Error Correction by Natural Redundancy for Long Term Storage

Anxiao (Andrew) Jiang
CSE Department
Texas A&M University
College Station, TX 77843
ajiang@cse.tamu.edu

Pulakesh Upadhyaya
CSE Department
Texas A&M University
College Station, TX 77843
pulakesh@tamu.edu

Erich F. Haratsch
Seagate Technology
Fremont, CA 94538

Jehoshua Bruck
EE & CNS Dept.
Caltech
Pasadena, CA 91125
bruck@caltech.edu

***Abstract*—Non-volatile memories are increasingly important for big-data storage. However, their long-term data reliability has significant challenges. This work studies how to use the *natural redundancy* in data for error correction. The natural redundancy can be combined with error-correcting codes to effectively improve data reliability. This work studies several aspects of natural redundancy: effective discovery of natural redundancy in compressed data, error-correction capability of error-correcting codes with natural redundancy, and efficient decoding of random codes that model data with natural redundancy.**

I. Introduction

Non-volatile memories (NVMs) have become increasingly important for big-data storage. With this great success, a significant challenge also emerges: how to recover data from errors as effectively as possible, especially for long term storage? Many NVMs, such as flash memories and phase-change memories, have data retention problems, where charge leakage or cell-level drifting makes data more noisy over time. Operations such as reads and writes cause accumulative disturbance in NVM data. Furthermore, erasures of NVM cells degrade cell quality and make cells more prone to errors over time. There is a strong motivation in elevating the long-term data reliability in NVMs to the next level.

The most effective way to protect data conventionally has been error-correcting codes (ECCs). The recent advancement in learning and the availability of big data for study have offered a new opportunity for error correction: *to use the natural redundancy in data for error correction*. By natural redundancy, we refer to the inherent redundancy in data that is not artificially added for error correction, such as various features in languages and images, structural features in databases, etc. Due to practical reasons (e.g., high complexity for compression, and lack of precise models for data), even after compression, lots of redundancy still exists.

This work studies how to use the natural redundancy in data for error correction. It is a topic related to joint source-channel coding and denoising. It extends the work in [1], [2], [3], [4], where texts compressed by Huffman coding were studied. Several new aspects of the topic are studied in this paper.

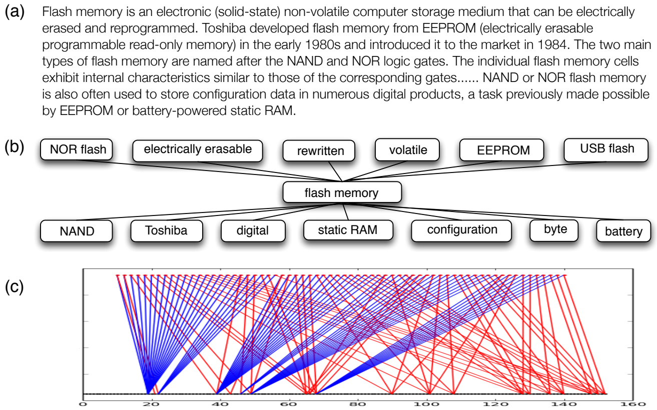
II. Efficient Discovery of Natural Redundancy

We illustrate the potential of natural redundancy with an example.

Example 1: Natural redundancy in languages. Take the English language as an example. Character-wise Huffman coding for English texts achieves the rate of 4.59 bits/character; a Markov model for 3-grams (which uses dependency between three consecutive characters) achieves 3.06 bits/character; and when LZW compression is used based on a dictionary of $2^{20} \approx 1$ million patterns (which is much larger than practically used LZW dictionaries), the rate is further reduced to 2.94 bits/character. However, Shannon has estimated that the true entropy of English texts is upper bounded by 1.34 bits/character. That means over 54% of the data after LZW compression is still redundant, which is a great resource for error correction.

We have applied natural language processing (NLP) techniques to the English language, which discovered different forms of redundancy useful for error correction. We illustrate here one type of such redundancy, *co-location*. Consider the sample text from Wikipedia in Fig. 1 (a). The co-location relationship (which means certain phrases appear in similar contexts much more frequently than usual) can be obtained from training data. When applied to testing, they reveal lots of dependency in texts, which often span whole articles. For example, for the above sample text, given the phrase "flash memory", its closely related phrases by the co-location relationship are illustrated in Fig. 1 (b). They exist in different places of the text, not necessarily beside the phrase "flash memory". It means natural redundancy can be *global*. The *global natural redundancy* is illustrated more clearly in Fig. 1 (c). Let us first partition the sample text into phrases (such as "Flash memory", "is an", "electronic", $\cdots$) by NLP, and show those phrases as dots (in the same order as in the text) at the bottom of the Tanner graph in Fig. 1 (c). If two phrases have the co-location relationship, they are connected by a red dot at the top through a blue edge and a red edge. Such relationships resemble parity checks in ECCs. □

We study LZW coding that uses a fixed dictionary of 2^{20} patterns, where every 20-bit LZW codeword represents a compressed variable-length string of text characters. We have designed an efficient decoding algorithm for error correction by detecting valid words, phrases, and co-location relationships, and use them to filter candidate solutions for each LZW codeword. Then a hard-decision decoding method is used: if all remaining candidate solutions agree on a bit, decode the



Flash memory is an electronic (solid-state) non-volatile computer storage medium that can be electrically erased and reprogrammed. Toshiba developed flash memory from EEPROM (electrically erasable programmable read-only memory) in the early 1980s and introduced it to the market in 1984. The two main types of flash memory are named after the NAND and NOR logic gates. The individual flash memory cells exhibit internal characteristics similar to those of the corresponding gates...... NAND or NOR flash memory is also often used to store configuration data in numerous digital products, a task previously made possible by EEPROM or battery-powered static RAM.

Fig. 1. (a) A sample text (part of which was omitted due to space limit). (a) Phrases in it that have the co-location relationship with "flash memory". (b) Tanner graph for phrases with co-location relationship in the sample text.

bit to that value; otherwise, keep it as an erasure. Note that the algorithm does not use any redundancy from ECC.

We illustrate its performance for the binary-erasure channel (BEC). The output of the decoding algorithm has both erasures and errors. (It will be given to ECC for further decoding.) Let $\epsilon \in [0,1]$ be the erasure probability before decoding. After the decoding by natural redundancy, let $\delta \in [0,1]$ denote the probability that an originally erased bit remains as an erasure, and let $\rho \in [0, 1-\delta]$ denote the probability that an originally erased bit is decoded to 0 or 1 incorrectly. Then the amount of noise after decoding can be measured by the entropy of the noise (erasures and errors) per bit: $E_{NR}(\epsilon) \triangleq \epsilon(\delta + (1-\delta)H(\frac{\rho}{1-\delta}))$, where $H(p) = -p\log p - (1-p)\log(1-p)$ is the entropy function. Some typical values of $E_{NR}(\epsilon)$ are shown below. The reduction in noise by the algorithm is $\frac{\epsilon - E_{NR}(\epsilon)}{\epsilon}$. The table shows that noise is reduced effectively (from 88.0% to 91.6%) for the LZW compressed data (without any help from ECC), for raw bit-erasure rate (RBER) from 5% to 30%, which is a wide range.

ϵ	0.05	0.15	0.30
$E_{NR}(\epsilon)$	0.00418	0.0142	0.0360
Noise reduction	91.6%	90.6%	88.0%

We have also designed a decoding algorithm for compressed images. Due to space limitation, we skip the details here.

III. ECC with Natural Redundancy

We can protect compressed data with a systematic ECC, and concatenate the natural-redundancy (NR) decoder with the ECC decoder. Given a noisy ECC codeword, the NR decoder uses natural redundancy to not only correct some errors, but also recognize some bits as being probably correct (because they form likely patterns). The NR decoder then sends the updated codeword bits with updated *a priori* error probabilities to the ECC decoder for further decoding. Note that the NR decoder can decode not only information bits, but also parity-check bits due to parity-check constraints. The achievable rate of ECC with natural redundancy can be analyzed accordingly. We present a result for binary-symmetric channel (BSC).

Theorem 1. *Consider a binary symmetric channel (BSC) with error probability p. Assume a natural-redundancy decoder can examine the output bits of the channel and adjust some bits' error probability to $q < p$. Assume every bit is adjusted this way independently with probability r. Then the capacity of this compound channel is $C = r(1-H(q)) + (1-r)(1-H(p))$. Furthermore, for the case $q = 0$ (namely, the natural-redundancy decoder can fully determine the values of some bits), consider an ECC of length n that can correct up to t errors given that the natural-redundancy decoder can determine the values of π codeword bits right before ECC decoding. Then the size of the ECC, $|C|$, is upper bounded by $|C| \le \frac{2^n}{\sum_{i=0}^{t}\binom{n-\pi}{i}}$.*

IV. Efficient Decoding of Random Codes

We study how to optimize the efficiency of the NR decoder. For languages, the compressed bits of valid words/phrases (e.g., by Huffman coding) can be seen as a random code. Given a noisy binary word $\mathbf{y} = (y_1, \cdots, y_n)$, we need to find a valid codeword $\mathbf{x} = (x_1, \cdots, x_n)$ at a small Hamming distance from $\mathbf{y}$ (for MAP decoding) without checking many candidate words despite the code's lack of structures (for low computational complexity). We have designed data structures to achieve this objective.

Theorem 2. *Let $\mathcal{C} \subseteq \{0,1\}^n$ be a random code whose codewords are chosen independently and uniformly at random. Let $\mathbf{x} \in \mathcal{C}$ be a codeword, and let $\mathbf{y} \in \{0,1\}^n$ be its received noisy word at Hamming distance t. Let $m \le n-t$ and $k \ge 1$ be positive integers. There exists a data structure with which one can on average check just $k\left(\frac{\binom{n-t}{m}}{\binom{n}{m}} + (|\mathcal{C}|-1)(\frac{1}{2})^m\right)$ candidate words in $\{0,1\}^n$ such that the codeword $\mathbf{x}$ is among the checked words with probability $1 - \left(1 - \frac{\binom{n-t}{m}}{\binom{n}{m}}\right)^k$.*

Note that without the data structure, an exhaustive search of $\binom{n}{t}$ candidate words is needed, which is very costly. For example, when $n = 48$ and $t = 10$, for n-bit English words compressed by Huffman coding, we can choose parameters m and k for the data structure such that only less than 250 (instead of $\binom{48}{10} \gg 250$) candidate words need to be checked on average, and the correct codeword is included in the checked words with probability at least 0.99.

References

[1] A. Jiang, Y. Li, and J. Bruck, "Enhanced Error Correction via Language Processing," in *Proc. Non-Volatile Memories Workshop*, 2015.

[2] Y. Li, Y. Wang, A. Jiang and J. Bruck, "Content-assisted File Decoding for Nonvolatile Memories," in *Proc. 46th Asilomar Conference on Signals, Systems and Computers*, pp. 937–941, Pacific Grove, CA, 2012.

[3] J. Luo, Q. Huang, S. Wang and Z. Wang, "Error Control Coding Combined with Content Recognition," in *Proc. 8th International Conference on Wireless Communications and Signal Processing*, pp. 1–5, 2016.

[4] Y. Wang, M. Qin, K. R. Narayanan, A. Jiang and Z. Bandic, "Joint Source-channel Decoding of Polar Codes for Language-based Sources," in *Proc. IEEE Global Communications Conference (Globecom)*, 2016.