UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Karl Kristjan Puusepp

Model for Saving Energy in Distributed Systems Using QC-LDPC Codes

Bachelor's Thesis (9 ECTS)

Supervisors:
Vitaly Skachek, PhD
Irina Bocharova, PhD
Boris Kudryashov, PhD

Tartu 2025

Model for Saving Energy in Distributed Systems Using QC-LDPC Codes

Abstract:

Modern distributed data storage systems incur substantial energy costs, largely driven by the need to keep large numbers of disks spinning to guarantee data availability. In this thesis, an energy-aware storage architecture is proposed based on a quasi-cyclic LDPC code. A hybrid single-bit recovery procedure is introduced that combines peeling decoding with Gaussian elimination to minimize complexity. Disk popularity skew is modeled under various parameters and Monte Carlo experiments demonstrate that the model yields potentially significant energy savings without sacrificing reliability.

Keywords: Energy-aware storage, Quasi-Cyclic LDPC codes, coding theory, distributed systems

CERCS: P170: Computer science, numerical analysis, systems, control

Kvaasi-tsüklilistel LDPC koodidel põhinev mudel hajussüsteemide energiakulu vähendamiseks

Lühikokkuvõte:

Kaasaegsed hajusad salvestussüsteemid põhjustavad märkimisväärseid energiakulusid, kuna suure hulga ketaste pidev jooksutamine on vajalik andmete kättesaadavuse tagamiseks. Käesolevas töös pakutakse energiasäästlikku salvestusarhitektuuri, mis põhineb kvaasi-tsüklilisel LDPC-koodil. Tutvustatakse hübriidset ühebitist taastamisprotseduuri, mis kombineerib *peeling*-dekodeerimise ja Gaussi eliminatsioonimeetodi arvutusliku keerukuse minimeerimiseks. Ketaste populaarsuse ebaühtlust modelleeritakse erinevate parameetrite alusel ning Monte Carlo eksperimendid näitavad, et mudel võib pakkuda olulist energiakokkuhoidu, säilitades samal ajal töökindluse.

Võtmesõnad: Energiateadlik salvesussüsteem, kvaasi-tsüklilised LDPC koodid, kodeerimisteooria, hajussüsteemid

CERCS: P170: Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimis- teooria)

Contents

1. Introduction

Energy consumption in today's data centers is dominated by storage hardware: hard disks draw power when idle and incur even larger surges during spin-up. Frequent spin-up events can negate any power-down savings, so naively turning off cold disks often proves counterproductive. To address this challenge, coding-theoretic redundancy is leveraged to allow more disks to remain off, treating powered-down disks as erasures in an LDPC code and recovering requested data through efficient decoding procedures.

This thesis aims to propose a model for saving energy by utilizing coding theory concepts to satisfy requests even when the requested disk is turned off. A new algorithm is introduced to do this effectively, using a combination of peeling decoding and Gaussian elimination for a reliable and fast single-bit decoding strategy. A simple probabalistic model is introduced, that models the popularity of disks in a disk array and Monte-Carlo simulations are used to show potentially significant energy savings in the proposed system.

The remainder of the thesis is organized as follows. Chapter 2 surveys storage-system energy models and prior power-management strategies. Chapter 3 introduces the necessary coding-theory preliminaries, including linear codes, LDPC and quasi-cyclic constructions, and Reed-Solomon codes. Chapter 4 develops erasure-decoding algorithms: Gaussian elimination, peeling, and a proposed combined single-bit recovery. Chapter 5 presents the energy-saving model. Finally, Chapter 6 summarizes the findings and outlines directions for future work.

2. Distributed Systems

Distributed systems form the backbone of modern data centers and cloud services. However, the ever-growing demand for high availability and large-scale storage has resulted in significant energy consumption. Despite their central role in contemporary IT infrastructure, these facilities are remarkably energy intensive, with estimates from the International Energy Agency (IEA) suggesting that data centers account for up to 1.3% of the annual global energy demand [1]. Due to the increasing demand for AI, this figure is also projected to nearly double by 2030 as shown in Figure 1. This significant footprint has raised concerns both from environmental and economic perspectives.

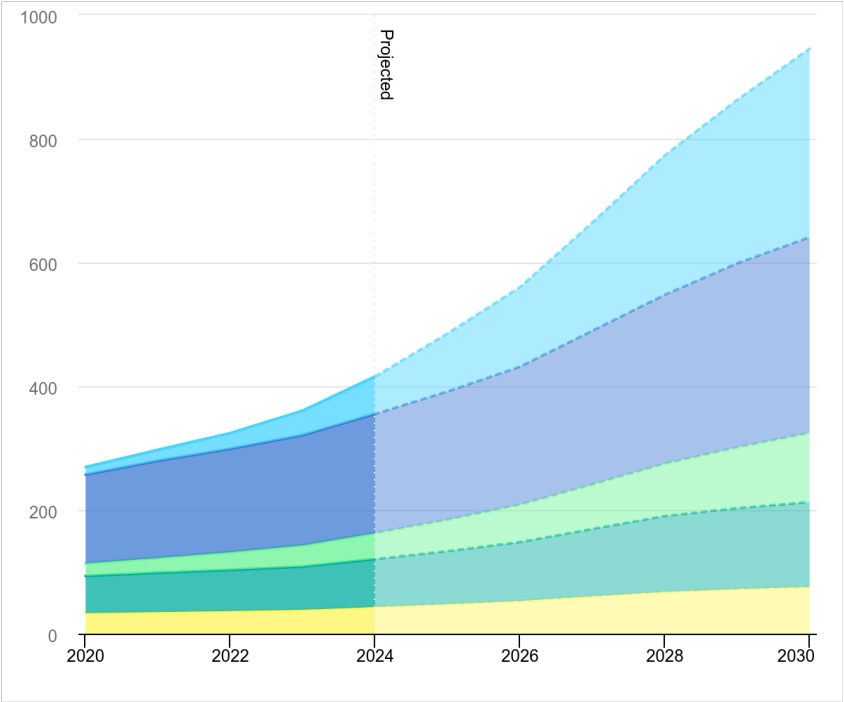


Figure 1. Global data centre electricity consumption in TWh, by equipment, Base Case, 2020-2030. Source: IEA (2025) [2].

The systems, which underpin many data center operations, commonly rely on vast arrays of storage devices, such as hard drives, to manage and ensure the availability of data. These storage systems are typically configured for reliability and performance, generally employing techniques like redundant array of independent disks (RAID) to mitigate the risk of data loss due to hardware failures. However, ensuring that data is constantly available usually involves keeping a large number of these disks running simultaneously, leading to substantial energy consumption.

A significant portion of the energy used by these systems can be attributed not just to the power drawn during standard operations, but also to the inefficiencies of managing the hardware state. Hard drives, for instance, are energy-hungry when active, and attempts to conserve energy by frequently switching them off are complicated. The act of spinning a disk back up is not only slow but also demands a surge of power, which can negate any potential energy savings.

Given these challenges, there is a need for new strategies that simultaneously ensure data reliability and achieve significant energy savings. This thesis aims to propose a model for utilizing redundant disks for energy saving purposes by leveraging coding methods.

2.1 Modelling the Energy Consumption of Disk Arrays

To efficiently manage energy in disk-based storage, a model for the storage system must be established. A 2020 paper by Arora and Bala [3] introduces the PAP (Power Aware Prediction) framework, which aims to reduce energy usage by intelligently switching disks between various operational states based on workload prediction. For the purposes of this thesis, a modified version of the model proposed in the paper is considered.

In conventional disk arrays, the drives typically remain in an **active** or **idle** state to ensure rapid data access. However, these states consume considerably more energy than a drive that is completely switched off. The challenge arises from the fact that while a switched-off drive would ideally use zero energy, turning it back on incurs a high energy cost and a delay. Moreover, disks in an active state often draw more power than when they are merely idling.

Table 1 lists representative wattage values adapted from the referenced article. Note, these values are representative and will be abstracted.

Table 1. Representative Power Consumption Values for Disk States (in Watts)

State	Power (W) for HDD
Active	∼8.0
Idle	∼5.0
Standby (Switched Off)	0.4
Spin-Up	14.63
Spin-Down	1.83

Figure 2 provides a schematic representation of the state transitions using a directed graph. In the figure, nodes correspond to the various operational states of the disk while directed edges illustrate the transitions between these states.

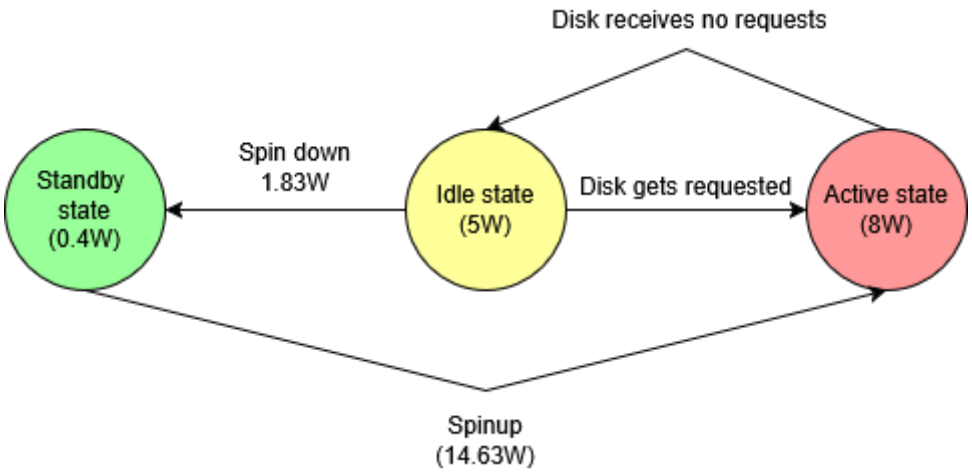


Figure 2. Directed graph of disk state transitions. Adapted from [3]

Based on the above considerations, the energy consumption E_d of a disk is modeled as a function of both the duration spent in various states and the corresponding power consumption levels:

$$E_d = t_{\text{active}} \cdot P_{\text{active}} + t_{\text{idle}} \cdot P_{\text{idle}} + t_{\text{spinup}} \cdot P_{\text{spinup}} + t_{\text{spindown}} \cdot P_{\text{spindown}}$$

where:

- t_{active}, t_{idle}, t_{spinup}, and t_{spindown} denote the time intervals spent in the active, idle, spin-up, and spin-down states, respectively.
- P_{active}, P_{idle}, P_{spinup}, and P_{spindown} denote the power consumption in the corresponding states.

For the purposes of this thesis, a simplified model is used by assuming that:

1. A disk that is completely switched off consumes no energy.
2. Spin-down operations incur negligible energy costs and, therefore, $P_{\text{spindown}} \approx 0$.
3. The time interval considered is always of equal duration and instead probabilities of different states are used to model the energy consumption of the disk.

Therefore, we model the average power consumption as

$$\overline{P_d} = p_{\text{active}} \cdot P_{\text{active}} + p_{\text{idle}} \cdot P_{\text{idle}} + p_{\text{spinup}} \cdot P_{spinup}$$

where p_{active}, p_{idle}, and p_{spinup} denote the probability of the disk being in the active, idle and spin-up states, respectively. Then the total power drawn in a single time interval on average by the proposed system is simply $\sum_{d \in D} \overline{P_d}$ where D is the set of all disks in our system. For further simplification, the idle state may be excluded in favor of modelling only ”hot” active and ”cold” turned-off disks.

2.2 Previously Proposed Strategies for Energy-Saving

In recent years, a number of strategies have been proposed to reduce energy consumption in disk-based storage systems. In the following, two examples of such approaches are reviewed: the PAP (Power Aware Prediction) framework and Pelican. While PAP relies on workload prediction to control disk state transitions, Pelican adopts a resource right-provisioning strategy and leverages redundancy utilizing the Massive Arrays of Idle Disks (MAID) paradigm and erasure coding to save power.

2.2.1 PAP - Power Aware Prediction Framework

The PAP framework [3] seeks to reduce the energy consumption of disk arrays by predicting idle intervals in disk activity and then dynamically switching disks from high-power active states into lower-power standby states.

PAP employs machine learning models to forecast the inter-arrival times of disk I/O requests. Using these predictions, the system determines whether the expected idle time is long enough to warrant a disk spin-down. By lowering the number of unnecessary spin-up events, PAP is able to achieve significant energy reductions while maintaining overall system performance.

However, the effectiveness of PAP heavily depends on the accuracy of the prediction models. Under highly variable workloads, the prediction error may lead to premature spin-ups or increased latency. The method also does not make use of redundant disks, which means it has potential for use in conjunction with the method proposed in this thesis.

2.2.2 Pelican

Pelican [4] is a rack-scale storage system designed specifically for cold data storage and is one of the few other systems that use redundancy for power saving. Unlike traditional designs that provision hardware for peak performance, Pelican is right-provisioned to provide only as much power, cooling, and network bandwidth as required by its workload. Its distinctive features include:

- **Resource right-provisioning.** Pelican supports only a small fraction (approximately 8%) of its disks in the active state at any given time. The remaining disks are kept in a low-power standby state. This drastically reduces the average power consumption of the storage system.
- **Redundancy through erasure coding.** Much like the system proposed in this thesis, Pelican incorporates redundancy using erasure codes. In contrast to approaches that rely solely on spinning down disks, Pelican uses redundancy in the form of MAID to ensure data reliability while having many disks idle. This allows the system to tolerate a certain number of disk failures and to serve data requests by reconstructing data from multiple disks.

However, as only a small subset of disks are active, requests for data requires spinning disks up more often, leading to higher latency compared to systems where more disks remain active. Furthermore, the right-provisioning approach necessitates sophisticated scheduling and data layout strategies to ensure that the redundancy and resource constraints are met without sacrificing performance. This makes it not as suitable for storing differing data that must be accessed fast and frequently.

3. Preliminaries. Codes

Codes are fundamental to modern digital communications and storage systems. They serve as mathematical tools that protect against data loss and corruption by incorporating redundancy into the transmitted or stored information. In essence, coding schemes enable the detection and correction of errors and erasures introduced by noisy channels or imperfect storage hardware, thereby ensuring the integrity and reliability of data.

Modern coding theory began with Claude E. Shannon in 1948, whose paper ”A Mathematical Theory of Communication” laid the foundation for modern information theory [5]. In this work, Shannon introduced fundamental concepts such as entropy, channel capacity, and the noisy-channel coding theorem, which collectively set the theoretical limits for error-free communication over noisy channels. We begin by looking at the communication model proposed by Shannon.

3.1 Communication System

The communication model in Figure 3 shows the transmission of information from a *source* to a *destination* through a *channel* [6]. *Source encoding* and *decoding* are usually utilized for compression and signal type transformations and will not be relevant to the subject of this thesis. The *channel encoder* and *channel decoder* are used to encode the data and add redundancy to pass the infomation through some *channel*. Since the output of a noisy channel often differs from its input, codes are used to overcome the limitations of different noisy channels.

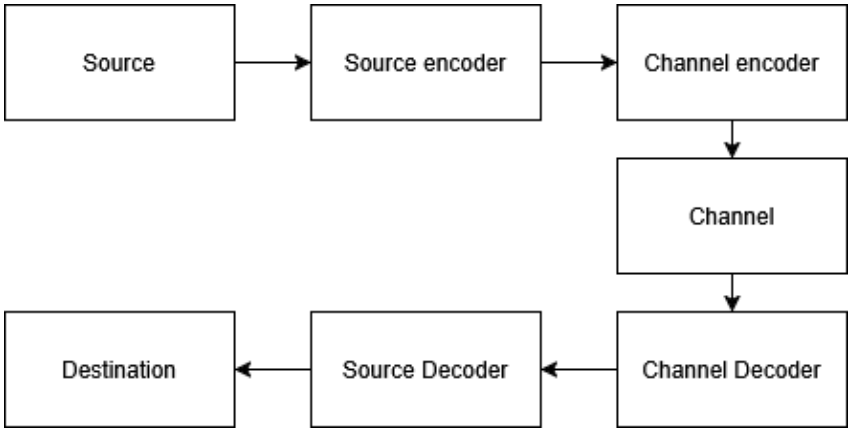


Figure 3. The communication model [6].

We will concentrate on the channel coding part of the introduced model and consider the *binary erasure channel* (BEC). This is the simplest channel model where the input alphabet is $\{0, 1\}$ and the output alphabet is $\{0, 1, ?\}$ where ”?” stands for a bit erasure. Any single input

symbol is erased with the probability p as shown in Figure 4. In essence, when transmitting a signal consisting of strictly ones and zeroes, each symbol in the output has a probability p of being erased.

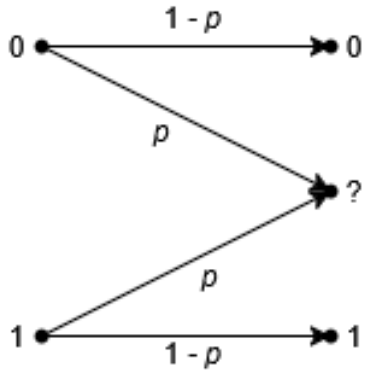


Figure 4. The binary erasure channel.

3.2 Linear Codes

In the scope of this thesis we will only consider codes in terms of binary information vectors. That is, our alphabet will always be $\{0, 1\}$. Furthermore, we only consider erasure-correcting codes, as our model deals with available and unavailable data, where unavailable data will be treated as an erasure. The definitions for this chapter were adapted from textbooks [6, 7].

Definition 3.1 (Binary Linear Code)**.** Any k-dimensional subspace of $\mathbb{F}_2^n$, is called a binary linear $[n, k]$-code $\mathcal{C}$. We call n the code length and k the code dimension.

Definition 3.2 (Codeword)**.** Given a binary linear $[n, k]$-code $\mathcal{C}$, a *codeword* is any vector $\mathbf{c} \in \mathbb{F}_2^n$ such that $\mathbf{c} \in \mathcal{C}$. In other words, the codewords of $\mathcal{C}$ are precisely the elements of the k-dimensional subspace that defines the code.

We also present some parameters for codes such as the code rate and minimum distance.

Definition 3.3 (Code rate)**.** The rate of the linear $[n, k]$-code is equal to $R = k/n$.

The code rate can be interpreted as the proportion of the codeword that is made up of information. The value 1-R is thus the proportion of the codeword made up of redundant bits. Next, we cover the Hamming distance of codes.

Definition 3.4 (Hamming Distance)**.** The Hamming distance between two words $\mathbf{x} = (x_1, \ldots, x_n)$ and $\mathbf{y} = (y_1, \ldots, y_n)$ of length n is defined as

$$d(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{n} d(x_i, y_i),$$

where

$$d(x, y) = \begin{cases} 0, & x = y \\ 1, & x \neq y \end{cases}$$

It satisfies the axioms

- $d(\mathbf{x}, \mathbf{y}) \geq 0$ and $d(\mathbf{x}, \mathbf{y}) = 0$ iff $\mathbf{x} = \mathbf{y}$
- $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$ (symmetry)
- $d(\mathbf{x}, \mathbf{y}) \leq d(\mathbf{x}, \mathbf{z}) + d(\mathbf{z}, \mathbf{y})$ (triangle inequality).

Definition 3.5 (Minimum distance)**.** The minimum distance of a code $\mathcal{C}$ is the least pairwise Hamming distance between its nonequal codewords

$$d = \min_{\mathbf{x}, \mathbf{y} \in \mathcal{C}, \mathbf{x} \neq \mathbf{y}} d(\mathbf{x}, \mathbf{y})$$

The minimum distance is an important measure of the recovery properties of the code. For any code $\mathcal{C}$ with minimum distance d, any pattern of up to $d - 1$ erasures is always recoverable, since at least d erasures could result in two different codewords colliding after erasures. Thus, a larger minimum distance is generally more desireable. For a linear code $\mathcal{C}$ with length n, dimension k and minimum distance d, we call it an $[n, k, d]$ linear code.

Next, we define generator and parity-check matrices through which codes are defined.

Definition 3.6 (Generator Matrix)**.** A generator matrix of a linear $[n, k]$ code is a $k \times n$ matrix whose rows are basis vectors of the linear subspace. We denote the generator matrix of a code by G.

Codewords of a linear code are linear combinations of basis vectors. Thereby, a codeword can be obtained by multiplying a message by the generator matrix G. Let $\mathbf{m} = (m_1, m_2, \ldots, m_k) \in \mathbb{F}_2^k$ be a message. Then the corresponding codeword $\mathbf{c} = (c_1, c_2, \ldots, c_n)$ can be obtained by $\mathbf{c} = \mathbf{m}G$. This is the process of encoding k message bits into a codeword of length n bits.

Definition 3.7 (Parity-Check Matrix)**.** For a linear $[n, k]$ code $\mathcal{C}$ over $\mathbb{F}_2$, the parity-check matrix $H = (\mathbf{h_1}, \ldots, \mathbf{h_n})^T$ of $\mathcal{C}$ is an $(n-k) \times n$ matrix such that for every $\mathbf{c} \in \mathbb{F}_2^n$,

$$\mathbf{c} \in \mathcal{C} \iff H\mathbf{c}^T = \mathbf{0}.$$

Definition 3.8 (Bit Error Rate (BER))**.** The bit error rate (BER) of a digital communication system is the expected proportion of transmitted bits that are received in error. Formally, if N bits are sent and N_e denotes the random number of bit errors, then

$$\mathrm{BER} = \lim_{N \to \infty} \frac{\mathrm{E}[\mathrm{N_e}]}{N}.$$

Example 3.1. Define a simple $[n = 3, k = 2]$ binary linear code of rate $R = 2/3$ whose generator and parity-check matrices are

$$G = \begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{pmatrix}, \qquad H = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix}.$$

We see that $G\,H^T = 0$, so indeed $\mathcal{C} = \{\, \mathbf{m}\,G \mid \mathbf{m} \in \mathbf{F}_2^2 \}$.

- **Minimum distance:** Every nonzero codeword has weight ≥ 2, so $d_{\min} = 2$.
- **Encoding:** Given an information vector $\mathbf{m} = (1, 0)$,

$$\mathbf{c} = (1, 0)\,G = (1, 0, 1).$$

  Likewise $(0, 0) \mapsto (0, 0, 0)$, $(0, 1) \mapsto (0, 1, 1)$ and $(1, 1) \mapsto (1, 1, 0)$.

3.2.1 Low-Density Parity-Check Codes

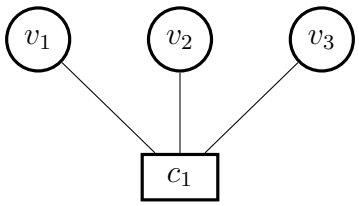
Introduced by R.Gallager in the early 1960s [8], *low-density parity-check* (LDPC) codes are linear codes whose parity-check matrices contain relatively few non-zero entries. This class of codes is mainly useful due to being generally decodeable with low complexity. LDPC codes have found many uses over the years, some of which include SSD data encoding and the IEEE802 standard for Ethernet and WiFi communication [9, 10].

Definition 3.9 ((J, K)-Regular LDPC Code)**.** A binary linear $[n, k]$-code determined by a parity-check matrix H is called (J, K)-regular if each column of H contains J ones and each row contains K ones.

Due to the nature of LDPC codes, we assume that J and K are small compared to n and k. A popular way to represent LDPC codes is via graphs. These are called Tanner graphs.

Definition 3.10 (Tanner Graph)**.** The Tanner Graph of a linear code determined by the parity-check matrix $H = \{h_{ij}\}, i = 1, \ldots, r, j = 1, \ldots, n$ is a bipartite graph whose one set of nodes corresponds to the checks of H (check nodes correspond to bits encoded by linear combinations of information bits) and the other set of nodes corresponds to the set of code symbols (variable nodes). A check node c_i is connected with a variable node v_j if $h_{ij} \neq 0$.

Example 3.2. Recall that the parity-check matrix $H = (1\ 1\ 1)$ of the $[3, 2]$ code gives a single check node connected to the three variable nodes. Its Tanner graph is thus



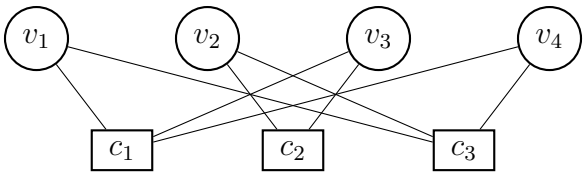
This matches Example 3.1: each valid codeword (c_1, c_2, c_3) must satisfy $c_1+c_2+c_3 = 0$ in $\mathbb{F}_2$ at the single check node.

Example 3.3. Alternatively, consider the binary linear code whose parity-check matrix is

$$H = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix}.$$

The Tanner graph of this code has

- 4 variable–nodes v_1, v_2, v_3, v_4, one for each column of H,
- 3 check–nodes c_1, c_2, c_3, one for each row of H,
- an edge between v_j and c_i exactly when $h_{ij} = 1$.



3.2.2 Quasi-Cyclic LDPC Codes

Quasi-cyclic (QC) LDPC codes form a very popular class of structured low-density parity-check codes. Their "quasi-cyclic'' nature makes both hardware implementation and analysis significantly simpler, while still preserving most of the good error-correction performance of LDPC ensembles.

Definition 3.11 (Circulant Matrix)**.** An $M \times M$ *circulant matrix* is one in which each row is a right-cyclic shift of the previous row. Equivalently, it is generated by a single row vector, and every subsequent row is obtained by rotating that vector one position to the right.

Circulant matrices are convenient building blocks because they can be implemented with very little memory and because cyclic shifts correspond to simple wiring patterns in hardware.

Definition 3.12 (Quasi-Cyclic Linear Code)**.** Let $\mathcal{C} \subseteq \mathbb{F}_q^n$ be a linear $[n, k]$ code. We say $\mathcal{C}$ is ℓ-quasi-cyclic if for every

$$\mathbf{c} = (c_1, c_2, \ldots, c_n) \in \mathcal{C},$$

the ℓ-position cyclic shift

$$(c_{n-\ell+1},\ c_{n-\ell+2},\ \ldots,\ c_n,\ c_1,\ c_2,\ \ldots,\ c_{n-\ell})$$

also lies in $\mathcal{C}$.

Definition 3.13 (Base Graph)**.** The base graph $\mathcal{G}_B = \{\mathcal{E}_B, \mathcal{C}_B\}$ is a graph, whose incidence matrix is given by the base matrix B. The girth of the base graph is denoted by g_B and is the length of the shortest cycle in $\mathcal{G}_B$.

One designs B (and later the circulant shifts) to maximize g_B, since short cycles degrade iterative decoding performance.

Concretely, a QC-LDPC code is often described by a *polynomial* parity-check matrix

$$H(D) \ = \ \big[h_{ij}(D)\big] \quad i = 1, \ldots, c-b, \ \ j = 1, \ldots, c, \quad h_{ij}(D) \in \{0,\, D^{w_{ij}}\},$$

where D is a formal indeterminate and each exponent $w_{ij} \in \{0, 1, \ldots, M-1\}$ (or -1 for $h_{ij} = 0$). To obtain the actual binary matrix of size $(c-b)M \times cM$, one replaces each monomial $D^{w_{ij}}$ by the $M \times M$ circulant permutation matrix $I_M(w_{ij})$ (and each -1 by the all-zero block).

The underlying base matrix is simply

$$B \ = \ H(D)\Big|_{D=1},$$

and the *degree matrix* $W = [\, w_{ij} \,]$ records all the shifts, using $w_{ij} = -1$ to indicate the zero-block positions.

Example 3.4. Consider the simplest nontrivial QC construction with $M = 2$, $c = 2$, $b = 1$, so rate $1/2$. Let the degree matrix be

$$W \ = \ \big[\, 0 \ \ 1 \,\big],$$

meaning the polynomial parity-check is $H(D) = [1 \;\; D]$. Replacing

$$1 \;\longmapsto\; I_2(0) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad D \;\longmapsto\; I_2(1) = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},$$

we obtain the 2×4 parity-check matrix of binary block QC-LDPC code

$$H \;=\; \big[I_2(0) \;\big|\; I_2(1)\big] = \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}.$$

3.3 Reed–Solomon Codes

Reed-Solomon (RS) codes are a class of non-binary maximum-distance-separable (MDS) codes defined over finite fields. To introduce them, we first recall the necessary algebraic structures of fields and extension fields.

Definition 3.14 (Field)**.** A *field* $(F, +, \times)$ is a set F equipped with two binary operations (addition and multiplication) such that

- $(F, +)$ is an abelian group with identity 0,
- $(F \setminus \{0\}, \times)$ is an abelian group with identity 1,
- multiplication distributes over addition.

Definition 3.15 (Finite Field)**.** A finite field (or Galois field) of size q, denoted $\mathrm{GF}(q)$, is a field with exactly q elements. Such a field exists if and only if $q = p^m$ for some prime p and integer $m \geq 1$. In the special case $m = 1$, one has

$$\mathrm{GF}(p) \cong \mathbb{Z}/p\mathbb{Z},$$

i.e. it is isomorphic to the prime field of integers modulo p.

Example 3.5 (Extension Field)**.** If $q = p^m$ with $m > 1$, one constructs GF(q) as the residue class ring

$$\mathrm{GF}(p)[x] \;/\; \langle p(x) \rangle$$

where $p(x) \in \mathrm{GF}(p)[x]$ is an irreducible (often primitive) polynomial of degree m. Its elements are represented by polynomials of degree $< m$, and arithmetic is performed modulo $p(x)$.

For example, if α is a root of a primitive polynomial of degree 3 over $\mathrm{GF}(2)$, then $\mathrm{GF}(2^3) = \{0, 1, \alpha, \ldots, \alpha^6\}$ and the nonzero elements form a cyclic multiplicative group of order 7. Table 2 lists this group.

Table 2. Multiplicative group of GF(2^3) generated by a primitive element α, with $\alpha^3 = \alpha + 1$.

Power	Polynomial form	Binary form
α^0	1	001
α^1	α	010
α^2	α^2	100
α^3	$1+\alpha$	011
α^4	$\alpha+\alpha^2$	110
α^5	$1+\alpha+\alpha^2$	111
α^6	$1+\alpha^2$	101

Definition 3.16 (Reed–Solomon Code)**.** Let GF(q) be a finite field and choose $n \leq q-1$ distinct nonzero evaluation points $\beta_1, \ldots, \beta_n \in$ GF(q). An $[n, k]$ Reed-Solomon code over GF(q) is the set of all codewords

$$\big(c_1, \ldots, c_n\big) \;=\; \big(f(\beta_1),\, \ldots,\, f(\beta_n)\big)$$

where $f(x)$ ranges over all polynomials in GF$(q)[x]$ of degree $< k$. Such a code has dimension k, length n, and minimum distance $d_{\text{min}} = n - k + 1$.

Equivalently, one can define the RS code as the cyclic code with generator polynomial

$$g(x) \;=\; (x-\alpha^b)(x-\alpha^{b+1})\cdots(x-\alpha^{b+n-k-1}) \quad \subset \text{GF}(q)[x]/(x^n-1),$$

where α is a primitive nth root of unity in $GF(q)$ and b is a design parameter (often $b = 1$).

Example 3.6. Over GF$(8) = \{0, 1, \alpha, \ldots, \alpha^6\}$ with $\alpha^3 = \alpha + 1$, an $[8, 6]$ RS code evaluates all polynomials of degree < 6 at the eight field elements $\{0, \alpha^0, \alpha^1, \ldots, \alpha^6\}$. For instance, the message

$$(m_0, m_1, m_2, m_3, m_4, m_5) = (1, \alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5)$$

corresponds to

$$f(x) = m_0 + m_1 x + m_2 x^2 + m_3 x^3 + m_4 x^4 + m_5 x^5 \quad \Longrightarrow \quad \big(f(0), f(\alpha^0), f(\alpha^1), \ldots, f(\alpha^6)\big).$$

Minimum distance of this code is $d = 8 - 6 + 1 = 3$, so it can correct up to 2 symbol erasures or 1 symbol error.

Usually, Reed-Solomon codes are the most used erasure-correcting method in storage such as disk arrays or compact disks [11]. They offer good recovery at the cost of relatively high

decoding complexity. However, for the current application, LDPC codes were chosen instead since they allow for faster single-bit recovery without having to read too many other disks (in other words, they have better *locality*). This means requests can be serviced faster without having to process too much data. Additionally, a Monte Carlo simulation was ran, comparing the bit error rates of the chosen LDPC code and an $[8, 6]$ RS code (both with rate $3/4$). Simulations were run for each erasure probability from 0.01 to 0.5 in increments of 0.01. For each, 10^6 random erasure patterns were generated and decoding attempted. The bit error rate (BER) was then recorded and averaged for both codes. The results can be seen in figure 5. Since the RS code showed higher error rates in the comparison of these short codes, LDPC codes are used for the remainder of this thesis.

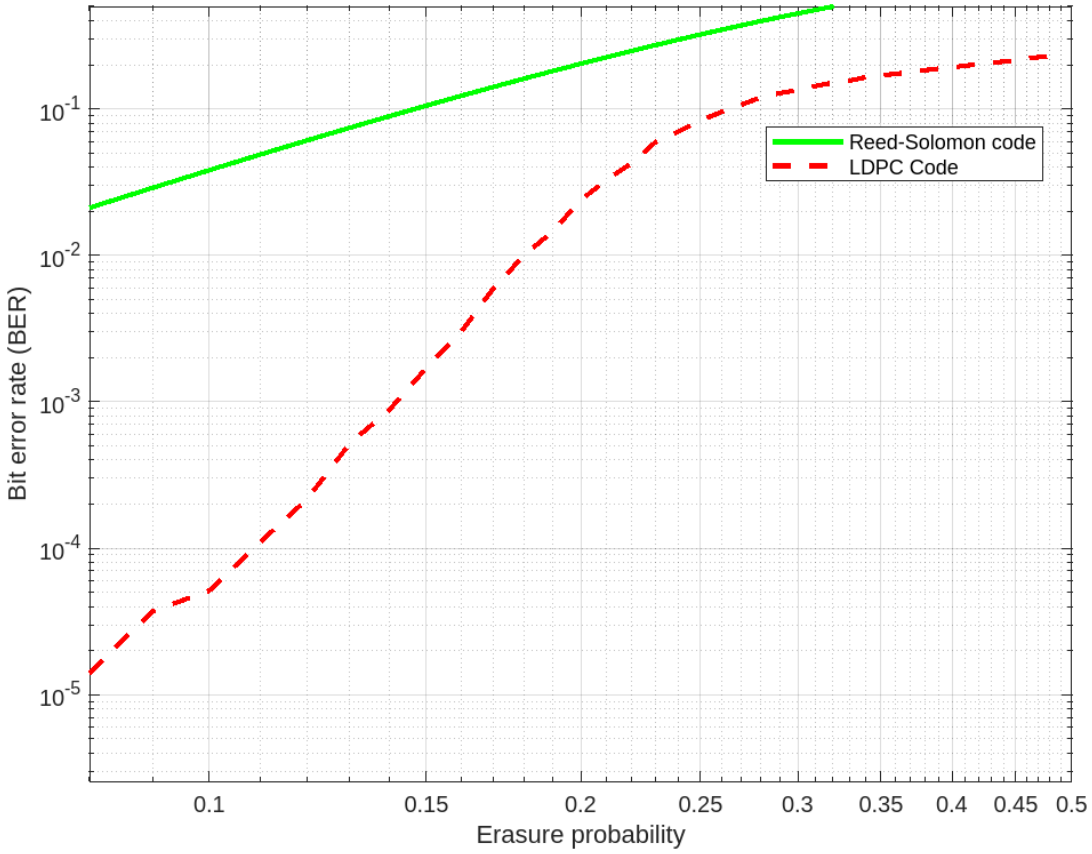


Figure 5. BER comparison of the chosen LDPC code and $[8, 6]$ RS code.

3.4 Chosen Code for Modelling

For simplicity of implementation and decoding, this thesis applies a QC-LDPC code for all of the following analysis. The chosen code is from the paper [12]. Specifically, the $(156, 119)$ QC-LDPC code constructed with a circulant size $M = 13$ and row weight $K = 12$ is utilized. This code is detailed in Table II of the referenced paper, which presents degree matrices for QC-LDPC codes with girth $g = 6$. The parameters of this code are as follows:

- **Code length**: $n = 156$
- **Code dimension**: $k = 119$
- **Code rate**: $R = \frac{k}{n} = \frac{119}{156} \approx \frac{3}{4}$
- **Circulant size**: $M = 13$
- **Row weight**: $K = 12$
- **Column weight**: $J = 3$
- **Girth**: $g = 6$
- **Minimum distance**: $d_{min} = 6$

The degree matrix W for this code is given as:

$$W = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 10 & 11 & 12 \\ 0 & 3 & 1 & 8 & 2 & 9 & 12 & 4 & 11 & 5 & 7 & 6 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

Each entry w_{ij} in the matrix W corresponds to a circulant permutation matrix $I_{13}(w_{ij})$, which is the 13×13 identity matrix cyclically shifted to the right by w_{ij} positions. The full parity-check matrix H of size $(3 \cdot 13) \times (12 \cdot 13) = 39 \times 156$ is constructed by replacing each entry w_{ij} in W with its corresponding permutation block $I_{13}(w_{ij})$. This specific code was chosen for two practical reasons.

Firstly, the code is of moderate length. With a length of 156 bits, the code is short enough to allow manageable simulations and practical implementations in disk arrays, yet long enough to demonstrate meaningful error-correcting performance.

Secondly, it has a practical rate. A rate of approximately $3/4$ ensures efficient use of bandwidth while still providing sufficient redundancy for error correction.

However, it should be emphasized that the particular LDPC code selected here serves only as an example. In practical implementations, an alternative code could and should be chosen to meet the specific performance, complexity, and rate requirements of the system. This may mean using another code family altogether, for example Reed-Solomon codes should also be considered. The code presented above was selected solely for its relative simplicity and clarity in illustrating the principles of the proposed model.

4. Decoding

Usually, in the case of erasure correction, the aim of decoding is to recover the entire codeword. However, this is not the case for the proposed storage application. Any given decoder only needs to recover a single bit: the requested one. By focusing on just one recovery, known decoding algorithms can be altered to achieve a better average complexity.

Decoding erasures in a binary linear code is equivalent to solving a system of linear equations over the finite field $\mathbb{F}_2$. Indeed, each row of the parity-check matrix H imposes one linear constraint on the codeword symbols. Let $\mathcal{C} \subset \mathbb{F}_2^n$ be an $[n, k]$ binary linear code with parity-check matrix

$$H = [h_{a,i}] \in \mathbb{F}_2^{(n-k)\times n}.$$

Suppose the received word $\mathbf{r} \in \{0, 1, ?\}^n$ exhibits an erasure set $\mathbf{E} = \{i_1, \ldots, i_e\}$. Denote by $H_{\mathbf{E}}$ the submatrix of H consisting of the e columns indexed by $\mathbf{E}$, and let

$$\mathbf{s} = -\sum_{i \notin \mathbf{E}} H_{*,i}\, r_i = \sum_{i \notin \mathbf{E}} H_{*,i}\, r_i \in \mathbb{F}_2^{n-k} \quad \text{(syndrome from the known bits)}.$$

Then the erased bits $\{c_{i_j}\}_{j=1}^{e}$ must satisfy

$$H_{\mathbf{E}}\, \mathbf{c}_{\mathbf{E}}^T = \mathbf{s}, \qquad \mathbf{c}_{\mathbf{E}} = (c_{i_1}, \ldots, c_{i_e}) \in \mathbb{F}_2^e.$$

If $H_{\mathbf{E}}$ has full column rank and $e \leq n - k$, this system has a unique solution, recovering all erasures. Here, $H_{*,i}$ denotes the i-th column of H, i.e. the vector $(h_{1,i}, h_{2,i}, \ldots, h_{n-k,i})^T \in \mathbb{F}_2^{n-k}$.

4.1 Gaussian Elimination

Gaussian elimination is a classical direct method for solving systems of linear equations over a field [13]. In the context of erasure decoding, it allows recovery of erased symbols by solving a small linear system derived from the parity-check constraints. This approach is costly with regards to complexity, but ensures, that if a solution exists, it will be found.

Algorithm 1 Gaussian Elimination for Erasure Decoding

Input: Parity-check submatrix $H_{\mathbf{E}} \in GF_2^{m \times e}$, syndrome $\mathbf{s} \in GF_2^m$

Output: Estimated erasure vector $\mathbf{c_E} \in GF_2^e$ solving $H_{\mathbf{E}}\, \mathbf{c}_{\mathbf{E}}^T = \mathbf{s}$

Form augmented matrix $\left[H_{\mathbf{E}} \mid \mathbf{s}\right]$ of size $m \times (e+1)$
for $j = 1$ to $\min(e, m)$ **do** ▷ Eliminate column j
    Find pivot row $i \geq j$ with entry $(i, j) = 1$; **fail** if none exists
    Swap rows $i \leftrightarrow j$
    for $i' = j + 1$ to m **do** ▷ Clear below pivot
        if $(i', j) = 1$ **then**
            $\text{Row}_{i'} \leftarrow \text{Row}_{i'} + \text{Row}_j$
        end if
    end for
end for

for $j = \min(e, m)$ down to 1 **do** ▷ Back-substitution

$$c_{\mathbf{E}}[j] \leftarrow \left(\left[\text{entry in col } (e+1) \text{ of row } j \right] - \sum_{k=j+1}^{e} H_{\mathbf{E}}[j,k]\, c_{\mathbf{E}}[k] \right) \bmod 2$$

end for
return $\mathbf{c_E}$

In binary arithmetic each row addition costs $O(e)$. We perform at most e pivot steps, each clearing up to m rows, so the forward elimination takes $O(m\, e^2)$. Back-substitution adds another $O(e^2)$, for a total of $O(m\, e^2 + e^2) \approx O(e^3)$ when m is on the order of e.

For the current application however, a single erased symbol c_j is only ever needed to be recovered, rather than the entire vector $\mathbf{c_E}$. A faster method utilizing this factor for this is proposed here.

By reordering columns so that our target j is first, and then performing elimination only on that one column, using the original parity-check rows to clear off all other entries in row 1, we can recover c_j faster.

Algorithm 2 One-Bit Erasure Decoding

Input: Parity-check submatrix $H_{\mathbf{E}} \in \mathbb{F}_2^{m \times e}$, syndrome $\mathbf{s} \in \mathbb{F}_2^m$, target index $j \in \mathbf{E}$

Output: Value of the erased bit c_j

Permute the columns of $H_{\mathbf{E}}$ so that j becomes column 1.
Make copies $H^{(\text{orig})} \leftarrow H_{\mathbf{E}}$, $\mathbf{s}^{(\text{orig})} \leftarrow \mathbf{s}$.
Find any row p with $H_{\mathbf{E}}[p, 1] = 1$. **fail** if none exists.
Swap rows $1 \leftrightarrow p$ in both $H_{\mathbf{E}}$ and $\mathbf{s}$. ▷ Now row 1 has a pivot in column 1
for $k = 2, \ldots, e$ **do**
    if $H_{\mathbf{E}}[1, k] = 1$ **then**
        Find a row $q > 1$ with $H^{(\text{orig})}[q, k] = 1$. **fail** if none.
        $H_{\mathbf{E}}[1, *] \leftarrow H_{\mathbf{E}}[1, *] + H^{(\text{orig})}[q, *]$
        $s_1 \leftarrow s_1 + s_q^{(\text{orig})}$
    end if
end for
return $c_j = s_1$

By targeting only the single requested symbol instead of the entire erasure set, Algorithm 2 replaces the $min(e, m)$ pivot steps and full forward-elimination over all e columns by a single pivot search in the target column and at most one row addition per other column. It then returns the recovered bit immediately, with no back-substitution loop over all erased symbols. In practice this means performing only $O(e)$ simple row-addition and lookup operations rather than the $O(e^2)$ operations of full Gaussian elimination, yielding a lower average decoding time.

4.2 Belief Propagation. The Peeling Decoder

Belief propagation, also known as the sum-product algorithm, is an iterative message-passing procedure [7, 14]. In the coding context one views the Tanner graph of an LDPC code as a bipartite factor graph, with variable nodes corresponding to code symbols and check nodes to parity constraints. At each iteration, variable nodes send "belief" messages to their neighboring check nodes, passing along their own symbol values. Each check node then computes and returns updated belief messages to its neighboring variables by enforcing its parity equation. These two steps are repeated until the beliefs converge. This approach tends to be fast, but performs rather poorly with regards to codeword recovery.

On the binary erasure channel, belief-propagation decoding specializes to the peeling decoder, which iteratively removes ("peels") known symbols from the Tanner graph. In each iteration one finds a check node whose current erasure-degree is exactly 1, recovers its single unknown neighbor by equating it to the syndrome at that check, and then updates all adjacent checks. Repeating until no degree-one checks remain either fully recovers all erasures or halts in a stopping set. [7, 14]

The full peeling decoder processes each edge at most once, so it runs in $O(E)$ time where E is the number of edges in the subgraph induced by the erasures. To recover only one target bit, we simply halt as soon as that variable node is resolved. Since the decoder explores only the portion of the graph reachable via degree-one checks before hitting the target, the average number of edges visited is much smaller than E, giving a substantially lower expected complexity for single-bit recovery.

The peeling decoder will not always recover the full codeword (or, in our case, the target bit) whenever the remaining erased variable subgraph contains a stopping set. A stopping set is a nonempty subset of variable nodes such that every check node adjacent to that subset has degree at least two within the induced subgraph. Once the peeling decoder removes all degree-one checks, it halts. Any remaining erasures form a stopping set and cannot be resolved by further message-passing, since no check provides a single pivot equation.

A slightly modified single-bit decoder implementation is shown here, that iteratively checks if the requested bit was recovered and halts as soon as the value is known. This helps keep the peeling decoder fast for single-bit recovery.

Algorithm 3 Single-Bit Peeling Decoder

Input: Tanner subgraph of erasures: variable-to-check adjacency lists for $H_{\mathbf{E}}$, syndrome vector $\mathbf{s}$, target variable $j \in \mathbf{E}$

Output: Recovered bit c_j or **fail**

Initialize queue $\mathcal{Q} \leftarrow \{\text{all checks of erasure-degree } 1\}$
Mark all erased variables as "unknown"
while $\mathcal{Q} \neq \emptyset$ **do**
    Remove a check node c from $\mathcal{Q}$
    Let v be the unique erased neighbor of c
    Recover $c_v \leftarrow s_c$ ▷ all other incident bits to c are known
    if $v = j$ **then return** c_j
    end if
    for all check nodes c' adjacent to v (other than c) **do**
        Update syndrome $s_{c'} \leftarrow s_{c'} + c_v$
        Remove edge (v, c') from the graph
        if new erasure-degree of c' is 1 **then**
            add c' into $\mathcal{Q}$
        end if
    end for
end while
fail ▷ no degree-one check reached j

4.3 A Combined Algorithm for Single-Bit Recovery

To leverage the speed of peeling decoding while retaining the guaranteed recovery of Gaussian elimination, a new combined decoder is proposed in Algorithm 4. It first attempts to resolve the target erasure via the fast peeling decoder; if that fails, it collects all variables solved during peeling, forms a reduced linear system over the remaining erasures, and then applies the one-bit Gaussian elimination (Algorithm 2) to recover the requested bit. This hybrid approach runs faster than Gaussian elimination, but in all cases finds the correct bit whenever it is recoverable.

Algorithm 4 Combined Single-Bit Recovery

Input: Parity-check submatrix $H_{\mathbf{E}}$, $\mathbf{s}$, $j \in \mathbf{E}$

Output: Recovered bit c_j or **fail**

(Peeling phase) Run Algorithm 3 on $(H_{\mathbf{E}}, \mathbf{s}, j)$
if peeling returns c_j **then**
    return c_j
end if

(Reduction) Let $\mathcal{V}_{\text{solved}}$ be all variables recovered during peeling
Remove columns $\mathcal{V}_{\text{solved}}$ from $H_{\mathbf{E}}$ and update $\mathbf{s}$ accordingly
Denote the remaining submatrix and syndrome by H' and $\mathbf{s}'$, and let $e' = |\mathbf{E} \setminus \mathcal{V}_{\text{solved}}|$

(Gaussian fallback) Run Algorithm 2 on $(H', \mathbf{s}', j)$
return result of one-bit Gaussian elimination

Because most erasure patterns are peeled out in the first phase, the combined decoder typically terminates in $O(E_{\text{peel}}) \ll O(e^3)$ time. Only for the few patterns that form stopping sets does it incur the $O(e^3)$ cost of Gaussian elimination on the much smaller residual system. And since this system is usually reduced significantly after peeling, the average case runtime is greatly improved. Hence the average runtime is much better than standard Gaussian elimination while still guaranteeing recovery whenever any direct elimination approach would succeed. Figure 6 compares the average microsecond run-times of pure peeling, pure Gaussian elimination, and the combined decoder as the number of erasures increases.

The runtime values were obtained by generating 10^6 erasure patterns for all numbers of erasures from 1 to $n/2 = 78$ for the QC-LDPC code introduced previously. Then, each erased bit in the pattern was requested separately and the runtime of the different algorithms was computed.

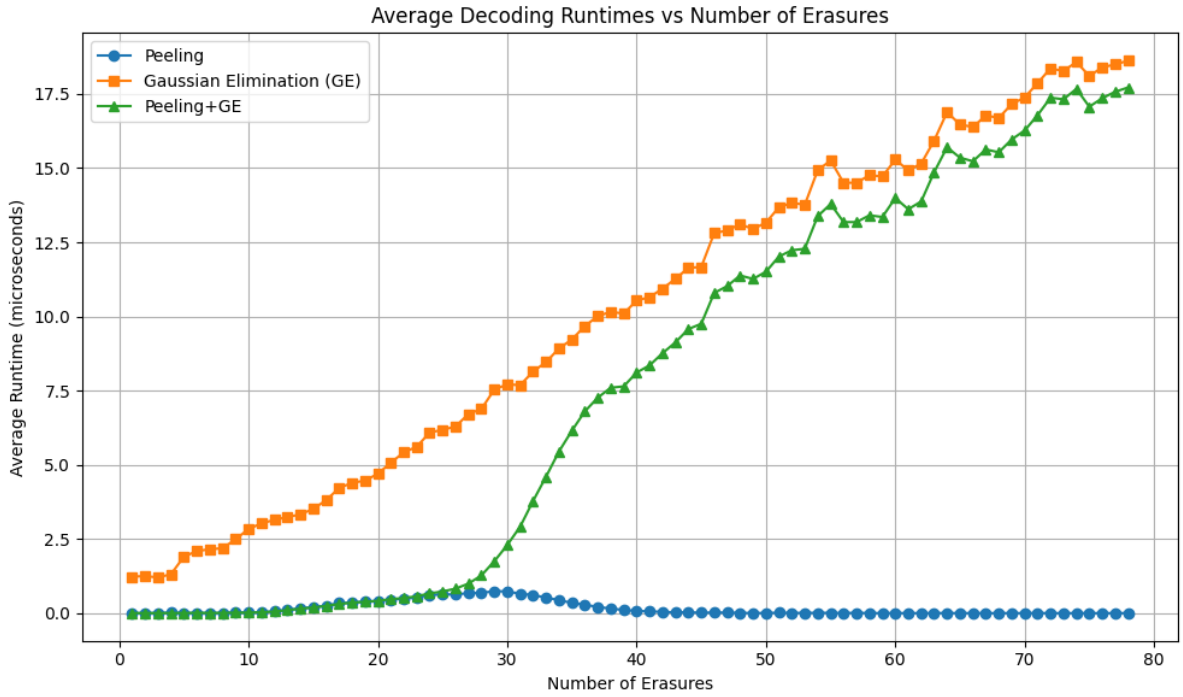


Figure 6. Runtime comparison of models

These were then averaged to produce the final figures. It can be seen in Figure 6 that for a relatively small number of erasures (up to about 25), the peeling decoder is often sufficient, since the runtime curves of the combined approach and peeling decoding coincide. Then, for some number of erasures (about 25-40), Gaussian elimination is used more often, affecting the runtime of the combined approach. After about 40 erasures, both decoders are mostly insufficient at recovering the requested bit, but since the peeling decoder fails fast and very often passes a significantly reduced system to Gaussian elimination, the fail condition is still evaluated faster on average than pure Gaussian elimination.

5. Proposed Model for Energy Saving

To integrate coding-theoretic redundancy into an energy-aware storage array, we map each codeword bit to a physical disk. The encoding operation thus corresponds to XOR-combinations across disks, and erasure decoding corresponds to activating only those disks necessary to reconstruct requested data. A turned off disk therefore corresponds to an erasure in the codeword. For the chosen QC-LDPC code we have $n = 156$ disks in total, $k = 119$ data disks and $n - k = 37$ parity disks.

Due to having to service write-requests, we only consider turning off some subset of information disks since writing data always also incurs a write-request to disks that hold redundancy (XOR of disks must be reevaluated). By keeping the 37 parity disks always running, we mitigate the need for more complex modelling. Note that further energy saving could be possible, if we only handled read-requests. Then it would be possible to switch off the subset of redundant disks whose corresponding information disks are all turned off and therefore render the redundant disk unimportant. However, this is not feasible in a practical system where write-requests must also be serviced.

5.1 Modelling Probability of Requests for Disks

In practical storage deployments, access patterns are usually skewed: a small set of disks holding "hot" data receives the bulk of I/O requests over any given time interval, while the majority store "cold" data that is accessed only sporadically. By continuously profiling, or predicting the per-disk access frequency, a storage controller can identify those cold disks and transition them into low-power or standby states. This allows us to exploit the non-uniformity of real-world workloads to realize substantial energy savings while maintaining target performance guarantees. For further analysis it is assumed that the popularity of disks in the model is known.

To capture skew in request popularity across disks, we model the index of the requested disk in some discrete time interval as a geometric random variable. Let $I \in \{1, 2, \ldots, k\}$ denote the disk index (with $k = 119$) sorted in descending order of popularity. We can then model the probability of a disk being requested within the given time interval as

$$\Pr[I = i] = \frac{p^{i-1}\,(1-p)}{1-p^k}, \qquad 0 < p < 1,$$

where $i = 1, 2, ..., n$, so that disk 1 is most likely to be requested, disk 2 slightly less, and so on, with the normalizing denominator accounting for the finite population. Let $\alpha = 1 - p$ be the parameter of the geometric distribution. By tuning α we can change how much the "hot" and

”cold” ends of the distribution differ. For higher values of α, the distribution is steeper, meaning the discrepancy between the popularity of ”hot” and ”cold” disks is increased. Consequently, as α increases, more disks can be turned off.

5.2 Approach to Energy Saving

In order to exploit the non-uniform request distribution for energy savings, we introduce a cutoff threshold $\tau \in (0, 1)$ such that any disk with estimated access probability $\Pr[I = i] < \tau$ is transitioned into a low-power (off) state. If a request arrives for a powered-down disk, we first attempt reconstruction via the proposed decoding algorithm. Only if decoding fails do we incur the cost of spinning the disk up.

Let P_{spinup} and P_{idle} denote the disk power draw in active and spinup modes, respectively. Leaving a disk idle for a time interval costs P_{idle}, whereas powering it down and incurring a spin-up event with probability p_{spinup} yields an expected energy cost of

$$p_{\text{spinup}} \cdot P_{\text{spinup}}.$$

Thus, disabling the disk is advantageous whenever

$$p_{\text{spinup}} \cdot P_{\text{spinup}} < P_{\text{active}} \quad \Longleftrightarrow \quad p_{\text{spinup}} < \frac{P_{\text{active}}}{P_{\text{spinup}}}.$$

With values $P_{\text{active}} \approx 5\,\text{W}$ and $P_{\text{spinup}} \approx 15\,\text{W}$, this inequality becomes $p_{\text{spinup}} < \frac{1}{3}$. However, this is a very naive figure, that does not take into account time spent on recovery for requests or other overheads. In practical systems, we wish to keep spinup events to a minimum, so spinning up a disk for approximately every third request is not feasible. To accommodate for workload uncertainty and spin-up overheads more conservatively, markers $p_{\text{spinup}} = 0.1$ and $p_{\text{spinup}} = 0.01$ are chosen instead, allowing at most 10% or 1% of accesses to necessitate a disk spin-up, respectively.

To assess the practicality of the energy-saving scheme, a Monte Carlo simulation was conducted under three geometric distributions with p values for the distribution

$$p \in \{0.98,\, 0.96,\, 0.94\}$$

that correspond to α values $\alpha \in \{0.02,\, 0.04,\, 0.06\}$. For each value of α, the cutoff threshold τ was discretized over the interval $[0,\, 1/k]$ into 50 equally-spaced points. At each threshold:

1. k independent requests from the geometric distribution were generated.

2. All disks whose access probability fell below τ were turned off.

3. Each of the requests was fulfilled with three strategies:

   (a) **No decoding:** If the requested disk is active, return the value immediately, otherwise, incur a spin-up and then return.

   (b) **Peeling only:** Attempt recovery via the peeling decoder; if it fails, spin up the disk.

   (c) **Combined (marked GE):** Attempt recovery via the hybrid peeling + one-bit Gaussian elimination, if that fails, spin up the disk.

4. The above was repeated for 10^5 independent trials per threshold τ and recorded:

   - The *error rate* p_{spinup}, defined as the fraction of requests that required a disk spin-up.
   - The *disk-off fraction*, i.e. the proportion of disks powered down at threshold τ.

Figure 7 plots the tradeoff between p_{spinup} and the diskoff fraction for each α.

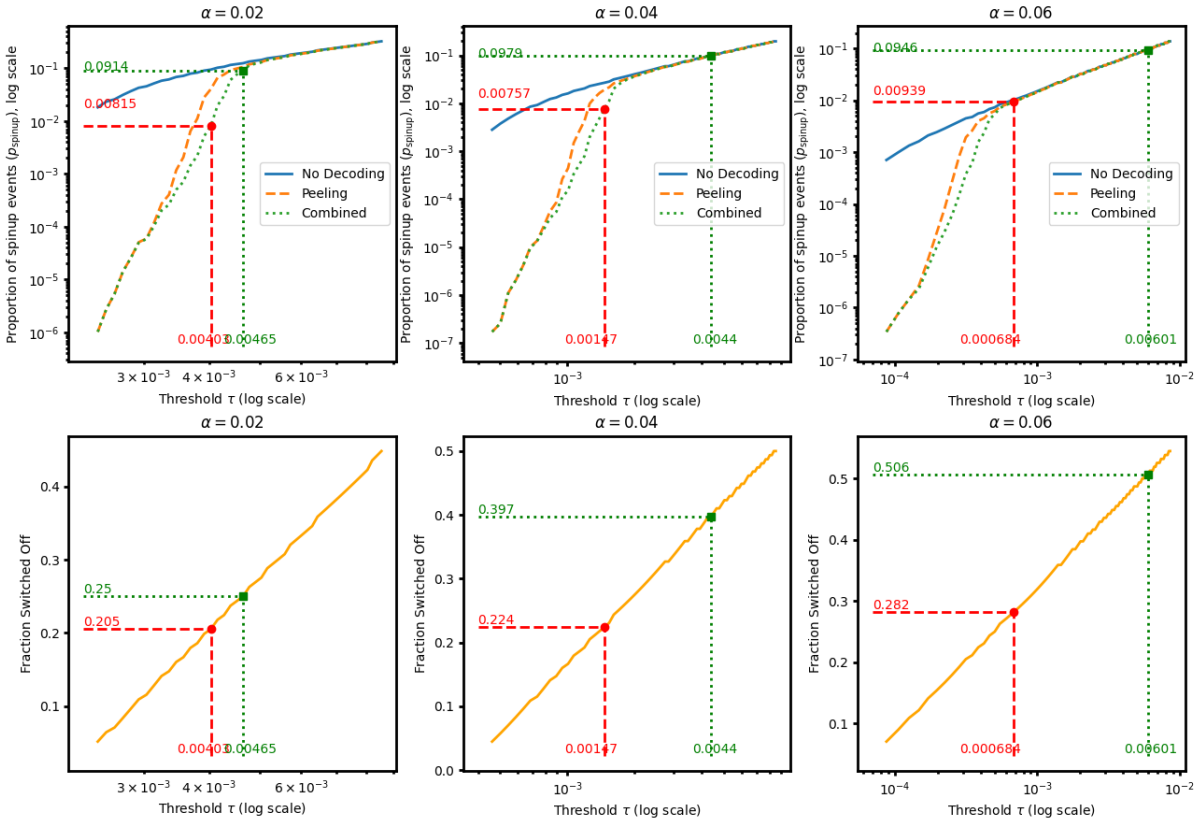


Figure 7. Fraction of disks powered down versus spin-up probability p_{spinup} for decoding policies under geometric distributions $\alpha = 0.02, 0.04, 0.06$. Marked points indicate the cutoff τ achieving $p_{\text{spinup}} = 1\%$ and 10%.

It can be seen that the combined decoder outperforms simple peeling and thus further analysis will use values from the performance of the combined decoder. The marked points mark the cutoff thresholds achieving $p_{\text{spinup}} = 1\%$ and 10%. For example, when $\alpha = 0.04$ the 10% spin-up constraint permits deactivating approximately 39.7% of the disks.

To calculate the saved energy with this approach, one must first calculate the total power usage of the system without energy saving. This is simply the sum of the power drawn by all active disks

$$P_{\text{all}} = n \cdot P_{\text{active}} = 156 \times 5\text{W} = 780\text{W}$$

for all the disks and

$$P_{\text{info}} = k \cdot P_{\text{active}} = 119 \times 5\text{W} = 595\text{W}$$

for a naive scheme that only includes the information disks. For $\alpha = 0.04$, with $p_{\text{spinup}} = 0.1$ and the fraction of disks turned off as 0.397 we can find the average power consumption of the model as

$$\begin{aligned} P_{\text{model}} &= (1 - 0.397)\, n\, P_{\text{active}} + 0.397\, n\, p_{\text{spinup}}\, P_{\text{spinup}} \\ &= 0.603 \times 156 \times 5\,\text{W} + 0.397 \times 156 \times 0.1 \times 15\,\text{W} \approx 563.24\,\text{W}. \end{aligned}$$

This amounts to around 28% and 5% power savings when compared to P_{all} and P_{info} respectively. Table 3 shows the performance metrics for all calculated values for $p_{\text{spinup}} = 0.1$ and $p_{\text{spinup}} = 0.01$. The table shows the percentage of turned off disks, power consumed by the system and power saved in comparison to P_{all} and P_{info}. Bold values indicate cases where energy-saving was achieved.

Table 3. Performance metrics under varying spin-up probabilities and geometric request distributions

	$p_{\text{spinup}} = 0.1$				$p_{\text{spinup}} = 0.01$			
Geom. dist. parameter α	Off%	P_{model}	vs P_{all}	vs P_{info}	Off%	P_{model}	vs P_{all}	vs P_{info}
$\alpha = 0.02$	25.0%	643.50W	**-17%**	+4%	20.5%	624.90W	**-20%**	+5%
$\alpha = 0.04$	39.7%	563.24W	**-28%**	**-5%**	22.4%	610.52W	**-22%**	+3%
$\alpha = 0.06$	59.6%	454.58W	**-42%**	**-24%**	28.2%	566.64W	**-27%**	**-5%**

Note that being able to deactivate more than 25% of the disks (more than the redundancy) almost always provides strictly greater energy savings than a naive scheme in which all parity disks are permanently powered down and only the k information disks remain active. And since, in real-world deployments one must also guard against unplanned hardware failures, which means some amount of redundancy is always required, the naive approach of no redundant disks is not feasible. The proposed storage architecture employs a QC-LDPC code of minimum Hamming distance $d_{min} = 6$, which guarantees recovery from any combination of up to 5 simultaneous erasures. Consequently, the system preserves full resilience against up to five unexpected disk failures in the disk array.

6. Conclusion

A hybrid single-bit recovery strategy has been developed that combines the speed of peeling decoding with the guaranteed correctness of Gaussian elimination. By attempting a lightweight peeling pass first and reverting to a reduced one-bit Gaussian solve only when necessary, the combined decoder achieves near-peeling runtimes on typical erasure patterns while still recovering any bit that lies in the column-rank of the induced parity-check submatrix. Monte Carlo experiments on a $(156, 119)$ QC-LDPC code confirm that average decoding latency closely tracks the fast peeling curve up to moderate erasure counts.

Energy-aware storage was modeled by mapping codeword bits to disks and exploiting skewed disk-request distributions to power down cold disks. Simulation under geometric popularity models demonstrates that incorporating the hybrid decoder allows a large fraction of disks to remain off while bounding spin-up rates, yielding energy savings up to 42% under certain distributions. Additionally, The QC-LDPC code's distance guarantee further ensures resilience against up to five simultaneous unplanned failures.

Further Research

- **Alternative code families.** Investigation of other erasure-correcting codes such as Reed–Solomon codes may yield different trade-offs in decoding cost, storage overhead and failure resilience.
- **Integration with predictive frameworks.** Coupling the hybrid decoder with workload prediction methods that utilize AI (e.g. PAP) could further reduce spin-up events by adapting decoding versus spin-up decisions to anticipated request bursts.
- **Temporal constraints.** Extending the model to handle time-varying request intensities or batching of multiple requests may improve energy-performance trade-offs in practical workloads.
- **Adaptive thresholding.** Dynamic adjustment of disk-off thresholds based on recent workloads and predictions may further increase the energy saved. Additionally, separate thresholds could be introduced for redundant disks, to also save energy in parity disks.
- **Modeling on a physical system.** A physical application of the proposed model should be tested to assess the applicability and bottlenecks of the approach.

References

[1] IEA. Tracking Clean Energy Progress 2023. Tech. rep. Licence: CC BY 4.0. Paris: IEA, 2023. https://www.iea.org/reports/tracking-clean-energy-progress-2023.

[2] IEA. Energy and AI. Tech. rep. Licence: CC BY 4.0. Paris: IEA, 2025. https://www.iea.org/reports/energy-and-ai.

[3] Arora S. and Bala A. PAP: Power Aware Prediction based Framework to Reduce Disk Energy Consumption. *Cluster Computing* 23 (2020), pp. 3157–3174. DOI: 10.1007/s10586-020-03077-3.

[4] Balakrishnan S., Black R., Donnelly A., England P., Glass A., Harper D., Legtchenko S., Ogus A., Peterson E., and Rowstron A. Pelican: A Building Block for Exascale Cold Data Storage. *Proceedings of the 11th USENIX Symposium on Operating Systems Design and Implementation*. Broomfield, CO: USENIX Association, Oct. 2014. https://www.usenix.org/conference/osdi14/technical-sessions/presentation/balakrishnan.

[5] Shannon C. E. A Mathematical Theory of Communication. *Bell System Technical Journal*. Vol. 27. Reprinted with corrections from The Bell System Technical Journal. July 1948, pp. 379–423, 623–656. https://www.itsoc.org/publications/shannons-mathematical-theory-of-communication.

[6] Roth R. M. Introduction to Coding Theory. Cambridge, UK: Cambridge University Press, 2006.

[7] Bocharova I. and Kudryashov B. Practical Error Correcting Coding. April, 2025. University of Tartu, Faculty of Science and Technology, Institute of Computer Science, 2025.

[8] Gallager R. G. Low-Density Parity-Check Codes. *IRE Transactions on Information Theory* 8.1 (1962), pp. 21–28. DOI: 10.1109/TIT.1962.1057683.

[9] Zhao K., Zhao W., Sun H., Zhang T., Zhang X., and Zheng N. LDPC-in-SSD: Making Advanced Error Correction Codes Work Effectively in Solid State Drives. *Proceedings of the 11th USENIX Conference on File and Storage Technologies (FAST '13)*. USENIX Association, 2013, pp. 243–256.

[10] IEEE Standard for Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specific requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 5: Enhancements for Higher Throughput. IEEE Std.

802.11n-2009. IEEE, 2009. http://standards.ieee.org/getieee802/download/802.11n-2009.pdf.

[11] Immink K. A. S. Reed–Solomon Codes and the Compact Disc. Ed. by Wicker S. B. and Bhargava V. K. New York, NY: Wiley–IEEE Press, 1994.

[12] Bocharova I. E., Hug F., Johannesson R., Kudryashov B. D., and Satyukov R. V. Searching for Voltage Graph–Based LDPC Tailbiting Codes with Large Girth. *IEEE Transactions on Information Theory* (2012). DOI: 10.1109/TIT.2011.2176717.

[13] Atkinson K. A. An Introduction to Numerical Analysis. 2nd. New York, NY: John Wiley & Sons, 1989.

[14] Narayanan K. R. The Peeling Decoder: Theory and Some Applications. Tech. rep. Lecture notes, NASIT 2016. Durham, NC: North American School of Information Theory, Duke University, 2016. http://pfister.ee.duke.edu/nasit16/Narayanan.pdf.

License

Non-exclusive licence to reproduce Thesis and make Thesis public

I, Karl Kristjan Puusepp,

1. grant the University of Tartu a free permit (non-exclusive licence) to reproduce, for the purpose of preservation, including for adding to the digital archives of the University of Tartu until the expiry of the term of copyright, my thesis

   Model for Saving Energy in Distributed Systems Using QC-LDPC Codes,

   supervised by Vitaly Skachek, PhD, Irina Bocharova, PhD, Boris Kudryashov, PhD;

2. grant the University of Tartu a permit to make the thesis specified in point 1 available to the public via the web environment of the University of Tartu, including via the digital archives, under the Creative Commons licence CC BY NC ND 4.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright;

3. am aware of the fact that the author retains the rights specified in points 1 and 2;

4. confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Karl Kristjan Puusepp

15/05/2025