Data Structures for Storing Small Sets in the Bitprobe Model

Jaikumar Radhakrishnan[1], Smit Shah[2] ⋆ , Saswata Shannigrahi[1] ⋆⋆

[1] Tata Institute of Fundamental Research, Mumbai, India.
{jaikumar,saswata}@tifr.res.in

[2] Institute of Technology, Nirma University, Ahmedabad, India.
{05bce106}@nirmauni.ac.in

Abstract. We study the following set membership problem in the bit probe model: given a set S from a finite universe U, represent it in memory so that membership queries of the form "Is x in S?" can be answered with a small number of bitprobes. We obtain explicit schemes that come close to the information theoretic lower bound of Buhrman et al. [STOC 2000, SICOMP 2002] and improve the results of Radhakrishnan et al. [ESA 2001] when the size of sets and the number of probes is small.

We show that any scheme that stores sets of size two from a universe of size m and answers membership queries using two bitprobes requires space $\Omega(m^{4/7})$. The previous best lower bound (shown by Buhrman et al. using information theoretic arguments) was $\Omega(\sqrt{m})$. The same lower bound applies for larger sets using standard padding arguments. This is the first instance where the information theoretic lower bound is found to be not tight for adaptive schemes.

We show that any non-adaptive three probe scheme for storing sets of size two from a universe of size m requires $\Omega(\sqrt{m})$ bits of memory. This extends a result of Alon and Feige [SODA 2009] to small sets.

1 Introduction

This paper addresses the following set membership problem in the bit probe model: given a set S from a finite universe U, represent it in memory so that membership queries of the form "Is x in S?" can be answered by reading a few bits. This problem was first studied by Buhrman, Miltersen, Radhakrishnan and Venkatesh [2]. Let (n, m, s, t)-scheme be a solution to this problem that uses s bits of memory and answers queries correctly for all n-element subsets of an m-element universe using t probes. Let $s_N(n, m, t)$ denote the minimum space s such that there exists a deterministic (n, m, s, t)-scheme that answers queries with t *non-adaptive* probes. (We replace the subscript N by A when we wish to emphasize that the probes are allowed to be adaptive.) Using this terminology,

⋆ A part of the work was done when the author was visiting TIFR Mumbai.

⋆⋆ The author has been partially supported by IBM PhD Fellowship 2009-2010.

the results of Buhrman et al. [2] can be stated as

$$s_N(n,m,2t+1) = O(ntm^{2/(t+1)});$$
$$\text{and } s_A(n,m,t) = \Omega(nt(\frac{m}{n})^{1/t}).$$

In this work, we will primarily be concerned with bounds when n and t are small (say constants, or at most $O(\log m)$), and focus on the constant in front of $\frac{1}{t}$ in the exponent of m; in particular, we would like to know if that constant could be 1 (matching the lower bound).

The upper bound shown by Buhrman et al. [2] used probabilistic existence arguments. Explicit constructions were studied by Radhakrishnan et al. [3]. In particular, they showed that $s_A(n,m,\lceil\lg(n+1)\rceil+1) \leq (n+\lceil\lg(n+1)\rceil)\sqrt{m}$. Note that the dependence on m in this result is in general inferior to the $m^{4/t}$ dependence in the bound of above. Our first result improves both these results when t is much larger than n.

Result 1 **(a)** *We give an explicit scheme that shows that $s_A(n,m,t) \leq Ct^n \cdot m^{1/(t-n+1)}$, for $t > n \geq 2$, and a constant C independent of n and t.*
(b) *Using a probabilistic argument we obtain a deterministic scheme that shows that $s_A(n,m,t) \leq Cntm^{1/(t-nt2^{-t+1})}$, for $t > n \geq 2$, and a constant C independent of n and t.*

The main point to note is that for small n and t somewhat larger than n, the exponent of m in these bounds approaches $\frac{1}{t}$, as in the lower bound above.

The above results use space close to the lower bound when the number of probes is large. Are savings possible with a small number of probes? Alon and Feige [1] studied the "power of two, three and four probes." They showed the following upper bounds for adaptive schemes.

Two probes: For $n < \log m$, $s_A(n,m,2) = O(\frac{mn\log\frac{\log m}{n}}{\log m})$.
Three probes: $s_A(n,m,3) = O(n^{\frac{1}{3}}m^{\frac{2}{3}})$.
Four probes: $s_A(n,m,4) = O(n^{\frac{1}{4}}m^{\frac{3}{4}})$.

These three upper bounds are remarkable in that they show that the space requirement can be $o(m)$ even with a small constant number of probes (disproving a conjecture of [2]).

Even for the simple case of $t = 2$ and $n = 2$, the current bounds [2] are not tight:

$$\sqrt{m} \leq s_A(2,m,2) \leq m^{2/3}.$$

We conjecture that the upper bound is tight. For non-adaptive probes, Buhrman et al. determined that $s_N(2,m,2) = m$; this, in particular, shows that $s_A(2,m,2) = o(s_N(2,m,2))$. Our next result shows an improved lower bound even when the queries are allowed to be adaptive.

Result 2 $s_A(2,m,2) = \Omega(m^{4/7})$.

This implies a similar lower bound as long as $n = o(m)$. This is the first lower bound for general schemes (previous bounds apply only to non-adaptive schemes) that beats the information theoretic lower bound of Buhrman et al. mentioned above.

Alon and Feige [1] showed the following lower bound for non-adaptive three probe schemes with $n \geq 16 \log m$.

$$s_N(n, m, 3) = \Omega(\sqrt{\frac{nm}{\log m}}).$$

We obtain the following result for $n = 2$.

Result 3 $s_N(2, m, 3) = \Omega(\sqrt{m})$.

Note that this result shows that the $\sqrt{m}$ dependence on m shows up already for sets of size 2, and extends the result of Alon and Feige above to sets of size less than $\log m$. Furthermore, Result 1 (b) shows that $s_A(2, m, 3) = O(m^{0.4})$. Thus, adaptive schemes are more efficient than non-adaptive schemes for $n = 2, t = 3$. A similar result was observed by Buhrman et al. for $n = 2, t = 2$.

Buhrman et al. [2] showed a non-explicit construction of a non-adaptive t probe scheme with $O(ntm^{4/t})$ space. Ta-Shma [4] gave explicit constructions of non-adaptive schemes. One can using the techniques of Buhrman et al. obtain an explicit non-adaptive (n, m, s, t) scheme with $s = 2tm^{\frac{n}{t+n-1}}$. For $n = 2, 3$, this is an improvement over the $O(ntm^{4/t})$ scheme of Buhrman et al. The details are omitted from this version of the paper.

Techniques used

Upper bounds: The two upper bounds results (Results 1 (a) and 1 (b)) are shown using different methods. Result 1 (a) is elementary, but it uses adaptiveness in a careful way. It employs schemes for storing sets of various sizes, and distributes the set to be stored among these schemes. Using the first few probes the membership algorithm is led to the scheme where this element may be found.

Result 1 (b) uses a probabilistic argument. The query scheme first make $t-1$ non-adaptive probes and determines the section of the memory it must finally probe to determine if the given element is in the set. The first $t-1$ probes can be explicitly described, it is only the location of the last probe is assigned probabilistically.

Lower bounds: The proof of Result 2 uses a graph theoretic formulation of the problem. It was shown by Radhakrishnan et al. [3] that for certain restricted schemes, $s(2, m, 2) = \Omega(m^{2/3})$.

While not resorting to the artificial restriction of the previous proof, our proof only yields a weaker bound of $m^{4/7}$. As stated the result beats the routine information-theoretic lower bound of $\Omega(\sqrt{m})$ proved earlier, where one shows that the small data structure leads to a short encoding of sets of size 2; since any such encoding must use at least $\log \binom{m}{2}$ bits, one infers a lower bound on the size of data structure. The proof in this paper departs from this information theoretic framework, in that it does not directly derive encodings of sets from the efficient data structure. The proof in this paper uses the graph theoretic formulation similar to that of Radhakrishnan et al. [3], but needs a more technical and complicated argument to deal with what intuitively appear to be easy cases.

Result 3 in contrast applies only to non-adaptive schemes but allows for three probes. As stated above, Buhrman et al. showed that $s_N(2, m, 2) = m$. We extend their argument to three probes. This involves considering cases based on the functions used by the query algorithm. In most cases, we are able to show that the scheme contains a $(2, m', s, 2)$-non-adaptive scheme on a universe of size $m' = \Omega(\sqrt{m})$; the results of Buhrman et al. then immediately gives us the desired lower bound.

2 Result 1: Adaptive schemes

In this section, we obtain explicit schemes that come close to the information theoretic lower bound of Buhrman et al. [2] and improve the results of Radhakrishnan et al. [3] when the size of sets and the number of probes is small. We first mention a result of Radhakrishnan et al. that we will use to prove Theorem 1 and Theorem 2.

Lemma 1. *(Theorem 1 of [3]) There is an explicit adaptive (n, m, s, t) scheme with $t = \lceil \lg(n + 1)\rceil + 1$ and $s = (n + \lceil \lg(n + 1)\rceil)\sqrt{m}$.*

We now present a proof of Result 1 (a) for the case of $n = 2$.

Theorem 1. *For any $t > 2$, there is an explicit adaptive $(2, m, s, t)$ scheme with $s = t^2 m^{\frac{1}{t-1}}$.*

Proof: We show how a multiprobe scheme can be built up from some schemes that use fewer probes.

Claim. Let $(1, m_2, s_1, t-2)$ and $(2, m_2, s_2, t-1)$ be two adaptive schemes. Then, there exists a $(2, m_1 m_2, s, t)$ adaptive scheme such that $s \le s_1 + s_2 + 2m_1$.

We will justify this claim below. Let us first verify that this yields our theorem by induction. For $t = 3$, the claim follows from Lemma 1 which tells that there is an explicit adaptive $(2, m, 4m^{\frac{1}{2}}, 3)$ scheme; this gives $s_A(2, m, 3) \le 4m^{\frac{1}{2}}$. Also, it is easy to see that $s_A(1, m, t) \le tm^{\frac{1}{t}}$. (To each element of the universe assign a distinct t-tuple from $f_x \in [m^{1/t}]^t$. The memory will consist of t tables $T_1, \ldots, T_t$, each with $m^{1/t}$ bits, and the query "Is x in S?" will be answered yes iff $T_j[f_x[j]] = 1$ for all j.) Now, let $m_1 = m^{\frac{1}{t-1}}$ and $m_2 = m^{\frac{t-2}{t-1}}$. Then, the claim gives: $s_A(2, m_1 m_2, t) \le (t-2)m^{\frac{1}{t-1}} + (t-1)^2 m^{\frac{1}{t-1}} + 2m^{\frac{1}{t-1}} \le t^2 m^{\frac{1}{t-1}}$ for any $t > 2$.

It remains to justify the claim. Partition the universe into m_1 blocks B_1, B_2, …, of size at most m_2 each. We have a table T with m_1 rows, each with two bits (this accounts for the $2m_1$ term), and space for one $(1, m_2, s_1, t-2)$-scheme N_1 and another adaptive $(2, m_2, s_2, t-1)$-scheme N_2. To answer the query "Is x in

S?" for an element $x \in B_i$, the query algorithm reads the first bit of $T[i]$, and if it is 1, it invests the remaining $t-1$ probes in N_2 assuming a $(2, m_2, s_2, t-1)$-scheme for B_i is implemented there. Otherwise, it reads the second bit and if it is 1, invests the remaining $t-2$ probes in the scheme N_1 assuming an $(1, m_2, s_1, t-2)$-scheme for B_i is implemented there. If the two bits read from T are 00, it answers No.

Now, given a set $\{x, y\}$ the storage is determined as follows.

(i) If there are distinct blocks B_i and B_j with one element each, we set $T[i] = 10$, $T[j] = 01$ and set the other entries of T to 00. We store the elements of B_i using N_2 and the elements of B_j using N_1.
(ii) If $\{x, y\} \subseteq B_i$ for some block B_i, we set $T[i] = 10$ and set the other entries of T to 00. The elements in B_i are now represented in the scheme N_2. □

We next generalize this result to larger sets. This will complete the proof of Result 1 (a).

Theorem 2. *For every $n \geq 2$ and $t > n$, there is an explicit adaptive (n, m, s, t)-scheme with $s = t^n m^{\frac{1}{t-n+1}}$.*

Proof: We demonstrate a scheme inductively. Lemma 1 shows that there exists an explicit adaptive $(n, m, 2nm^{\frac{1}{2}}, n+1)$ scheme for all $n \geq 2$. (Lemma 1 actually demonstrates a scheme with smaller space and fewer probes, but this scheme is sufficient for our proof.) Moreover, it is proven in Theorem 1 above that there is an explicit adaptive $(2, m, t^2 m^{\frac{1}{t-1}}, t)$ scheme for every $t > 2$. We use these two schemes as the base cases for our induction argument (note that $2n \leq (n+1)^2$ for all $n \geq 2$). We also note that there is an non-adaptive $(1, m_1, t_1 m_1^{\frac{1}{t_1}}, t_1)$ scheme for every m_1 and every $t_1 \geq 2$.

For induction, we assume that an explicit adaptive $(i_2, m_2, {t_2}^{i_2} {m_2}^{\frac{1}{t_2 - i_2 + 1}}, t_2)$ scheme exists for every m_2, and every i_2 and t_2 satisfying $2 \leq i_2 < n$ and $i_2 < t_2 \leq t$. We also assume that an explicit adaptive $(n, m_3, {i_3}^n {m_3}^{\frac{1}{i_2 - n + 1}}, i_3)$ scheme exists for every m_3 and every i_3 satisfying $n < i_3 < t$.

We now demonstrate our storage scheme. We divide the universe of size m into $m^{\frac{1}{t-n+1}}$ blocks of size at most $m^{\frac{t-n}{t-n+1}}$ each. The first part of the storage scheme consists of a table T with $m^{\frac{1}{t-n+1}}$ entries, each with an entry which is at most n bits long. Each entry of this table corresponds to a block. Assume that there are l blocks each of which contain at least one element inside it (note that l is at most n). We order these blocks by the number of elements inside them; let the ordering be $B_1, B_2, \ldots, B_l$ where B_k contains at least as many elements as in B_{k+1}, for all $1 \leq k \leq l-1$. Note that B_k contains at most $n-k+1$ elements. At the entry of T that corresponds to B_k, we store a string of $(k-1)$ zeroes followed by an 1. If a block does not contain any element, we store n zeroes in the corresponding entry of the table.

The second part of the storage scheme consists of $(n-1)$ adaptive schemes and one non-adaptive scheme. We denote them by $S_1, S_2, \ldots, S_n$. For every $1 \leq j \leq (n-1)$, S_j is an explicit adaptive $(n-j+1, m^{\frac{t-n}{t-n+1}}, (t-j)^{n-j+1} m^{\frac{1}{t-n+1}}, t-j)$

scheme whose existence is guaranteed by the induction hypothesis. S_n is a non-adaptive $(1, m^{\frac{t-n}{t-n+1}}, (t-n)m^{\frac{1}{t-n+1}}, t-n)$ scheme. For all $1 \leq j \leq l$ (note that l is the number of blocks that contain at least one element inside them), S_j is used to store the block B_j.

Let $S_{\{t,n\}}$ be the total space required by the above storage scheme. We note that the first part of the storage scheme takes $nm^{\frac{1}{t-n+1}}$ amount of space. Hence, $S_{\{t,n\}} = nm^{\frac{1}{t-n+1}} + (t-1)^n m^{\frac{1}{t-n+1}} + (t-2)^{n-1} m^{\frac{1}{t-n+1}} + \ldots + (t-n)m^{\frac{1}{t-n+1}}$. This number is less than $t^n m^{\frac{1}{t-n+1}}$ for all $n \geq 2$ and $t > n$.

The query scheme for an element x in the universe is as follows. The scheme first finds the block x belongs to and then probes the corresponding location in the table. It sequentially probes the binary string stored at that location till it either finds an 1 or finds a string containing n zeroes. If it finds an 1 at the i-th position, it gets directed to the scheme S_i. If that scheme returns "yes", the query scheme answers "yes" to the query x. If it finds a string of n zeroes in the table, it answers that the element x is not present in the universe. $\square$

We use a probabilistic argument to show that there exists a scheme which uses less space than the space in the above theorems. This proves Result 1 (b).

Theorem 3. *Suppose n and t are integers satisfying $n \geq 2$ and $t > n$. Then, there exists an adaptive (n, m, s, t) scheme with $s = 8nt \cdot (2m)^{\frac{2^{t-1}}{t2^{t-1}-(n-1)(t-1)}}$. This scheme is non-explicit.*

Proof: We prove the theorem for $n = 2$. The proof for $n > 2$ uses similar ideas. (See below for the complete proof.)

The storage scheme consists of two parts: the index part and the actual storage part. Let s' be an integer (whose value will be determined later) such that the universe of size m is divided into $\left(s'2^{t-1}\right)^{t-1}$ blocks of size at most $\frac{m}{(s'2^{t-1})^{t-1}}$ each. The index part consists of a $(t-1)$-partite complete $(t-1)$-uniform hypergraph H with $s'2^{t-1}$ vertices in each partition. This hypergraph has $\left(s'2^{t-1}\right)^{t-1}$ distinct hyperedges. Each block is assigned a unique hyperedge, called its *index hyperedge*. The vertices of the hyperedges are used to store either 0 or 1. We say that the index hyperedge of a block contains k if the sequentially stored bits at the vertices of the hyperedge represent the integer k.

The actual storage part consists of 2^{t-1} tables T_0, T_1, T_2, $\ldots$, $T_{2^{t-1}-1}$, each of size s'. Each of these tables is divided into $\frac{(s')^t 2^{(t-1)^2}}{m}$ sub-tables of size $\frac{m}{(s')^{t-1}2^{(t-1)^2}}$ each. For every $0 \leq i \leq 2^{t-1} - 1$, each block of the universe is randomly assigned one sub-table of T_i. We call two blocks to be *colliding* in T_i if they are assigned the same sub-table in T_i.

If there is a block B that contains at most two elements (and others contain none), the storage scheme stores its characteristic vector inside a sub-table in one of the tables T_j and assigns j to B's index hyperedge. Any other block is assigned a number different from j in its index hyperedge and the storage scheme stores 0 in all locations of the remaining tables. If there are two blocks B_1 and B_2 that contain one element each, we probabilistically show below that it is possible to find a table T_k and a scheme S_k such that: (i) the index hyperedges of both B_1 and B_2 contain k, (ii) B_1 and B_2 do not collide in T_k and their sub-tables store the corresponding characteristic vectors, (iii) the index hyperedges that are not subsets of the union of the index hyperedges of B_1 and B_2 contain a number not equal to k, (iv) neither B_1 nor B_2 collides with at most $2^{t-1}-2$ other blocks whose index hyperegdes are subsets of the union of B_1's and B_2's index hyperedges and therefore contain k in their index hyperedges, and (v) all tables other that T_k store 0 in all locations inside them. We call (B_1, B_2) to be a *bad pair* if no table T_k satisfies the above conditions. We need to show that no such bad pair exists in our scheme. From the query scheme described below, it will be clear that such a storage scheme correctly stores the elements of the universe.

Given an element x in the universe, the query scheme first finds out the block B' it belongs to. It then makes $t-1$ sequential probes into B''s index hyperedge. If the index hyperedge contains i, then the query scheme looks at T_i's sub-table that contains the characteristic vector of B'. The query scheme returns a "yes" if and only if it finds an 1 at the location of x in the characteristic vector.

Let us now show that a scheme S_k (as described above) exists. For any i, the probability that a pair of blocks collide in T_i is at most $\frac{m}{(s')^t 2^{(t-1)^2}}$. Therefore, the probability that even one of $2 \cdot 2^{t-1} - 3$ pairs of blocks collide in T_i is at most $2 \cdot 2^{t-1} \frac{m}{(s')^t 2^{(t-1)^2}}$. Hence, the probability that (B_1, B_2) is a bad pair is at most $\left[2^t \frac{m}{(s')^t 2^{(t-1)^2}}\right]^{2^{t-1}}$. It follows that the expected number of bad pairs of blocks is at most $\left[s' 2^{t-1}\right]^{2(t-1)} \left[2^t \frac{m}{(s')^t 2^{(t-1)^2}}\right]^{2^{t-1}}$. So, there exists an assignment of sub-tables in which there are at most $\left[s' 2^{t-1}\right]^{2(t-1)} \left[2^t \frac{m}{(s')^t 2^{(t-1)^2}}\right]^{2^{t-1}}$ bad pairs of blocks. Even if half of all possible pairs of blocks are bad, we get a scheme for storing $\frac{m}{2}$ elements by deleting at most one block from each bad pair of blocks.

If we summarize the above discussion, it follows that s' has to satisfy the following inequality.

$$\left[s' 2^{t-1}\right]^{2(t-1)} \left[2^t \frac{m}{(s')^t 2^{(t-1)^2}}\right]^{2^{t-1}} \leq \frac{\left[s' 2^{t-1}\right]^{t-1}}{2}$$

This is satisfied when $s' \geq \frac{32}{2^t} m^{\frac{2^{t-1}}{t 2^{t-1} - t + 1}}$. Thus, the total space required for a scheme for $\frac{m}{2}$ sized universe is $t \cdot (2^{t-1}) s' = 16t \cdot m^{\frac{2^{t-1}}{t 2^{t-1} - t + 1}}$. Hence, there exists an adaptive $(2, m, s, t)$ scheme with $s = 16t \cdot (2m)^{\frac{2^{t-1}}{t 2^{t-1} - t + 1}}$.

Complete Proof of Theorem 3: We use a storage scheme similar to the one described in the proof for $n = 2$. The only difference now is that the scheme has to ensure that any n blocks are stored correctly. Each block can now collide with at most n^{t-1} other blocks. This number used to be 2^{t-1} in the case $n = 2$. In summary, we need to find an s' that satisfies

$$\left[s'2^{t-1}\right]^{n(t-1)}\left[n\cdot n^{t-1}\frac{m}{(s')^t 2^{(t-1)^2}}\right]^{2^{t-1}} \leq \frac{\left[s'2^{t-1}\right]^{t-1}}{2}$$

$$\Rightarrow 2^{(t-1)^2(n-1)+(2t-1)2^{t-1}}\cdot\left[\frac{m}{\left(\frac{2^t (s')^{1-\frac{(n-1)(t-1)}{t2^{t-1}}}}{n}\right)^t}\right]^{2^{t-1}} \leq \frac{1}{2} \qquad (1)$$

If

$$s' \geq \left(\frac{8n}{2^t}\right)^{\frac{t2^{t-1}}{t2^{t-1}-(n-1)(t-1)}}\cdot(m)^{\frac{2^{t-1}}{t2^{t-1}-(n-1)(t-1)}}\,, \qquad (2)$$

then the left hand side of the equation (1) is

$$\frac{2^{(t-1)^2(n-1)+(2t-1)2^{t-1}}}{(8)^{t2^{t-1}}} = \frac{1}{(2)^{(t+1)2^{t-1}-(t-1)^2(n-1)}}.$$

This is at most $\frac{1}{2}$ whenever $(t+1)2^{t-1}-(t-1)^2(n-1)\ \geq 1$ (since $n \geq 2$ and $t > n$). Note that in order to satisfy equation (2) it is enough to find an s' satisfying $s' \geq \left(\frac{16n}{2^t}\right)(m)^{\frac{2^{t-1}}{t2^{t-1}-(n-1)(t-1)}}$ (this follows from the fact $n \geq 2$ and $t > n$). From this, we get an $(n, \frac{m}{2}, s'', t)$ scheme where $s'' = ts'2^{t-1} = 8nt\,(m)^{\frac{2^{t-1}}{t2^{t-1}-(n-1)(t-1)}}$. $\square$

3 Result 2: Lower bound for two-probe adaptive schemes

In this section, we show that $s_A(2, m, 2) = \Omega(m^{4/7})$; this improves the $\Omega(\sqrt{m})$ bound which follows from a direct counting argument, but it falls short of the current best upper bound $O(m^{2/3})$, which we conjecture to be tight.

Preliminaries. Fix an adaptive $(2, m, s, 2)$-scheme. Our goal is to show that $s = \Omega(m^{4/7})$. A two-probe adaptive query algorithm is specified using three access functions $a, b, c : [m] \to [s]$, where the query "Is x in S?" is answered by first probing location $a(x)$, and then probing locations $b(x)$ or $c(x)$ depending on the bit the first probe returned. For adaptive schemes we may assume (with at most a constant factor increase in space) that the answer to the query is the last bit read. Such a scheme can be associated with a bipartite (multi-)graph

$$G\ =\ (B := [s], C := [s], E := \{\langle b(x), c(x)\rangle : x \in [m]\}).$$

The edge $\langle b(x), c(x)\rangle$ will be labeled by the element x. Note that E is naturally partitioned into sets $E_1, E_2, \ldots, E_s$, where

$$E_i = \{\langle b(x), c(x)\rangle : x \in [m] \text{ and } a(x) = i\}.$$

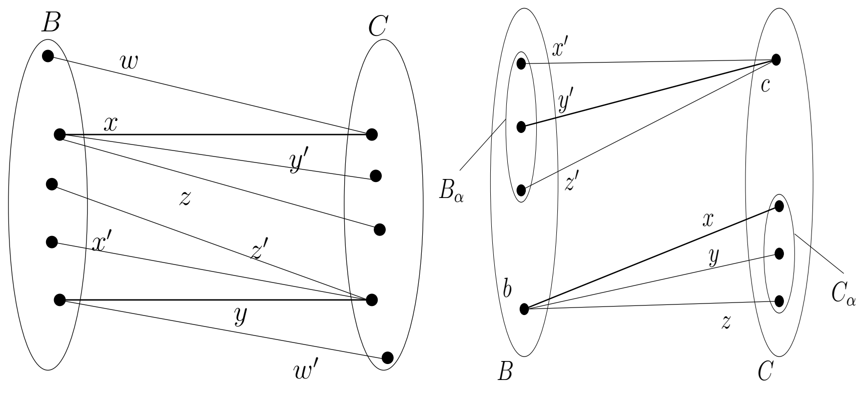

Fig. 1. (a) The pair $\{x, y\}$ cannot be stored (left), (b) The pair $\{x, y'\}$ cannot be stored (right).

Assumption: Each E_i is a matching. (We may restrict attention to a sub-universe of size $\Omega(m)$, so that this assumption always holds. Details omitted.)

We will label edges from the same partition using variants of the same letter. For example, in Figure 1 (a), the eight edges come from four matchings: e.g. the edges labeled x and x' both belong to a common matching, E_α, edges labeled y and y' both belong to a common matching, say E_β, etc.

The obstruction. The configuration of edges and labels depicted in Figure 1 (a) has a special significance for us, for if it were to exist in our graph, then the underlying scheme cannot store all sets of size two. In particular, it is easy to verify that if this configuration exists, then there is no way to store the set $\{x, y\}$.

Goal. Show that if $s < \frac{1}{10}m^{4/7}$, then there exists an obstruction to store some pair $\{x, y\}$.

In the rest of the proof we will address this question directly in graph theoretic terms. We will denote the vertices involved in E_α by $B_\alpha \subseteq B$ and $C_\alpha \subseteq C$. Let $F_\alpha = B_\alpha \times C_\alpha$, that is, F_α is the set of pairs in the graph obtained by replacing E_α by a complete bipartite graph between its end points. For a pair $(b, c) \in B \times C$, let $d(b, c) = |\{\alpha : (b, c) \in F_\alpha\}|$. Clearly,

$$\sum_{b,c} d(b,c) = \sum_{\alpha \in [s]} |E_\alpha|^2 \geq \frac{m^2}{s}, \tag{3}$$

where we used $\sum_{\alpha \in [s]} |E_\alpha| = m$ (and the Cauchy-Schwarz inequality) to justify the last inequality. For $\alpha \in [s]$, let $\Delta_\alpha = \sum_{(b,c) \in F_\alpha} d(b, c)$. Then,

$$\sum_{\alpha \in [s]} \Delta_\alpha = \sum_{(b,c)} d(b,c)^2 \geq \frac{m^4}{s^4},$$

where we used (3) to justify the last inequality. Thus, there is an $\alpha \in [s]$ such that $\Delta_\alpha \geq \frac{m^4}{s^5}$. Fix such an α.

Claim. $\Delta_\alpha - m \leq 2s^2$.

If this claim holds, then we have $\frac{m^4}{s^5} - m \leq 2s^2$, which implies that $s \geq \frac{1}{10} m^{\frac{4}{7}}$. It thus remains to prove the claim.

Proof of claim: We will now interpret Δ_α in another way. Let $B_{\alpha,\beta} = B_\alpha \cap B_\beta$ and $C_{\alpha,\beta} = C_\alpha \cap C_\beta$; let $C'_{\alpha,\beta}$ be the subset of C_β matched to $B_{\alpha,\beta}$ in the matching E_β, and similarly let $B'_{\alpha,\beta}$ be the subset of B_β matched to $C_{\alpha,\beta}$ in the matching E_β. Then,

$$\Delta_\alpha = \sum_{\beta \in [s]} |B_{\alpha,\beta}| \cdot |C_{\alpha,\beta}| = \sum_{\beta \in [s]} |B'_{\alpha,\beta}| \cdot |C'_{\alpha,\beta}|.$$

Let $F'_{\alpha,\beta} = B'_{\alpha,\beta} \times C'_{\alpha,\beta} - E_\beta$. It follows that $|F'_{\alpha,\beta}| \geq |B'_{\alpha,\beta}| \cdot |C'_{\alpha,\beta}| - |E_\beta|$, and $\sum_{\beta \in [s]} |F'_{\alpha,\beta}| \geq \Delta_\alpha - m$. Now, suppose $\Delta_\alpha - m > 2s^2$. Then, by averaging over all pairs $(b, c) \in [s] \times [s]$ we may fix a pair (b, c) that appears in $F'_{\alpha,\beta}$ for three different choices, β_1, β_2 and β_3, of β. (We may assume that $\beta_1, \beta_2 \neq \alpha$.) The resulting situation is described in Figure 1 (b), where we use x, x' to label edges from E_{β_1}, y, y' for edges from E_{β_2} and z, z' for edges from E_{β_3}. It is easy to verify that this results in an obstruction to storing the pair $\{x, y'\}$—a contradiction. □

4 Result 3: Lower bound for three-probe non-adaptive schemes

In this section, we show that any non-adaptive three probe scheme for storing sets of size two from a universe of size m requires $\Omega(\sqrt{m})$ bits of memory. This extends a result of Alon and Feige [1] to small sets. The following two lemmas will be used in the proof of Theorem 4 below.

Lemma 2. *Let G be a bipartite graph with v vertices on both sides. If it does not have a path of length three, then it can have at most $2v - 1$ edges.*

Lemma 3. *(Theorem 8(1) of [2]) $s_N(2, m, 2) = m$.*

We now prove Result 3.

Theorem 4. *For any m, a non-adaptive $(2, m, s, 3)$ scheme uses $s \geq \frac{1}{32}\sqrt{m}$ space.*

Proof: We divide the 16 functions from $\{0,1\}^2$ to $\{0,1\}$ into three classes.

1. OR-type functions: $x \vee y$, $x \vee \bar{y}$, $\bar{x} \vee y$, $\bar{x} \vee \bar{y}$
2. XOR-type functions: $x \oplus y$, $x \oplus \bar{y}$
3. AND-type functions: $x \wedge y$, $x \wedge \bar{y}$, $\bar{x} \wedge y$, $\bar{x} \wedge \bar{y}$, and 0, 1, x, y, $\bar{x}$, $\bar{y}$ (functions that depend on at most one variable can be assumed to have an 1 as the other variable of an AND function)

In a non-adaptive scheme S, each z belonging to the universe is assigned three locations l_1, l_2, l_3 in the storage and a boolean function $f_z : \{0,1\}^3 \to \{0,1\}$ that is used to decode whether z is a member or not. We prove that if the functions are the same for all z in the universe then for this type of $(2, m, s', 3)$ scheme (let us call it *type*-1 scheme), $s' \geq \frac{1}{2}m^{\frac{1}{2}}$. Note that there are 256 different functions mapping $\{0,1\}^3$ to $\{0,1\}$ and therefore there must be at least $\frac{m}{256}$ elements of the universe which use the same function. This proves that s is at least $\frac{1}{2}(\frac{m}{256})^{\frac{1}{2}}$ for the scheme S.

In order to prove $s' \geq \frac{1}{2}m^{\frac{1}{2}}$ for any type-1 scheme S' we prove a lower bound for the the following type of scheme (let us call it *type*-2 scheme). In this scheme all z in the universe use the same function and the 3 probes to determine the membership of every z are made to 3 distinct storages of size s'' each. We show below that $s'' \geq \frac{1}{2}m^{\frac{1}{2}}$. This proves that $s' \geq \frac{1}{2}m^{\frac{1}{2}}$ in any type-1 scheme because otherwise we can copy the storage of $(2, m, s', 3)$ scheme 3 times to obtain a type-2 scheme that uses less than $\frac{1}{2}m^{\frac{1}{2}}$ space in each of its three storages.

We assume to the contrary that there exists a type-2 scheme S'' that uses less than $\frac{1}{2}m^{\frac{1}{2}}$ space in each of its 3 storages. Let $f(l_1, l_2, l_3)$ denote the function for decoding any element of the universe where l_1, l_2, l_3 are the bits stored at the first, second and third storages respectively. In any such scheme, there exists a set U of at least $2m^{\frac{1}{2}}$ elements in the universe that probe the same location in the first probe. The function f is defined as follows. If l_1 is 0, f is a function $f_0(l_2, l_3)$ of the last two bits, and it is $f_1(l_2, l_3)$ otherwise.

We show that in both the following cases, our assumption of S'' using less than $\frac{1}{2}m^{\frac{1}{2}}$ space in each of its 3 storages leads to a contradiction. In each case, we show that there exists a subset of size 2 that cannot be stored using this scheme. We define a bipartite graph G on the last two storages of the scheme as follows: two locations are connected by an edge if and only if they are the last two storage locations of an element of the universe. Note that each element of L has a unique edge in this graph (because the first storage location is common).

Case 1. (one of f_0 or f_1 is either an OR-type function or an XOR-type function) We first assume that the function is f_0 and it is an OR-type function. Let F be a maximal acyclic subgraph of G and U_F be the elements whose edges appear in F. Then, $|U_F| \leq m^{\frac{1}{2}} - 1$. Let $U^* = U \setminus U_F$. Then, $|U^*| \geq m^{\frac{1}{2}} + 1$. Now, observe that to store any pair $\{x, y\} \subset U^*$, the scheme must place an 1 at the location of the first probe, i.e., at l_1. This yields a two-probe scheme for sets of size at most two for the universe U^*. However, by Lemma 3, any such scheme needs space at least $|U^*| \geq m^{\frac{1}{2}} + 1$, whereas G has $m^{\frac{1}{2}}$ vertices. This leads to a contradiction.

Let us now assume that f_0 is an XOR-type function. Let e_{i_1} be an edge of G that is not contained in F. Then, e_{i_1} must form an even length cycle $\{e_{i_1} = (v_1, v_2), e_{i_2} = (v_2, v_3), \ldots, e_{i_{2u}} = (v_{2u}, v_1)\}$ which corresponds to the set of elements $C = \{i_1, i_2, \ldots, i_{2u}\}$ respectively. Let us first consider the case when f_0 is $x \oplus y$. We denote by b_i the bit corresponding to a vertex v_i. In order to store all elements of $C \setminus \{i_1\}$ as non-members, the bits b_i's has to simultaneously satisfy the equations $b_2 + b_3 = 0$ (mod 2), $b_3 + b_4 = 0$ (mod

2), ..., $b_{2u}+b_1 = 0 \pmod 2$. Adding the equations gives us $b_1+b_2 = 0 \pmod 2$ which forces the element i_1 to be stored as a non-member.
Let us now consider the case when f_0 is $x \oplus \bar{y}$. In order to store all elements of C (except i_1) as non-members, the bits b_i's have to simultaneously satisfy the equations $b_2+b_3 = 1 \pmod 2$, $b_3+b_4 = 1 \pmod 2$, ..., $b_{2u}+b_1 = 1 \pmod 2$. Since there are odd number of equations, these adds up to $b_1 + b_2 = 1 \pmod 2$ which forces the element i_1 to be stored as a non-member.
Hence, we observe that to store any pair $\{x, y\} \subset U^*$, the scheme must place an 1 at the location of the first probe, i.e., at l_1. This leads to a contradiction as before.

Case 2. (Both f_0 and f_1 are AND-type functions) From Lemma 2, it follows that G must have a path of length 3. Let us denote it by $\{e_p, e_q, e_r\}$, where p, q, r are the corresponding elements of U. Consider the set $\{p, r\}$. If f_0 is of the form $x \wedge y$, both the endpoints of both e_p and e_r have to be assigned 1. This makes both the endpoints of e_q to be 1, which leads to q being wrongly stored as a member. It is easy to see that this argument works for other AND-type functions and for f_1. $\square$

5 Concluding remarks

We have studied the set membership problem in the bitprobe model with the aim of determining if the exponent of m in these bounds can be made to approach $\frac{1}{t}$.

- Even for small n and t, our lower bounds are not tight. We conjecture that $s_A(2, m, 2) = \Theta(m^{\frac{2}{3}})$.
- We have shown that $\Omega(m^{\frac{1}{3}}) \leq s_A(2, m, 3) \leq O(m^{0.4})$. The upper bound is probabilistic; it can be implemented using limited independence, but it is not fully explicit. We believe that a simple explicit scheme should match our upper bound.
- We conjecture that the information theoretic lower bound is not tight for any $n \geq 2$ and $t \geq 2$; we are able to show this for $n = 2, t = 2$, but believe this to be true in general.

References

1. N. Alon and U. Feige. On the power of two, three and four probes. *Proc. Symposium on Discrete Algorithms 2009*, 346-354.
2. H. Buhrman, P. B. Miltersen, J. Radhakrishnan and S. Venkatesh. Are bitvectors optimal?. *Proc. Symposium on Theory of Computing 2000*, 449-458. Also: *SIAM J. Computing* 31: 1723-1744, 2002.
3. J. Radhakrishnan, V. Raman and S. S. Rao. Explicit Deterministic Constructions for Membership in the Bitprobe Model. *Proc. European Symposium on Algorithms 2001*, 290-299.
4. A. Ta-Shma. Storing Information with Extractors. *Information Processing Letters* 83: 267-274, 2002.