Data Deduplication with Random Substitutions

Hao Lou, Farzad Farnoud
Electrical and Computer Engineering, University of Virginia, VA, USA. Email: {hl2nu,farzad}@virginia.edu

Abstract—**Data deduplication saves storage space by identifying and removing repeats in the data stream. In this paper, we provide an information-theoretic analysis of the performance of deduplication algorithms with data streams where repeats are not exact. We introduce a source model in which probabilistic substitutions are considered. Two modified versions of fixed-length deduplication are studied and proven to have performance within a constant factor of optimal with the knowledge of repeat length. We also study the variable-length scheme and show that as entropy becomes smaller, the size of the compressed string vanishes relative to the length of the uncompressed string.**

I. Introduction

The task of reducing data storage costs is gaining increasing attention due to the explosive growth of digital data, especially redundant data [1]–[3]. Compared with traditional data compression approaches, data deduplication is more efficient in dealing with large scale data. It has been widely used in mass data storage systems, e.g., LBFS (low-bandwidth network file system) [4] and Venti [5]. In this paper, we study the performance of data deduplication algorithms when repeated data segments are not necessarily exact copies from an information-theoretic point of view.

In general, a typical data deduplication system uses a chunking scheme to cut the data stream into multiple data "chunks" [6]. The chunks can be of equal length (fixed-length chunking) or of lengths that are content-defined (variable-length chunking) [7]. The fixed-length scheme has low complexity but suffers from the boundary-shift problem since the boundaries of the chunks do not necessarily align with the boundaries of repeated substrings, thus making the copies different from each other. In the variable-length scheme, chunk breakpoints are determined by some pre-defined patterns and therefore repeated data segments can be identified regardless of their positions. The chunks are processed sequentially. Each chunk is put into the dictionary at the first occurrence, and the duplicates are replaced by pointers to the dictionary.

An information-theoretic analysis of deduplication algorithms was first performed by Niesen [8]. Niesen's work introduced a source model, formalized deduplication algorithms in both fixed-length and variable-length schemes, and analyzed their performance. We adopt a similar strategy in this paper. The source model introduced by Niesen produces data strings that are composed of blocks with each block being an exact copy of one of the source symbols, where the source symbols are pre-selected strings. In practice it is often the case, however, that the copies of a block of data that is repeated many times are approximate, rather than exact. This may occur, for example, due to edits to the data, or in the case of genomic data[1], due to mutations. Thus, in our source model, we add probabilistic substitutions to each block, resulting in data strings composed of approximate copies of the source symbols.

We analyze data deduplication in both the fixed-length scheme and the variable-length scheme. For the fixed-length scheme, two simple modifications of the formalized algorithm in [8], the *double fixed-length* deduplication and the *fixed-length edit-distance* deduplication, are presented and analyzed. We show that when all source symbols have the same length, these modified versions of the fixed-length algorithm perform well with proper choices of parameters while the conventional algorithm fails. For sources with random symbol lengths, we show that the variable-length scheme can achieve large compression ratios relative to the uncompressed length.

A large amount of work has been done in the area of data deduplication and [6] serves as a comprehensive survey. Besides [8], [10] also analyzed deduplication from an information-theoretic point of view but with a different source model and algorithm. The problem of deduplication under edit errors was also considered in [11]. While [11] focuses on performing deduplication of two files, one being an edited version of another by insertions and deletions, we consider a single data stream with substitution errors.

The rest of the paper is organized as follows. Notation and preliminaries are given in the next section. In section III, we present the information source model. In Section IV, we formally describe the deduplication algorithms and give our metric of performance. Bounds on the performance of algorithms in fixed-length schemes and variable-length schemes are derived in Sections V and VI, respectively. Due to space limitation, some of the proofs are omitted or sketched.

II. Preliminary

Let the alphabet Σ be $\{0,1\}$. The set of all finite strings over Σ is denoted Σ^*. For some positive integer m, let Σ^m be the set of all strings of length m over Σ. Strings are denoted by boldface letters such as $\boldsymbol{x}$, while symbols by normal letters such as x. For any strings $\boldsymbol{u}, \boldsymbol{v} \in \Sigma^*$, the concatenation of $\boldsymbol{u}$ and $\boldsymbol{v}$ is denoted $\boldsymbol{uv}$ and the concatenation of i copies of $\boldsymbol{u}$ is denoted $\boldsymbol{u}^i$. The length of $\boldsymbol{u}$ is denoted by $|\boldsymbol{u}|$. The cardinality of some set S is also denoted by $|S|$.

In this paper, all logarithms are to base 2. For any $0 \leq x \leq 1$, we use $H(x)$ to denote the binary entropy function, i.e., $H(x) = x\log(1/x) + (1-x)\log(1/(1-x))$. For any event $\mathcal{E}$, we use $\mathcal{E}^c$ to denote the complement of $\mathcal{E}$.

[1] Repeats are common in genomic data. For example, a majority of the human genome consists of interspersed and tandem repeated sequences [9].

The k-runlength-limited (k-RLL) strings [12] are binary strings in which the runs of 0's have length at most $k-1$. For any nonnegative integer n, we denote the set of all k-RLL strings of length n by R_k^n.

Next, we give two lemmas that will be used in the rest of the paper. Both Lemmas 1 and 2 below can be proven by induction, but the proofs are omitted.

Lemma 1. *The number of k-RLL sequences of length n, $|R_k^n|$, is less than $2(2-1/2^k)^n$.*

Lemma 2. *For any $x \in (0,1)$ and positive integer n,*

$$\max(0, 1-nx) \le (1-x)^n \le \max(1/2, 1-nx/2).$$

III. Source model

The source model studied in this paper extends the one described in [8] by allowing probabilistic substitutions. We also adopt the notation of [8] to the extent possible for the sake of consistency. Let the source alphabet be $\mathcal{X}$, with $|\mathcal{X}| = A$. Specifically, the source alphabet $\mathcal{X}$ contains A strings over Σ, denoted $\mathsf{X}_1, \dots, \mathsf{X}_A$. Fix a probability distribution $\mathbb{P}_s$ over $\mathbb{N}_{>0}$ with mean L. The A source symbols $\mathsf{X}_1, \dots, \mathsf{X}_A$ are generated iid as follows. For each $1 \le a \le A$, X_a is chosen from Σ^{L_a} uniformly at random, where L_a is a positive integer drawn independently of other quantities from the distribution $\mathbb{P}_s$. To simplify some of the derivations, we adopt the same assumption as in [8] that $\mathbb{P}_s$ is concentrated around its mean, specifically, $\mathbb{P}_s(L/2 \le L_a \le 2L) = 1$ for all a.

After generating the source alphabet $\mathcal{X}$, we generate the source string $\boldsymbol{s}$ in the following way. Choose B strings $Y_1', \dots, Y_B'$ independently uniformly at random from $\mathcal{X}$ with replacement. For every Y_b', we flip each of its bits with probability δ as a way to simulate edits and other changes to the data in a simple manner. We will refer to δ as the *error rate*. The modified version of Y_b' will be denoted Y_b and referred to as a *source block*. The source string $\boldsymbol{s}$ is then constructed to be the concatenation of $Y_1, \dots, Y_B$, i.e., $\boldsymbol{s} = Y_1 \cdots Y_B$. The entropy of this source is denoted $H(\boldsymbol{s})$.

Note that each Y_b is an altered version of some source symbol in $\mathcal{X}$. We say that Y_b is a descendant of X_a if $Y_b' = \mathsf{X}_a$. For a fixed source alphabet $\mathcal{X} = \{\mathsf{X}_1, \dots, \mathsf{X}_A\}$ and source string $\boldsymbol{s}$, we denote the set of all descendants of X_a in $\boldsymbol{s}$ by $Y(\mathsf{X}_a)$.

In this paper, we study the asymptotic regime in which $B \to \infty$. The error rate δ is a constant. We assume that $L = B^{1/k}$ for some constant $k > 1$, reflecting the larger size of the data set relative to the length of individual elements. We also assume $A = o(B^{1-\epsilon})$ for some $0 < \epsilon < 1$ to ensure that, on average, each symbol is repeated many times and thus deduplication can be effective. Under these assumptions, we compute the entropy of our source model in the following lemma with the proof omitted.

Lemma 3. *As $B \to \infty$, the entropy of the above source model $H(\boldsymbol{s})$ satisfies*

$$H(\delta)BL \le H(\boldsymbol{s}) \le H(\delta)BL + o(BL).$$

Let $\mathcal{E}_l$ be the event that $|Y(\mathsf{X}_a)| \ge \frac{B}{2A}$ for all a and $\mathcal{E}_u$ be the event that $|Y(\mathsf{X}_a)| \le \frac{3B}{2A}$ for all a. We have

$$\Pr(\mathcal{E}_l) \ge 1 - Ae^{-\frac{B}{8A}}, \quad \Pr(\mathcal{E}_u) \ge 1 - Ae^{-\frac{B}{10A}}, \qquad (1)$$

as a direct consequence of the Chernoff bound and the union bound. Since $B/10A - \log A$ goes to infinity, the probability of $\mathcal{E}_l$ goes to 1 (also true for $\mathcal{E}_u$). Therefore, in the performance analysis of deduplication algorithms in Sections V and VI, we will focus on the case when $\mathcal{E}_l$ or $\mathcal{E}_u$ is true and show that the effect on the performances when $\mathcal{E}_l^c$ or $\mathcal{E}_u^c$ holds is asymptotically negligible.

IV. Deduplication schemes and performance metric

A. Deduplication schemes

We next describe the deduplication algorithms that are studied in this paper.

The *double fixed-length* deduplication algorithm has two parameters, segment length D and chunk length $\ell, \ell \le D$. The source string $\boldsymbol{s}$ is first parsed into segments of length D, denoted by $S_1, \dots, S_K$ with $K = \lceil |\boldsymbol{s}|/D \rceil$ (S_K may be of length less than D). Next, each segment S_j will be parsed into substrings of length ℓ, denoted $Z_1^j, \dots, Z_C^j$ with $C = \lceil D/\ell \rceil$ (the last substring may be of length less than ℓ). Note that substring number of the last segment may be less than C. The substrings $\{Z_c^j\}_{c,j}$ are then taken as chunks. The algorithm starts by initializing an empty dictionary for storing chunks. Next, the length of the source string $|\boldsymbol{s}|$ is encoded by a prefix-free code to make sure the whole scheme is prefix-free. Then the chunks $\{Z_c^j\}_{c,j}$ will be encoded sequentially. If any chunk Z_c^j is not in the dictionary, this chunk will be encoded with a 1 followed by itself, and Z_c^j is added into the dictionary. If chunk Z_c^j is already in the dictionary, then it will be encoded with a 0 followed by a pointer to the dictionary. The number of bits fixed-length deduplication takes to encode $\boldsymbol{s}$ is denoted $\mathcal{L}_F(\boldsymbol{s})$.

In the *fixed-length edit-distance* deduplication algorithm, two parameters, chunk length ℓ and edit distance $t, t \le \frac{1}{2}\ell$, are fixed. The source string $\boldsymbol{s}$ is parsed into substrings of length ℓ, denoted by $Z_1, \dots, Z_C$ with $C = \lceil |\boldsymbol{s}|/\ell \rceil$. The algorithm again initializes an empty dictionary and encodes the length of the source string $|\boldsymbol{s}|$. Next, for each chunk Z_c, it is encoded with a 1 followed by itself if there exist no string within Hamming distance t of Z_c in the dictionary. If there exists a string in the dictionary that is within Hamming distance t of Z_c as a reference, then Z_c will be encoded with a 0 followed by a pointer to the dictionary and the positions that the reference string and Z_c differ. Since we only encode the difference within Hamming distance t, we need at most $\log \sum_{i=0}^{t} \binom{\ell}{t} + 1 \le \ell H(t/\ell) + 1$ bits. The number of bits needed by fixed-length edit-distance deduplication for source string $\boldsymbol{s}$ is denoted by $\mathcal{L}_{ED}(\boldsymbol{s})$.

The *variable-length* deduplication algorithm is formalized in [8] and restated here. We fix the all-zero string of length M, 0^M, to be the marker. The source string $\boldsymbol{s}$ is then split into chunks by this marker. Specifically, the source string $\boldsymbol{s}$ is parsed as $\boldsymbol{s} = Z_1 \cdots Z_C$, where each Z_c (except perhaps the last one) contains a single appearance of 0^M, which appear at its end. After splitting $\boldsymbol{s}$ into the chunks $\{Z_c\}_c$, the same encoding process as in double fixed-length deduplication will be conducted. The expected number of bits for variable-length deduplication to encode $\boldsymbol{s}$ is denoted $\mathcal{L}_{VL}(\boldsymbol{s})$.

In all three algorithms, the pointer for some chunk $\boldsymbol{z}$ that is already in the dictionary can be encoded in $\log|T_{\boldsymbol{z}}| + 1$ bits, where $|T_{\boldsymbol{z}}|$ is the size of the dictionary at the time $\boldsymbol{z}$ is processed for the first time. In the following, when the chunking scheme is clear from the context, we use T to denote the final dictionary after the chunking phase is complete. For any string $\boldsymbol{w}$, we use $\boldsymbol{w} \in T$ to denote the event that $\boldsymbol{w}$ appears as a chunk.

B. Performance metric

We measure performance of the deduplication algorithms as source model parameters increase along with B. Since the entropy of the source model is asymptotically $H(\delta)\mathbb{E}[|\boldsymbol{s}|]$ by Lemma 3, we are particularly interested in the performance for different values (but fixed as $B \to \infty$) of δ, especially when δ is close to 0. For an algorithm with expected number of bits normalized by the expected length of the source string being R, e.g., $R = \mathbb{E}[\mathcal{L}_F(\boldsymbol{s})]/\mathbb{E}[|\boldsymbol{s}|]$, we consider it to behave poorly if there exists constant c such that $R \geq c$ for all δ. We say the algorithm is within a constant factor of optimal if there exists a constant c such that $R \leq cH(\delta)$ for all δ.

V. Fixed-length deduplication

In this section, we derive bounds on the performance of the fixed-length chunking schemes over the source model described in Section III. It is pointed out by [8] that when the source symbols all have the same length ($\mathbb{P}_s$ is degenerate), fixed-length deduplication preforms well, while when symbols have different lengths, the loss of synchronization leads to poor performance. The question of interest is then whether fixed-length deduplication can still perform well when symbols have the same length but their copies in $\boldsymbol{s}$ are not exact. Therefore, in the rest of this section, we assume the data string is produced by the source where source symbols are all of length L.

A. Double fixed-length deduplication

Consider the case when double fixed-length deduplication with parameters D and ℓ is performed on data string $\boldsymbol{s}$. If we pick $D = L$ with some foresight, then $\boldsymbol{s}$ is first parsed into segments being exactly the source blocks $Y_1, \ldots, Y_B$. We assume that ℓ divides L for simplification of the derivations, it will be clear from the proofs that this assumption leads to no loss of generality. Let $C = L/\ell$. Each Y_b, $1 \leq b \leq B$, is then parsed into chunks $Z_1^b, \ldots, Z_C^b$ with $\left|Z_c^b\right| = \ell$ for all $1 \leq c \leq C$. Therefore, for double fixed-length chunking, T is the set $\{Z_c^b\}_{b,c}$ and for any $\boldsymbol{w} \in \Sigma^\ell$, $\boldsymbol{w} \in T$ is equivalent to $\boldsymbol{w} = Z_c^b$ for some b, c.

Lemma 4. *Let $\boldsymbol{s}$ be parsed into chunks $\{Z_c^b\}_{b,c}$ by the double fixed-length chunking scheme with parameters $D = L$ and ℓ. We have*

$$\Pr(\boldsymbol{w} \in T|\mathcal{E}_l) \geq \frac{1}{2}\mathbb{E}_d\big[\min\big(1, BL\delta^d(1-\delta)^{\ell-d}/(2\ell)\big)\big], \quad (2)$$

$$\Pr(\boldsymbol{w} \in T|\mathcal{E}_u) \leq \frac{AL}{\ell}\mathbb{E}_d\big[\min\big(1, 3B\delta^d(1-\delta)^{\ell-d}/(2A)\big)\big], \quad (3)$$

where d is a random variable with distribution $\Pr(d = x) = \binom{\ell}{x}/2^\ell, x = 0, 1, \ldots, \ell$.

The proof of Lemma 4 is omitted due to space limitation. Note that as a direct consequence of Lemma 4, we have that the probability of $\boldsymbol{w}$ appearing as a chunk in the first half of $\boldsymbol{s}$ given $\mathcal{E}_l$ is at least

$$\frac{1}{2}\mathbb{E}_d\big[\min\big(1, BL\delta^d(1-\delta)^{\ell-d}/(4\ell)\big)\big]. \quad (4)$$

Theorem 5. *Consider the source model in which source symbols have the same length L. For the double fixed-length deduplication with $D = L$, we have*

$$\mathbb{E}[\mathcal{L}_F(\boldsymbol{s})]/\mathbb{E}[|\boldsymbol{s}|] \geq \frac{1}{32}(1 + o(1)), \quad \textit{as } B \to \infty,$$

if $\frac{\log BL}{H(\delta)} \leq \ell \leq L$ or $\omega(1) \leq \ell \leq \frac{\log BL}{\frac{1}{2}\log 1/(\delta(1-\delta))}$.

Proof: For all $\ell \geq \frac{\log BL}{H(\delta)}$, we have $\frac{BL}{2\ell}\delta^{\delta\ell}(1-\delta)^{\ell-\delta\ell} \leq 1$. It can be shown by Lemma 4 and [13, Theorem 1] that for any $\boldsymbol{w} \in \Sigma^\ell$, $\Pr(\boldsymbol{w} \in T|\mathcal{E}_l) \geq \frac{1}{16}\frac{BL}{\ell 2^\ell}$. The expected size of the dictionary in bits (i.e., those bits in the compressed string that describe a chunk observed for the first time) equals $\mathbb{E}[|T|]\ell$. It follows that as $B \to \infty$,

$$\begin{aligned}\mathbb{E}[\mathcal{L}_F(\boldsymbol{s})] &\geq \mathbb{E}[|T|]\ell \geq \Pr(\mathcal{E}_l)\mathbb{E}[|T||\mathcal{E}_l]\ell \\ &= \Pr(\mathcal{E}_l)\sum_{\boldsymbol{w}\in\Sigma^\ell}\Pr(\boldsymbol{w} \in T|\mathcal{E}_l)\ell \geq \frac{1}{32}BL(1 + o(1)).\end{aligned}$$

For all $\ell \leq \frac{\log BL}{\frac{1}{2}\log 1/(\delta(1-\delta))}$, it can be shown by (4) and Markov's inequality that given $\mathcal{E}_l$, with probability at least $\frac{1}{5}$, $\frac{1}{16}$ of the total 2^ℓ length-ℓ strings will appear in the first half of $\boldsymbol{s}$. Let $T_{1/2}$ denote the dictionary for the first half of $\boldsymbol{s}$. Any chunk in the second half is encoded either in full or via a pointer to the dictionary. As $B \to \infty$, $\mathbb{E}[\mathcal{L}_F(\boldsymbol{s})]$ is greater than

$$\Pr(\mathcal{E}_l)\mathbb{E}\big[\min\big(\log\big|T_{1/2}\big|, \ell\big) \cdot BL/(2\ell)|\mathcal{E}_l\big] \geq \frac{BL}{10}(1 + o(1)).$$

■

Note that by letting $\ell = L$, the double fixedlength deduplication becomes the standard fixed-length deduplication with chunk length L, which is shown in [8] to be close to optimal on the source model with 0 error rate. From Theorem 5 we can see that even with the knowledge of L, not choosing ℓ properly (e.g., setting $\ell = L$ as in the regular algorithm) can cause $\mathbb{E}[\mathcal{L}_F(\boldsymbol{s})]$ to be asymptotically larger than entropy by an arbitrarily large multiplicative factor as δ goes to 0. This poor behavior results from the fact that for large ℓ, the noisy copies are not likely to appear multiple times and so deduplication is ineffective, while small ℓ leads to an oversized dictionary.

Theorem 6. *Consider the source model with $A = o(B^{1-\epsilon}), L = B^{1/k}$. If source symbols all have the same length L, the performance of double fixed-length deduplication with $D = L$ and $\ell = \frac{\gamma \log B}{H((1+\alpha)\delta)}$ satisfies*

$$1 \leq \frac{\mathbb{E}[\mathcal{L}_F(\boldsymbol{s})]}{H(\boldsymbol{s})} \leq \frac{1}{\gamma}\left(1+\frac{1}{k}\right)\frac{H((1+\alpha)\delta)}{H(\delta)}(1+o(1)),$$

as $B \to \infty$, for any $0 < \gamma \leq \epsilon, 0 < \alpha \leq \frac{1}{2\delta} - 1$.

Proof: Since the Elias [14] code allows us to encode the length of $\boldsymbol{s}$ in $2\log BL + 3 = o(BL)$ bits, and since $H(\boldsymbol{s}) = \Theta(BL)$, the contribution of encoding the length of $\boldsymbol{s}$ to $\mathcal{L}_F$ is negligible (absorbed into the $o(1)$ term above).

Next, we compute the bits used for the dictionary (the first time some chunk appears) in two cases, $\mathcal{E}_u$ and $\mathcal{E}_u^c$. Since there are always at most BL/ℓ chunks in dictionary and each costs $\ell + 1$ bits, we have $\Pr(\mathcal{E}_u^c)\mathbb{E}[|T||\mathcal{E}_u^c](\ell+1) = o(1)$ by (1). While given $\mathcal{E}_u$, it can be shown by Lemma 4, Lemma 2 and the Chernoff bound that for any $\boldsymbol{w} \in \Sigma^\ell$, any $0 < \alpha \leq \frac{1}{2\delta} - 1$,

$$\begin{aligned}\Pr(\boldsymbol{w} \in T|\mathcal{E}_u) &\leq \frac{AL2^{\ell H((1+\alpha)\delta)}}{\ell 2^\ell} + \frac{3BL}{2\ell 2^\ell}e^{-\frac{\alpha^2\delta\ell}{2+\alpha}} \\ &= \frac{o(B)L}{\ell 2^\ell} + o(1). \qquad (5)\end{aligned}$$

Thus, $\mathbb{E}[|T||\mathcal{E}_u] \leq \sum_{\boldsymbol{w}\in\Sigma^\ell} \Pr(\boldsymbol{w} \in T|\mathcal{E}_u) = o(B)\frac{L}{\ell} + o(B)$. So the chunks in the dictionary, plus the bits indicating that each one is new, take $\mathbb{E}[|T||\mathcal{E}_u](\ell+1) = o(BL)$ bits. This is again negligible.

For the length contributed by chunks represented by pointers, it is obvious that the dictionary size is at most BL/ℓ so $\log|T|+1 \leq \log BL$. Moreover, there are at most $\frac{BL}{\ell}$ chunks. The total length of the pointers and their indicator bits is then at most

$$\mathbb{E}[(\log|T|+1)\cdot BL/\ell] = \frac{1}{\gamma}(1+\frac{1}{k})H((1+\alpha)\delta)BL,$$

completing the proof. ■

It can be shown that for any δ, there exist α such that $H((1+\alpha)\delta)/H(\delta) \leq 2$. If we pick $\gamma = \epsilon$, Theorem 6 shows that for any δ, there exists ℓ such that $\mathbb{E}[\mathcal{L}_F(\boldsymbol{s})]/H(\boldsymbol{s}) \leq \frac{2}{\epsilon}(1+1/k)$, which means that the double fixed-length algorithm is within a constant factor of optimal.

B. Fixed-length edit-distance deduplication

Theorem 7. *For the source model in which source symbols have the same length L, the performance of fixed-length edit-distance deduplication with $\ell = L$ and $t = 2(1+\beta)\delta L$ satisfies*

$$\frac{\mathbb{E}[\mathcal{L}_{ED}(\boldsymbol{s})]}{H(\boldsymbol{s})} \leq \frac{H(2(1+\beta)\delta)}{H(\delta)}(1+o(1)),$$

as $B \to \infty$ for any $\beta > 0$.

Proof: For the source string $\boldsymbol{s} = Y_1 \cdots Y_B$, we know that each Y_b is descendant of one of the source symbols. Let $\mathcal{E}_3$ be the event that every block Y_b has Hamming distance at most $(1+\beta)\delta L$ from its parent. We have, by applying the union bound and the chernoff bound, $1 - \Pr(\mathcal{E}_3) \leq Be^{-\frac{\beta^2}{2+\beta}\delta L}$.

Given $\mathcal{E}_3$, it can be shown that there are at most A strings in the dictionary. So for encoding all strings in the dictionary, we need at most $A(L+1)$ bits. The pointers plus the description of the difference between the current chunk and the referenced chunk need at most $B[1+(\log A+1)+(H(2(1+\beta)\delta)L+1)]$ bits. With the addition of the $2\log BL+3$ bits for encoding the length, we have $\mathbb{E}[\mathcal{L}_{ED}(\boldsymbol{s})|\mathcal{E}_3] \leq H(2(1+\beta)\delta)BL + o(BL)$.

If the complement of $\mathcal{E}_3$ holds, it can be shown in similar fashion that $\mathbb{E}[\mathcal{L}_{ED}(\boldsymbol{s})|\mathcal{E}_3^c] \leq 2BL + o(BL)$. Thus,

$$\begin{aligned}\mathbb{E}[\mathcal{L}_{ED}(\boldsymbol{s})] &= \Pr(\mathcal{E}_3)\mathbb{E}[\mathcal{L}_{ED}(\boldsymbol{s})|\mathcal{E}_3] + \Pr(\mathcal{E}_3^c)\mathbb{E}[\mathcal{L}_{ED}(\boldsymbol{s})|\mathcal{E}_3^c] \\ &\leq H(2(1+\beta)\delta)BL + o(BL).\end{aligned}$$

■

Similar to Theorem 6, we can always find β such that the fixed-length edit-distance deduplicccation achieves a constant factor of optimal.

VI. Variable-length deduplication

In this section, we study the variable-length algorithm, which is more widely applicable than fixed-length schemes and does not need any extra information about the boundaries of source blocks. We saw in the previous section for the double fixed-length algorithm, to achieve optimality, the chunk length should be chosen appropriately. For the variable-length scheme, we show in the following theorem that similar to the double fixed-length scheme, if the marker length of variable-length deduplication is not chosen properly, then the expected length of the compressed string will be lower bounded by a constant regardless of how small entropy is.

Theorem 8. *Consider the source model with error rate δ and variable-length deduplication algorithm with marker length M. Asymptotically, if $2^M = o(\log B)$ or $2^M = \omega(\log B)$, then there exists a constant c_1 such that for all $\delta > 0$, $\mathbb{E}[\mathcal{L}_{VL}(\boldsymbol{s})]/BL \geq c_1$.*

The proof is similar to the proof of Theorem 5 and omitted due to space limitation. Given the theorem, in the following, we only consider marker length M such that $2^M = \Theta(\log B)$ as $B \to \infty$.

Next, we give a lemma which is an analog of the upper bound in Lemma 4 with the proof omitted. We will use this lemma to bound the performance of the variable-length scheme with properly chosen marker length M.

Lemma 9. *Let $\boldsymbol{s} = Y_1Y_2\cdots Y_B$ denote the data string. For any string $\boldsymbol{w} \in \Sigma^*$ with $|\boldsymbol{w}| \leq \frac{1}{2}L$, let $F(\boldsymbol{w})$ denote the event that $\boldsymbol{w}$ appears as a substring of Y_i for some i. As $B \to \infty$,*

$$\Pr(F(\boldsymbol{w})|\mathcal{E}_u) \leq AL\mathbb{E}_d\Big[\min\Big(3B\delta^d(1-\delta)^{|\boldsymbol{w}|-d}/(2A), 1\Big)\Big],$$

where $\Pr(d = x) = \binom{|\boldsymbol{w}|}{x}/2^{|\boldsymbol{w}|}, x = 0, 1, \ldots, |\boldsymbol{w}|$.

Theorem 10. *Consider the source model with $A = o(B^{1-\epsilon})$, $L = B^{1/k}$ and error rate δ. For any $0 < \alpha \leq 1/(2\delta) - 1$, the* *performance of variable-length deduplication with the optimal marker length satisfies*

$$\frac{\mathbb{E}[\mathcal{L}_{VL}(\boldsymbol{s})]}{BL} \leq \left(c^2 + 6c + 6\right)e^{-c}(1 + o(1)), \text{ as } B \to \infty,$$

where $c = -W_{-1}\left(-(1+\frac{1}{k})\frac{2\mathcal{F}}{3}\right), \mathcal{F} = \min\left(1, \frac{H((1+\alpha)\delta)}{\epsilon}\right)$, *and* W_{-1} *is the lower branch of the Lambert W function.*

Proof: It can be shown that the expected number of bits needed for encoding the chunks intersecting the boundaries of source blocks (including the first and the last chunk), the chunks of the form 0^M and the chunks of length larger than $\frac{1}{2}L$, is $o(BL)$. The effect of these chunks will be absorbed into the $o(1)$ term and need not be considered further. Encoding the length of $\boldsymbol{s}$ also takes $o(BL)$ bits.

We first compute the expected number of bits needed for pointers. Note that the dictionary size $|T|$ is at most $|\boldsymbol{s}|/M$, so a pointer takes at most $\log\frac{|\boldsymbol{s}|}{M} + 1 \leq \log|\boldsymbol{s}|$ bits. It can also be shown that given $|\boldsymbol{s}|$, the expected number of chunks is at most $\frac{|\boldsymbol{s}|}{2^M}$. Therefore the expected number of bits used by pointers is at most

$$\mathbb{E}\left[(\log|\boldsymbol{s}| + 1)\cdot|\boldsymbol{s}|/2^M\right] \leq 2(\log BL + 1)BL/2^M. \tag{6}$$

Next, we compute the expected number of bits needed for encoding the dictionary. Similar to the proof of Theorem 10, it can be shown that the number of bits used by encoding the dictionary given $\mathcal{E}_u^c$ is $o(1)$. So in the following, we assume $\mathcal{E}_u$ is true. We will give the proof for the case when $H((1+\alpha)\delta) < \epsilon$. The proof for the case in which $H((1+\alpha)\delta) \geq \epsilon$ is similar. Let $\ell_0 = \epsilon\log B/H((1+\alpha)\delta)$ and note that $\ell_0 > \log B$. By assumption every chunk in the dictionary is of the form $\boldsymbol{v}10^M$ for some $(M-1)$-RLL string $\boldsymbol{v}$. Moreover, for every $\boldsymbol{v}10^M$ to be parsed as a chunk, there must be a 0^M right before it. Therefore, the probability of $\boldsymbol{v}10^M \in T$ is less than the probability of $F(0^M\boldsymbol{v}10^M)$, which is defined in Lemma 9.

Thus, let $\ell = \left|0^M\boldsymbol{v}10^M\right|$, $\Pr(\boldsymbol{v}10^M \in T|\mathcal{E}_u)$ is upper bounded by

$$\begin{aligned} &AL\mathbb{E}_d\left[\min(3B\delta^d(1-\delta)^{\ell-d}/(2A), 1)\right] \\ &\leq \begin{cases} 1, & \ell \leq \log B, \\ \frac{L}{2^\ell}o(B), & \log B \leq \ell \leq \ell_0, \\ \frac{3BL}{2^{\ell+1}}, & \ell > \ell_0. \end{cases} \end{aligned} \tag{7}$$

Next, we compute the expected number of bits needed to encode the dictionary T. We have

$$\mathbb{E}\left[\sum_{\boldsymbol{w}\in T}|\boldsymbol{w}|\middle|\mathcal{E}_u\right] = \sum_{\ell=0}^{L/2-M-1}\sum_{\boldsymbol{v}\in R_{M-1}^\ell}\Pr(\boldsymbol{v}10^M \in T|\mathcal{E}_u)(\ell + M + 2). \tag{8}$$

We compute (8) by considering two different ranges for ℓ. It can be shown by (7) and Lemma 1 that as $B \to \infty$,

$$\sum_{\ell=0}^{\ell_0-M-1}\sum_{\boldsymbol{v}\in R_{M-1}^\ell}\Pr(\boldsymbol{v}10^M \in T|\mathcal{E}_u)(\ell + M + 2) = o(BL), \tag{9}$$

and

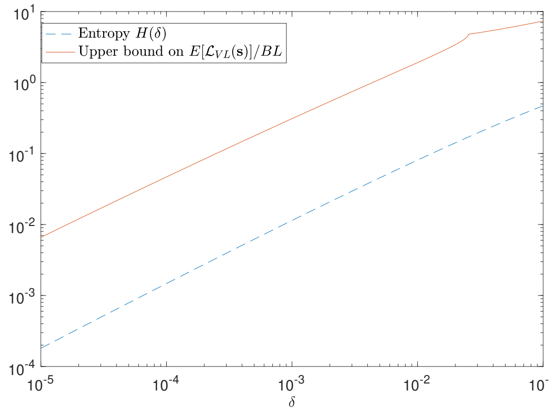

Figure 1. Upper bound on $\frac{\mathbb{E}[\mathcal{L}_{VL}(\boldsymbol{s})]}{BL}$ and $H(\delta)$ vs the error rate δ, with the optimal marker length, $A = L = B^{1/2}$, $\alpha = 0.1$ and δ ranging from 10^{-5} to 10^{-1}.

$$\begin{aligned} &\sum_{\ell=\ell_0-M}^{L/2-M-1}\sum_{\boldsymbol{v}\in R_{M-1}^\ell}\Pr(\boldsymbol{v}10^M \in T|\mathcal{E}_u)(\ell + M + 2) \\ &\qquad \leq 6(1 + c_M)e^{-c_M}BL + o(BL), \end{aligned} \tag{10}$$

where $c_M = \ell_0/2^{M+1}$. So by (6), (9), (10), and by ignoring the terms of the form $o(BL)$, we find that asymptotically $\mathbb{E}[\mathcal{L}_{VL}(\boldsymbol{s})]$ is upper bounded by

$$\left((1 + \frac{1}{k})\frac{H((1+\alpha)\delta)}{\epsilon}2c_M + 6(c_M + 1)e^{-c_M}\right)BL. \tag{11}$$

For any given c, there exists an integer value for M such that $c \leq c_M \leq 2c$. For this M, (11) is upper bounded by

$$\left((1 + \frac{1}{k})\frac{H((1+\alpha)\delta)}{\epsilon}4c + 6(c + 1)e^{-c}\right)BL.$$

The desired result is obtained by choosing the value of c (and thus M) minimizing the above expression. ■

For all δ such that $H(\delta) \leq \epsilon$, there exists α such that $\mathcal{F} = H((1+\alpha)\delta)/\epsilon$, and hence the upper bound on the normalized expected compressed length approaches 0 as δ approaches 0. This means that as the entropy becomes smaller, the compression ratio grows if the length of the marker is chosen appropriately. In particular, it can be shown that the optimal length of the marker depends on δ, which represents the degree of variability between the copies. For $A = L = B^{1/2}$, i.e., $\epsilon = \frac{1}{2}, k = 2$ and $\alpha = 0.1$, Figure 1 shows the upper bound on the ratio $\mathbb{E}[\mathcal{L}_{VL}(\boldsymbol{s})]/BL$ and $H(\delta)$ as δ ranges from 10^{-5} to 10^{-1}.

A Large compression ratio when entropy is small is desirable and variable-length deduplication achieves this. However, it can be shown and also observed in Figure 1 that the upper bound of the ratio $\mathbb{E}[\mathcal{L}_{VL}(\boldsymbol{s})]/H(\boldsymbol{s})BL$ given by Theorem 10 increases as δ decreases. Therefore, despite the large compression ratios, the gap to entropy may become large for small δ. Determining whether it is indeed the case or the bound provided here is loose is left to future work.

References

[1] J. Gantz and D. Reinsel, “The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east”, *IDC iView: IDC Analyze the future*, vol. 2007, no. 2012, pp. 1–16, 2012.

[2] D. T. Meyer and W. J. Bolosky, “A study of practical deduplication”, *ACM Transactions on Storage*, vol. 7, no. 4, p. 14, 2012.

[3] A. El-Shimi, R. Kalach, A. Kumar, A. Ottean, J. Li, and S. Sengupta, “Primary data deduplicationâlarge scale study and system design”, in *Presented as part of the 2012 USENIX Annual Technical Conference (USENIX ATC 12)*, 2012, pp. 285–296.

[4] A. Muthitacharoen, B. Chen, and D. Mazieres, “A low-bandwidth network file system”, in *ACM SIGOPS Operating Systems Review*, ACM, vol. 35, 2001, pp. 174–187.

[5] S. Quinlan and S. Dorward, “Venti: A new approach to archival storage”, in *FAST*, vol. 2, 2002, pp. 89–101.

[6] W. Xia, H. Jiang, D. Feng, F. Douglis, P. Shilane, Y. Hua, M. Fu, Y. Zhang, and Y. Zhou, “A comprehensive study of the past, present, and future of data deduplication”, *Proceedings of the IEEE*, vol. 104, no. 9, pp. 1681–1710, 2016.

[7] U. Manber, “Finding similar files in a large file system”, in *Usenix Winter*, vol. 94, 1994, pp. 1–10.

[8] U. Niesen, “An information-theoretic analysis of deduplication”, *IEEE Transactions on Information Theory*, vol. 65, no. 9, pp. 5688–5704, Sep. 2019.

[9] E. S. Lander, L. M. Linton, B. Birren, C. Nusbaum, M. C. Zody, J. Baldwin, K. Devon, K. Dewar, M. Doyle, W. FitzHugh, *et al.*, “Initial sequencing and analysis of the human genome”, *Nature*, vol. 409, no. 6822, pp. 860–921, 2001.

[10] R. Vestergaard, Q. Zhang, and D. E. Lucani, “Generalized deduplication: Bounds, convergence, and asymptotic properties”, *arXiv preprint arXiv:1901.02720*, 2019.

[11] L. Conde-Canencia, T. Condie, and L. Dolecek, “Data deduplication with edit errors”, in *2018 IEEE Global Communications Conference (GLOBECOM)*, IEEE, 2018, pp. 1–6.

[12] B. H. Marcus, R. M. Roth, and P. H. Siegel, “An introduction to coding for constrained systems”, *Lecture notes*, 2001.

[13] S. Greenberg and M. Mohri, “Tight lower bound on the probability of a binomial exceeding its expectation”, *Statistics & Probability Letters*, vol. 86, pp. 91–98, 2014.

[14] T. M. Cover and J. A. Thomas, *Elements of information theory*. John Wiley & Sons, 2012.