SIAM J. Discrete Math. © 2002 Society for Industrial and Applied Mathematics
Vol. 15, No. 1, pp. 73–85

NEIGHBORHOOD PRESERVING HASHING AND APPROXIMATE QUERIES[*]

DANNY DOLEV[†], YUVAL HARARI[†], NATHAN LINIAL[†], NOAM NISAN[†], and MICHAL PARNAS[‡]

Abstract. Let $D \subseteq \Sigma^n$ be a dictionary. We look for efficient data structures and algorithms to solve the following *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 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 at most k buckets and (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.

Key words. approximate query, hashing, isoperimetric inequality, error correcting code

AMS subject classifications. 68P20, 68P05, 68P10, 05B40, 05D99

PII. S089548019731809X

1. Introduction.

Consider a text with words that all belong to a given dictionary D. Due to limited reliability, the 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 the 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 also need to *preserve locality.* The requirements of hashing and of preserving locality seem to contradict one another. 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 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 $D \subseteq \{0,1\}^n$. Our approach is to look for a mapping $h : \{0,1\}^n \to \{1,\ldots,B\}$, where B is the number of entries (buckets) in the hash table. Given a query $u \in \{0,1\}^n$, we would like to retrieve all words in D that

[*]Received by the editors March 7, 1997; accepted for publication (in revised form) November 15, 2001; published electronically January 4, 2002.
http://www.siam.org/journals/sidma/15-1/31809.html

[†]Institute of Computer Science, The Hebrew University, Jerusalem, Israel (dolev@cs.huji.ac.il, nati@cs.huji.ac.il, noam@cs.huji.ac.il). The work of the first author was supported in part by the Yeshaya Horowitz Association. The work of the third author was supported in part by a grant from the Israeli Academy of Sciences.

[‡]The Academic College of Tel-Aviv-Yaffo, 4 Antokolsky st., Tel-Aviv, Israel (michalp@mta.ac.il).

are in the *c-neighborhood* of u (i.e., words 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 exhibit 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 (see also [H] and [P]). A similar problem called the partial-match retrieval problem is studied in [R]. Numerous interesting applications of such problems can be found in [SK].

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

Color the vertices of the n-dimensional cube C_n so that the c-neighborhood of each vertex is colored with at most k colors and such that the largest color class is as small as possible.

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 C_n and are described in section 3. Specifically we prove the following theorems.

THEOREM 1.1. *In any $(1, k)$-coloring of C_n there exists a color that appears at least $\sum_{i=0}^{t} \binom{n}{i}$ times, where $t = \frac{n+2}{k+1} - 1$.*

THEOREM 1.2. *In any (c, k)-coloring of C_n, there exists 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 coloring only some small subset of the cube. The following lower bound addresses this issue and can be found in section 4. Denote by $B_{n,r}$ the Hamming ball of radius r in C_n.

THEOREM 1.3. *In any $(1, k)$-coloring of $B_{n,r}$, there exists a color that appears at least $\binom{n/k}{r}$ times.*

The upper bounds for the (c, k)-problem 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, our lower bounds. Some of these colorings are derived using error correcting codes (see [PW] or [MS] for relevant information on error correcting codes). Examples of the upper bounds obtained are described in Table 1.1.

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. Section 6 contains a combinatorial formalization of this problem, as well as possible directions for further research.

2. The approximate query problem.

Let $D \subseteq \Sigma^n$ be a dictionary of words of length n over some alphabet Σ. We concentrate on the case where $\Sigma = \{0, 1\}$, as our combinatorial results directly translate to this case. The extensions to a larger alphabet size are straightforward. Our results apply to algorithms of the following type (see also [DHP]).

The data structure. The algorithm stores the dictionary D in a hash table using a hash function $h_D : \Sigma^n \rightarrow \{1 \dots B\}$, where B is the number of entries in the

TABLE 1.1
Upper bounds for the $(1, k)$-coloring problem.

n	k	Largest color	Remarks
any	n	2	
any	$n-d+1$	2^d	d is a 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	$2\binom{n}{n/2}$	

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

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

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 is given queries $u \in \Sigma^n$ and should determine $N_c(u,D)$. The algorithm 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 an 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 the algorithm when answering a query u.

We first show that there exist dictionaries D, for which M_D grows exponentially in n/K_D, for any hash function h_D.

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

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 cube C_n. The claim then follows from Theorems 1.1 and 1.2, using the Stirling approximation for the binomial coefficients. ∎

Note that if c is a constant, then the c-neighborhood of any query is polynomial in size (even for the large dictionary D described in the proof). Yet if K_D is small, then there will be a bucket which contains many words that are irrelevant to a query whose search sequence includes this bucket. For example, if $D = \{0,1\}^n$, then the 1-neighborhood of each query contains $n+1$ words. However, if K_D is a constant, then Corollary 2.1 states that there exists a bucket that contains $2^{\Omega(n)}$ words.

The proof of Corollary 2.1 holds only for big dictionaries. Are there any good hash functions for small dictionaries? Notice that for a small enough random dictionary D, the size of the c-neighborhood of any query u with respect to 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 (although 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, $M_D = \Omega\left(|D|/K_D^r\right)$.*

Proof. Let D be a dictionary that includes all the words in some Hamming ball of radius r. The claim follows from Theorem 1.3. $\square$

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 a hash function h_D, $u \in \Sigma^n$ is a query, and B_i is the ith bucket in the hash table. The efficiency of A in terms of its time complexity can then be measured as

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

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

An algorithm 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 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 with respect to Σ^n. Thus for the Hamming distance $N_c = \sum_{i=0}^{c} \binom{n}{i}(|\Sigma|-1)^i$. Note that usually $|N_c(u,D)| \ll N_c$. 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 the following corollary.

COROLLARY 2.3. *Let A be an optimal space D-oblivious algorithm. Then $T_A = \Omega(N_c/4^c)$.*

Proof. Since algorithm A is D-oblivious, 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 $K_A \le N_c/4^c$, then h is an $(N_c/4^c)$-coloring of the cube. Thus by Theorem 1.1 and Theorem 1.2, 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. The claim follows for any query u whose search sequence includes this set. ∎

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 any (c, k)-coloring of C_n.

This section will provide lower bounds on the size of the largest color set in any (c, k)-coloring of the n-dimensional cube C_n. If $k = 1$, it is clear that the whole cube should be colored using one color. For the remainder of this and the next section assume therefore that $k \geq 2$. We now prove Theorems 1.1 and 1.2.

Recall that we assume that $\Sigma = \{0, 1\}$. 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 1.1.

Assume by contradiction 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 $S \subseteq C_n$, not including the vertices in S. That is,

$$\Gamma_S = \{v \notin S | \exists u \in S, d(u, v) = 1\}.$$

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 $|\Gamma_S| \geq \binom{n}{r} - m + m \cdot \binom{n}{r+1} / \binom{n}{r}$.*

Proof. See [B, pp. 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 $\frac{|\Gamma_S|}{|S|} \geq \binom{n}{r+1} / \sum_{i=0}^{r} \binom{n}{i}$.*

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

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

is decreasing over $0 \leq m \leq \binom{n}{r}$. The inequality $f(m) \geq f(m + 1)$ reduces to showing that

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

This inequality follows by comparing the ith term on the left with the $(i + 1)$ term on the right. ∎

LEMMA 3.3.

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

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

$$\sum_{i=0}^{\alpha n}\binom{n}{i} < \binom{n}{\alpha n}\cdot\left(1+\frac{\alpha}{1-\alpha}+(\frac{\alpha}{1-\alpha})^2+\cdots\right) = \binom{n}{\alpha n}\cdot\frac{1-\alpha}{1-2\alpha}.$$

This completes the proof. □

As specified above, the size of each neighbor set is at least as large as the size of the set itself multiplied by some number 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 $|S| \leq \sum_{i=0}^{\alpha n}\binom{n}{i}$, and $0 < \alpha < 1/2$. Then $|\Gamma_S| > \beta \cdot |S|$ 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

$$\binom{n}{\alpha n+1} > \frac{1-2\alpha}{\alpha+1/n}\cdot\sum_{i=0}^{\alpha n}\binom{n}{i}.$$

By Lemma 3.3 and the fact that $\beta = \frac{1-2\alpha}{\alpha+1/n}$ we get

$$\binom{n}{\alpha n+1} = \binom{n}{\alpha n}\cdot\frac{1-\alpha}{1/n+\alpha} = \beta\cdot\binom{n}{\alpha n}\cdot\frac{1-\alpha}{1-2\alpha} > \beta\cdot\sum_{i=0}^{\alpha n}\binom{n}{i}$$

as claimed. □

Proof of Theorem 1.1. The theorem is obviously true if $k = 1$ (the cube must be colored with one color) or $k = n+1$ (each vertex is colored with a different color). Therefore assume that $1 < k < n+1$, and let $\alpha = \frac{1}{k+1} - \frac{k-1}{n(k+1)}$. Note that by our assumption on k, $0 < \alpha < 1/2$.

Let S_i be the color sets and assume by contradiction that all sets are smaller than $\Sigma_{i=0}^{\alpha n}\binom{n}{i}$. We will bound $\sum_i |\Gamma_{S_i}|$ from above and below and derive a contradiction.

Consider a vertex v colored j (i.e., $v \in S_j$). 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 to the color set S_j). Therefore each one of the 2^n vertices belongs to at most $k-1$ of the neighborhoods Γ_{S_i}. Thus

$$\sum_i |\Gamma_{S_i}| \leq (k-1)2^n.$$

On the other hand, we can bound the number of neighbors from below as follows. If we set $\beta = k-1$ in Lemma 3.4, then, by our choice of α, each color set S_i satisfies $|\Gamma_{S_i}| > (k-1)|S_i|$. Hence,

$$\sum_i |\Gamma_{S_i}| > \sum_i (k-1)|S_i| = (k-1)2^n,$$

which is 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 1.2. The same proof method will provide a lower bound for the more general (c,k)-coloring problem. We use the isoperimetric inequality for larger neighborhoods. Let Γ_S^c denote the c-neighborhood of a subset $S \subseteq C_n$, not including the vertices in S. That is,

$$\Gamma_S^c = \{v \notin S | \exists u \in S, d(u,v) \leq c\}.$$

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 $|\Gamma_S^c| \geq \sum_{i=1}^{c} \binom{n}{\alpha n + i}$.*

Proof. See [B, pp. 122–129]. □

The following lemma is central to the proof of Theorem 1.2.

LEMMA 3.6. *Let S be any subset of C_n, where $|S| \leq \sum_{i=0}^{\alpha n} \binom{n}{i}$, and $0 < \alpha < 1/2$. Then $|\Gamma_S^c| > \beta \cdot |S|$ for $\beta = (\frac{1-2\alpha}{\alpha + c/n})^c$.*

Proof. Again let S be a set of size $|S| = \sum_{i=0}^{\alpha n} \binom{n}{i}$. (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.) Using the isoperimetric inequality we have to show that

$$\sum_{i=1}^{c} \binom{n}{\alpha n + i} > \left(\frac{1-2\alpha}{\alpha + c/n}\right)^c \cdot \sum_{i=0}^{\alpha n} \binom{n}{i}.$$

By applying Lemma 3.3 it is enough to show that

$$\sum_{i=1}^{c} \binom{n}{\alpha n + i} \geq \left(\frac{1-2\alpha}{\alpha + c/n}\right)^c \cdot \frac{1-\alpha}{1-2\alpha} \cdot \binom{n}{\alpha n}.$$

Let $a = \alpha n$. Thus, we have to prove that

$$\sum_{i=1}^{c} \binom{n}{a+i} \geq \left(\frac{n-2a}{a+c}\right)^c \cdot \frac{n-a}{n-2a} \cdot \binom{n}{a}.$$

Assume first that $n \geq 2a + c$. In this case we will prove the stronger inequality

$$\binom{n}{a+c} \geq \left(\frac{n-2a}{a+c}\right)^c \cdot \frac{n-a}{n-2a} \cdot \binom{n}{a}.$$

Notice that

$$\binom{n}{a+c} = \frac{n-a}{a+c} \cdot \frac{n-a-1}{a+c-1} \cdots \frac{n-a-(c-1)}{a+1} \cdot \binom{n}{a}.$$

Since $n \geq 2a + c$, then for every $0 \leq j \leq c-1$

$$\frac{n-a-j}{a+c-j} \geq \frac{n-a}{a+c}.$$

Therefore

$$\binom{n}{a+c} \geq \left(\frac{n-a}{a+c}\right)^c \cdot \binom{n}{a}.$$

We thus have to show that

$$\left(\frac{n-a}{a+c}\right)^c \geq \left(\frac{n-2a}{a+c}\right)^c \cdot \frac{n-a}{n-2a}.$$

This is obviously true for $a < n/2$, and therefore the claim follows for $n \geq 2a + c$.

Assume now that $n < 2a + c$ and recall that $a < n/2$. Thus $\binom{n}{a} \leq \binom{n}{a+1}$, and so

$$\sum_{i=1}^{c} \binom{n}{a+i} \geq \binom{n}{a}.$$

Since $n < 2a + c$, then $n - 2a < c$ and $n - a < a + c$. Thus,

$$\left(\frac{n-2a}{a+c}\right)^{c} \cdot \frac{n-a}{n-2a} \cdot \binom{n}{a} \leq \left(\frac{c}{a+c}\right)^{c-1} \cdot \binom{n}{a} \leq \binom{n}{a}.$$

This completes the proof in this case. □

Proof of Theorem 1.2. 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 $\alpha = \frac{1}{\sqrt[c]{k-1}+2} - \frac{c}{n} \cdot \frac{\sqrt[c]{k-1}}{\sqrt[c]{k-1}+2}$. Again, we can assume that $0 < \alpha < 1/2$, for otherwise the theorem is trivial.

As in Theorem 1.1, the c-neighborhood of any vertex v is k-colored, and therefore v is a neighbor of at most $k-1$ color sets. Thus

$$\sum_i |\Gamma^c_{S_i}| \leq (k-1)2^n.$$

However, by Lemma 3.6 and the choice of α, each color set S_i satisfies $|\Gamma_{S_i}| > (k-1)|S_i|$ (simply set $\beta = k-1$ in Lemma 3.6). Again a contradiction is derived, and therefore the theorem is proved.

4. Lower bounds on the size of the largest color set in any (c,k)-coloring of $B_{n,r}$. Denote by L_r the rth layer of C_n, that is, the set of all vectors in C_n with exactly r coordinates which are 1. Let $B_{n,r} = \cup_{i=0}^{r} L_r$ be the Hamming ball of radius r in C_n. We now prove Theorem 1.3. A similar result can be proved for the general (c,k)-coloring. The details are omitted.

When proving Theorem 1.3, we will prove in fact a stronger claim: there exists a large color set in the rth layer of $B_{n,r}$. The proof is similar to that of Theorem 1.1; however, it uses the concept of *shadows* instead of the isoperimetric inequality. The *lower shadow* of a set $S \subseteq L_r$ is defined as

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

LEMMA 4.1 (Kruskal–Katona). *Let $\emptyset \neq S \subseteq L_r$ be a set of size $|S| = \binom{x}{r}$ for some real number $x \geq r$. Then $|\partial(S)| \geq \binom{x}{r-1}$.*

Proof. See [B, pp. 23–39]. □

COROLLARY 4.2. *Let $S \subseteq L_r$ be any set of size $|S| < \binom{x}{r}$ for some real number $x \geq r$. Then $|\partial(S)| > |S| \cdot r/(x-r+1)$.*

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

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

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

Proof of Theorem 1.3. Consider any $(1,k)$-coloring of $B_{n,r}$, and let S_i be the color sets in the rth layer of $B_{n,r}$. Assume by contradiction that all sets are smaller than $\binom{n/k}{r}$. We will estimate $\sum_i |\partial(S_i)|$ 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 rth layer. Thus v can belong to at most k shadows of sets S_i. The number of vertices in the $(r-1)$ layer is $\binom{n}{r-1}$ and thus

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

On the other hand, by Corollary 4.2, $|\partial(S_i)| > |S_i| \cdot r/(n/k - r + 1)$. Also $\sum_i |S_i| = \binom{n}{r}$. Hence

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

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

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

How tight is the lower bound given in Theorem 1.1? 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 we develop a general strategy to color the cube, which is based on *error correcting codes*.

$\boldsymbol{k = 1, n + 1}$. The two extreme cases, $k = 1$ (where the cube is colored with one color) and $k = n + 1$ (where each vertex is colored with a different color), are obviously completely solved. Let us turn to more interesting colorings.

$\boldsymbol{k = 2}$. In this case, we color layer j with color $\lfloor j/2 \rfloor$, $j \geq 0$. It is easy to verify that the 1-neighborhood of each vertex is colored with at most $k = 2$ colors. What is the largest color set? 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 Theorem 1.1 for $k = 2$ is $\sum_{i=0}^{(n-1)/3} \binom{n}{i}$. There is a gap between the upper and lower bounds, and we conjecture that the upper bound is the best possible in this case.

$\boldsymbol{k = n}$. Here we can show a coloring in which each color appears at most twice. This is of course tight, since every 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. However, 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.

$\boldsymbol{k = n - d + 1}$, where $\boldsymbol{d}$ is a constant. We can color the cube in a similar way such that each color appears 2^d times. Simply color every 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 every 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 try and match the lower bound. For this we use error correcting codes.

5.1. Error correcting codes.

Let $\mathcal{C}$ 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 as follows:

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 only for very restricted values of n and t. However, since the same proof method will be used subsequently 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 remains 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. Therefore consider only vectors x at 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 with the same color as it is.

In how many colors are the remaining $n-t$ neighbors of x (those not in $N_t(u)$) colored? We claim that they are colored in at most $(n-t)/(t+1)$ 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 $(n-t)/(t+1)$ 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 $k=(n-t)/(t+1)+1=(n+1)/(t+1)$ colors in the 1-neighborhood of x. We have thus proved the following:

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, where $t=\frac{n+1}{k}-1$.

As we can see, this result almost matches the lower bound stated in Theorem 1.1. 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 the spheres of radius t around code words are disjoint, and every vector is at distance at most $t+1$ from some code word. A subclass 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 5.2 (see [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 5.3. *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 that is 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 5.2 there are $\lfloor n/(t+1)\rfloor$ code words at 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 with the 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 have only 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 5.4. *The 1-neighborhood of each vector is colored with at most $k = O(n^2/t^2)$ 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 $k \le 1 + \binom{n}{1}/\binom{t+1}{1} + \binom{n}{2}/\binom{t+2}{2} = O(n^2/t^2)$ as stated. ∎

THEOREM 5.5. *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, where* $t = O(n/\sqrt{k})$.

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 $k = O((n/t)^{d+1})$. Then a similar theorem can be proved.

THEOREM 5.6. *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, where* $t = O\left(n/k^{1/(d+1)}\right)$.

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

6. Generalizations and further research.

We have proved lower and upper bounds on the (c, k)-coloring problem, in which each word was hashed exactly once to the hash table. It may be useful to 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 $S_{u_1}, \ldots, S_{u_k}$ such that $N_c(u) \subseteq \cup_{i=1}^{k} S_{u_i}$.

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

The solution we showed for the (c, k)-coloring problem may give insight into 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 a tradeoff between k and the size of the color sets (or the radius t of the spheres) (see Theorems 1.1 and 1.2).

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

For example, suppose we want to find a $(1, 1, s)$-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 as follows.

Let $\mathcal{C}$ be an (n, t)-perfect code. Define a subset $S_u = N_{t+1}(u)$ for each $u \in \mathcal{C}$. 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] to design algorithms for the retrieval of neighbors from dictionaries. See also [H] and [P].) 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.

Several interesting open questions remain. The extensions of the upper bounds for the (c, k)-coloring problem and the (c, k, s)-covering problems use a general (n, t, d)-code, and their efficiency depends on d. Thus an important open question is to try and 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 asymptotically 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.

Acknowledgment. We would like to thank the anonymous referees for their comments, which improved the presentation of this paper.

REFERENCES

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

[CKMS] G.D. Cohen, M.G. Karpovsky, H.F. Mattson, and J.R. Schatz, *Covering radius—survey and recent results*, IEEE Trans. Inform. Theory, 31 (1985), pp. 328–343.

[DHP] D. Dolev, Y. Harari, and M. Parnas, *Finding the neighborhood of a query in a dictionary*, in Proceedings of the 2nd Israel Symposium on Theory of Computing and Systems, 1993, pp. 33–42.

[GS] J.M. Goethals and S.L. Snover, *Nearly perfect binary codes*, Discrete Math., 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, Jerusalem, Israel, 1992 (in Hebrew).

[L] K. Lindstrom, *The nonexistence of unknown nearly perfect codes*, Ann. Univ. Turku., Ser. A I, 169 (1975), pp. 3–28.

[LV1] G.M. Landau and U. Vishkin, *Efficient string matching in the presence of errors*, in Proceedings of the 26th IEEE Symposium on Foundations of Computer Science, 1985, pp. 126–136.

[LV2] G.M. Landau and U. Vishkin, *Introducing efficient parallelism into approximate string matching and a new serial algorithm*, in Proceedings of the 18th Annual ACM Symposium on Theory of Computing, 1986, pp. 220–230.

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

[P] M. Parnas, *Robust Algorithms and Data Structures for Information Retrieval*, Ph.D. thesis, Hebrew University, Jerusalem, Israel, 1994.

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

[R] R.L. Rivest, *On hash-coding algorithms for partial-match retrieval*, in Proceedings of the 15th Annual Symposium on Switching and Automata Theory, IEEE, 1974, pp. 95–103.

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

[V] J.H. Van Lint, *A survey of perfect codes*, Rocky Mountain J. Math., 3 (1975), pp. 199–224.

[VS] S.G. Vladuts and A.N. Skorobogatov, *Covering radius for long BCH codes*, Problemy Peredachi Informatsii, 25 (1989), pp. 38–45.