $z \in \{0, \ldots, d\}$. To obtain the required upper bound (4.2), we will show that for any $z \in \{0, \ldots, d\}$,

$$\alpha(z) \leq \left\lfloor \min\left(\frac{(d-z)c}{d\,s}, 1\right)\binom{d}{z}\right\rfloor. \tag{4.23}$$

Suppose that $z \in \{0, \ldots, d\}$. Consider the first $m_{\mathsf{E}} + n - 1$ messages $\{1, \ldots, m_{\mathsf{E}} + n - 1\}$, and the union of their (overlapping) coding windows $T_{m_{\mathsf{E}}+n-1}$, where $n \in \mathbb{Z}^+$. Let $\tilde{\mathcal{E}}_z$ be the collection of all $\binom{d}{z}$ possible subsets $\tilde{E} \subseteq \{1, \ldots, d\}$ of size z, i.e.,

$$\tilde{\mathcal{E}}_z \triangleq \big\{\tilde{E} \subseteq \{1, \ldots, d\} : |\tilde{E}| = z\big\}.$$

From each $\tilde{E} \in \tilde{\mathcal{E}}_z$, we derive a periodic erasure pattern $E \subseteq T_{m_{\mathsf{E}}+n-1}$ by concatenating copies of $\tilde{E}$; let $\mathcal{E}_z$ be the set of these $\binom{d}{z}$ erasure patterns, i.e.,

$$\mathcal{E}_z \triangleq \Big\{\big\{(j-1)d + i \in T_{m_{\mathsf{E}}+n-1} : j \in \mathbb{Z}^+, i \in \tilde{E}\big\} : \tilde{E} \in \tilde{\mathcal{E}}_z\Big\}.$$

Note that because of the periodicity of each erasure pattern $E \in \mathcal{E}_z$, there are exactly z erased time steps and therefore exactly $d - z$ unerased time steps in each coding window W_k, i.e.,

$$\big|W_k \backslash E\big| = d - z \quad \forall\, k \in \{1, \ldots, m_{\mathsf{E}} + n - 1\}, E \in \mathcal{E}_z. \tag{4.24}$$

Furthermore, for a fixed choice of $k \in \{1, \ldots, m_{\mathsf{E}} + n - 1\}$, the set of $d - z$ unerased time steps in the coding window for message k, i.e., $W_k \backslash E$, is distinct under each erasure pattern $E \in \mathcal{E}_z$; in other words,

$$(E_1, E_2 \in \mathcal{E}_z) \wedge (W_k \backslash E_1 = W_k \backslash E_2) \Longrightarrow E_1 = E_2 \quad \forall\, k \in \{1, \ldots, m_{\mathsf{E}} + n - 1\}. \tag{4.25}$$

Suppose that $k \in \{1, \ldots, m_{\mathsf{E}} + n - 1\}$ and $E \in \mathcal{E}_z$. From the definition of conditional mutual information, we have

$$\begin{aligned} &I\big(M_k\,;\, X[W_k \backslash E] \,\big|\, M_1^{k-1}\big) \\ &= H\big(M_k \,\big|\, M_1^{k-1}\big) - H\big(M_k \,\big|\, M_1^{k-1}, X[W_k \backslash E]\big) \\ &= H\big(X[W_k \backslash E] \,\big|\, M_1^{k-1}\big) - H\big(X[W_k \backslash E] \,\big|\, M_1^{k}\big). \end{aligned}$$

Rearranging terms produces

$$H\big(X[W_k \backslash E] \,\big|\, M_1^{k}\big) = H\big(X[W_k \backslash E] \,\big|\, M_1^{k-1}\big) - H\big(M_k \,\big|\, M_1^{k-1}\big) + H\big(M_k \,\big|\, M_1^{k-1}, X[W_k \backslash E]\big). \tag{4.26}$$

Since messages are independent, we have

$$H\big(M_k \,\big|\, M_1^{k-1}\big) = H\big(M_k\big) = s. \tag{4.27}$$

Now, if

$$H\big(M_k \,\big|\, M_1^{k-1}, X[W_k \backslash E]\big) = 0, \tag{4.28}$$

then, by substituting (4.27) and (4.28) into (4.26), we obtain

$$H\big(X[W_k\backslash E]\,\big|\,M_1^k\big) = H\big(X[W_k\backslash E]\,\big|\,M_1^{k-1}\big) - s. \tag{4.29}$$

On the other hand, if condition (4.28) is not satisfied, then we have the inequality

$$H\big(X[W_k\backslash E]\,\big|\,M_1^k\big) \leq H\big(X[W_k\backslash E]\,\big|\,M_1^{k-1}\big), \tag{4.30}$$

which is always true.

Suppose that $k \in \{1, \ldots, m_{\mathsf{E}} + n - 1\}$. According to the definition of $\alpha_k(z)$, there are $\alpha_k(z)$ subsets $U_k^{(2)} \subseteq W_k$ of size $d - z$ for which

$$H\big(M_k\,\big|\,M_{k-m_{\mathsf{E}}+1}^{k-1}, X[U_k^{(2)}]\big) = 0.$$

Equivalently, it follows from properties (4.24) and (4.25) of the set of erasure patterns $\mathcal{E}_z$ that there are $\alpha_k(z)$ erasure patterns $E \in \mathcal{E}_z$ for which

$$H\big(M_k\,\big|\,M_{k-m_{\mathsf{E}}+1}^{k-1}, X[W_k\backslash E]\big) = 0.$$

Now, since

$$H\big(M_k\big|M_1^{k-1}, X[W_k\backslash E]\big) \leq H\big(M_k\big|M_{k-m_{\mathsf{E}}+1}^{k-1}, X[W_k\backslash E]\big),$$

there are therefore at least $\alpha_k(z)$ erasure patterns $E \in \mathcal{E}_z$ for which condition (4.28) is satisfied. Summing over all erasure patterns and applying (4.29) and (4.30) the appropriate number of times produces the following inequality:

$$\sum_{E\in\mathcal{E}_z} H\big(X[W_k\backslash E]\,\big|\,M_1^k\big) \leq \left(\sum_{E\in\mathcal{E}_z} H\big(X[W_k\backslash E]\,\big|\,M_1^{k-1}\big)\right) - s\cdot\alpha_k(z). \tag{4.31}$$

We now proceed to prove by induction that the following inequality holds for any $k \in \{m_{\mathsf{E}}, \ldots, m_{\mathsf{E}} + n - 1\}$:

$$\sum_{E\in\mathcal{E}_z} H\big(X[W_k\backslash E]\,\big|\,M_1^k\big) \leq \left(\sum_{E\in\mathcal{E}_z} \big|T_k\backslash E\big|\right) - (k - m_{\mathsf{E}} + 1)s\cdot\alpha(z). \tag{4.32}$$

(Base case) Consider the case of $k = m_{\mathsf{E}}$. According to (4.31), we have

$$\begin{aligned}
&\sum_{E\in\mathcal{E}_z} H\big(X[W_{m_{\mathsf{E}}}\backslash E] \,\big|\, M_1^{m_{\mathsf{E}}}\big) \\
&\le \bigg(\sum_{E\in\mathcal{E}_z} H\big(X[W_{m_{\mathsf{E}}}\backslash E] \,\big|\, M_1^{m_{\mathsf{E}}-1}\big)\bigg) - s\cdot\alpha_{m_{\mathsf{E}}}(z) \\
&\overset{\text{(a)}}{\le} \bigg(\sum_{E\in\mathcal{E}_z} H\big(X[W_{m_{\mathsf{E}}}\backslash E]\big)\bigg) - s\cdot\alpha(z) \\
&\overset{\text{(b)}}{\le} \bigg(\sum_{E\in\mathcal{E}_z} \big|W_{m_{\mathsf{E}}}\backslash E\big|\bigg) - s\cdot\alpha(z) \\
&\overset{\text{(c)}}{\le} \bigg(\sum_{E\in\mathcal{E}_z} \big|T_{m_{\mathsf{E}}}\backslash E\big|\bigg) - s\cdot\alpha(z),
\end{aligned}$$

as required, where

(a) follows from the removal of conditioned random variables $M_1^{m_{\mathsf{E}}-1}$ in the entropy term, and the application of (4.22);

(b) follows from the fact that $H(X_t) \le 1$ for any t because of the unit packet size;

(c) follows from the fact that $W_{m_{\mathsf{E}}} \subseteq T_{m_{\mathsf{E}}}$.

(Inductive step) Suppose that (4.32) holds for some $k \in \{m_{\mathsf{E}}, \ldots, m_{\mathsf{E}} + n - 2\}$. According to (4.31), we have

$$\begin{aligned}
&\sum_{E\in\mathcal{E}_z} H\big(X[W_{k+1}\backslash E] \,\big|\, M_1^{k+1}\big) \\
&\le \bigg(\sum_{E\in\mathcal{E}_z} H\big(X[W_{k+1}\backslash E] \,\big|\, M_1^{k}\big)\bigg) - s\cdot\alpha_{k+1}(z) \\
&\overset{\text{(a)}}{\le} \bigg(\sum_{E\in\mathcal{E}_z} H\big(X[(W_k\backslash E)\cup(W_{k+1}\backslash E)] \,\big|\, M_1^{k}\big)\bigg) - s\cdot\alpha(z) \\
&\overset{\text{(b)}}{\le} \bigg(\sum_{E\in\mathcal{E}_z} H\big(X[W_k\backslash E] \,\big|\, M_1^{k}\big)\bigg) + \bigg(\sum_{E\in\mathcal{E}_z} H\Big(X\big[(W_{k+1}\backslash E)\backslash(W_k\backslash E)\big]\Big)\bigg) - s\cdot\alpha(z) \\
&\overset{\text{(c)}}{\le} \bigg(\sum_{E\in\mathcal{E}_z} \big|T_k\backslash E\big|\bigg) - (k - m_{\mathsf{E}} + 1)s\cdot\alpha(z) + \bigg(\sum_{E\in\mathcal{E}_z} \big|(W_{k+1}\backslash E)\backslash(W_k\backslash E)\big|\bigg) - s\cdot\alpha(z) \\
&\overset{\text{(d)}}{=} \bigg(\sum_{E\in\mathcal{E}_z} \big|T_{k+1}\backslash E\big|\bigg) - (k - m_{\mathsf{E}} + 2)s\cdot\alpha(z),
\end{aligned}$$

as required, where

(a) follows from the addition of random variables $X[W_k \backslash E]$ in the entropy term, and the application of (4.22);

(b) follows from the chain rule for joint entropy, and the removal of conditioned random variables $X[W_k \backslash E]$, M_1^k in the second entropy term;

(c) follows from the inductive hypothesis, and the fact that $H(X_t) \leq 1$ for any t because of the unit packet size;

(d) follows from the fact that

$$\big|T_k \backslash E\big| + \big|(W_{k+1} \backslash E) \backslash (W_k \backslash E)\big| \;=\; \big|T_k \backslash E\big| + \big|(T_{k+1} \backslash T_k) \backslash E\big| \;=\; \big|T_{k+1} \backslash E\big|.$$

Now, since the conditional entropy term in (4.32) is nonnegative, it follows that for the choice of $k = m_{\mathsf{E}} + n - 1$, we have

$$0 \leq \left(\sum_{E \in \mathcal{E}_z} \big|T_{m_{\mathsf{E}}+n-1} \backslash E\big| \right) - n\, s \cdot \alpha(z),$$

which implies

$$\begin{aligned} \alpha(z) &\leq \frac{1}{n\, s} \sum_{E \in \mathcal{E}_z} \big|T_{m_{\mathsf{E}}+n-1} \backslash E\big| \\ &\leq \frac{1}{n\, s} \binom{d}{z} \left(\frac{(m_{\mathsf{E}} + n - 2)c + d}{d} + 1 \right) (d - z) \\ &= \frac{d - z}{d\, s} \binom{d}{z} \left(c + \frac{m_{\mathsf{E}} c - 2c + 2d}{n} \right). \end{aligned}$$

Furthermore, since $\alpha(z)$ is independent of n, this upper bound must also hold in the limit $n \to \infty$, i.e.,

$$\alpha(z) \leq \frac{(d - z)c}{d\, s} \binom{d}{z}.$$

Finally, taking into account the fact that $\alpha(z)$ is an integer that is at most $\binom{d}{z}$, we arrive at (4.23). Applying (4.22) and (4.23) to (4.21) produces the required upper bound (4.2) on $\mathbb{P}\left[S_k\right]$ for $k \geq m_{\mathsf{E}}$.

■

4.9 Acknowledgment

The author and his coauthors A. Qureshi and T. Ho would like to thank Yury Polyanskiy for sharing his work with them. They also thank Ashish Khisti for the interesting discussions.

Chapter 5

Routing-Caching for Named Data Networking

5.1 Introduction

Named data networking (NDN), or content-centric networking (CCN), is a proposed network architecture for the Internet that replaces the traditional client-server model of communications with one based on the identity of data or content [18]. This abstraction more accurately reflects how the Internet is primarily used today: instead of being concerned about communicating with specific nodes, end users are mainly interested in obtaining the data they want. Jacobson *et al.* [18] refer to this approach as replacing *where* with *what*.

Content delivery in an NDN is accomplished using two types of packets, and specific data structures in nodes. Requests for data objects by end users lead to the creation of *Interest* packets (IPs), which are forwarded along routes determined by the *Forwarding Information Base* (FIB) at each node. The FIB tells the node which neighbor node(s) to transmit each IP to. Received IPs are recorded in the *Pending Interest Table* (PIT) at each node, thus allowing repeated requests for the same object to be suppressed. When a node receives an IP that it can fulfill, it creates a *Data* packet (DP) containing the requested data object. The DP is subsequently transmitted back along the path taken by the corresponding IP, as recorded by the PIT at each node. Nodes may optionally cache the data objects contained in received DPs. Consequently, a request for a data object can be fulfilled not only by the origin server but also by any node with a copy of that object in its cache.

Assuming the prevalence of caches, the usual approaches to routing and caching that treat the two as separate tasks may no longer be effective for NDN. For best performance, the routing and caching policies should work in concert when responding to dynamic changes in end user request patterns and in network topology. To address the joint problems of routing and caching for NDN, we propose a routing-caching policy based on the backpressure algorithm [19, 20]. The policy applies the backpressure algorithm to *Virtual Interest* packets (VIPs), and makes routing and caching decisions according to local statistics collected on these VIPs.

We begin with a formal description of the network model in Section 5.2, and present our backpressure-based routing-caching policy in Section 5.3. The performance of the proposed policy is evaluated against a basic protocol using a packet-level simulation in Section 5.4.

5.2 Network Model

Assume that each IP has a normalized size of one, and that the size of each DP is equal to the size of the object contained in it. For brevity, we adopt the following notation to describe various aspects of the network.

Nodes: Each node v in the network has the following properties: $\texttt{NodeName}(v)$ is the unique identifier or name of the node; $\texttt{CacheSize}(v)$ is the size of the cache in the node. Let $\texttt{LinkBandwidth}(v_1, v_2)$ be the bandwidth (e.g., in bits per second) of the directed link from node v_1 to node v_2.

Objects: Each object w in the network has the following properties: $\texttt{ObjectName}(w)$ is the unique identifier or name of the object; $\texttt{ObjectSize}(w)$ is the size of the object; $\texttt{OriginServer}(w)$ is the unique node that is the origin server for the object.

Requests: Each request u, which is created by an end user for an object, has the following properties: $\texttt{Node}(u)$ is the node that is collocated with the requesting end user; $\texttt{Object}(u)$ is the object being requested; $\texttt{TimeCreated}(u)$ is the time at which the request is created; $\texttt{TimeFulfilled}(u)$ is the time at which the request is fulfilled, i.e., the earliest time $t \geq \texttt{TimeCreated}(u)$ at which $\texttt{Node}(u)$ receives a DP containing $\texttt{Object}(u)$. For a newly created request u, we initialize $\texttt{TimeFulfilled}(u) \leftarrow \infty$. Let $\texttt{PendingRequestTable}(v, w)$ be the set of unfulfilled requests u that are created by end users at node v for object w.

Forwarding Information Base (FIB): The FIB data structure is represented by the function $\texttt{ForwardingNodes}(v, w)$, which gives the subset of neighbor nodes of node v to which node v may transmit IPs for object w.

Pending Interest Table (PIT): The PIT data structure is represented by the function $\texttt{RequestingNodes}(v, w)$, which gives the set of nodes that have directly requested from node v the object w. Each requesting node $v' \in \texttt{RequestingNodes}(v, w)$ is either a neighbor node of node v (that has transmitted to node v an IP for object w), or node v itself (because a collocated end user has created a request for object w).

Caches: Let $\texttt{CachedObjects}(v)$ be the set of objects that are cached in node v; because of the limited cache space, we have

$$\sum_{w \in \texttt{CachedObjects}(v)} \texttt{ObjectSize}(w) \leq \texttt{CacheSize}(v)$$

at all times.

A *cache hit* occurs when a node receives an IP for an object that is currently in its cache. Each cache hit h has the following properties: $\texttt{Node}(h)$ is the node in which the cache hit occurs; $\texttt{Object}(h)$ is the cached object $w \in \texttt{CachedObjects}\big(\texttt{Node}(h)\big)$ being requested; $\texttt{Time}(h)$ is the time at which the cache hit occurs.

A *cache eviction* occurs when an object is removed from the cache in a node. Each cache eviction e has the following properties: $\texttt{Node}(e)$ is the node in which the cache eviction occurs; $\texttt{Object}(e)$ is the cached object $w \in \texttt{CachedObjects}\big(\texttt{Node}(e)\big)$ being evicted; $\texttt{Time}(e)$ is the time at which the cache eviction occurs.

5.2.1 Creation of Requests

Suppose that an end user at node v creates a request u for object w at time t.

The node adds the request to the corresponding set of unfulfilled requests, i.e.,

$$\texttt{PendingRequestTable}(v, w) \leftarrow \texttt{PendingRequestTable}(v, w) \cup \{u\}.$$

If $\texttt{PendingRequestTable}(v, w)$ was previously empty, then the node creates an IP for object w, and

transmits it to itself (see Section 5.2.2 for subsequent actions taken by the node); otherwise, the creation of an IP is suppressed because an IP for the same object has already been created earlier and the node is currently waiting for the arrival of the corresponding DP.

5.2.2 Handling of Interest Packets (IPs) and Routing

Suppose that node v receives an IP for object w from a requesting node v' at time t. Note that the requesting node v' can be a neighbor node of node v, or node v itself. Node v does one of three things depending on the availability of the requested object.

First, if the node is the origin server for the requested object, i.e., $\texttt{OriginServer}(w) = v$, then it creates a DP containing object w and transmits it to the requesting node v'.

Second, if the requested object is currently cached in the node, i.e., $w \in \texttt{CachedObjects}(v)$, then a cache hit is recorded, and the node creates a DP containing object w and transmits it to the requesting node v'.

Third, if the IP cannot be fulfilled locally, then the node adds the requesting node to the corresponding PIT entry, i.e.,

$$\texttt{RequestingNodes}(v, w) \leftarrow \texttt{RequestingNodes}(v, w) \cup \{v'\}.$$

If $\texttt{RequestingNodes}(v, w)$ was previously empty, then the node transmits a copy of the received IP to one or more forwarding nodes from the set $\texttt{ForwardingNodes}(v, w)$, in accordance with the assumed *routing policy*; otherwise, the transmission of the IP is suppressed because an IP for the same object has already been transmitted earlier and the node is currently waiting for the arrival of the corresponding DP.

5.2.3 Handling of Data Packets (DPs) and Caching

Suppose that node v receives a DP containing object w at time t.

First, the node checks the corresponding PIT entry to find out which neighbor nodes have requested the object. It transmits a copy of the DP to each requesting node $v' \in \texttt{RequestingNodes}(v, w)$ that is a neighbor node of node v. The PIT entry is subsequently cleared, i.e., $\texttt{RequestingNodes}(v, w) \leftarrow \{\}$.

Next, the node checks if there are unfulfilled requests by end users for the object. All such requests are fulfilled at the same time, i.e.,

$$\texttt{TimeFulfilled}(u) \leftarrow t \quad \forall\, u \in \texttt{PendingRequestTable}(v, w),$$

with the requested object being served to the respective end users. The set of unfulfilled requests is subsequently cleared, i.e., $\texttt{PendingRequestTable}(v, w) \leftarrow \{\}$.

Finally, if the node is not the origin server for the object, i.e., $\texttt{OriginServer}(w) \neq v$, and the object is not currently cached in the node, i.e., $w \notin \texttt{CachedObjects}(v)$, and the cache is large enough to accommodate the object, i.e., $\texttt{CacheSize}(v) \geq \texttt{ObjectSize}(w)$, then the node decides whether to cache object w and possibly evict one or more currently cached objects, in accordance with the assumed *caching policy*.

5.2.4 Performance Metrics

To evaluate the performance of a given routing-caching policy, we compute the following quantities at periodic instances of time t.

1) Number of unfulfilled requests:

$$\left|\left\{u : \texttt{TimeCreated}(u) \leq t, \texttt{TimeFulfilled}(u) > t\right\}\right|.$$

2) Average delay incurred by a fulfilled request:

$$\frac{\sum\limits_{u\,:\,\texttt{TimeFulfilled}(u) \leq t} \texttt{TimeFulfilled}(u) - \texttt{TimeCreated}(u)}{\left|\left\{u : \texttt{TimeFulfilled}(u) \leq t\right\}\right|}.$$

3) Average rate of data transmission:

$$\frac{1}{t} \sum_{v_1, v_2} \texttt{AmountDataTransmitted}(v_1, v_2, 0, t),$$

where $\texttt{AmountDataTransmitted}(v_1, v_2, t_1, t_2)$ is the amount of data transmitted over the directed link from node v_1 to node v_2 between time t_1 and time t_2.

4) Data transmission queue length:

$$\sum_{v_1,v_2} \texttt{DataTransmissionQueueLength}(v_1, v_2, t),$$

where $\texttt{DataTransmissionQueueLength}(v_1, v_2, t)$ is the instantaneous length (e.g., in number of bits) of the data transmission queue for the directed link from node v_1 to node v_2 at time t.

5) Average rate of data access from cache hits:

$$\frac{1}{t} \sum_{h\,:\,\texttt{Time}(h)\leq t} \texttt{ObjectSize}\big(\texttt{Object}(h)\big).$$

6) Average rate of data removal from cache evictions:

$$\frac{1}{t} \sum_{e\,:\,\texttt{Time}(e)\leq t} \texttt{ObjectSize}\big(\texttt{Object}(e)\big).$$

These performance metrics reflect how promptly user requests are being fulfilled, and how efficiently the links, queue buffers, and caches are being utilized.

5.3 Virtual Backpressure Routing-Caching Policy

The Virtual Backpressure routing-caching policy introduces a *virtual* control plane for the handling of *Virtual Interest* packets (VIPs); in contrast, the handling of IPs and DPs is said to occur in the *actual* plane. The routing and caching decisions stipulated by this policy are based on local statistics collected on these VIPs.

VIPs are manipulated by nodes in a distributed asynchronous manner according to a backpressure-like algorithm [19, 20]. To deal with heterogeneous object sizes and link bandwidths, we shall assume that VIPs can be quantified in continuous amounts, as opposed to discrete units. Each node maintains a separate VIP queue for each object. Because no actual object data is contained in these VIPs, each VIP queue can simply be represented by a numerical variable; VIP transmissions between nodes are nothing more than messages about changing the values of these variables. Unlike IPs which may get suppressed (e.g., when multiple IPs for the same object arrive

in quick succession), VIPs are never suppressed.

The creation and transmission of VIPs occur during Virtual Backpressure *iterations* in each node, which may be executed asynchronously and at irregular time intervals. VIPs are created by each node v for each object w at a rate that matches that at which requests are created by end users at node v for object w. The VIPs for object w are eventually removed from the network at the origin server, i.e., $\texttt{OriginServer}(w)$, or at nodes that have cached object w; the corresponding VIP queues at these nodes are empty. VIPs are transmitted across each link according to the backpressure algorithm, subject to the bandwidth constraint of the *reverse* link in anticipation of the corresponding DPs. For faster convergence of the algorithm, these Virtual Backpressure iterations can be executed more frequently, at the expense of greater overhead.

The routing policy for the actual plane stipulates that a node should transmit an IP to a randomly selected forwarding node, with each such node weighted by the corresponding rate of VIPs transmitted to it. The caching policy for the actual plane stipulates that a node should cache and evict objects opportunistically so as to increase the rate of VIPs received for the cached objects, with each VIP weighted by the size of the corresponding object.

For brevity, we adopt the following notation to describe various aspects of the policy.

Policy Parameter: `CacheWeight` is the weight applied to received VIP flows for cached objects.

VIP Queues: Let $\texttt{VIPQueueLength}(v, w)$ be the length of the VIP queue at node v for object w.

VIP Flows: Let $\texttt{AmountVIPsTransmitted}(v_1, v_2, w, t)$ be the amount of VIPs transmitted over the directed link from node v_1 to node v_2, for object w, up till time t. Let $\texttt{CacheWeightedAmountVIPsTransmitted}(v_1, v_2, w, t)$ be the *cache-weighted* amount of VIPs transmitted over the directed link from node v_1 to node v_2, for object w, up till time t, where the amount of transmitted VIPs is multiplied by `CacheWeight` whenever object w is cached in node v_2.

5.3.1 Creation of Virtual Interest Packets (VIPs)

Let $\texttt{RequestRate}(v, w)$ be the rate at which requests are created by end users at node v for object w; this is an exogenous parameter provided to the Virtual Backpressure policy that can be estimated using actual observations or other information about user request patterns.

Consider a Virtual Backpressure iteration in node v that occurs T time units after the previous

iteration. For each object w, the node creates x amount of VIPs for the object, where

$$x = \texttt{RequestRate}(v, w) \times T,$$

and transmits them to itself (see Section 5.3.2 for subsequent actions taken by the node).

5.3.2 Handling of Virtual Interest Packets (VIPs)

Suppose that node v receives x amount of VIPs for object w. If the node is the origin server for the requested object, i.e., $\texttt{OriginServer}(w) = v$, or if the requested object is currently cached in the node, i.e., $w \in \texttt{CachedObjects}(v)$, then the received VIPs are immediately discarded; otherwise, the received VIPs are immediately added to the corresponding queue, i.e.,

$$\texttt{VIPQueueLength}(v, w) \leftarrow \texttt{VIPQueueLength}(v, w) + x.$$

Consider a Virtual Backpressure iteration in node v that occurs T time units after the previous iteration. Let $\texttt{AllForwardingNodes}(v)$ be the set of all the forwarding nodes of node v, taken over all objects, i.e.,

$$\texttt{AllForwardingNodes}(v) = \bigcup_{w} \texttt{ForwardingNodes}(v, w).$$

For each forwarding node $v' \in \texttt{AllForwardingNodes}(v)$, taken in arbitrary order, node v executes the following steps to decide which VIPs to transmit to node v'.

First, the node determines the set of candidate objects $\texttt{CandidateObjects}(v, v')$ whose VIPs can be transmitted to node v'. These are the objects for which node v' is a forwarding node of node v, and for which the VIP queue differential is positive, i.e.,

$$\begin{aligned}\texttt{CandidateObjects}(v, v') = \big\{w : v' \in \texttt{ForwardingNodes}(v, w),& \\ \texttt{VIPQueueLength}(v, w) > \texttt{VIPQueueLength}(v', w)\big\}.&\end{aligned}$$

If $\texttt{CandidateObjects}(v, v')$ is empty, then node v transmits zero VIPs to node v', and proceeds to consider the next forwarding node. Otherwise, the node selects from $\texttt{CandidateObjects}(v, v')$ an

object w^* with the largest positive VIP queue differential, weighted by the object size, i.e.,

$$w^* = \underset{w \in \texttt{CandidateObjects}(v,v')}{\arg\max} \left(\texttt{VIPQueueLength}(v,w) - \texttt{VIPQueueLength}(v',w)\right) \times \texttt{ObjectSize}(w).$$

Finally, the node transmits a maximal amount y of VIPs for object w^* to node v', subject to the bandwidth constraint of the *reverse* link in anticipation of the corresponding DPs, where

$$y = \min\left(\frac{\texttt{LinkBandwidth}(v',v) \times T}{\texttt{ObjectSize}(w^*)}, \texttt{VIPQueueLength}(v,w^*)\right).$$

The length of the corresponding VIP queue is updated accordingly, i.e.,

$$\texttt{VIPQueueLength}(v,w^*) \leftarrow \texttt{VIPQueueLength}(v,w^*) - y.$$

5.3.3 Routing Policy for the Actual Plane

Suppose that node v receives an IP for object w from a requesting node at time t. Recall from Section 5.2.2 that if the IP cannot be fulfilled locally, then the node may need to transmit a copy of the received IP to one or more forwarding nodes from the set $\texttt{ForwardingNodes}(v,w)$. In this event, the Virtual Backpressure policy stipulates that the received IP should be transmitted to a forwarding node v^* selected randomly from $\texttt{ForwardingNodes}(v,w)$, with each forwarding node weighted by the average rate of VIPs transmitted to it. Specifically, for each forwarding node $\widehat{v} \in \texttt{ForwardingNodes}(v,w)$, we have

$$\mathbb{P}\left[v^* = \widehat{v}\right] = \frac{\frac{1}{t}\,\texttt{AmountVIPsTransmitted}(v,\widehat{v},w,t)}{\sum\limits_{\overline{v} \in \texttt{ForwardingNodes}(v,w)} \frac{1}{t}\,\texttt{AmountVIPsTransmitted}(v,\overline{v},w,t)}.$$

5.3.4 Caching Policy for the Actual Plane

Suppose that node v receives a DP containing object w^* at time t. Recall from Section 5.2.3 that if the node is not the origin server for the object, and the object is not currently cached in the node, and the cache is large enough to accommodate the object, then it may decide whether to cache object w^* and possibly evict one or more currently cached objects. In this event, the Virtual Backpressure policy stipulates that object w^* should be cached if doing so increases the sum of

the cache-weighted average rates of VIPs received for the cached objects, with the rates further weighted by the corresponding object sizes. Specifically, node v executes the following steps.

First, the node determines which cached objects, if any, should be candidates for eviction to make room for the new object. To account for the unused cache space in subsequent calculations, we define a dummy object w_0 such that

$$\texttt{ObjectSize}(w_0) = \texttt{CacheSize}(v) - \sum_{w \in \texttt{CachedObjects}(v)} \texttt{ObjectSize}(w),$$

with $\texttt{CacheWeightedAmountVIPsTransmitted}(v', v, w_0, t) = 0$ for any node v'. Let $(w_0, w_1, w_2, \ldots)$ be a vector of the objects w from the set $\texttt{CachedObjects}(v) \cup \{w_0\}$, sorted in ascending order of the cache-weighted average rate of VIPs received, which is given by

$$\frac{1}{t} \sum_{v'} \texttt{CacheWeightedAmountVIPsTransmitted}(v', v, w, t).$$

The set of candidate objects for eviction $\texttt{CandidateObjects}(v)$ is greedily selected according to this rate, i.e.,

$$\texttt{CandidateObjects}(v) = \left\{w_i\right\}_{i=1}^{k^*},$$

where

$$k^* = \min \left\{ k : \sum_{i=0}^{k} \texttt{ObjectSize}(w_i) \geq \texttt{ObjectSize}(w^*) \right\}.$$

Next, the node determines whether it is beneficial to evict the set of objects $\texttt{CandidateObjects}(v)$ and cache object w^*, by comparing the cache-weighted average rates of VIPs received, with the rates further weighted by the corresponding object sizes. If

$$\frac{1}{t} \sum_{v'} \texttt{CacheWeightedAmountVIPsTransmitted}(v', v, w^*, t) \times \texttt{ObjectSize}(w^*)$$
$$\geq \sum_{w \in \texttt{CandidateObjects}(v)} \frac{1}{t} \sum_{v'} \texttt{CacheWeightedAmountVIPsTransmitted}(v', v, w, t) \times \texttt{ObjectSize}(w),$$

then the node proceeds to evict the set of objects $\texttt{CandidateObjects}(v)$ and cache object w^*, i.e.,

$$\texttt{CachedObjects}(v) \leftarrow \big(\texttt{CachedObjects}(v) \big\backslash \texttt{CandidateObjects}(v)\big) \cup \{w^*\},$$

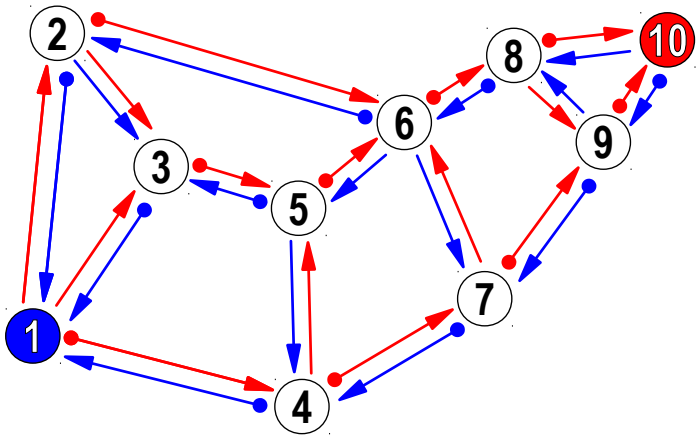


Figure 5.1. Network used in the simulation. IPs for the odd-numbered objects (for which node 1 is the origin server) are forwarded in the direction of the blue arrows, while IPs for the even-numbered objects (for which node 10 is the origin server) are forwarded in the direction of the red arrows. The shortest path to each origin server is indicated by arrows with a dotted tail; the forwarding nodes used in the basic protocol are given by these arrows.

and clears the VIP queue for the newly cached object, i.e.,

$$\texttt{VIPQueueLength}(v, w^*) \leftarrow 0.$$

Otherwise, the cache remains unchanged.

5.4 Simulation

As a preliminary evaluation of the proposed Virtual Backpressure routing-caching policy, we compared its performance against a basic protocol that combines shortest-path routing with least-recently-used (LRU) cache replacement [21], using a packet-level discrete event simulator.

This basic protocol handles IPs and DPs in the following manner: When a node receives an IP that cannot be fulfilled locally, it transmits the IP to a forwarding node on the shortest path to the corresponding origin server. Objects in the cache are arranged according to when they were last used or accessed. When a node receives a DP containing an object eligible for caching, it always caches the object; if necessary, one or more of the least recently used objects in the cache are evicted to make room for the new object.

5.4.1 Simulation Setup

We considered a 10-node network based on a simplified abstraction of the Internet2 network topology [65], with the nodes numbered 1 through 10, as shown in Figure 5.1. Each directed link in the network has a bandwidth of 1 000 000 data units per second. Nodes 1 and 10 are noncaching nodes (i.e., with a cache size of zero), while each of the other eight nodes has a cache size of 2 000 data units. The network contains 10 distinct objects, numbered 1 through 10, each of size 1 000 data units (recall that an IP has a normalized size of 1 data unit). Node 1 is the origin server for the odd-numbered objects, while node 10 is the origin server for the even-numbered objects. The request rates for the objects are the same across nodes; objects 1 and 2 are the most popular with a rate of 1 000 requests per second at each node, while objects 9 and 10 are the least popular with a rate of 100 requests per second at each node. The color-coded arrows in the figure describe the FIB at each node.

For the Virtual Backpressure policy, we set `CacheWeight` $= 1.0$, and have the iterations occurring at one-second intervals asynchronously across nodes. For each of the two policies, we ran the simulation for a duration of 3 hours, and recorded the six performance metrics defined in Section 5.2.4 at 10-second intervals.

5.4.2 Simulation Results and Discussion

Figure 5.2 summarizes the simulation results obtained for the two policies. The plots show that the Virtual Backpressure policy performed significantly better than the basic protocol in terms of request fulfillment (i.e., the number of unfulfilled requests and the average delay incurred by a fulfilled request) and cache utilization (i.e., the average rate of data access from cache hits and the average rate of data removal from cache evictions).

The superior performance of the Virtual Backpressure policy may be explained by its use of information about user request patterns (the rate of VIP creation reflects the popularity of the object), and its ability to perform load balancing automatically (by making probabilistic routing and caching decisions according to backpressure-influenced VIP flows).

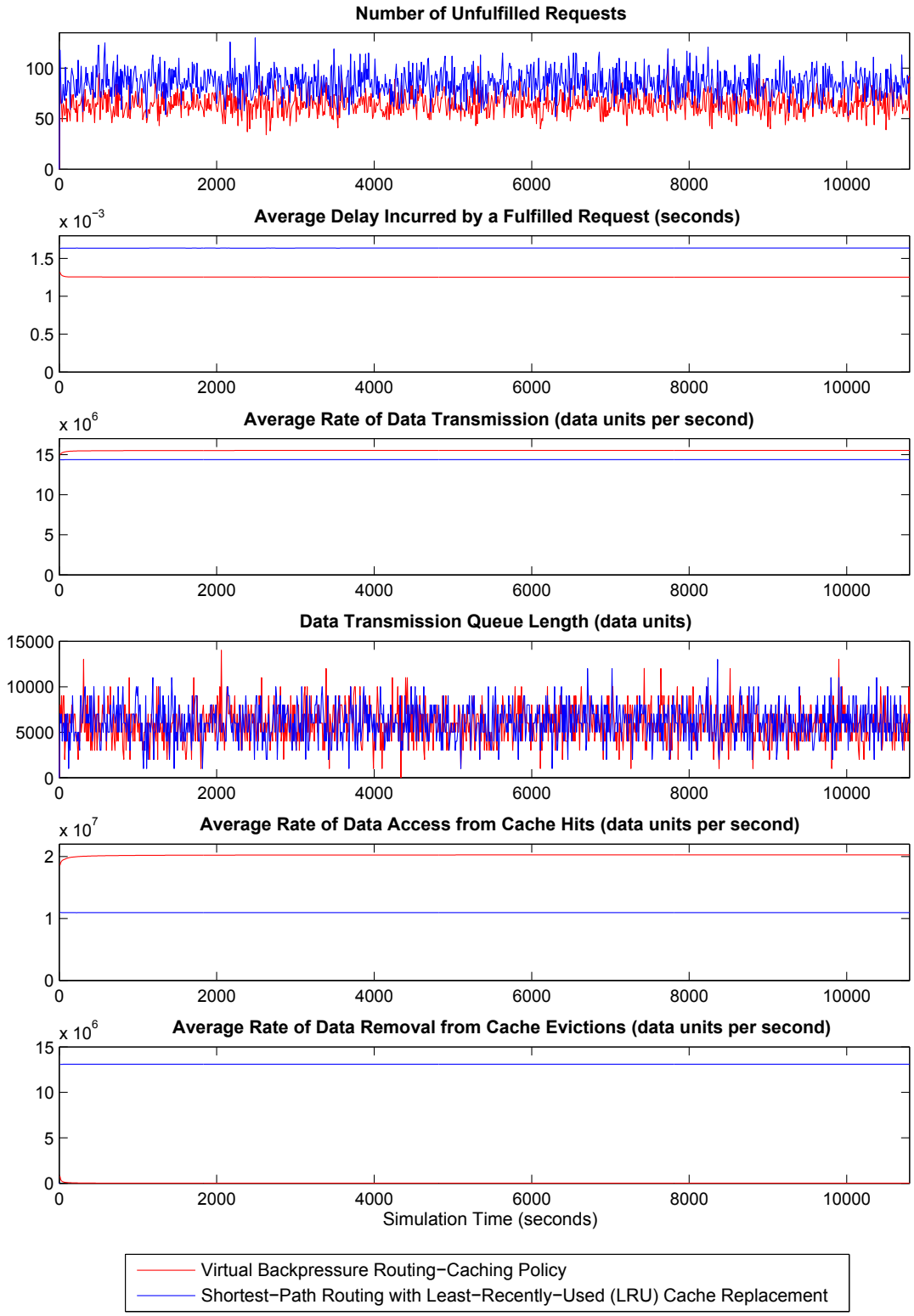


Figure 5.2. Simulation results for the Virtual Backpressure routing-caching policy, compared against a basic protocol that combines shortest-path routing with least-recently-used (LRU) cache replacement. Definitions of the performance metrics are given in Section 5.2.4.

5.5 Conclusion and Future Work

We considered the joint problems of routing and caching for named data networking, and proposed a backpressure-based policy that employs virtual interest packets to make routing and caching decisions. In a packet-level simulation, the proposed policy outperformed a basic protocol that combines shortest-path routing with least-recently-used (LRU) cache replacement.

Although the proposed policy performed well in simulations, many technical challenges relating to its implementation in a large-scale network like the Internet have yet to be addressed. For example, a good protocol would need to scale well with the number of nodes, objects, and users, react promptly to changes in network topology and user request patterns, and handle the creation and expiration of objects over time. It would also be interesting to study how such a policy would interact in practice with the other components of the named data networking architecture.

5.6 Acknowledgment

The author and his coauthors M. Burd and T. Ho would like to thank their collaborators Edmund Yeh and Kyle Dumont for the enriching discussions. They also thank Caroline Kim and Emil Ibrishimov for their programming assistance.

Chapter 6

Summary and Future Work

In the preceding chapters, we considered a series of basic erasure coding problems for distributed storage and streaming communications, and explored a number of fundamental limits and trade-offs associated with these systems.

6.1 Summary

In Chapter 2, we examined three variations of the distributed storage allocation problem, and determined the optimal allocation or optimal symmetric allocation for a variety of cases. Although the optimal allocation can have nonintuitive structure and can be difficult to find in general, our results suggest a simple heuristic for achieving reliable storage: when the budget is small, spread it minimally; when the budget is large, spread it maximally. In other words, coding is unnecessary when the budget is small, but is beneficial when the budget is large.

In Chapter 3, we studied the effects of distributed storage allocations on the recovery delay performance in a network of mobile nodes. We showed that the maximization of the probability of successful recovery by a given deadline is closely related to the allocation problem of Chapter 2. For the minimization of the expected recovery delay, we solved for the optimal symmetric allocation, and found that the optimal solutions for the two problems can be quite different. A simulation study on a simple data dissemination and storage protocol demonstrated that the choice of allocation can have a significant impact on the recovery delay.

In Chapter 4, we considered a real-time streaming problem for a packet erasure link, where each

message must be decoded within a given delay from its creation time. We showed that a symmetric intrasession code is asymptotically optimal over all codes for two variations of the window-based erasure model, and for the bursty erasure model when the maximum erasure burst length is sufficiently short or long. We also showed that diagonally interleaved codes derived from specific systematic block codes are asymptotically optimal over all codes for the bursty erasure model in several other cases. For the i.i.d. erasure model, we derived an upper bound on the decoding probability for any time-invariant code, and showed that the gap between this bound and the performance of the family of symmetric intrasession codes is small when the message size and packet erasure probability are small. In a simulation study, these symmetric codes performed well against a family of random time-invariant convolutional codes under a number of scenarios.

In Chapter 5, we considered the joint problems of routing and caching for named data networking, and proposed a backpressure-based policy that employs virtual interest packets to make routing and caching decisions. In a packet-level simulation, the proposed policy outperformed a basic protocol that combines shortest-path routing with least-recently-used (LRU) cache replacement.

6.2 Future Work

Many interesting questions remain unanswered for each of the problems.

In Chapter 2, while the optimal allocation is known for a number of special cases, a general solution for the distributed storage allocation problem remains elusive. We conjectured that a symmetric optimal allocation always exists for the fixed-size subset variation of the problem. To describe real-world applications more accurately, the simple allocation and access models assumed for the problem can be extended in several ways. For example, apart from the budget constraint, we can impose additional storage constraints on individual nodes to limit the amount of data stored in them. Also, the independent probabilistic access model can be generalized to handle heterogeneous access, e.g., nonuniform failure probabilities for individual nodes. We can assign different costs to each node for data storage and access so as to reflect the capabilities of the node and the ease of communicating with it. It would also be interesting to find reliable allocations for specific codes with efficient encoding and decoding algorithms.

In Chapter 3, we conjectured that a symmetric optimal allocation always exists for the mini-

mization of the expected recovery delay. The simple contact model assumed for the problem can be generalized to one that allows a variable amount of data to be transmitted between nodes when they meet. It would also be interesting to study how distributed storage allocations can affect the recovery delay in more sophisticated mobility models, e.g., where nodes have different rates of contact with the data collector.

In Chapter 4, while optimal real-time streaming codes have been constructed for both variations of the window-based erasure model, such codes have yet to be found for the bursty erasure model in a number of cases. The i.i.d. erasure model also offers many interesting problems for future work. In an effort to find the optimal code, it may be useful to consider hybrid code constructions that capture the strengths of both the symmetric intrasession codes and the random time-invariant convolutional codes that were examined in the simulation study.

In Chapter 5, although the proposed backpressure-based routing-caching policy performed well in simulations, many technical challenges relating to its implementation in a large-scale network like the Internet have yet to be addressed. It would also be interesting to study how such a policy would interact in practice with the other components of the named data networking architecture.

Bibliography

[1] D. A. Patterson, G. Gibson, and R. H. Katz, "A case for redundant arrays of inexpensive disks (RAID)," in *Proc. ACM Intl. Conf. Management Data (SIGMOD)*, 1988, pp. 109–116.

[2] Amazon S3, Amazon. [Online]. Available: http://aws.amazon.com/s3/

[3] S. Acedánski, S. Deb, M. Médard, and R. Koetter, "How good is random linear coding based distributed networked storage?" in *Proc. Workshop Netw. Coding, Theory, Appl. (NetCod)*, Apr. 2005, pp. 4057–4073.

[4] 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, pp. 111–117.

[5] A. Kamra, V. Misra, J. Feldman, and D. Rubenstein, "Growth codes: Maximizing sensor network data persistence," in *Proc. ACM SIGCOMM*, Sep. 2006, pp. 255–266.

[6] Y. Lin, B. Liang, and B. Li, "Data persistence in large-scale sensor networks with decentralized fountain codes," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, May 2007, pp. 1658–1666.

[7] 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, pp. 171–182.

[8] R. Kleinberg, R. Karp, C. Papadimitriou, and E. Friedman, *Personal correspondence between R. Kleinberg and A. G. Dimakis*, Oct. 2006.

[9] A. Tsirigos and Z. J. Haas, "Analysis of multipath routing—Part I: The effect on the packet delivery ratio," *IEEE Transactions on Wireless Communications*, vol. 3, no. 1, pp. 138–146, Jan. 2004.

[10] ——, "Analysis of multipath routing, Part 2: Mitigation of the effects of frequently changing network topologies," *IEEE Transactions on Wireless Communications*, vol. 3, no. 2, pp. 500–511, Mar. 2004.

[11] 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, pp. 109–120.

[12] W. K. Lin, D. M. Chiu, and Y. B. Lee, "Erasure code replication revisited," in *Proc. Int. Conf. Peer-to-Peer Comput. (P2P)*, Sep. 2004.

[13] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and Wait: An efficient routing scheme for intermittently connected mobile networks," in *Proc. ACM SIGCOMM Workshop Delay-Tolerant Netw.*, Aug. 2005, pp. 252–259.

[14] Y. Polyanskiy, "Low-latency codes for packet-erasure networks," *Unpublished technical report*, Apr. 2009.

[15] L. J. Schulman, "Coding for interactive communication," *IEEE Transactions on Information Theory*, vol. 42, no. 6, pp. 1745–1756, Nov. 1996.

[16] A. Sahai, "Anytime information theory," Ph.D. dissertation, Massachusetts Institute of Technology, 2001.

[17] R. T. Sukhavasi, "Distributed control and computing: Optimal estimation, error correcting codes, and interactive protocols," Ph.D. dissertation, California Institute of Technology, 2012.

[18] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, "Networking named content," in *Proc. Int. Conf. Emerging Netw. Experiments and Technol. (CoNEXT)*, Dec. 2009.

[19] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Transactions on Automatic Control*, vol. 37, no. 12, pp. 1936–1948, Dec. 1992.

[20] L. Georgiadis, M. J. Neely, and L. Tassiulas, "Resource allocation and cross-layer control in wireless networks," *Foundations and Trends in Networking*, vol. 1, no. 1, pp. 1–144, 2006.

[21] S. Podlipnig and L. Böszörmenyi, "A survey of Web cache replacement strategies," *ACM Comput. Surveys (CSUR)*, vol. 35, no. 4, pp. 374–398, Dec. 2003.

[22] D. Leong, A. G. Dimakis, and T. Ho, "Distributed storage allocation problems," in *Proc. Workshop Netw. Coding, Theory, Appl. (NetCod)*, Lausanne, Switzerland, Jun. 2009, pp. 86–91.

[23] ——, "Distributed storage allocation for high reliability," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Cape Town, South Africa, May 2010.

[24] ——, "Symmetric allocations for distributed storage," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Miami, Florida, USA, Dec. 2010.

[25] ——, "Distributed storage allocations," *IEEE Transactions on Information Theory*, vol. 58, no. 7, pp. 4733–4752, Jul. 2012.

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

[27] A. Jiang, "Network coding for joint storage and transmission with minimum cost," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2006, pp. 1359–1363.

[28] 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 Transactions on Information Theory*, vol. 52, no. 10, pp. 4413–4430, Oct. 2006.

[29] 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.

[30] J.-S. Wu and R.-J. Chen, "An algorithm for computing the reliability of weighted-k-out-of-n systems," *IEEE Transactions on Reliability*, vol. 43, no. 2, pp. 327–328, Jun. 1994.

[31] Y. Chen and Q. Yang, "Reliability of two-stage weighted-**k**-out-of-n systems with components in common," *IEEE Transactions on Reliability*, vol. 54, no. 3, pp. 431–440, Sep. 2005.

[32] M. Sardari, R. Restrepo, F. Fekri, and E. Soljanin, "Memory allocation in distributed storage networks," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2010, pp. 1958–1962.

[33] N. Alon, P. Frankl, H. Huang, V. Rödl, A. Ruciński, and B. Sudakov, "Large matchings in uniform hypergraphs and the conjectures of Erdős and Samuels," *Journal of Combinatorial Theory, Series A*, vol. 119, no. 6, pp. 1200–1215, Aug. 2012.

[34] P. Erdős, "A problem on independent r-tuples," *Ann. Univ. Sci. Budapest*, vol. 8, pp. 93–95, 1965.

[35] M. Naor and R. M. Roth, "Optimal file sharing in distributed networks," *SIAM J. Comput.*, vol. 24, no. 1, pp. 158–183, Feb. 1995.

[36] A. Jiang and J. Bruck, "Network file storage with graceful performance degradation," *ACM Trans. Storage*, vol. 1, no. 2, pp. 171–189, May 2005.

[37] D. Leong, T. Ho, and R. Cathey, "Optimal content delivery with network coding," in *Proc. Annu. Conf. Inf. Sci. Syst. (CISS)*, Baltimore, Maryland, USA, Mar. 2009, pp. 414–419.

[38] J. H. van Lint and R. M. Wilson, *A Course in Combinatorics*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 2001.

[39] G. O. H. Katona, "Extremal problems for hypergraphs," in *Combinatorics*, M. Hall, Jr. and J. H. van Lint, Eds. Dordrecht, Holland: D. Reidel, 1974.

[40] V. Ntranos, G. Caire, and A. G. Dimakis, "Allocations for heterogenous distributed storage," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Cambridge, Massachusetts, USA, Jul. 2012.

[41] L. G. Valiant, "The complexity of computing the permanent," *Theoretical Comput. Sci.*, vol. 8, no. 2, pp. 189–201, 1979.

[42] M. Sipser, *Introduction to the Theory of Computation*, 2nd ed. Boston, MA: Thomson Course Technology, 2006.

[43] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. New York: Cambridge Univ. Press, 2005.

[44] W. Feller, *An Introduction to Probability Theory and Its Applications, Vol. 1*, 3rd ed. New York: Wiley, 1968.

[45] R. Kaas and J. M. Buhrman, "Mean, median and mode in binomial distributions," *Statistica Neerlandica*, vol. 34, no. 1, pp. 13–18, 1980.

[46] D. Leong, A. G. Dimakis, and T. Ho, "Distributed storage allocations for optimal delay," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Saint Petersburg, Russia, Jul. 2011, pp. 1447–1451.

[47] S. Jain, K. Fall, and R. Patra, "Routing in a delay tolerant network," in *Proc. ACM SIGCOMM*, Aug. 2004.

[48] Y. Wang, S. Jain, M. Martonosi, and K. Fall, "Erasure-coding based routing for opportunistic networks," in *Proc. ACM SIGCOMM Workshop Delay-Tolerant Netw.*, Aug. 2005.

[49] M. Piórkowski, N. Sarafijanovoc-Djukic, and M. Grossglauser, "A parsimonious model of mobile partitioned networks with clustering," in *Proc. Intl. Conf. Commun. Syst. Netw. (COMSNETS)*, Jan. 2009.

[50] J. Havil, *Gamma: Exploring Euler's Constant*. Princeton, NJ: Princeton Univ. Press, 2003.

[51] D. Leong and T. Ho, "Erasure coding for real-time streaming," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Cambridge, Massachusetts, USA, Jul. 2012.

[52] E. Martinian and C.-E. W. Sundberg, "Low delay burst erasure correction codes," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2002, pp. 1736–1740.

[53] E. Martinian and M. Trott, "Delay-optimal burst erasure code construction," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2007, pp. 1006–1010.

[54] A. Badr, A. Khisti, W.-T. Tan, and J. Apostolopoulos, "Streaming codes for channels with burst and isolated erasures," *arXiv:1208.0072*, Aug. 2012. [Online]. Available: http://arxiv.org/abs/1208.0072

[55] Ö. F. Tekin, S. Vyetrenko, T. Ho, and H. Yao, "Erasure correction for nested receivers," in *Proc. Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Sep. 2011, pp. 1454–1461.

[56] Y.-K. Wang, R. Even, T. Kristensen, and R. Jesup, "RTP payload format for H.264 video," IETF RFC 6184, May 2011. [Online]. Available: http://datatracker.ietf.org/doc/rfc6184/

[57] "UMTS; LTE; MBMS; Protocols and codecs," 3GPP TS 26.346 version 11.2.0 Release 11, Oct. 2012. [Online]. Available: http://www.3gpp.org/ftp/Specs/html-info/26346.htm

[58] A. H. Li, "RTP payload format for generic forward error correction," IETF RFC 5109, Dec. 2007. [Online]. Available: http://datatracker.ietf.org/doc/rfc5109/

[59] L. Rizzo, "Effective erasure codes for reliable computer communication protocols," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 27, no. 2, pp. 24–36, Apr. 1997.

[60] Y. Wang and Q.-F. Zhu, "Error control and concealment for video communication: A review," *Proceedings of the IEEE*, vol. 86, no. 5, pp. 974–997, May 1998.

[61] B. W. Wah, X. Su, and D. Lin, "A survey of error-concealment schemes for real-time audio and video transmissions over the Internet," in *Proc. Int. Symp. Multimedia Software Eng.*, Dec. 2000, pp. 17–24.

[62] J. Rasmussen, A. Shokrollahi, S. Lassen, G. Horn, V. Goyal, B. Dobyns, and M. Luby, "System and method for reliably communicating the content of a live data stream," U.S. Patent 7 249 291, Jul. 24, 2007.

[63] M. Luby, "LT codes," in *Proc. Annu. IEEE Symp. Found. Comput. Sci. (FOCS)*, Nov. 2002.

[64] M. G. Luby, M. Watson, and M. A. Shokrollahi, "Forward error-correcting (FEC) coding and streaming," U.S. Patent 7 676 735, Mar. 9, 2010.

[65] Internet2 Network Services, Internet2. [Online]. Available: http://www.internet2.edu/network/services