An Improved Two Query Adaptive Bitprobe Scheme Storing Three Elements

A B. Tech Project Report Submitted
in Partial Fulfillment of the Requirements
for the Degree of

Bachelor of Technology

by

Kethavath Naveen
(170101032)

under the guidance of

Deepanjan Kesh

to the

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI
GUWAHATI - 781039, ASSAM

CERTIFICATE

*This is to certify that the work contained in this thesis entitled “**An Improved Two Query Adaptive Bitprobe Scheme Storing Three Elements**” is a bonafide work of **Kethavath Naveen (Roll No. 170101032)**, carried out in the Department of Computer Science and Engineering, Indian Institute of Technology Guwahati under my supervision and that it has not been submitted elsewhere for a degree.*

Supervisor: **Deepanjan Kesh**

Assistant/Associate

Professor,

May, 2020

Department of Computer Science & Engineering,

Guwahati.

Indian Institute of Technology Guwahati, Assam.

Contents

1 Introduction **1**

1.0.1 Explicit Scheme: 1

1.0.2 Non-Explicit Scheme: 1

1.0.3 Adaptive Scheme: 2

1.0.4 Non-Adaptive Scheme: 2

1.1 Problem Statement 2

1.2 Previous Works 2

1.3 Our Contribution 2

1.4 Arrangement Of Elements 4

1.5 An Adaptive Scheme for Subset size Three and two bitprobes 7

1.6 Conclusion 13

1.7 Future Work 14

References **15**

Abstract

In this paper, We are going to Describe a scheme in the bitprobe model of a set membership problem. In Previous scheme, subset size less than or equal to two *i.e* ($n \leq 2$) that can be handled using $\mathcal{O}(m^{2/3})$ amount of space and answers membership questions using minimum number of bitprobes *i.e*, $t = 2$. This scheme is proposed by Radhakrishnan *et al.* [SV01]. Our scheme That we are going to present will describe that, it will stores maximum of three elements *i.e* ($n \leq 3$) using $\mathcal{O}(m^{2/3})$ amount of space and answers membership questions using two bitprobes ($t = 2$). This scheme improves upon the Radhakrishnan scheme *et al.* [SV01] and already existing scheme in the liturature.

Chapter 1

Introduction

Let us consider a universe U of size m. In this model, We are going to learn a problem, that given a set S containing n elements ($n \leq m$) drawn from the universe$U$ of m elements, represent in memory of size s ($i.e$ in data structure) so that the membership questions of is the form "$Is\ x$ in S" (where, x is an element of universe U) can be answered correctly and quickly using a minimum number of bitprobes in our data structure. This scheme can be denoted by (m, n, s, t) where, m is our universe size and n is Subset size S, s is size of the data structure used for membership problem, and t is amount of bitprobes are used to answer membership questions. Bitprobe Model Schemes are classified in to two sub schemes.As follows,

1.0.1 Explicit Scheme:

Size of the Datastructure can be figure out in polynomial time s.And the bits that should be questioned can be figure out in polynomial time in log$m$ and t.

1.0.2 Non-Explicit Scheme:

It does not gives an intuition for how query algo decides the bits to probe. But in this scheme we can show that storage a scheme and query algorithm exists, but it does not gives an intuition of how to build it efficiently.

1.0.3 Adaptive Scheme:

To decide a membership queries, the location of bitprobe might be depend on previous bitprobe.

1.0.4 Non-Adaptive Scheme:

To decide a membership queries, the location of bitprobe is Independent on previous bitprobe.

1.1 Problem Statement

let us assume that the problem of storing three elements (*i.e*, $n = 3$) and answering membership questions using two adaptive bitprobes ($t = 2$). *i.e* In this paper, we build an adaptive scheme for the situation when three elements and two bitprobes (*i.e* $n \leq 3$,t = 2).

1.2 Previous Works

Some previous results on the set membership problem, has mainly concentrated on small size of subsets(*i.e*, n is small). The Space complexity of storing size of subset is one ($n = 1$) the lower bound is $\Omega(m^{2/3})$ and answers membership questions using two adaptive bitprobes ($t = 2$) [HBV17]. And for storing two elements ($n \leq 2$), there is a scheme that was proposed by Radhakrishnan *et al.* [SV01] that will take $\mathcal{O}(m^{2/3})$ amount of memory and answers membership questions using two adaptive btiprobes ($t = 2$).

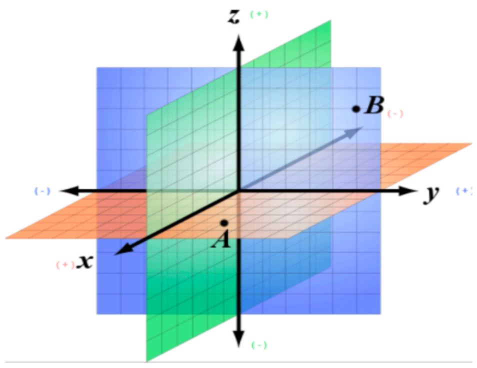
1.3 Our Contribution

In this paper I will be presenting an explicit adaptive scheme for storing subset size three ($n \leq 3$) in $s = \mathcal{O}(m^{2/3})$ bits of space and answers membership questions using two bitprobes (t

$= 2$). This scheme improves upon the Radhakrishnan *et al.* [SV01] scheme (m, 2,$\mathcal{O}(m^{2/3})$, 2). Here,the main idea of this scheme is to set the elements of the universe (U) into integral points of 3D cube, then we see into the projection of the several points onto the faces of the 2D cube.

1.4 Arrangement Of Elements

we take a cube and place it on the first octant of 3D space with coordinates x, y, z. Cube can be place so that any one vertices of the cube lies on the origin and sides of three dimensional cube are parallel to the coordinate axis. Sides of magnitude is $m^{1/3}$. let us define m is the total number of points in the three dimensional cube and on the 3D cube with all integer coordinates. we set all of the m elements of (U) on those m points within the cube and on the cube. Coordinates of the point on which it lies can be refer as element of universe (U). For our illustration, An element of universe (U) lies on the point (p, q, r) we call this as an element (p, q, r).



We define 4 partitions of U based on geometric constructions. And those Partitions are named as X, Y, Z, D. Lets, define partitions one by one.

Partition X : For better understanding let us assume An element (i, j, k) of universe U and set $X(i, j, k)$ of universe U is defined as follows: Draw a line through the point (p, q, r) which is parallel to x-axis and perpendicular to yz-plane. An element of our universe U that falls on this line belongs to the set $X(i, j, k)$. *i.e*,

$$X(p, q, r) = \{ \ (f, g, h) \in U \mid g = q \textbf{ and } h = r \ \}$$

Observation 1: If an element (f, g, h) does not belongs to the set $X(p, q, r)$ then the set $X(p, q, r)$ $X(f, g, h)$ are disjoint.

Proof : According to geometry, the sets X(p, q, r) and X(f, g, h) will tells us that the point (f, g, h) does not lie on the line that belongs to the set X(p, q, r).Then the line belongs to the set X(f, g, h) is parallel to the line belongs to the set X(p, q, r). So, point (f, g, h) does not lie on the line that belongs to the set X(p, q, r) it tells us that either $g \neq q$, or $h \neq r$, or both $i.e$, $g \neq q$, and $h \neq r$. For an arbitrary point (i, j, k) that belongs to the set X(f, g, h), WLOG, let us consider that two points of the z-coordinates are not equal ($i.e$, $h \neq k$). it tells us that $k = h \neq r$.so, we can easily conclude that the point (i, j, k) not belongs to the set X(p, q, r). This is enough to prove that the sets X(p, q, r) and X(f, g, h) are disjoint.

Observation 2: If an element (f, g, h) does belongs to the set $X(p, q, r)$ then the set $X(p, q, r)$ and $X(f, g, h)$ are equal.

Proof : According to geometry, the sets X(p, q, r) and X(f, g, h) these two sets should be equal.It has only one line that passes through both the points and parallel to the x-axis and normal to yz-plane.

As, point (f, g, h) does lie on the line that belongs to the set X(p, q, r) it tells us that $g = q$ and $h = r$. So, here the point $(f, g, h) = (f, q, r)$. it means that the point (f, g, h) is actually the point (f, q, r). Without loss of generality, let us consider point $i, j, k)$ that belongs to the set X(f, g, h). it is clear that, we have $j = q$ and $k = r$. So, the point (i, j, k) becomes (i, q, r). Hence, it is clearly says that the point (i, q, r) does belongs to the set X(p, q, r). Therefore it concludes that the set X(f, g, h) is a subset of X(p, q, r).

We, have proved that the point (f, g, h) belongs to the set X(p, q, r). but we have not proved reverse. if we show that the point (p, q, r) belongs to the set X(f, g, h) then we can easily prove that Sets X(f, g, h) X(p, q, r) are equal.

As given in the problem, the point (f, g, h) is a subset of the set X(p, q, r), it is clear that, we have $g = q$ and $h = r$. So, the point (p, q, r) becomes (p, g, h) $i.e$, $(p, q, r) = (p, g, h)$ that belongs to the set X(f, g, h) from the definition of the set X(f, g, h). Therefore, we have proved the set X(p, q, r) is a subset of the set X(f, g, h).

Therefore, from the above two results *i.e*, the set X(f,g,h) is a subset of X(p,q,r) and the set X(p,q,r) is a subset of the set X(f,g,h) it says that both the sets X(f,g,h) and X(p,q,r) are equal.

So, we define partition X as,

$$X = \{\ Y(0,p,q) \mid \mathbf{0} \leq p,\ q < m^{1/3}\ \}$$

Lets define size of the partitions As follows,

Lemma1: *The size of partition X is $m^{2/3}$.*

Proof : This is a result of partitions X.

Let start defining partition$Y$ and partition Z. these definitions are no too dissimilar to the definition of partition X.

$$Y(p,q,r) = \{\ (f,g,h) \in U \mid f = p \quad h = r\ \}$$

Similarly,

$$Z(p,q,r) = \{\ (f,g,h) \in U \mid f = p \quad g = q\ \}$$

set Y$(p.q.r)$ **::** A line passes through the point (p,q,r) and the line is parallel to the $y - axis$ and normal to the $xz - plane$.

$$Y = \{\ \mathbf{Y}(p,0,q) \mid \mathbf{0} \leq p,\ q < m^{1/3}\ \}$$

set Z(p,q,r) **::** A line passes through the point (p,q,r) and the line is parallel to the $z - axis$ and normal to the $xy - plane$. *i.e*,

$$Z = \{\ Z(p,q,0) \mid \mathbf{0} \leq p,\ q < m^{1/3}\ \}$$

Lemma2: The size of partition Y and Z are $m^{2/3}$ and $m^{2/3}$ respectively.

Proof : proof is not too dissimilar to the we argued for size of Partition X.

Partition D : Let start defining partition D, The set $D(p,q,r)$ includes all the points who lies on that line passes through the point (p,q,r) and it has 45° slope on that plane. The set $D(p,q,r)$ is defined as,

$$D(p,q,r) = \{\ (f,g,h) \in U \mid h = r \textbf{ and } f - p = g - q\ \}$$

Partition D cab be defined as,

$$D = \{\ D(p,0,q) \mid \mathbf{0} \leq p,\ q < m^{1/3}\ \}\ \cup \{\ D(0,p,q) \mid \mathbf{0} \leq p,\ q < m^{1/3}\ \}$$

1.5 An Adaptive Scheme for Subset size Three and two bitprobes

Here, I will be providing an explicit adaptive bitprobe model for subset size three and two bitprobes.

DataStructure: This datastructure is made up of three tables, out of three tables, one table for partitions X, Y, and Z. to avoid too many notations Here, we will be using these partition notations ($i.e$, X, Y, and Z) for denoting the tables in this datastructure. We reserve one bit for every set in the different partitions in the correlated table in the datastructure. We again curse the notation and use it for a set to refer to its correspanding bit in our datastructure.

Lemma 3. Size of Datstructure is $3 \times m^{1/3}$.

Proof : This Lemma can simply proved using Lemma[1] and Lemma[2].

Query Scheme : In this model, the query scheme is represented by a binary tree, we call it as "*decisiontree*". This decision tree will tells us that the result of previous bitprobe known and the current location of a bitprobe is known. So, As we know, for adaptive scheme result of previous bitprobe should be known. You can refer figure 1 for decision tree.

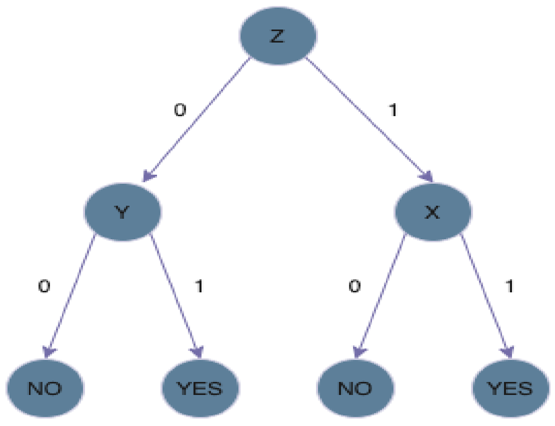


Figure 1: A Decision Tree For Adaptive BitProbe Model

Let us consider an example for how this query scheme works. Let consider an element (I, J, K) of our universe U, here we need to determine whether the subset S contains an element (I, J, K) or not. By definition of partitions and sets, we easily know that the set X$(0, J, K)$ contains an element in the table X. Similarly, the set $Z(I, J, 0)$ in table Z and the set Y$(I, J, 0)$ in table Y. So, initial query will be build in table Z according to decision tree as shown in figure[1].So, the location of bitprobe is at Z$(I, J, 0)$.if the bit Z$(I, J, 0)$ gives '1' then we go to right child (table$X$) and query the location at X$(0, J, K)$ in table X.On other hand, if the bit Z$(I, J, 0)$ gives '0' then we follow left child (table Y) and query the location Y$(I, 0, K)$ in table Y.We prove that the subset S contains an element (I, J, K) iff the second query returns bit '1'. Otherwise an element does not belongs to the subset S ($i.e$, second query returns '0').

Storage Scheme : Storage Scheme tells us that how to we can set the bits of our datastructure so that query scheme can work efficiently and gives answer correctly. We describe Storage scheme by considering an example for better understanding. From universe U consider an arbitrary subset $S = (P1, Q1, R1), (P2, Q2, R2), (P3, Q3, R3)$. Here, bits depends on how the Subset S members are chosen from the Universe U.Below we describe each and every case along with $proof of correctness$.

$Case1$: Consider the situation when All of three points of x-coordinate are equal. $i.e$, $P1 = P2 = P3 = P$ (Case1 Assumption). So, In this situation, Subset S can be looks like $(P, Q1, R1), (P, Q2, R2), (P, Q3, R3)$. As we know initial query will be build in table Z. So, we set the bits in table Z. As follows Z$(P, Q1, 0)$ = 1,Z$(P, Q2, 0)$ =1, Z$(P, Q3, 0)$ = 1 And rest bits are set to '0'. Next we set the bits in table X. As X$(0, Q1, R1)$ =1, X$(0, Q2, R2)$ =1, X$(0, Q3, R3)$ = 1, and the rest bits in table X are set to '0'. at last, we move to table Y, and we set all the bits to '0' ($i.e$, $Y(P', Q', R') = 0$.

Proof of Correctness : If any member of Subset S queried, then the work of the bits in the table Z tells us that it will get 1.So, '1' means it will query next in right child $i.e$, table X . In the table X the bits corresponding to three members of Subset S are equals to '1'. Let us assume that an Element of Our Universe U $i.e$, (I, J, K). This query got "YES" in our Datastructure.So, it means that it should get '1' from table X because in table Y all the bits are set to '0'. And to query in table X it should get '1' from table Z, this is how simple it is. let Assume that it get the bit Z$(P, Q3, 0)$ is '1'.So, it should be the case that $I = PJ = Q3$ and K is still unknown. So, now the element becomes $(I, J, K) = (P, Q3, K)$. Now, we query in table X As we know previous bitprobe location in table Z is 1. it will query in table X only for such bits whose y-coordinate is $Q3$ and that bit has been set to 1. So, Such bit is X$(0, Q3, R3)$. Therefore $K = R3$. Finally element is (I, J, K) = $(P, Q3, R3)$ which is a member of Subset S. Similar Argument for other cases as well.

$Case2$: Consider the situation when two points of x-coordinate is same and the third $x-$coordinate is distinct from first two (remaining two points of $x-$ coordinates). WLOG, $P1 = P2 = P$ and $P1 = P2 = P$!= $P3$. So, now subset S looks like $(P, Q1, R1), (P, Q2, R2), (P3, Q3, R3)$. Here it might arise two sub cases, $i.e$, a.Third element of $Y-$Coordinate may equals to the first two elements of $Y-$coordinate. b. third element of the $Y-$coordinate is different from first two elements of the $Y-$coordinate.

$SubCase2a$: Third element of the $Y-$coordinate is equals to one of the first two elements of Y coordinates.WLOG, consider $Q1 = Q3 = Q$ (Assume).Now Subset looks like $S = (P, Q, R1), (P, Q2, R2), (P3, Q, R3)$ here again might arise the complication when $Q = Q2$. we consider this as a another sub case we will be discuss below.

$SubCase2a.1$: Let us assume the situation when $Q! = Q2$. then subset S Becomes $(P, Q, R1), (P, Q2, R2), (P3, Q, R3)$. We set the bits of $Z(P, Q', 0) =' 1'$,for all $Q'! = Q$, and $Z(P3, Q, 0) =' 1'$ in table Z. And the remaining bits in the table Z are set to '0'. We set the bits of Y$(P, 0, R1)$ = '1', And the remaining bits in the table Y are set to '0'.Lastly, we set X$(0, Q2, R2)$ = '1', X$(0, Q, R3)$ = '1', and And the remaining bits in the table X are set to '0'.

Proof of Correctness : . For giving correctness for above case let take randomly a member from subset S, $i.e$, $(P3, Q, R3)$. we will query first in table$Z$ and we will get '1' from table$Z$ (Since Z$(P3, Q, 0)$ has been set to '1'), because of previous bit '1' we will next query in (Right child) table$X$. and we will get '1' from table$X$ (Since Z$(0, Q, R3)$ has been set to '1') And Returns YES. Therefore this is enough to prove that an element $(P3, Q, R3)$ is a member of Subset S. Similarly, take another member from subset S, $(P, Q2, R2)$. we will query first in table$Z$ and we will get '1' from table$Z$ (Since Z$(P, Q', 0)$ has been set to '1' for all $Q'! = Q$), because of previous bit '1' we will next query in (Right child) table$X$. and we will get '1' from table$X$ (Since Z$(0, Q2, R2)$ has been set to '1') And Return YES. Therefore this is enough to prove that an element $(P, Q2, R2)$ is a member of Subset S. Similarly, the query Scheme for all members of Subset$S$ will gives answer "YES" $i.e$, correctly and efficiently.

Let Assume an element (I, J, K) from Universe U got YES. Here table Y and table X both have more than one bits are set to '1'. It is difficulty to say that an element (I, J, K)

got YES from either table X or table Y.but it could have got YES from either of them. Lets Assume that an element (I, J, K) got YES from table Y. In table Y only one bit of Y$(P, 0, R1)$ set to '1'.So, our element becomes $(I, J, K) = (P, J, R1)$. To query in table$Y$ it should get '0' bit from table$Z$. And table$Z$ has only one bit $i.e$, Z$(P, Q, 0)$ is set to '0'. and it must query this bit ($i.e$,) Z$(P, Q, 0)$. Now an element becomes $(I, J, K) = (P, Q, R1)$, and it indeed a member of Subset S. lets Assume another case that an element (I, J, K) got YES from table X. In table X two bits are set to '1' ($i.e$,) X$(0, Q2, R2)$ and X$(0, Q, R3)$ both are set to '1'.So, an element $(I, J, K$ have two options, So, an element could be either of $(I, Q2, R2)$ or $(I, , Q, R3)$. To query in table$X$ it should get '1' bit from table$Z$.here, an element $(I, Q2, R2)$ will get '1' by querying the bit Z$(P, Q', 0)$ for all $Q'! = Q$ and this bit is set to '1'.So, we have $I = P$ and our element looks like $(P, Q2, R2)$. Similarly, an element $(I, Q, R3)$ will get '1' by querying the bit Z$(P3, Q, 0)$ and this bit is set to '1'.So, we have $I = P3$ and our element looks like $(P3, Q, R3)$. Therefore,Now an element becomes $(I, J, K) = (P, Q2, R2)$ or $(I, J, K) = (P3, Q, R3)$. So,here all the cases, An element (I, J, K) will result to be a member of Subset S.

$SubCase2a.2$: Let us Assume the case when B = B2. Now, the elements of subset S Becomes $(P, Q, R1), (P, Q, R2), (P3, Q, R3)$. This case is nearly same as case2.a.1 ($i.e$, When Q not equal to Q2).We set the bits of Z$(P, Q', 0) =$ '1',for all $Q'! = Q$, and Z$(P3, Q, 0)$ = '1' in table Z. And the remaining bits in the table Z are set to '0'. We set the bits of Y$(P, 0, R1)$ = '1', And Y$(P, 0, R2)$ = '1'.And the remaining bits in the table Y are set to '0'.Lastly, we set $X(0, Q, R3) =' 1'$, and And the remaining bits in the table X are set to '0'. For this case to prove the correctness you can refer above case as it is nearly same as case2.a.1 ($i.e$, case when Q is not equal to Q2).

$SubCase2b$: Let us Assume the case when a third element of the Y coordinate if distinct from other two elements of the Y coordinate.So, now our Subset S looks like

$(P, Q1, R1), (P, Q2, R2), (P3, Q3, R3)$.In this case first in the table Z, we set the bits of Z$(P, Q1, 0)$ = '1',Z$(P, Q2, 0)$ = '1',Z$(P3, Q3, 0)$ = '1', and the remaining bits of table Z are set to '0'.Next, In table X, we set the bits of X$(0, Q1, R1)$ = '1',X$(0, Q2, R2)$ = '1',X$(0, Q3, R3)$ = '1', and the remaining bits of table X are set to '0'.Lastly, In table Y, we set all bits of table Y are set to '0'.

Proof of Correctness : Let us assume that an Element of Our Universe U $i.e$, (I, J, K). And it got "YES" in our Datastructure.It means that it should get '1' from table X as in table Y all the bits are set to '0'. And to query in table X it should get '1' from table Z. Let Assume the case that an element (I, J, K) got '1' by querying the bit X$(0, Q2, R2)$ in table X. So, we have $J = Q2$ and $K = R2$.Now an element becomes $(I, Q2, R2)$.Next, the element $(I, Q2, R2)$ got '1' by querying the bit $(P, Q2, 0)$ in table Z. It implies that $I = P$, Now our element becomes $(I, J, K) = (P, Q2, R2)$. Therefore an element $(P, Q2, R2)$ is a member of subset S. Similarly, We can prove that if (I, J, k) got '1' by querying any other bits in table X, It will be a member of Subset S.

$Case3$: Final Case, let us Assume the case when All the three members (of subset S) of $x-$coordinates are distinct $.i.e$, $P1$ not equals to $P2$ not equals to $P3$. Now, our Subset S look likes $(P1, Q1, R1), (P2, Q2, R2), (P3, Q3, R3)$. In this case, initially, we set the bits of Z$(P1, Q1, 0)$ = '0',Z$(P2, Q2, 0)$ = '0',Z$(P3, Q3, 0)$ = '0', and the remaining bits of table Z are set to '1'.Next, In table X, we set all the bits of table X to '0'.Finally, In table Y, we set the bits of Y$(P1, 0, R1)$ = '1',Y$(P2, 0, R2)$ = '1',Z$(P3, 0, R3)$ = '1', and the remaining bits of table$Y$ are set to '0'. Here, initial query will be build in table Z and As in table Z all the three bits are set to 0, next, we will build query in table Y. So, In table Y All the bits of three members of Subset S are set to '1'. Therefore it will give 'YES' if an element is a member of Subset$S$, otherwise it gives 'NO'.

Proof of Correctness : Let us Assume an element (I, J, k) got 'YES' by query in our datastructure. It must get YES from table$Y$ as all the elements in table X are set to '0'. WLOG, Let us consider an element (I, J, K) got '1' by querying the bit Y$(P3, 0, R3)$. It implies that $I = P3andK = R3$ and the element becomes $(I, J, K) = (P3, J, R3)$. To query in table$Y$ it should get '0' from table Z. So, next element$(P3, J, R3)$ will query in table$Z$. And there is only one bit whose $x-$coordinate is equals to $P3$.So, an element $(P3, J, R3)$ got '0' by querying the bit Z$(P3, Q3, 0)$.Which implies that $J = Q3$. So, finally element becomes $(P3, J, R3) = (P3, Q3, R3)$. Therefore an element $(P3, Q3, R3)$ is a member of Subset S.

Idea of Storing At Most Four Elements

I have Conclude that it is not possible to store four elements of our universe U in space $s = \mathcal{O}(m^{2/3})$. So, Our idea is If more than one element will share the same location then we can achieve the space bound o(m). Here an idea of subset size two comes in which was proposed by Radhakrishnan [SV01]. *i.e*, using blocks and super blocks.

- Given an universe of m elements. We divide the universe into blocks of size $(m^{1/6})$. So, we have total $(m^{5/6})$ number of blocks. And numbered as 1,2,3,4,5,....$(m^{1/6})$. We consider these as $Index$ of element.

- Again we divide the blocks in our universe$U$ into superblocks of size $(m^{4/6})$. So, we have total $(m^{1/6})$ number of superblocks. And numbred as 1,2,3,4,5,....$(m^{1/6})$.

1.6 Conclusion

- In this paper, I have provided an Improved scheme in bitprobe model for set membership problem of storing Subset size $(n \leq 3)$ and using two bitprobes $(t = 2)$. The main idea is geometrical arrangement of elements of universe U on the 3D cube. And the scheme (m, 3, $\mathcal{O}(3 \times m^{2/3})$, 2) also gives us non-adaptive scheme. Irrespective of initial query made in table z. If we probe all three tables of our datastructure. Then it becomes a three bitprobe scheme i, e $t = 3$.Here every bitprobe is a independent of previous bitprobe.So, our query scheme , decide membership queries on getting result of three queries.

- I conclude that Subset size four($n \leq 4$) is cannot be stored using space $= \mathcal{O}(3 \times m^{2/3})$, and two bitprobes ($t = 2$).

1.7 Future Work

- We will Determine the scheme that can store at most four elements using $\mathcal{O}(m^{5/6})$ amount of space and answers using 2 bitprobes.
- We will try to Determine the scheme that can store more elements using lower bound space and using minimum number of bitprobes.

References

[D.K17] D.Kesh. On adaptive bitprobe schemes for storing two elements. *in - Combinatorial Optimization and Applications - 11th International Conference*, page 471–479., 2017.

[HBV17] J. Radhakrishnan H. Buhrman, P. B. Miltersen and S. Venkatesh. Siam. *on Computing*, page 471–479., 2017.

[JR15] M.Garg J. Radhakrishnan. Set membership with a few bit probes. *in - Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, page 776–784, 2015.

[R.B09] R.Blue. The bit probe model for membership queries. *in - Non-adaptive bit queries, Master's thesis, Graduate School of the University of Maryland,*, pages 346–354, 2009.

[SV01] J.Radhakrishnan S.S.Rao and V.Raman. Explicit deterministic constructions for membership in the bitprobe model. *in - Algorithms - ESA 2001*, pages 290–299, 2001.

[U.F09] N.Alon U.Feige. On the power of two three and four probes. *in - Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 346–354, 2009.

[SV01]. [U.F09]. [R.B09]. [JR15]. [D.K17]. [HBV17].