# A Coloring Approach to Constructing Deletion Correcting Codes from Constant Weight Subgraphs

Daniel Cullina, Ankur A. Kulkarni, and Negar Kiyavash
Dept. of Electrical and Computer Eng., Coordinated Science Laboratory, Dept. of Industrial and Enterprise Systems Eng.
University of Illinois at Urbana-Champaign, Urbana, Illinois 61801
Email: {cullina,akulkar3,kiyavash}@illinois.edu

***Abstract*—We take a graph theoretic view of deletion correcting codes. The problem of finding an $n$-bit $s$-deletion correcting code is equivalent to finding an independent set in a particular graph. We discuss the relationship between codes and colorings and demonstrate that the VT codes are optimal in a coloring sense. We describe a method of partitioning the set of bit strings by Hamming weight and finding codes within each partition. In the single deletion case, we find an optimal coloring of the constant Hamming weight induced subgraphs. We show that the resulting code is asymptotically optimal. We also prove a lower bound on size of codes constructed using these partitions for any number of deletions.**

## I. Introduction

Deletion channels output only a subsequence of their input while preserving the order of the transmitted symbols. They have applications in synchronization problems, communication of information over packet networks and biology. This paper concerns deletion channels for strings of bits, with fixed input bit string length and a fixed number of deletions. Despite significant effort on this case, there still are many fundamental open problems, pertaining specifically to the design of codes and the size of the largest codebook.

Levenshtein approached the design of codes from a combinatorial and number theoretic perspective [4]. He showed that the Varshamov Tenengolts (VT) codes, which had been designed for a different channel [8], functioned as codes for the single deletion channel. In fact, the VT codes are conjectured to optimal for the single deletion channel [7]. Levenshtein also derived an upper bound and a nonconstructive lower bound on the sizes of codes for any number of deletions. Much less is known for channels with larger number of deletions. Helberg generalized the VT construction for any number of deletions, but the sizes of resulting codes grow very slowly, far below Levenshtein's bound [2].

Another direction for the construction of codes is computational. It is well known that the problem of finding deletion correcting codes is equivalent to finding an independent set in a particular graph. But since, for general graphs, finding the maximum independent set is NP-hard, exact algorithms rapidly become intractable with increasing input string length ($n$). For the case of the single deletion, the computational approach has established that VT codes are optimal for $n \leq 11$ (graph with $2^{11}$ vertices) [6]. For multiple deletions, the best known codes have all been found through search algorithms. Butenko et al. found two-deletion correcting codes of maximum size for $n \leq 11$ [1]. Khajouei et al. used a heuristic algorithm to find the largest known two deletion correcting codes for $n \leq 25$ [3].

This paper takes a graph theoretic perspective on this problem and contributes to both the combinatorial and computational approaches. Our first contribution is on the theoretical understanding VT codes: we show that VT codes optimally solve the *coloring problem* in the single deletion graph (while they have been conjectured to solve the *independent set problem*). Second, we present a new method of constructing codes by solving several problems on smaller graphs, a computationally less intensive task. The method decomposes the graph of $2^n$ possible bit strings into subgraphs based on their Hamming weight, finds codes in selected subgraphs, and takes the union of these codes. In the single deletion case, this construction is asymptotically optimal; we show this by constructing an optimal coloring of the subgraphs. For larger number of deletions, we prove a lower bound on the size of codes constructed using these subgraphs.

The paper is organized as follows. In Section II, we give some notation and definitions related to the deletion channel and review the graph theoretic terminology and results. We also discuss the VT codes in graph theoretic terms. In Section III, we present our construction of codes for single and multiple deletions. Section IV contains some computational results on the sizes of some two deletion codes found by computer search using our partitioning strategy. In Section V, we discuss future work.

## II. Preliminaries

### A. Notation

Let $[n]$ be the set of nonnegative integers less that $n$, $\{0, 1..n-1\}$. Let $[2]^n$ be the set of bit strings of length $n$. Let $H(x)$ be the Hamming weight of a string $x$. By $\binom{[n]}{k}$ we denote the set of bit strings of length $n$ with $k$ ones. We will need the following asymptotic notation: let $a(n) \lesssim b(n)$ denote that $\lim_{n\to\infty} \frac{a(n)}{b(n)} \leq 1$.

### B. The deletion channel and related graphs

We will formalize the problem of correcting deletions by defining the deletion channel. The deletion channel takes a bit string of length $n$ and outputs a substring of length $n-s$. For bit strings $x$ and $y$, write $x < y$ if $x$ is a substring of $y$ and define the following sets.

**Definition 1.** *For $x \in [2]^n$, define $D_s(x) = \{z \in [2]^{n-s} : z < x\}$, the set of substrings of $x$ that can be produced by $s$ deletions. Define $I_s(x) = \{w \in [2]^{n+s} : w > x\}$, the set of superstrings of $x$ that can be produced by $s$ insertions.*

If $x$ is the input to an $n$ bit $s$ deletion channel, $D_s(x)$ is the set of possible outputs. If $x$ is the *output* from the channel, $I_s(x)$ is the set of possible inputs.

We are interested in zero error codes for the deletion channel. Consequently, a code is a set $C$ of bit strings of length $n$ such that for any two distinct bit strings $x$ and $y$ in $C$, the intersection $D_s(x) \cap D_s(y)$ is empty. We can restate this in another way by defining a distance measure between bit strings.

**Definition 2.** *Let $x, y \in [2]^n$. Let $l$ be the largest integer for which there exists some $z \in [2]^l$ such that $z < x$ and $z < y$. Define the deletion distance between $x$ and $y$ to be $d_L(x, y) = n - l$.*

An $s$-deletion correcting code is a set where the deletion distance between any two codewords is at least $s + 1$. Two codewords cannot both appear in a code if their deletion distance is $s$ or less. We can capture this condition by defining the following graph.

**Definition 3.** *For every distance $s$ and length $n$, both positive integers, let $L_{s,n}$ be a graph with $[2]^n$ as its vertices. Vertices $x$ and $y$ are adjacent if and only if $d_L(x, y) \leq s$.*

A code that can correct $s$ deletions is a set of vertices in $L_{s,n}$ that have no edges between them.

### C. Independent Sets, Colorings, and Cliques

Now we will briefly define some graph notation and review a few concepts that will be useful later. All of these are sourced from West [9]. Given a graph $G$, let $V(G)$ denote its vertex set and let $E(G)$ denote its edge set. Given $S \subseteq V(G)$, the *subgraph induced by* $S$ contains the vertices in $S$ and the edges in $E(G)$ that have both endpoints in $S$.

An *independent set* in a graph is a set of vertices that are all nonadjacent. The size of a largest independent set in a graph $G$ is denoted by $\alpha(G)$. The degree of a vertex is the number of adjacent vertices. The maximum degree of any vertex in $G$ is denoted by $\Delta(G)$. It is easy to argue that $\alpha(G) \geq |V(G)|/(\Delta(G) + 1)$.

A *coloring* of a graph assigns a color (a number) to each vertex. The coloring is proper if it never assigns the same color to both endpoints of an edge. Thus a proper coloring of a graph partitions its vertices into independent sets; each independent set is assigned a single color and called a *color class*. The chromatic number of a graph $G$, denoted $\chi(G)$, is the smallest $k$ for which a proper $k$-coloring of $G$ exists. An argument based on greedy coloring of $G$ shows that $\chi(G) \leq \Delta(G) + 1$.

A coloring gives us several independent sets to choose from. At least one of these color classes must be at least as large as the average size of a color class. Consequently, $\alpha(G) \geq |V(G)|/\chi(G)$. However, properly coloring a graph using the minimum number of colors is not equivalent to finding the largest independent set. In general there is no guarantee that the largest color class in a particular coloring is a maximum independent set or that any minimal coloring has a maximum independent set as a color class.

A *clique* in a graph is a set of vertices that are all adjacent. The size of a largest clique in a graph $G$ is denoted by $\omega(G)$. In a proper coloring, each vertex in a clique must be assigned a different color, so for any graph $G$, $\chi(G) \geq \omega(G)$.

### D. The Varshamov-Tenengolts coloring

For each string length $n$, the Varshamov-Tenengolts construction provides $n + 1$ distinct single deletion correcting codes. The largest of these codes is conjectured to be optimal, i.e., it is conjectured to solve the independent set problem on $L_{1,n}$. We show that together they optimally solve the coloring problem.

**Definition 4.** *For any $x \in [2]^n$, let $w(x) = \sum_{i=0}^{n-1}(i + 1)x_i$. Call $w(x) \mod n + 1$ the VT weight of $x$.*

The VT contruction partitions $[2]^n$ into $n+1$ sets, each with a particular VT weight. Each of these sets is a code (cf. [4]) and an independent set in $L_{1,n}$. This makes the VT weight a coloring of $L_{1,n}$ that uses $n + 1$ colors, although it has not usually been described in this language. To demonstrate that one cannot use fewer colors in any coloring of $L_{1,n}$, we will find cliques of size $n + 1$ in $L_{1,n}$ and use $\omega(G) \leq \chi(G)$.

**Lemma 1.** *For each $x \in [2]^{n-s}$, $I_s(x)$ induces a clique in $L_{s,n}$. Furthermore $|I_s(x)| = \sum_{i=0}^{s} \binom{n}{i}$.*

*Proof:* Any two vertices in $I_s(x)$ have a common substring of length $n - s$, $x$, so their deletion distance is at most $s$ and they are adjacent in $L_{s,n}$. The size of $I_s(x)$ is sourced from Levenshtein [5]. ∎

This gives us the result that an optimal coloring of $L_{1,n}$ uses $n + 1$ colors.

**Theorem 1.** *For any $n$, $\chi(L_{1,n}) = \omega(L_{1,n}) = n + 1$.*

*Proof:* The VT coloring uses $n+1$ colors, and by taking $s = 1$ in Lemma 1, we see that there are cliques of $n + 1$ vertices in $L_{1,n}$. So $n + 1 \leq \omega(L_{1,n}) \leq \chi(L_{1,n}) \leq n + 1$. ∎

The largest color class (corresponding to VT weight zero) in the VT coloring of $L_{1,n}$ always contains at least $\frac{2^n}{n+1}$ codewords. This sequence of independent sets is asymptotically maximum [4].

## III. Code construction by weight partitioning

We now describe a new strategy for code construction for any number of deletions. For single deletion channels, the codes we construct are asymptotically optimal (Section III-A). In Section III-B we prove lower bounds on the sizes of our codes for any number of deletion. This strategy is inspired by a simple bound on deletion distance.

**Lemma 2.** *For all $x, y \in [2]^n$, $d_L(x, y) \geq |H(x) - H(y)|$.*

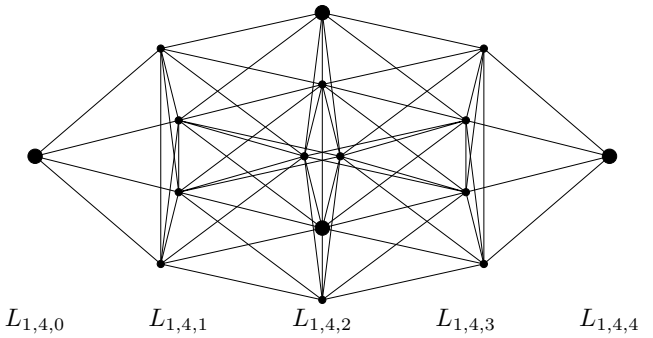


Fig. 1. $L_{1,4}$ partitioned by Hamming weight. An independent set in each even weight layer is highlighted.

*Proof:* If $z < x$ and $z < y$, then $z$ must have fewer ones than either $x$ or $y$ as well as fewer zeros. ∎

Let $L_{s,n,k}$ be the subgraph of $L_{s,n}$ induced by the vertices with exactly $k$ ones. The endpoints of any edge in $L_{s,n}$ differ in Hamming weight by at most $s$. Suppose we find an independent set composed entirely of vertices of Hamming weight $k$, i.e. an independent set in $L_{s,n,k}$, and another independent set entirely of vertices of weight $k+s+1$, we can guarantee that their union is an independent set in $L_{s,n}$. Then we can add another independent set in $L_{s,n,k+2(s+1)}$ and continue until we have exhausted the weights that are equal to $k \mod s+1$. This procedure gives us an independent set in $L_{s,n}$. Figure 1 illustrates this for $L_{1,4}$.

More formally, we have the following result.

**Lemma 3.** *For each possible remainder $a \in [s+1]$, we have* $\alpha(L_{s,n}) \geq \sum_{i=0}^{n/(s+1)} \alpha(L_{s,n,i(s+1)+a})$

Another way to describe this process is that we start by throwing out all the vertices whose Hamming weights do not equal $a \mod s+1$. We keep only about $\frac{1}{s+1}$ of the vertices, The remaining graph is disconnected. It has broken up into a component for each weight.

### A. Explicit construction of a single deletion correcting code

The strategy outlined above reduces the problem of finding an independent set in $L_{s,n}$ to the problem of finding independent sets in each of $L_{s,n,k}, k = 0, \ldots, n$. In the single deletion case ($s = 1$), we show an explicit construction of independent sets in the graphs $L_{1,n,k}$. We construct these independent sets by finding an optimal coloring of $L_{1,n,k}$. This coloring is closely related to the optimal VT coloring of $L_{1,n}$. The code that results is asymptotically optimal.

**Lemma 4.** *The modified VT weight* $f(x) = w(x) \bmod (\max(k, n-k)+1)$ *gives a proper coloring of* $L_{1,n,k}$.

*Proof:* Let $x$ and $y$ be adjacent vertices in $L_{1,n,k}$. To show $f(x) \neq f(y)$, we will show that $0 < |w(y) - w(x)| \leq \max(k, n-k)$. Let $i$ be the smallest index where $x_i \neq y_i$ and let $j$ be the largest such index. Because $d_L(x, y) = 1$, either $x_{[n]\setminus i} = y_{[n]\setminus j}$ or $x_{[n]\setminus j} = y_{[n]\setminus i}$. Without loss of generality assume the latter. Because $H(x) = H(y) = k$, $x_j = y_i$. The interval $x_{\{i..j-1\}}$ shifts right by one space to become $y_{\{i+1..j\}}$ so the contribution to the weight of each one in the interval increases by one. The bit $x_j$ moves $j-i$ spaces to the left, so its contribution decreases by that amount. If $l$ is the number of ones in $x_{\{i..j-1\}}$, then $w(y) - w(x) = l - x_j(j-i)$.

If $x_i = 0$. then $w(y) - w(x) = l \leq k$. Since $x \neq y$, $l > 0$. On the other hand, if $x_i = 1$, then $w(y) - w(x) = l + i - j$. There are $j - i - l$ zeros in $x_{[i,j)}$ and only $n - k$ zeros in all of $x$, so $w(y) - w(x) \geq k - n$. Because $x \neq y$, $l < j - i - 1$ and $w(y) - w(x) < 0$. ∎

To prove optimality, we need $\omega(L_{1,n,k})$. As in $L_{1,n}$ we will look at cliques whose vertices have a single common substring. Let us introduce some notation.

**Definition 5.** *For $x \in \binom{[n]}{k}$, let $I_{s,r}(x) = I_s(x) \cap \binom{[n+s]}{k+r}$. This is the number of superstrings of $x$ of length $n+s$ with exactly $k+r$ ones.*

Just as the size of $I_s(x)$ only depends on the length of $x$, the size of the set $I_{s,r}(x)$ only depends on the length and weight of $x$. To prove this we will need the following lemma. Due to space limitations, we only sketch the proof.

**Lemma 5.** *For all $x \in \binom{[n]}{k}$, all $s$, and all $r \in [s+1]$,* $|I_{s,r}(x)| = \sum_{a=0}^{\min(r,s-r)} \binom{n-k+r}{r-a}\binom{k+s-r}{s-r-a}$.

*Sketch of proof:* For any $x \in [2]^{n-s}$, there is a bijection between $I_s(x)$ and $\bigcup_{i=0}^{s} \binom{[n+i-1]}{i} \times [2]^{s-i}$. Each superstring in $I_s(x)$ is produced by two types of operations: insertions of some number of negated bits before a bit of $x$ and insertion of bits at the end of $x$. The sets $\binom{[n+i-1]}{i}$ encode the former operation and $[2]^{s-i}$ encode the latter. For any superstring in $I_{s,r}(x)$, let $p$ be the number of ones and $q$ be the number of zeros in the appended bit string, so there are $\binom{p+q}{p}$ such bit strings. There are $r-p$ remaining new ones to insert before the $n-k$ existing zeros, which can be done in $\binom{n-k+r-p-1}{r-p}$ ways. The remaining $s-r-q$ zeros can be inserted before the $k$ existing ones, in $\binom{k+s-r-q-1}{s-r-q}$ ways. Summing over all possible values of $p$ and $q$ gives the size of $I_{s,r}(x)$ as

$$\sum_{p=0}^{r}\sum_{q=0}^{s-r}\binom{n-k+r-p-1}{r-p}\binom{k+s-r-q-1}{s-r-q}\binom{p+q}{p}.$$

This sum can be simplified using the Vandermonde identity and a variation of it for multisets to give the result. ∎

**Lemma 6.** *$L_{s,n,k}$ contains cliques of sizes* $\sum_{i=0}^{\min(r,s-r)} \binom{n-k-s+2r}{r-i}\binom{k+s-2r}{s-r-i}$ *for all $r \in [s+1]$.*

*Proof:* In Lemma 5, we fix the length and weight of the substring to be $n$ and $k$. Here we would like the length and weight of the superstrings to be $n$ and $k$, so we substitute $n-s$ for $n$ and $k-r$ for $k$ in the previous result. ∎

**Theorem 2.** $\chi(L_{1,n,k}) = \omega(L_{1,n,k}) = \max(k, n-k)+1$.

*Proof:* By Lemma 6, $L_{1,n,k}$ contains cliques of sizes $k+1$ and $n-k+1$. Lemma 4 gives the coloring. ∎

We now show that this strategy produces independent sets in $L_{1,n}$ that are asymptotically of optimal size. Let $C_{n,k}$ be

a largest color class of $L_{1,n,k}$ using the coloring described above. Our code is the set $D_{n,a}$,

$$D_{n,a} := \bigcup_{0 \le i \le n/2} C_{n,2i+a}.$$

**Lemma 7.** *$|D_{n,a}| \ge \frac{1}{n+1}\left(2^n - \binom{n}{k^*}\right)$ where $k^*$ is $(n-1)/2$ if $n$ is odd, $(n-2)/2$ if $n$ is even and $a \equiv n/2 \bmod 2$ and $n/2$ otherwise.*

*Proof:* We only consider the case where $n$ is odd; the other case follows similarly. In each graph $L_{1,n,k}$, some color class must be at least as large as the average.

$$|D_{n,a}| = \sum_{\substack{0 \le k \le n \\ k \equiv a \bmod 2}} |C_{n,k}| \ge \sum_{\substack{0 \le k \le n \\ k \equiv a \bmod 2}} \frac{|V(L_{1,n,k})|}{\chi(L_{1,n,k})}$$

There are $\binom{n}{k}$ vertices in $L_{1,n,k}$ and $\chi(L_{1,n,k}) = \max(k, n-k)+1$. Without loss of generality suppose $(n-1)/2 \equiv a \bmod 2$. Thus $|D_{n,a}|$ is at least

$$\sum_{\substack{k=0 \\ k \equiv a \bmod 2}}^{(n-1)/2} \binom{n}{k}\frac{1}{n-k+1} + \sum_{\substack{k=(n+3)/2 \\ k \equiv a \bmod 2}}^{n} \binom{n}{k}\frac{1}{k+1}$$
$$= \frac{1}{n+1}\sum_{\substack{0 \le k \le n \\ k \ne (n-1)/2}} \binom{n}{k} = \frac{1}{n+1}\left(2^n - \binom{n}{(n-1)/2}\right)$$

■

**Theorem 3.** *$D_{n,a}$ is asymptotically optimal.*

*Proof:* By Stirling's formula, $\binom{n}{n/2} \sim 2^n\sqrt{\frac{2}{\pi n}}$, so

$$|D_{n,a}| \sim \frac{2^n}{n+1}\left(1 - \sqrt{\frac{2}{\pi n}}\right) \sim \frac{2^n}{n}.$$

Levenshtein showed that $\alpha(L_{1,n}) \lesssim \frac{2^n}{n}$ [4], hence the claim. ■

Note that $\max_k \chi(L_{1,n,k}) = n$, which is barely better than $\chi(L_{1,n}) = n+1$. However, most of the vertices are in the subgraphs with Hamming weight $\approx n/2$, and $\chi(L_{1,n,n/2}) = n/2+1$. Thus, half the vertices have been thrown out, but the middle layers are colored about twice as efficiently as they were in the original graph. This explains the asymptotic optimality.

### *B. A lower bound for multiple deletion code sizes*

For $s > 1$, we do not have optimal explicit colorings of $L_{s,n,k}$. However, we can use the maximum degrees of $L_{s,n,k}$ to lower bound the sizes of their maximum independent sets. Recall the relation from Section II-C, $\alpha(G) \ge |V(G)|/(\Delta(G)+1)$. This is equivalent to considering the performance of greedy colorings on these graphs.

First we will obtain an asymptotic expression for the number of superstrings of a particular weight. We will use that to bound the degree of a vertex in $L_{s,n,k}$. This will translate into a bound on independent set size.

**Lemma 8.** *Let $k = pn$ and $x \in \binom{[n]}{k}$. For fixed $p$, $s$, and $r$, $|I_{s,r}(x)| \sim \frac{n^s}{s!}\binom{s}{r}(1-p)^r p^{s-r}$*

*Proof:* We start with the result of Lemma 5. Only the first term of $\sum_{a=0}^{\min(r,s-r)} \binom{n-k+r}{r-a}\binom{k+s-r}{s-r-a}$ is of degree $s$. Since $\binom{n}{c} \sim \frac{n^c}{c!}$, this term becomes $\frac{(n-pn)^r(pn)^{s-r}}{r!(s-r)!}$ which we can rearrange into $\frac{n^s}{s!}\binom{s}{r}(1-p)^r p^{s-r}$. ■

As $n$ becomes large, the weight distribution of vertices in $L_{s,n}$ concentrates around $n/2$, so we need to bound the number of insertions in that region only.

**Lemma 9.** *Let $k = pn$, $\frac{s}{2s+2} \le p \le \frac{s+2}{2s+2}$, and $x \in \binom{[n]}{k}$. Fix $s$ and $r \in [s+1]$. If $s$ is even, then $|I_{s,r}(x)| \lesssim \frac{1}{2^s}\binom{s}{s/2}\binom{n}{s}$. If $s$ is odd, then $|I_{s,r}(x)| \lesssim \frac{1}{2^{s-1}}\binom{s-1}{(s-1)/2}\binom{n}{s}$.*

*Proof:* Due to space limitations, we prove this only for the case where $s$ is even. For each $r \in [s+1]$, let $f_r(p) = \binom{s}{r}(1-p)^r p^{s-r}$. In the interval $1 - \frac{r+1}{s+1} \le p \le 1 - \frac{r}{s+1}$, $f_r(p)$ is the largest of the $s+1$ polynomials. The maximum of $f_r(p)$ occurs at $p = 1 - \frac{r}{s}$ and the value achieved there is $\binom{s}{r}\frac{r^r(s-r)^{s-r}}{s^s}$.

If $s$ is even, then $f_{s/2}(s/2) = \frac{1}{2^s}\binom{s}{s/2}$. For all $r$, $f_r(p) \le \frac{1}{2^s}\binom{s}{s/2}$ in the interval $\frac{s}{2s+2} \le p \le \frac{s+2}{2s+2}$. ■

Now we can apply this result to get a bound on degree.

**Lemma 10.** *Let $k = pn$. Fix $s$, $r \in [s+1]$, and $\frac{s}{2s+2} \le p \le \frac{s+2}{2s+2}$. If $s$ is even, $\Delta(L_{s,n,k}) \lesssim \frac{1}{2^s}\binom{s}{s/2}\binom{n}{s}^2$ and if $s$ is odd, $\Delta(L_{s,n,k}) \lesssim \frac{1}{2^{s-1}}\binom{s-1}{(s-1)/2}\binom{n}{s}^2$.*

*Proof:* For $x \in \binom{[n]}{k}$, let $d(x)$ be the degree of $x$ in $L_{s,n,k}$. Each vertex adjacent to $x$ shares at least one substring of length $n-s$ with it. We bound degree by considering the superstrings of the substrings of $x$. i.e.,

$$d(x) \le \sum_{y \in [2]^{n-s} : y < x} |I_{s,r}(y)|$$

Since each vertex has at most $\binom{n}{s}$ substrings of length $n-s$, there are at most $\binom{n}{s}$ terms in the sum. We can use Lemma 9 to bound $|I_{s,r}(y)|$, which results in the desired bound. ■

Finally, we can use the upper bound on degree to get a lower bound on code size.

**Theorem 4.** *For fixed even $s$, codes produced by the constant weight strategy contain asymptotically at least $\frac{2^{n+s}}{(s+1)\binom{s}{s/2}\binom{n}{s}^2}$ codewords. For odd $s$, they contain at least $\frac{2^{n+s-1}}{(s+1)\binom{s-1}{(s-1)/2}^2\binom{n}{s}^2}$ codewords. For even $s$ this size is a factor of $(s+1)\binom{s}{s/2}$ less than Levenstein's asymptotic lower bound and for odd $s$ it is a factor of $\frac{s+1}{2}\binom{s-1}{(s-1)/2}$ less.*

*Proof:* There must be some $a \in [s+1]$ such that

$$\sum_{\substack{0 \le k \le n \\ k \equiv a \bmod s+1}} \frac{|V(L_{s,n,k})|}{1+\Delta(L_{s,n,k})} \ge \frac{1}{s+1}\sum_{0 \le k \le n} \frac{|V(L_{s,n,k})|}{1+\Delta(L_{s,n,k})}$$

We drop the values of $k$ that are outside the interval in the condition for Lemma 10.

$$\ge \frac{1}{s+1}\sum_{\frac{s}{2s+2}n \le k \le \frac{s+2}{2s+2}n} \binom{n}{k}\frac{1}{1+\Delta(L_{s,n,k})}$$

| $n$ | Our code | Best known | $n$ | Our code | Best known |
|---|---|---|---|---|---|
| 3 | 2 | 2 | 12 | 27 | 32 |
| 4 | 2 | 2 | 13 | 40 | 49 |
| 5 | 2 | 2 | 14 | 60 | 78 |
| 6 | 4 | 4 | 15 | 100 | 126 |
| 7 | 5 | 5 | 16 | 161 | 201 |
| 8 | 6 | 7 | 17 | 264 | 331 |
| 9 | 8 | 11 | 18 | 449 | 546 |
| 10 | 12 | 16 | 19 | 744 | 911 |
| 11 | 17 | 24 | 20 | 1244 | 1539 |

TABLE I
COMPARISON OF CODE SIZES FOR CORRECTING TWO DELETIONS. SIZES OF BEST KNOWN CODES TAKEN FROM KHAJOUEI ET AL. [3]

The bound in the previous lemma does not depend on $k$. In the case where $s$ is even, we get

$$\gtrsim \frac{2^s}{(s+1)\binom{s}{s/2}\binom{n}{s}^2} \sum_{\frac{s}{2s+2}n \le k \le \frac{s+2}{2s+2}n} \binom{n}{k}.$$

The sum is asymptotic to $2^n$. The factor is found by comparison with Levenshtein's lower bound, $2^{n+s}/\binom{n}{s}^2$ [4]. ■

## IV. Computational Searches for Two Deletion Codes

To demonstrate how well our construction performs, we applied this strategy to the cases of one and two deletions. We used a greedy algorithm to find maximal independent sets in $L_{2,n,k}$ for $n \le 20$. One advantage of working with the constant weight subgraphs $L_{s,n,k}$ is that they are much smaller than $L_{s,n}$, which makes experiments more tractable. For each $n$, there is a set of layers for each remainder modulo three. We computed code sizes for each set and took the largest. The sizes of the codes that we found and the best known are given in Table I and their ratio is plotted in Figure 2.

For comparison, we computed the exact sizes of the codes given by our construction for $s = 1$. To do this, we determined which color class in $L_{1,n,k}$ was largest for each $n$ and $k$. The sizes of these codes are shown in Figure 3. For very small $n$, the codes are significantly larger than the lower bound. This is because the gap between the size of largest color class and the average size is proportionally largest for small $n$. The effect is large enough that the ratio between these codes and the VT codes is fairly flat across the plot even though the codes are asymptotically optimal.

Consequently, it is difficult to conclude much from the plot for $s = 2$. As in the $s = 1$ plot, the ratio is about 0.8 throughout. There might be a constant factor gap between the performance of the two strategies, convergence that is too slow to observe, or perhaps something else.

## V. Conclusion

We translated the problem of finding deletion correcting codes into one of finding independent sets in $L_{s,n}$. We discussed coloring as means of constructing independent sets and demonstrated the the VT codes are optimal in a coloring sense. We described a strategy of decomposing the problem of finding codes into a set of smaller problems by partitioning $L_{s,n}$ by Hamming weight and finding codes within each partition. In the single deletion case, we found an optimal coloring of $L_{1,n,k}$ and showed that the code is asymptotically optimal. We proved a lower bound on size of codes constructed using these partitions that applies to any number of deletions. In the two deletion case, we compared the performance of the best known codes, which were found by searching all of $L_{2,n}$, and codes found using our strategy of searching each of $L_{2,n,k}$ separately.

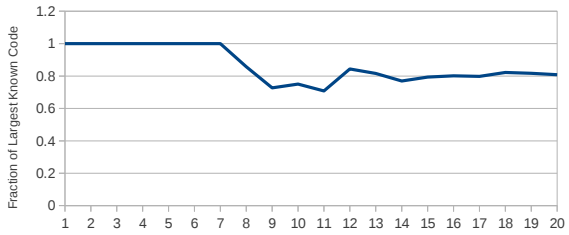


Fig. 2. Ratio of the size of our codes for two deletions to the best known.

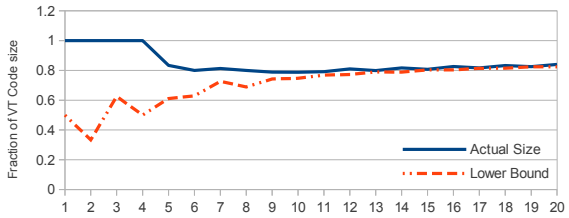


Fig. 3. Ratio of the size of our codes for one deletion to the VT codes.

## Acknowledgment

This work was supported in part by AFOSR under grants FA 9550-11-1-0016 and FA 9550-10-1-0573; and by NSF grant CCF 10-54937 CAR.

## References

[1] S. Butenko, P. Pardalos, I. Sergienko, V. Shylo, and P. Stetsyuk., "Finding maximum independent sets in graphs arising from coding theory," *Proceedings of the 2002 ACM Symposium on Applied Computing*, pp. 542–546, 2002.

[2] A. S. J. Helberg and H. C. Ferreira, "On multiple insertion/deletion correcting codes," *IEEE Trans. Inform. Theory*, vol. 48, pp. 305–308, January 2002.

[3] F. Khajouei, M. Zolghadr, and N. Kiyavash, "An algorithmic approach for finding deletion correcting codes," *Information Theory Workshop (ITW), 2011 IEEE*, pp. 25 –29, oct. 2011.

[4] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions and reversals," *Soviet Physics - Doklady*, vol. 10, no. 8, pp. 707–710, February 1966.

[5] ——, "Efficient reconstruction of sequences from their subsequences or supersequences," *J. Comb. Theory*, vol. 93, no. 2, pp. 310 – 332, 2001.

[6] N. Sloane, "Challenge problems: Independent sets in graphs http://www2.research.att.com/ njas/doc/graphs.html," January 2012. [Online]. Available: http://www2.research.att.com/ njas/doc/graphs.html

[7] ——, "On single-deletion-correcting codes," *Codes and Designs, Ohio State University*, pp. 273–292, May 2002.

[8] R. R. Varshamov, "On an arithmetic function with an application in the theory of coding," no. 3, pp. 540–543, 1965.

[9] D. B. West, *Introduction to graph theory*. Upper Saddle River, NJ: Prentice Hall Inc., 1996.