Neighborhood Preserving Hashing and Approximate Queries

Danny Dolev* Yuval Harari Nathan Linial † Noam Nisan Michal Parnas

Institute of Computer Science
Hebrew University, Jerusalem

Abstract

Let $D \subset \Sigma^n$ be a dictionary. We look for efficient data structures and algorithms to solve the *approximate query* problem: Given a query $u \in \Sigma^n$ list all words $v \in D$ that are close to u in Hamming distance.

The problem reduces to the the following combinatorial problem: hash the vertices of the n-dimensional hypercube into buckets so that (1) the $c-neighborhood$ of each vertex is mapped into no more than k buckets; (2) no bucket is too large.

Lower and upper bounds are given for the tradeoff between k and the size of the largest bucket. These results are used to derive bounds for the approximate query problem.

1 Introduction

Consider a text whose words all belong to a given dictionary D. Due to limited reliability, words may contain errors. Our task is to provide for every word from the text a list of alternative similar words from the dictionary, while minimizing search time and storage space. When the size of D is small enough, it may be feasible to search all of it for each such query (as done for example, by algorithms for the problem of string matching with k-differences; see [LV1], [LV2]). However, when D is large, a more efficient approach is needed. This problem, called the *approximate query problem*, can arise in many different fields. The most obvious examples come from the design of efficient spellers and speech-recognizers. Variations of the problem arise in fields such as the analysis of DNA sequences and proteins in Chemistry and Biology.

Hashing is a powerful tool in handling many similar problems. However, as we shall see, the hash functions we require need also to *preserve locality.* The requirements of hashing and of preserving locality seem to contradict. Informally, the objective of hashing is to "scatter" the given data, while preserving locality means doing the opposite. Both hashing and locality preservation are used quite widely in theory and in practice, but the tradeoffs between these two requirements are not well

*Supported in part by Yeshaya Horowitz Association.
†Supported in part by a grant from the Israeli Academy of Sciences.

understood. The combinatorial problems raised in this paper concern the extent to which hashing and preservation of locality can be satisfied together.

In the scenario under consideration, the dictionary D may change over time, and we desire a data structure that requires only minimal modifications upon changes in D. For concreteness, let $D \subset \{0,1\}^n$. Our approach is to look for a mapping $h : \{0,1\}^n \to B$, where B is a set of "buckets". Given a query $u \in \{0,1\}^n$, we would like to retrieve all words in D that are in the $c-neighborhood$ of u (i.e., vertices of Hamming distance at most c from u). Thus, in order to minimize the search time, h should map each $c - neighborhood$ into a small number, k, of buckets. This requirement reflects our desire for h to *preserve locality.* At the same time, we expect h to exihibt good *hashing* properties, in that each bucket should contain relatively few words. This requirement reduces the amount of redundant search among words not in the $c-neighborhood$ of u. We provide both upper and lower bounds on the tradeoff between k, c and the size of the largest bucket. These tradeoffs are used in Section 2, to derive bounds on the complexity of the approximate query problem.

A framework for questions of this type was developed in [DHP], where the problem is called the *Approximate Query Retrieval* problem. A similar problem called the Partial-Match Retrieval problem is studied in [R]. Numerous interesting applications of such problems can be found in [KS].

We can now state formally our problem and results. The **(c,k)-coloring** problem is defined as:

Color the vertices of the n-dimensional hypercube C_n, so that the $c - neighborhood$ of each vertex is colored with no more than k colors, and such that the largest color class is as small as possible.

n	k	max color	remarks
any	n	2	
any	$n-d+1$	2^d	d constant
2^r-1	$(n+1)/2$	$n+1$	uses Hamming codes
4^r-1	$1+n/3$	$O(n^3)$	uses Nearly perfect codes
see	remarks	$\sum_{i=0}^{t+1}\binom{n}{i}$, $t=O(n/\sqrt{k})$	when an (n,t)-quasi perfect code exists
any	any	$\sum_{i=0}^{t+d}\binom{n}{i}$, $t=O(n/k^{1/(d+1)})$	$d \le t$, uses a - general (n,t,d)-code
any	2	$2\binom{n}{n/2}$	

Table 1: Upper bounds for the $(1,k)$-coloring problem.

We call such a coloring a (c,k)-coloring, and say that each $c - neighborhood$ is $k - colored$. Our first results are lower bounds on the size of the largest color class. These bounds rely on isoperimetric inequalities for the cube, and are described in Section 3:

Theorem 1: *In any (1,k)-coloring of C_n there is a color that appears at least $\sum_{i=0}^{t}\binom{n}{i}$ times, where $t = \frac{n+2}{k+1} - 1$.*

Theorem 2: *In any (c,k)-coloring of C_n, there is a color that appears at least $\sum_{i=0}^{t}\binom{n}{i}$ times, where $t = \frac{n-c\sqrt[c]{k-1}}{\sqrt[c]{k-1}+2}$.*

Some of the applications require to color only some small subset of the hypercube. The following lower bound addresses this issue, and can be found in Section 4. Denote by $B_{n,r}$ the Hamming ball in C_n of radius r.

Theorem 3: *In any $(1,k)$-coloring of $B_{n,r}$, there is a color that appears at least $\binom{n/k}{r}$ times.*

We then turn to prove upper bounds, which are described in Section 5. Although we do not have a full answer, we show various colorings of the cube that match, or nearly match, the previous lower bounds. Some of these colorings are derived from error correcting codes, (see [PW] or [MS] for the definition of error correcting codes). Examples of the upper bounds obtained are described in Table 1.

We conclude by introducing a more general problem called the **(c,k,s)-covering problem**, which allows each word to be hashed to s entries in the hash table instead of to a single entry, using s hash functions instead of only one. A combinatorial formalization of it appears in Section 6, and possible directions towards the solution of this more general problem are given.

2 The Approximate Query Problem

Let $D \subset \Sigma^n$ be a dictionary of words of length n over some alphabet Σ. We concentrate on the case where the alphabet Σ is of size 2, as our combinatorial results directly translate to this case. The extensions to larger alphabet size are straightforward. Our results apply to algorithms A of the following type (see also [DHP]):

The Data Structure
The algorithm A stores the dictionary D in a hash table using a hash function $h_D : \Sigma^n \to \{1 \ldots B\}$, where B is the number of entries in the hash table. Each entry (bucket) in the hash table contains all the words in D that were hashed to it.

Answering queries
Define the $c-$*neighborhood* of $u \in \Sigma^n$ with respect to Σ^n to be:

$$N_c(u) = \{v | v \in \Sigma^n, \quad d(u,v) \leq c\},$$

where $d(u,v)$ is the Hamming distance of u and v. The $c-neighborhood$ of $u \in \Sigma^n$ with respect to D is:

$$N_c(u,D) = N_c(u) \cap D.$$

The algorithm A is given queries $u \in \Sigma^n$ and should determine $N_c(u,D)$. The algorithm A answers a query u by probing those buckets in the hash table that contain $N_c(u,D)$. Let S_u be the sequence of buckets searched by the algorithm as answer to query u. The sequence S_u is called the *search sequence* of u.

Complexity measures

The total time to answer a query u depends on the length $|S_u|$ of the search sequence of u, and the total number of words found in the buckets searched by S_u. We thus have two complexity measures:

1. The length $K_D = \max_u |S_u|$ of the longest search sequence used by the algorithm. Small search sequences are advantageous, since each access to a different bucket may result in a costly disk access.
2. The size M_D of the largest bucket created by the algorithm, i.e. the maximum number of words in D that were mapped by h_D to the same bucket. A small M_D reduces the number of irrelevant words read by A when answering a query u.

The first result we show is that there exist dictionaries D, for which the tradeoff between M_D and K_D is bad, for any hash function h_D, That is, M_D grows exponentially in n/K_D.

Corollary 2.1 *There exists a dictionary D, such that for every hash function h_D, $M_D \geq 2^{\Omega(n/K_D)}$.*

Proof: Let $D = \Sigma^n$. Then any hash function h_D for which the length of the search sequences is bounded by K_D is a $K_D - coloring$ of the the cube. The claim then follows from Theorems 1 and 2, using the Stirling approximation for the binomial coefficients. $\square$

The above corollary holds only for big dictionaries. Do there exist good hash functions for small dictionaries? Notice that for a random small enough dictionary D, the size of the $c-neighborhood$ of any query u w.r.t D is $|N_c(u,D)| = O(1)$ with high probability. Thus it is possible to map each word of D to a separate bucket, and there exists a short search sequence for each query (computing this search sequence may not be easy). We exhibit however, small dictionaries with no good hash function h. That is, either the search sequence is long or there exists a large bucket:

Corollary 2.2 *There exists a dictionary D of size $|D| = O(n^r)$, such that for every hash function h_D, $M_D \geq \frac{|D|}{K_D^r}$.*

Proof: Let D be a dictionary that includes all the words in some Hamming ball of radius r. The claim follows from Theorem 3. $\square$

It is possible to define a number of natural complexity measures in terms of K_D and M_D. In [DHP] one such possible time complexity measure is defined:

$$Time_A(h_D, u) = |S_u| + \sum_{i \in S_u} |B_i|,$$

where A is an algorithm that uses hash function h_D, and $u \in \Sigma^n$ is a query. The efficiency of A in terms of its time complexity can then be measured as:

$$T_A = \max_{D,u} \frac{TIME(h_D, u)}{|N_c(u, D)| + 1}.$$

The algorithm is *time optimal* if $T_A = O(1)$, that is, the algorithm gives an answer in time linear in the answer size.

An algorithm A for which the hash function h is fixed for any dictionary D, and the search sequence of a query u depends only on h and u, will be called $D - oblivious$. Such algorithms are of course preferred since they are easier to design, and can handle a dynamically changing dictionary D, without having to change the data structure with every change in the D. For such algorithms we give tight lower bounds on the time complexity defined above. Denote by N_c the size of a $c - neighborhood$ w.r.t. Σ^n. Thus for the Hamming distance $N_c = \sum_{i=0}^{c} \binom{n}{i} (\Sigma - 1)^i$. Note that usually $|N_c(u, D)| << N_c$. Let A be a $D - oblivious$ algorithm that is *space optimal* (i.e., uses space $|D|$). Then [DHP] show that $T_A = \Omega(\sqrt{N_c})$. We improve their result and show:

Corollary 2.3 *For any optimal space $D - oblivious$ algorithm A, $T_A = \Omega(N_c/4^c)$.*

Proof: The algorithm A is $D - oblivious$. Therefore it uses some fixed hash function h for all dictionaries D. If $K_A > N_c/4^c$, then there exists a query u for which $S_u > N_c/4^c$. Let D be any dictionary for which the $c - neighborhood$ of u is empty; that is, $|N_c(u, D)| = 0$. The claim follows.

If $K_A \leq N_c/4^c$, then h is an $(N_c/4^c) - coloring$ of the cube. Thus by Theorem 1 and Theorem 2, there exists a color set of size $\Omega(N_c^2/4^c)$. Let D be a dictionary that includes most of the words in this color set. Then the claim follows for any query u whose search sequence includes this set. $\square$

This bound is tight up to a factor of 4^c. Consider for example the $D - oblivious$ algorithm that maps each word to a separate bucket. The search sequence of a query u will include the buckets of all its possible $c - neighbors$ in Σ^n. Thus $|S_u| = |N_c(u)| = N_c$ and the time complexity is $N_c + |N_c(u, D)| = O(N_c)$.

3 Lower Bounds on the Size of the Largest Color Set in C_n in any (c, k)-Coloring

This section will provide lower bounds on the size of the largest color set in any (c, k)-coloring. If $k = 1$, it is clear that the whole cube should be colored in one color. For the remainder of this and the next section assume therefore that $k \geq 2$. The main theorems we prove in this section are:

Theorem 1: *In any (1,k)-coloring of C_n, there is at least one color that appears at least $\sum_{i=0}^{t} \binom{n}{i}$ times, where $t = \frac{n+2}{k+1} - 1$.*

Theorem 2: *In any (c,k)-coloring of C_n, there is at least one color that appears at least $\sum_{i=0}^{t} \binom{n}{i}$ times, where $t = \frac{n - c\sqrt[c]{k-1}}{\sqrt[c]{k-1}+2}$.*

Similar results can be obtained for a general alphabet Σ, using an extension of the Isoperimetric Inequality for larger alphabets. The details are omitted.

3.1 Proof of Theorem 1

Assume that all color sets are smaller than the size stated in the theorem. The Isoperimetric Inequality provides a lower bound on the number of neighbors of a set of vertices in C_n, as a function of the cardinality of the set. Thus every color set S, has at least $\beta|S|$ neighbors (where β is a decreasing function of $|S|$). The number of neighbors of all color sets can thus be bounded from below. On the other hand, the fact that the $1 - neighborhood$ of each vertex is $k - colored$, can be used to bound the number of neighbors of all color sets from above. An appropriate choice of parameters leads to a contradiction. We proceed with the detailed proof.

Denote by Γ_S the $1 - neighborhood$ of a subset $S \subset C_n$.

Lemma 3.1 (Isoperimetric Inequality) *Let S be a subset of C_n of size $|S| = \sum_{i=0}^{r-1} \binom{n}{i} + m$, where $0 \leq m \leq \binom{n}{r}$. Then $|\Gamma_S| \geq \binom{n}{r} - m + m\frac{\binom{n}{r+1}}{\binom{n}{r}}$.*

Proof: See [B], pages 122-129. □

Using this inequality, we can derive a lower bound on the ratio $|\Gamma_S|/|S|$ as a function of $|S|$.

Lemma 3.2 *Let S be any set of size $|S| \leq \sum_{i=0}^{r} \binom{n}{i}$. Then $\frac{|\Gamma_S|}{|S|} \geq \frac{\binom{n}{r+1}}{\sum_{i=0}^{r} \binom{n}{i}}$.*

Proof: Let S be any set with size $|S| = \sum_{i=0}^{r-1} \binom{n}{i} + m$, where $0 \leq m \leq \binom{n}{r}$.

By Lemma 3.1, the claim holds for $m = \binom{n}{r}$, and so it suffices to show that the function

$$f(m) = \frac{\binom{n}{r} - m + m\frac{\binom{n}{r+1}}{\binom{n}{r}}}{\sum_{i=0}^{r-1} \binom{n}{i} + m}$$

is decreasing over $0 \leq m \leq \binom{n}{r}$.

The inequality $f(m) \geq f(m+1)$ reduces to showing that

$$\binom{n}{r+1} \sum_{i=0}^{r-1} \binom{n}{i} \leq \binom{n}{r} \sum_{i=0}^{r} \binom{n}{i}.$$

This inequality follows by comparing the $i-th$ term on the left with the $i+1$ on the right. □

Lemma 3.3

- $\binom{n}{\alpha n+1} = \frac{1-\alpha}{1/n+\alpha} \binom{n}{\alpha n}$.
- $\sum_{i=0}^{\alpha n} \binom{n}{i} < \binom{n}{\alpha n} \frac{1-\alpha}{1-2\alpha}$ *where* α *is a constant,* $0 < \alpha < 1/2$.

Proof: The first part can be verified easily using the definition of the binomial coefficients. For the second part note that:

$$\begin{aligned} \sum_{i=0}^{\alpha n} \binom{n}{i} &< \binom{n}{\alpha n} (1 + \frac{\alpha}{1-\alpha} + (\frac{\alpha}{1-\alpha})^2 + ...) \\ &= \binom{n}{\alpha n} \frac{1-\alpha}{1-2\alpha}. \end{aligned}$$

□

As specified above, the size of each neighbor set is at least as large as the size of the set itself multiplied by some variable β, which depends on the size of the set. The following lemma shows the relationship between β and the size of the set.

Lemma 3.4 *Let S be any subset of C_n, where $|S| \leq \sum_{i=0}^{\alpha n} \binom{n}{i}$. Then $|\Gamma_S| > \beta |S|$ for $\beta = \frac{1-2\alpha}{\alpha+1/n}$.*

Proof: By Lemma 3.2 it is enough to prove the claim for sets S with size $|S| = \sum_{i=0}^{\alpha n} \binom{n}{i}$. By the Isoperimetric Inequality
$$|\Gamma_S| \geq \binom{n}{\alpha n + 1}.$$
Thus we have to show that:
$$\binom{n}{\alpha n + 1} > \beta \sum_{i=0}^{\alpha n} \binom{n}{i}.$$
If $\beta = 0$ this is obviously true. Therefore assume $\beta > 0$ (i.e., $\alpha < 1/2$), and then by Lemma 3.3 it is enough that:
$$\binom{n}{\alpha n} \frac{1-\alpha}{1/n + \alpha} \geq \beta \binom{n}{\alpha n} \frac{1-\alpha}{1-2\alpha}$$
as claimed. □

Proof of Theorem 1:

Let S_i be the color sets, and assume by contradiction that all sets are smaller than $\Sigma_{i=0}^{\alpha n} \binom{n}{i}$, for $\alpha = \frac{1}{k+1} - \frac{k-1}{n(k+1)}$. We will estimate the sum $\sum_i |\Gamma_{S_i}|$, and derive a contradiction.

Consider a vertex v colored i (i.e., $v \in S_i$). The $1-$ *neighborhood* of v is $k-$ *colored* and therefore v can be a neighbor of at most $k-1$ color sets (since it belongs itself to the color set S_i). Thus
$$\sum_i |\Gamma_{S_i}| \leq (k-1)2^n.$$

On the other hand, the Isoperimetric Inequality bounds the number of neighbors from below. By Lemma 3.4 and the choice of α, each color set S_i satisfies $|\Gamma_{S_i}| > (k-1)|S_i|$. Hence -
$$\sum_i |\Gamma_{S_i}| > \sum_i (k-1)|S_i| = (k-1)2^n,$$
a contradiction.

Therefore there exists at least one color set with size at least $\sum_{i=0}^{\alpha n} \binom{n}{i}$, where $\alpha n = \frac{n+2}{k+1} - 1$. □

3.2 Proof of Theorem 2

The same proof method will provide a lower bound for the more general (c,k)-coloring problem. Again we use the Isoperimetric Inequality for larger neighborhoods. Let Γ_S^c denote the $c-$ *neighborhood* of a subset $S \subset C_n$.

Lemma 3.5 (Isoperimetric Inequality) *Let S be a subset of C_n of size $|S| = \sum_{i=0}^{\alpha n} \binom{n}{i}$. Then $|\Gamma_S^c| \geq \sum_{i=1}^{c} \binom{n}{\alpha n + i}$.*

Lemma 3.6 *Let S be any subset of C_n, where $|S| \le \sum_{i=0}^{\alpha n} \binom{n}{i}$. Then $|\Gamma_S^c| > \beta|S|$ for $\alpha \le \frac{1}{\sqrt[c]{\beta}+2} - \frac{c}{n}\frac{\sqrt[c]{\beta}}{\sqrt[c]{\beta}+2}$.*

Proof: As in the proof of Lemma 3.4, it is possible to show using the Isoperimetric Inequality, that $|\Gamma_S^c| > \beta|S|$ for any set S with size $|S| = \sum_{i=0}^{\alpha n} \binom{n}{i}$, where $\alpha \le \frac{1}{\sqrt[c]{\beta}+2} - \frac{c}{n}\frac{\sqrt[c]{\beta}}{\sqrt[c]{\beta}+2}$. The proof that for any smaller set S, the ratio $\frac{|\Gamma_S^c|}{|S|}$ can only grow, is similar to that of Lemma 3.2. $\square$

Proof of Theorem 2:

Let S_i be the color sets, and assume by contradiction that all sets are smaller than $\Sigma_{i=0}^{\alpha n}\binom{n}{i}$, for $\alpha \le \frac{1}{\sqrt[c]{k-1}+2} - \frac{c}{n}\frac{\sqrt[c]{k-1}}{\sqrt[c]{k-1}+2}$.

In a similar way to the proof of Theorem 1, it is possible to estimate the sum $\sum_i |\Gamma_{S_i}^c|$, and derive a contradiction. $\square$

4 Lower Bounds on the size of the Largest Color Set in $B_{n,r}$ in any (c, k)-Coloring

Denote by L_r the $r'th$ layer of the cube, that is the set of all vertices of Hamming weight r, and let $B_{n,r} = \cup_{i=0}^{r} L_r$. We prove:

Theorem 3: *In any (1,k)-coloring of $B_{n,r}$, there is at least one color that appears $\Omega(\binom{n/k}{r})$ times.*

A similar theorem can be proved for the general $(c, k) - coloring$. The details are omitted. When proving Theorem 3, we will prove in fact a stronger claim, that there exists a large color set in the $r'th$ layer of $B_{n,r}$. The proof is similar to that of Theorem 1, but uses the concept of *shadows*, instead of the Isoperimetric Inequality. The *lower shadow* of a set $S \subset L_r$ is defined as:

$$\partial(S) = \{v \in L_{r-1} | \exists u \in S, \quad v \subset u\}.$$

Lemma 4.1 (Kruskal-Katona) *Let $\emptyset \ne S \subset L_r$ be a set of size $|S| = \binom{x}{r}$ for some real number $x \ge r$. Then $\partial(S) \ge \binom{x}{r-1}$.*

Proof: See [B] pages 23-39. $\square$

Corollary 4.1 *Let $S \subset L_r$ be any set of size $|S| < \binom{x}{r}$ for some real number $x \geq r$. Then $\partial(S) > |S|r/(x-r+1)$.*

Proof: Assume the size of S is $|S| = \binom{y}{r}$ for $y < x$. Then by Lemma 4.1,

$$\begin{aligned} \partial(S) &\geq \binom{y}{r-1} \\ &= |S|\frac{r}{y-r+1} > |S|\frac{r}{x-r+1}, \end{aligned}$$

where the equality follows from the definition of the binomial coefficients and the last inequality is true since $y < x$. $\square$

Proof of Theorem 3:

Consider any $(1,k)-coloring$ of $B_{n,r}$ and let S_i be the color sets in the $r'th$ layer of $B_{n,r}$. Assume by contradiction that all sets are smaller than $\binom{n/k}{r}$. We will estimate the sum $\sum_i |\partial(S_i)|$, and derive a contradiction.

Consider a vertex $v \in L_{r-1}$. The $1-neighborhood$ of v is $k-colored$ and therefore v can be a neighbor of at most k color sets in the $r'th$ layer. Thus v can belong to at most k shadows of sets S_i. The number of vertices in the $(r-1)'st$ layer is $\binom{n}{r-1}$ and thus:

$$\sum_i |\partial(S_i)| \leq k\binom{n}{r-1}.$$

On the other hand, by Corollary 4.1, each color set S_i satisfies $|\partial(S_i)| > |S_i|r/(n/k-r+1)$. Also $\sum_i |S_i| = \binom{n}{r}$. Hence

$$\sum_i |\partial(S_i)| > \sum_i |S_i|\frac{r}{n/k-r+1} = \binom{n}{r}\frac{r}{n/k-r+1}.$$

This is a contradiction for $r \geq 1$. $\square$

5 Upper Bounds for the (1,k)-Coloring Problem

How tight is the lower bound given in Theorem 1? Although we do not have a full answer to this question, we show colorings of the cube for certain values of k, for which the bound is tight or almost tight. We start with a few simple cases for specific values of k. In the next subsection, a more general strategy to color the cube will be developed, which is based on *Error correcting codes.*

The two extreme cases, $k = 1$ (where all the cube is colored in one color) and $k = n + 1$ (where each vertex is colored with a different color) are obviously completely solved.

Now let us turn to more interesting colorings. Define the $i'th$ layer of the cube to be all the vectors in C_n with exactly i coordinates that are 1. For $k = 2$, color layer j with color $\lfloor j/2 \rfloor$, $j \geq 0$. The color that appears the most times is the color of layers $\lfloor n/2 \rfloor$ and $\lfloor n/2 \rfloor - 1$. Thus for $k = 2$ there is a coloring of the cube in which each color appears at most $\binom{n}{\lfloor n/2 \rfloor} + \binom{n}{\lfloor n/2 \rfloor - 1}$ times. The lower bound proved in Section 3 for $k = 2$ is $\sum_{i=0}^{n/3} \binom{n}{i}$. There is a gap between the upper and lower bounds, and we conjecture that the upper bound is the best possible in this case.

For $k = n$ we can show a coloring in which each color appears at most twice. This is of course tight, since each $1 - neighborhood$ should contain at most $k = n$ colors, and the size of a $1 - neighborhood$ is $n + 1$. Thus there should be at least one color that appears twice. The coloring that achieves this bound is as follows:

Color every two vectors that agree on the first $n - 1$ coordinates and differ only in the last coordinate, with the same color. This coloring uses 2^{n-1} different colors, and every color appears exactly twice. It remains to show that each $1 - neighborhood$ is $n - colored$. But this is clear since the neighbor of a vector x, which differs from it only in the last coordinate, is colored the same as x.

For any $k = n - d + 1$, where d is a constant, we can color the cube in a similar way, such that each color appears 2^d times. Simply color each set of vectors that agree on all $n - d$ first coordinates, with the same color. Each vector has exactly d neighbors colored as itself. Thus each $1 - neighborhood$ is *(n-d+1)-colored.*

This method of coloring is efficient as long as d is constant. However when $k \leq n/2$, a different coloring is needed in order to match the lower bound. For this we use *Error correcting codes.*

5.1 Error Correcting Codes

Let C be a binary error correcting code of length n, with minimum distance $2t + 1$ between code words. Denote by $N_t(u)$ the sphere of radius t around a code word u, i.e., all vectors of distance at most t from u. Such a code is called an $(n, t) - code$ (see [PW] or [MS] for a comprehensive description of error-correcting codes). Given any $(n, t) - code$ we define the *induced coloring* of C_n:

Color each code word with a different color. Vectors not in the code are colored with the color of the nearest code word, breaking ties arbitrarily.

Notice that each sphere $N_t(u)$ around a code word u, is colored with the color of the code word u. Thus each color appears at least $\Sigma_{i=0}^{t} \binom{n}{i}$ times, and the $1 - neighborhoods$ of vectors at distance less than t from some code word are $1 - colored$. The question is what happens with vectors at distance greater than t from any code word. We first look at some special codes that provide better upper bounds.

5.1.1 Perfect Codes

An *(n,t)-perfect code*, is a code for which the spheres of radius t around the code words partition C_n. Thus every vector is at distance at most t from some code word. Perfect codes exist for only very restricted values of n and t. However since the same proof method will be used later to color the cube using a general code, assume for the moment that we have a perfect code for any n and t.

Begin with the *induced coloring* of some *(n,t)-perfect code*. Since this is a perfect code, each color appears exactly $\sum_{i=0}^{t}\binom{n}{i}$ times. It is left to bound the number of colors in the $1-neighborhood$ of each vector.

The $1-neighborhood$ of any vector x at distance less than t from some code word is $1-colored$. So consider only vectors x with distance exactly t from some code word u (i.e., x is on the boundary of the sphere $N_t(u)$). The vector x is colored the same as u, and it has exactly t neighbors in $N_t(u)$ that are colored in the same color as it is.

In how many colors are the other $n-t$ neighbors of x (those not in $N_t(u)$) colored? We claim that they are colored in at most $\frac{n-t}{t+1}$ colors. To see this note that each one of these neighbors belongs to some sphere $N_t(v)$ around a code word v, where v is at distance $t+1$ from x. Furthermore, there are exactly $t+1$ neighbors of x in any such sphere. Thus the neighbors of x that are not in $N_t(u)$, can be divided into equivalence classes of size $t+1$, according to the spheres around code words to which they belong.

Therefore the $n-t$ neighbors of x that are not in $N_t(u)$, are colored in $\frac{n-t}{t+1}$ colors. If we add to this the color of x and the t neighbors that are in $N_t(u)$, we get a total of $k = \frac{n-t}{t+1} + 1 = \frac{n+1}{t+1}$ colors in the $1 - neighborhood$ of x. We have thus proved:

Any (n,t)-perfect code induces a (1,k)-coloring of C_n, in which each color appears $\Sigma_{i=0}^{t}\binom{n}{i}$ times, and $t = \frac{n+1}{k} - 1$.

As we can see, this result almost matches the lower bound stated in Theorem 1. Unfortunately, perfect error correcting codes are rather rare. The Hamming code is perfect for $t = 1$ (i.e., $k = \frac{n+1}{2}$), and n of the form $n = 2^r - 1$, for some r. The Golay code is perfect for $n = 23$ and $t = 3$.

Corollary 5.1

1. *For any $n = 2^r - 1$, there exists a $(1, \frac{n+1}{2})$-coloring of C_n, in which each color appears $n+1$ times.*

2. *There exists a $(1,6)$-coloring of C_{23}, in which each color appears 24 times.*

However for larger values of t the situation is much worse. In fact it was proven that no other nontrivial perfect codes exist (see [V]). The resort is to examine larger classes of codes.

5.1.2 Quasi Perfect Codes

An *(n,t)-quasi perfect code* is a code for which spheres of radius t around code words are disjoint, and every vector is at distance at most $t+1$ from some code word. A sub-class of quasi perfect codes are *nearly perfect codes.* These codes were defined by Goethals and Snover, and their exact definition can be found in [GS].

Goethals and Snover ([GS]) list a few nearly perfect codes. For $t = 1$ there exists a nearly perfect code for any $n = 2^r - 2$. For $t = 2$ there exists a nearly perfect code for any $n = 4^r - 1$. Lindstrom proved that no other nearly perfect codes exist ([L]). The following lemma is from [GS]:

Lemma 5.1 ([GS]) *For any (n,t)-nearly perfect code:*

1. *Any vector at distance greater than t from any code word, is at distance t+1 from exactly* $\lfloor n/(t+1)\rfloor$ *code words.*
2. *Any vector at distance t from some code word, is at distance t+1 from exactly* $\lfloor (n-t)/(t+1)\rfloor$ *other code words.*

Corollary 5.2 *For any* $n = 4^r - 1$*, there exists a* $(1, 1+\frac{n}{3})$*-coloring of* C_n*, in which each color appears* $O(n^3)$ *times.*

Proof: Take an (n,t)-nearly perfect code, where $t = 2$ and $n = 4^r - 1$, and consider the *induced coloring.* Since this is a nearly perfect code, each word is at distance at most $t+1$ from some code word. Thus each color appears at most $\Sigma_{i=0}^{t+1}\binom{n}{i} = O(n^3)$ times.

We now have to show how many colors appear in each $1 -$ *neighborhood.* The interesting cases are those of vectors at distance t and $t+1$ from some code word. Let x be a vector at distance $t+1$ from some code word. Notice that each code word u at distance $t+1$ from x, forces $t+1$ of the neighbors of x to be colored in the same color as u. By Lemma 5.1 there are $\lfloor n/(t+1)\rfloor$ code words of distance $t+1$ from x. Since $n = 4^r - 1$ and $t+1 = 3$, it is easy to verify that $t+1$ divides n. Therefore the neighborhood of x is colored with $k = 1 + n/(t+1)$ colors (we add the color of x). The proof for vectors at distance t from some code word is similar. □

Similar results can be obtained for any *quasi-perfect code.* It is not known yet whether quasi perfect codes exist for large values of t, therefore we state the general theorem for any (n,t)-quasi perfect code, in hope that more codes will be found.

As before take any *(n,t)-quasi perfect code* and look at the *induced coloring.* Each vector is at distance at most $t+1$ from some code word, and so each color appears at most $\Sigma_{i=0}^{t+1}\binom{n}{i}$ times. We only have to bound the number of colors in each $1-neighborhood$. Again, the interesting cases are those of vectors at distance t or $t+1$ from some code word.

Lemma 5.2 *The 1-neighborhood of each vector is colored with at most $k = O(\frac{n^2}{t^2})$ colors.*

Proof: Let x be some vector. Denote by Z_i the number of code words at distance $t+i$ from x, where $i = 1, 2$. The number of colors in the $1-neighborhood$ of x does not exceed $1 + Z_1 + Z_2$, since the neighbors of x receive their colors from code words at distance $t+1$ and $t+2$ from x, and we have to add the color of x itself.

There are $\binom{n}{i}$ vectors at distance i from x. Call them the $i-neighbors$ of x. Let v be some code word at distance $t+i$ from x. Then, exactly $\binom{t+i}{i}$ of the $i-neighbors$ of x belong to $N_t(v)$. Since the spheres of radius t around code words are disjoint, there are at most $\binom{n}{i}/\binom{t+i}{i}$ code words at distance $t+i$ from x. Thus the number of colors in the $1-neighborhood$ of x is at most $k \leq 1 + \binom{n}{1}/\binom{t+1}{1} + \binom{n}{2}/\binom{t+2}{2} = O(n^2/t^2)$. □

Theorem 4: *Any (n,t)-quasi perfect code induces a (1,k)-coloring of C_n, in which each color appears at most $\Sigma_{i=0}^{t+1}\binom{n}{i}$ times, and $t = O(\frac{n}{\sqrt{k}})$.*

5.1.3 General Codes

The method of proof used for quasi perfect codes can be generalized for any code. An *(n,t,d)-code* is a code in which the spheres of radius t around code words are disjoint and the spheres of radius $t+d$ around code words cover the whole cube. Thus for perfect codes $d = 0$, and for quasi perfect codes $d = 1$.

It is again possible to look at the *induced coloring* of such a code, and bound the number of colors in each $1-neighborhood$ by $k = O((\frac{n}{t})^{d+1})$. Then a similar theorem can be proved:

Theorem 5: *Any (n,t,d)-code induces a (1,k)-coloring of C_n, in which each color appears at most $\Sigma_{i=0}^{t+d}\binom{n}{i}$ times, and $t = O(\frac{n}{k^{1/(d+1)}})$.*

Similar techniques can be used to show upper bounds for the *(c,k)-coloring* problem. The details are omitted.

6 The (c,k,s)-Covering Problem

In the previous sections we proved lower and upper bounds on the (c,k)-coloring problem, for which each word was hashed exactly once to the hash table. We now allow each word to be hashed to s entries in the hash table, using s hash functions. More formally, a $(c,k,s)-covering$ of Σ^n is a collection of subsets S_i that cover Σ^n such that:

- Each word $u \in \Sigma^n$, is contained in at most s of the subsets S_i.
- For any $u \in \Sigma^n$, there exist k subsets $S_{u_1}, ..., S_{u_k}$ such that $N_c(u) \subset \cup_{i=1}^{k} S_{u_i}$.

The **(c,k,s)-covering problem** is to minimize the size of the largest set S_i.

The solution of the (c,k)-coloring problem may give insight to the solution of the general (c,k,s)-covering problem.

The lower bound for the (c,k)-coloring problem was proved using the Isoperimetric Inequality. This inequality is tight for sets that are spheres and only for them (see [B]). On the other hand, the upper bound shows that a perfect code that covers the cube with disjoint spheres, induces a *(1,k)-coloring* that almost matches the lower bound. These observations make it plausible that in an optimal *(1,k)-coloring*, all color sets have the structure of spheres.

We also showed tradeoffs between the size of the color sets (or the radius t of the spheres) and k (see Theorem 1 and Theorem 2).

All this suggests that in order to find a *(c,k,s)-covering*, it may be wise to try to cover Σ^n with spheres (possibly overlapping). By changing the radius of the spheres - it may be possible to find a general tradeoff between the size of the largest subset, s and k.

For example, suppose we want to find an *(1,1,s)-covering* of C_n. It is possible to simply cover the whole cube with one subset. In this case $s = 1$ but there is one large subset. Or we can cover the cube by defining a different subset for the $1 - neighborhood$ of each vector. In this case the size of each subset is optimal, but s is large. Instead, we can try to combine these two methods:

Let C be an *(n,t)- perfect code*. Define a subset $S_u = N_{t+1}(u)$ for each $u \in C$. The coverings described above are extreme cases of this covering, with $t = n$ and $t = 0$. The size of each subset is $\sum_{i=0}^{t+1} \binom{n}{i}$, and it is possible to show that $s = (n+1)/(t+1)$. Thus by choosing t, it is possible to get the desired tradeoff between the size of each subset and s. (A similar method was used by [DHP] and [H] to design algorithms for the retrieval of neighbors from dictionaries.)

This method can be generalized to cover the cube using a general $(n,t,d) - code$, where the efficiency of the cover depends on d.

7 Conclusion

Several interesting open questions remain. The extensions of the upper bounds for the (c,k)-coloring problem and the (c,k,s)-covering problems used a general $(n,t,d) - code$, and their efficiency depended on d. Thus an important open question is to try to determine an upper bound on d that would guarantee the existence of an $(n,t,d) - code$, for any given n and t. An upper bound of $d = t$ is easy to show, but no better asymptoticly general bound is known. This question is of independent interest to the study of Error correcting codes, and some bounds were given for special types of codes (see [CKMS],[VS]).

There are also open problems concerning the (1,k)-coloring problem. The upper bound for this problem can probably be improved when k is a constant. The lower bound seems to be tight or almost tight for all cases, except for $k = 2$. We conjecture that for $k = 2$, the upper bound we showed of $O(\binom{n}{n/2})$ is the best possible.

Finally, it would be interesting to check what happens with different distance measures (not necessarily the Hamming distance). Different distance measures induce other graphs (other than the n-dimensional cube), for which one can try to solve the Coloring or Covering problems. Expanders may be an interesting class of graphs to check.

References

[B] B. Bollobas. Combinatorics: Set Systems, Hypergraphs, Families of Vectors, and Combinatorial Probability. *Cambridge University Press*, 1986.

[CKMS] G.D. Cohen, M.G. Karpovsky, H.F. Mattson and J.R. Schatz. Covering Radius - Survey and Recent Results. *IEEE Transactions on Information Theory,* Vol. 31, No. 3, 1985.

[DHP] D. Dolev, Y. Harari, M. Parnas. Finding the Neighborhood of a Query in a Dictionary. *The 2'nd Israel Symposium on Theory of Computing and Systems,* 1993.

[GS] J.M. Goethals and S.L. Snover. Nearly Perfect Binary Codes. *Discrete Mathematics 3*, 1972, pp. 65-88.

[H] Y. Harari. Algorithms and Lower Bounds for the Retrieval of Neighbors from a Dictionary. *M.Sc. thesis. Hebrew University, 1992. In Hebrew.*

[L] K. Lindstrom. The Nonexistence of Unknown Nearly Perfect Codes. *Ann. Univ. Turku., Ser. A, No. 169,* 1975, pp. 3-28.

[LV1] G.M. Landau and U. Vishkin. Efficient String Matching in the Presence of Errors. *Proc. 26th IEEE Symposium on Foundations of Computer Science,* 1985.

[LV2] G.M. Landau and U. Vishkin. Introducing Efficient Parallelism into Approximate String Matching and a New Serial Algorithm. *Proc. 18th Annual ACM Symposium on Theory of Computing,* 1986.

[MS] F.J. MacWilliams and N.J.A. Sloane. The Theory of Error Correcting Codes. *North-Holland Publishing Company,* 1977.

[PW] W.W. Peterson and E.J. Weldon. Error Correcting Codes. *MIT Press,* 1972.

[R] R.L. Rivest. On Hash-Coding Algorithms for Partial-Match Retrieval. *15'th Annual Symposium on Switching aand Automata Theory,* 1974.

[SK] D. Sankoff and J.B. Kruskal. Time Warps, Strings Edits and Macromolecules: The Theory and Practice of Sequence Comparison. *Addison-Wesley Publishing Company, Inc.*, 1983.

[V] J.H. Van Lint. A Survey of Perfect Codes. *Rocky Mountain Journal of Mathematics, Vol. 3, No. 2*, 1975, pp. 199-224.

[VS] S.G. Vladuts and A.N. Skorobogatov. Covering Radius for Long BCH Codes. *Problemy Peredachi Informatsii, Vol.25, pp. 38-45*, 1989.