Improved Bounds for Two Query Adaptive Bitprobe Schemes Storing Five Elements

Mirza Galib Anwarul Husain Baig and Deepanjan Kesh

Indian Institute of Technology Guwahati, Guwahati, Assam 781039, India
{mirza.baig,deepkesh}@iitg.ac.in

Abstract. In this paper, we study two-bitprobe adaptive schemes storing five elements. For these class of schemes, the best known lower bound is $\Omega(m^{1/2})$ due to Alon and Feige [1]. Recently, it was proved by Kesh [8] that two-bitprobe adaptive schemes storing three elements will take at least $\Omega(m^{2/3})$ space, which also puts a lower bound on schemes storing five elements. In this work, we have improved the lower bound to $\Omega(m^{3/4})$. We also present a scheme for the same that takes $\mathcal{O}(m^{5/6})$ space. This improves upon the $\mathcal{O}(m^{18/19})$-scheme due to Garg [6] and the $\mathcal{O}(m^{10/11})$-scheme due to Baig *et al.* [4].

Keywords: Data structure · Set membership problem · Bitprobe model · Adaptive Scheme.

1 Introduction

The *static membership problem* involves the study and construction of such data structures which can store an arbitrary subset $\mathcal{S}$ of size at most n from the universe $\mathcal{U} = \{1, 2, 3, \ldots, m\}$ such that membership queries of the form "Is x in $\mathcal{S}$?" can be answered correctly and efficiently. A special category of the static membership problem is the *bitprobe model* in which we evaluate our solutions w.r.t. the following resources – the size of the data structure, s, required to store the subset $\mathcal{S}$, and the number of bits, t, of the data structure read to answer membership queries. It is the second of these resources that lends the name to this model.

In this model, the design of data structures and query algorithms are known as *schemes*. For a given universe $\mathcal{U}$ and a subset $\mathcal{S}$, the algorithm to set the bits of our data structure to store the subset is called the *storage scheme*, whereas the algorithm to answer membership queries is called the *query scheme*. Schemes are divided into two categories depending on the nature of our query scheme. Upon a membership query for an element, if the decision to probe a particular bit depends upon the answers received in the previous bitprobes of this query, then such schemes are known as *adaptive schemes*. If the locations of the bitprobes are fixed for a given element of $\mathcal{U}$, then such schemes are called *non-adaptive schemes*.

For any particular setting of n, m, s, and t, the corresponding scheme is referred to in the literature as a (n, m, s, t)-scheme [5,10,11]. Radhakrishnan *et*

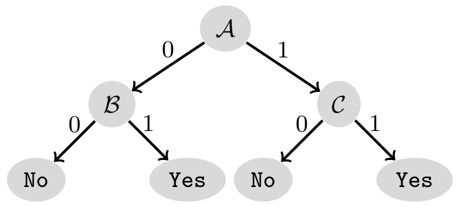

Fig. 1: The decision tree of an element.

al. [11] also introduced the convenient notations $s_A(n, m, t)$ and $s_N(n, m, t)$ to denote the space required by an adaptive or a non-adaptive scheme, respectively.

1.1 Two-bitprobe Adaptive Schemes

In this paper, we consider those schemes that use two bitprobes ($t = 2$) to answer membership queries. Data structures in such schemes consist of three tables, namely $\mathcal{A}, \mathcal{B}$, and $\mathcal{C}$. The first bitprobe is always made in table $\mathcal{A}$; the location of the bit being probed, of course, depends on the element which is being queried. The second bitprobe is made in table $\mathcal{B}$ or in table $\mathcal{C}$, depending on whether 0 was returned in the first bitprobe or 1 was returned. The final answer of the query scheme is `Yes` if 1 is returned by the second bitprobe, otherwise it is `No`. The data structure and the query scheme can be succintly denoted diagrammatically by what is known as the *decision tree* of the scheme (Figure 1).

1.2 Our Contribution

In this paper, we study schemes when the number of allowed bitprobes is two ($t = 2$) and the subset size is at most five ($n = 5$). Some progress has been made for subsets of smaller sizes.

When the subset size is odd, the problem is well understood. More particularly, when $n = 1$, there exists a scheme that takes $\mathcal{O}(m^{1/2})$ amount of space, and it has been shown that it matches with the lower bound $\Omega(m^{1/2})$ [1,5,9]. When $n = 3$, Baig and Kesh [2] have shown that there exists a $\mathcal{O}(m^{2/3})$-scheme, and Kesh [8] has proven that it matches with the lower bound $\Omega(m^{2/3})$.

For even sized subsets, tight bounds are yet to be proven. For $n = 2$, Radhakrishnan *et al.* [10] have proposed a scheme that takes $\mathcal{O}(m^{2/3})$ space, and for $n = 4$, Baig *et al.* [3] have presented a $\mathcal{O}(m^{5/6})$-scheme, but it is as of yet unknown whether these bounds are tight.

For subsets of size five ($n = 5$), the best known lower bound was due to Alon and Feige [1] which is $\Omega(m^{1/2})$. The $\Omega(m^{2/3})$ lower bound for $n = 3$ also puts an improved bound for the $n = 5$ case. Our first result improves the bound to $\Omega(m^{3/4})$.

Result 1 (Theorem 8). $s_A(5, m, 2) = \Omega(m^{3/4})$.

We also propose an improved scheme for the problem. The best known upper bound was due to Garg [6] which was $\mathcal{O}(m^{18/19})$, which was improved by Baig *et al.* [4] to $\mathcal{O}(m^{10/11})$. In this paper, we improve the bound to $\mathcal{O}(m^{5/6})$.

Result 2 (Theorem 9). $s_A(5, m, 2) = \mathcal{O}(m^{5/6})$.

One thing to note is that the space for the scheme storing five elements now matches the space for the scheme storing four elements. Moreover, the two results stated above combined together significantly reduces the gap between the upper and lower bounds for the problem under consideration.

2 Lower Bound

In this section, we present our proof for the lower bound of $s_A(5, m, 2)$. So, the size of our subset $\mathcal{S}$ that we want to store in our data structure is at most five.

In table $\mathcal{A}$ of our data structure, multiple elements must necessarily map to the same bit to keep the table size to $o(m)$. The set of elements that map to the same bit in this table is referred to in the literature as a *block* (Radhakrishnan *et al.* [10]). We refer by $\mathcal{A}(e)$ to the block to which the element e belongs. Elements mapping to the same bit in tables $\mathcal{B}$ and $\mathcal{C}$ will be referred to as just *sets*. That set of table $\mathcal{B}$ to which the element e belongs will be denoted by $\mathcal{B}(e)$. $\mathcal{C}(e)$ is similarly defined.

Storing a member e of our subset $\mathcal{S}$ in table $\mathcal{B}$ is an informal way to state the following – the bit corresponding to $\mathcal{A}(e)$ is set to 0, and $\mathcal{B}(e)$ is set to 1. So, upon query for the element e, we will get a 0 in our first bitprobe, query table $\mathcal{B}$ at location $\mathcal{B}(e)$ to get 1, and finally answer `Yes`. Similarly, storing an elememt f which is not in $\mathcal{S}$ in table $\mathcal{C}$ would entail assigning 1 to $\mathcal{A}(f)$ and 0 to $\mathcal{C}(f)$.

To start with, we make the following simplifying assumptions about any scheme for the aforementioned problem.

1. All the tables of our datastructure have the same size, namely s, and hence the size our data structure is $3 \times s$.
2. If two elements belong to the same block in table $\mathcal{A}$, they do not belong to the same sets in either of tables $\mathcal{B}$ or $\mathcal{C}$.

In the conclusion of this section, we will show that these assumptions do not affect the space asymptotically, but rather by constant factors.

2.1 Universe of an Element

We now define the notion of the *universe* of an element. This is similar to the definition of the universe of a set in Kesh [8].

Definition 1. *The universe of an element e w.r.t. to table $\mathcal{B}$, denoted by $\mathcal{U}_{\mathcal{B}}(e)$, is defined as follows.*

$$\mathcal{U}_{\mathcal{B}}(e) = \bigcup_{f \in \mathcal{B}(e) \setminus \{e\}} \mathcal{A}(f) \setminus \{f\}.$$

Similarly, the universe of an element e w.r.t. to table $\mathcal{C}$, denoted by $\mathcal{U}_{\mathcal{C}}(e)$, is defined as follows.

$$\mathcal{U}_{\mathcal{C}}(e) = \bigcup_{f \in \mathcal{C}(e) \setminus \{e\}} \mathcal{A}(f) \setminus \{f\}.$$

A simple property of the universe of an element, which will be useful later, is the following.

Observation 1. *1. $\mathcal{A}(e) \cap \mathcal{U}_{\mathcal{B}}(e) = \phi$ and $\mathcal{B}(e) \cap \mathcal{U}_{\mathcal{B}}(e) = \phi$.*
2. $\mathcal{A}(e) \cap \mathcal{U}_{\mathcal{C}}(e) = \phi$ and $\mathcal{C}(e) \cap \mathcal{U}_{\mathcal{C}}(e) = \phi$.

Proof. Due to Assumption 2, e is the only element of the block $\mathcal{A}(e)$ in set $\mathcal{B}(e)$. So, it follows from the definition that no element of $\mathcal{A}(e)$ is part of $\mathcal{U}_{\mathcal{B}}(e)$. No element of $\mathcal{B}(e)$ is part of $\mathcal{U}_{\mathcal{B}}(e)$ as we specifically preclude those elements from $\mathcal{U}_{\mathcal{B}}(e)$. The other scenarios can be similarly argued. □

We make the following simple observations about the universe of an element to help illustrate the constraints any storage scheme must satisfy to correctly store a subset $\mathcal{S}$.

Observation 2. *If $\mathcal{B}(e) \cap \mathcal{S} = \{e\}$, and we want to store e in table $\mathcal{B}$, then all the elements of $\mathcal{U}_{\mathcal{B}}(e)$ must be stored in table $\mathcal{C}$.*

Proof. As e has been stored in table $\mathcal{B}$, the bit corresponding to the set $\mathcal{B}(e)$ must be set to 1. Consider an element f, different from e, in $\mathcal{B}(e)$. According to Assumption 2, e and f cannot belong to the same block. As $f \notin \mathcal{S}$, so f cannot be stored in table $\mathcal{B}$; if we do so, the query for f will look at the bit $\mathcal{B}(e)$ and incorrectly return 1. So, the element f and its block $\mathcal{A}(f)$ must be stored in table $\mathcal{C}$.

The above argument applies to any arbitrary element of $\mathcal{B}(e) \setminus \{e\}$. So, according to Definition 1, all of the elements of $\mathcal{U}_{\mathcal{B}}(e)$ must be stored in table $\mathcal{C}$. □

We make the same observation, without proof, in the context of table $\mathcal{C}$.

Observation 3. *If $\mathcal{C}(e) \cap \mathcal{S} = \{e\}$, and we want to store e in table $\mathcal{C}$, then all the elements of $\mathcal{U}_{\mathcal{C}}(e)$ must be stored in table $\mathcal{B}$.*

Next, we define, what could be referred to as, a higher-order universe of an element, built on top of the universe of the element.

Definition 2. *The 2-universe of an element e w.r.t. table $\mathcal{B}$, denoted by $\mathcal{U}^2_{\mathcal{B}}(e)$, is defined as follows.*

$$\mathcal{U}^2_{\mathcal{B}}(e) = \bigcup_{f \in \mathcal{U}_{\mathcal{B}}(e)} \mathcal{C}(f) \setminus \{f\}.$$

Similarly, the 2-universe of an element e w.r.t. table $\mathcal{C}$, denoted by $\mathcal{U}^2_{\mathcal{C}}(e)$, is defined as follows.

$$\mathcal{U}^2_{\mathcal{C}}(e) = \bigcup_{f \in \mathcal{U}_{\mathcal{C}}(e)} \mathcal{B}(f) \setminus \{f\}.$$

The following observations provide more constraints for our storage schemes.

Observation 4. *Consider an element e such that $\mathcal{B}(e) \cap \mathcal{S} = \{e\}$, and suppose we want to store e in table $\mathcal{B}$. If f is a member of $\mathcal{U}_{\mathcal{B}}(e)$ such that $\mathcal{C}(f) \cap \mathcal{S} = \{f\}$, then all the other members of $\mathcal{C}(f)$ must be stored in table $\mathcal{B}$.*

Proof. If e, a member of $\mathcal{S}$, is stored in table $\mathcal{B}$, then Observation 2 tells us that all members of $\mathcal{U}_{\mathcal{B}}(e)$ must be stored in table $\mathcal{C}$. As $f \in \mathcal{U}_{\mathcal{B}}(e) \cap \mathcal{S}$, so f must be stored in table $\mathcal{C}$, and consequently, the bit corresponding to $\mathcal{C}(f)$ must be set to 1. As the other members of $\mathcal{C}(f)$ do not belong to $\mathcal{S}$, they cannot be stored in table $\mathcal{C}$, and hence must be stored in table $\mathcal{B}$. $\square$

Observation 5. *Consider an element e such that $\mathcal{B}(e) \cap \mathcal{S} = \{e\}$, and suppose we want to store e in table $\mathcal{B}$. If f is a member $\mathcal{U}_{\mathcal{B}}(e)$ such that $\mathcal{C}(f) \cap \mathcal{S} = \{x\}$, where $x \neq f$, then x must be stored in table $\mathcal{B}$.*

Proof. As e, a member of $\mathcal{S}$, is stored in $\mathcal{B}$, Observation 2 tells us that all the members of $\mathcal{U}_{\mathcal{B}}(e)$, and f in particular, must be stored in table $\mathcal{C}$. As $f \notin \mathcal{S}$, so the bit corresponding to $\mathcal{C}(f)$ has to be 0. As $x \in \mathcal{C}(f)$ belongs to $\mathcal{S}$, then storing x in table $\mathcal{C}$ would imply that $\mathcal{C}(f)$ must be set to 1, which is absurd. So, x must be stored in table $\mathcal{B}$. $\square$

We next state the same observations in the context of table $\mathcal{C}$.

Observation 6. *Consider an element e such that $\mathcal{C}(e) \cap \mathcal{S} = \{e\}$, and suppose we want to store e in table $\mathcal{C}$. If f is a member of $\mathcal{U}_{\mathcal{C}}(e)$ such that $\mathcal{B}(f) \cap \mathcal{S} = \{f\}$, then all the other members of $\mathcal{B}(f)$ must be stored in table $\mathcal{C}$.*

Observation 7. *Consider an element e such that $\mathcal{C}(e) \cap \mathcal{S} = \{e\}$, and suppose we want to store e in table $\mathcal{C}$. If f is a member $\mathcal{U}_{\mathcal{C}}(e)$ such that $\mathcal{B}(f) \cap \mathcal{S} = \{x\}$, where $x \neq f$, then x must be stored in table $\mathcal{C}$.*

2.2 Bad Elements

We now define the notion of *good* and *bad* elements. These notions are motivated by the notions of *large* and *bounded sets* from Kesh [8].

Definition 3. *e is a bad element w.r.t. table $\mathcal{B}$ if one of the following holds.*

1. *Two elements of $\mathcal{U}_{\mathcal{B}}(e)$ share a set in table $\mathcal{C}$.*
2. *The size of $\mathcal{U}^2_{\mathcal{B}}(e)$ is greater than $2s$, i.e. $|\mathcal{U}^2_{\mathcal{B}}(e)| > 2 \cdot s$.*

Otherwise, it is said to be good. Bad and good elements w.r.t. to table $\mathcal{C}$ are similarly defined.

The next claims state the consequences of an element being bad due to any of the above properties getting satisfied.

Claim. If two elements of $\mathcal{U}_{\mathcal{B}}(e)$ share a set in table $\mathcal{C}$, then $\exists$ a subset $\mathcal{S}$ that contains e and has size two such that to store $\mathcal{S}$, e cannot be stored in $\mathcal{B}$.

Proof. Suppose the elements $x, y \in \mathcal{U}_{\mathcal{B}}(e)$ share the set Y in table $\mathcal{C}$. We would prove that to store the subset $\mathcal{S} = \{e, x\}$, e cannot be stored in table $\mathcal{B}$.

Let us say that e indeed can be stored in table $\mathcal{B}$. According to Observation 2, all elements of $\mathcal{U}_{\mathcal{B}}(e)$, including x and y, must be stored in table $\mathcal{C}$. As $x \in \mathcal{S}$, the bit corresponding to the set Y must be set to 1. As $y \notin \mathcal{S}$, the bit corresponding to Y must be set to 0. We thus arrive at a contradiction. So, e cannot be stored in table $\mathcal{B}$. □

Claim. If the size of $\mathcal{U}^2_{\mathcal{B}}(e)$ is greater than $2s$, then $\exists$ a subset $\mathcal{S}$ that contains e and has size at most three such that to store $\mathcal{S}$, e cannot be stored in table $\mathcal{B}$.

Proof. Consider the set of those elements f of $\mathcal{U}^2_{\mathcal{B}}(e)$ such that it is the only member of its block to belong to $\mathcal{U}^2_{\mathcal{B}}(e)$. As there are a total of s blocks, there could be at most s such elements. Removing those elements from $\mathcal{U}^2_{\mathcal{B}}(e)$ still leaves us with more than s elements in $\mathcal{U}^2_{\mathcal{B}}(e)$. These remaining elements have the property that there is at least one other element from its block that is present in $\mathcal{U}^2_{\mathcal{B}}(e)$. Let this set be denoted by Z.

As the size of Z is larger than the size of table $\mathcal{B}$, there must exist at least two elements $x, y \in Z$ that share a set X in table $\mathcal{B}$. According to Definition 2, this implies that there exists elements $z, z' \in \mathcal{U}_{\mathcal{B}}(e)$ such that $x \in \mathcal{C}(z) \setminus \{z\}$ and $y \in \mathcal{C}(z') \setminus \{z'\}$. It might very well be that $z = z'$.

If $x \in \mathcal{A}(e)$, as e has been stored in table $\mathcal{B}$, so all the elements of $\mathcal{A}(e)$, including x, must have been stored in table $\mathcal{B}$. Consider the subset $\mathcal{S} = \{e, x, z'\}$. As $x \in \mathcal{S}$, the bit corresponding to set X must be set to 1. As e is stored in table $\mathcal{B}$, Observation 2 tells us that $z' \in \mathcal{U}_{\mathcal{B}}(e)$ must be stored in table $\mathcal{C}$. As $\mathcal{C}(z') \cap \mathcal{S} = \{z'\}$, Observation 4 tells us that y must be stored in table $\mathcal{B}$. So, the bit corresponding to set X must be set to 0, which is absurd. So, to store $\mathcal{S}$, e cannot be stored in table $\mathcal{B}$. This argument holds even if $x = e$. Similar is the case if $y \in \mathcal{A}(e)$.

If $x \in \mathcal{B}(e) \setminus \{e\}$, and as e has been stored in table $\mathcal{B}$, Observation 2 tells us that x must be stored in table $\mathcal{C}$. Consider the subset $\mathcal{S} = \{e, z\}$. Observation 4 tells us that as $z \in \mathcal{U}_{\mathcal{B}}(e)$ is in $\mathcal{S}$, $x \in \mathcal{C}(z)$ cannot be stored in table $\mathcal{C}$, which is absurd. So, to store $\mathcal{S}$, e cannot be stored in table $\mathcal{B}$. We can similarly argue the case $y \in \mathcal{B}(e)$.

We now consider the case when $x, y \notin \mathcal{A}(e)$ and $\notin \mathcal{B}(e)$. If $\mathcal{S}$ contains e and x, and we store e in table $\mathcal{B}$, Observation 2 tells us that $z \in \mathcal{U}_{\mathcal{B}}(e)$ must be stored in table $\mathcal{C}$, and as $x \in \mathcal{C}(z)$, Observation 5 tells us that x must be stored in table $\mathcal{B}$. As $x \in \mathcal{S}$, hence the bit corresponding to set of x in table $\mathcal{B}$, which is X, must be set to 1.

If $z \neq z'$, we include z' in $\mathcal{S}$, and according to Observation 4, $y \in \mathcal{C}(z')$ must be stored in table $\mathcal{B}$. As $y \in X$ is not in $\mathcal{S}$, X must be set to 0, and we arrive at a contradiction for the subset $\mathcal{S} = \{e, x, z'\}$.

It could also be the case that $z = z'$. As $y \in Z$, there exists an element $y' \in \mathcal{A}(y) \cap \mathcal{U}^2_{\mathcal{B}}(e)$. Let $y' \in C(z'')$, where $z'' \in \mathcal{U}_{\mathcal{B}}(e)$. In this scenario, we consider storing the subset $\mathcal{S} = \{e, x, z''\}$. As, $z'' \in \mathcal{S} \cap \mathcal{U}_{\mathcal{B}}(e)$, and $y' \notin \mathcal{S}$, Observation 4 implies that y', and hence the whole of block $\mathcal{A}(y')$, including y, must be stored in table $\mathcal{B}$. As $y \notin \mathcal{S}$, the set of y in table $\mathcal{B}$, which is X, must be set to 0, and we again arrive at a contradiction.

So, we conclude that e in either of the cases cannot be stored in table $\mathcal{B}$. $\square$

The two claims above imply the following – if an element e is bad w.r.t. table $\mathcal{B}$ (Definition 3) due to Property 1, or if this property does not hold but Property 2 does, then there exists a subset, say $\mathcal{S}_1$, of size at most three containing e such that to store $\mathcal{S}_1$, e cannot be stored in table $\mathcal{B}$. The claims above also hold w.r.t. table $\mathcal{C}$. So, we can claim that if e is bad w.r.t. to table $\mathcal{C}$, then there exists a subset $\mathcal{S}_2$ containing e of size at most three such that to store $\mathcal{S}_2$, e cannot be stored in table $\mathcal{C}$.

Consider the set $\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2$. As e is common in both the subsets, size of $\mathcal{S}$ is at most five. If e is bad w.r.t. to table $\mathcal{B}$ and table $\mathcal{C}$, then to store subset $\mathcal{S}$, we cannot store e in either of the tables, which is absurd. We summarise the discussion in the following lemma.

Lemma 1. *If an element e is bad w.r.t. $\mathcal{B}$, then it must be good w.r.t $\mathcal{C}$.*

2.3 Good Schemes

Based on the above lemma, we can partition our universe $\mathcal{U}$ into two sets $\mathcal{U}_1$ and $\mathcal{U}_2$ – one that contains all the good elements w.r.t. to table $\mathcal{B}$, and one that contains the bad elements. We now partition each block and each set of the three tables of our datastructure into two parts, one containing elements from $\mathcal{U}_1$, and one containing the elements from $\mathcal{U}_2$. For elements of $\mathcal{U}_1$, only those blocks and sets that contain elements of $\mathcal{U}_1$ will be affected; similarly for the elements of $\mathcal{U}_2$.

In effect, we have two independent schemes, one for $\mathcal{U}_1$ and one for $\mathcal{U}_2$. In the scheme for $\mathcal{U}_1$, all the elements in table $\mathcal{B}$ are good. In the scheme for $\mathcal{U}_2$, all the elements in table $\mathcal{B}$ are bad, and consequently, Lemma 1 tells us that all the elements of table $\mathcal{C}$ are good. In the scheme for $\mathcal{U}_2$, we now relabel the table $\mathcal{B}$ to $\mathcal{C}$ and relabel the table $\mathcal{C}$ to $\mathcal{B}$. To make the new scheme for $\mathcal{U}_2$ work, we now have to store 0 in the blocks of table $\mathcal{A}$ for $\mathcal{U}_2$ when earlier we were storing 1, and have to store 1 when earlier we were storing 0.

This change gives us a new scheme with two important properties – the size of the datastructure has doubled from the earlier scheme, and all the elements in table $\mathcal{B}$ are now good.

Lemma 2. *Given a $(5, m, s, 2)$-scheme, we can come up with a $(5, m, 2 \times s, 2)$-scheme such that all the elements of $\mathcal{U}$ are good w.r.t. to table $\mathcal{B}$ in the new scheme.*

2.4 Space Complexity

Consider a $(5, m, 3 \times s, 2)$-scheme all of whose elements are good w.r.t. table $\mathcal{B}$. The table sizes then are each equal to s. According to Lemma 2, the 2-universe of each element w.r.t. to $\mathcal{B}$ will be at most $2s$. So, the sum total of all the 2-universe sizes of all the elements is upper bounded by $m \times 2s$.

We now consider how much each set of table $\mathcal{C}$ contribute to the total. From Definition 2, we have the following –

$$\sum_{e \in \mathcal{U}} \mid \mathcal{U}^2_{\mathcal{B}}(e) \mid = \sum_{e \in \mathcal{U}} \mid \bigcup_{f \in \mathcal{U}_{\mathcal{B}}(e)} \mathcal{C}(f) \setminus \{f\} \mid = \sum_{e \in \mathcal{U}} \left(\sum_{f \in \mathcal{U}_{\mathcal{B}}(e)} \mid \mathcal{C}(f) \setminus \{f\} \mid \right).$$

As all the elements are good, and hence for every element e, no two elements of $\mathcal{U}_{\mathcal{B}}(e)$ share a set in table $\mathcal{C}$, we can thus convert the union in Definition 2 to summation.

Resolving the above equation, the details of which can be found in the Appendix A, we have

$$\sum_{e \in \mathcal{U}} \mid \mathcal{U}^2_{\mathcal{B}}(e) \mid \geq \ c \cdot \frac{m^4}{s^3},$$

for some constant c. The proof show that the minimum value is achieved when all the blocks and the sets in the three tables are of the same size, i.e. m/s. This combined with the upperbound for total sum of the sizes of all 2-universes gives us

$$c \cdot \frac{m^4}{s^3} \leq m \times 2s.$$

Resolving the equation gives us

$$s = \Omega(m^{3/4}).$$

This bound applies to good schemes that respect the two assumptions declared at the beginning of this section.

Suppose we have an arbitrary adaptive $(5, m, s, 2)$-scheme. If we want to make all the tables in this scheme of the same size, we can add extra bits which will make the size of the data structure at most $3 \cdot s$. So, we get a $(5, m, 3 \times s, 2)$-scheme that respect Assumption 1.

In Kesh [8], sets which have multiple elements from the same block were referred to as *dirty sets*. *Clean sets* were those which contain elements from distinct blocks. It was shown in Section 3 that any scheme with dirty sets can be converted into a scheme with only clean sets by using twice amount of space. Though the final claim was made in context of $n = 3$, but the proof applies to any n. So, we can now have a $(5, m, 6 \times s, 2)$-scheme that respects both of our assumptions.

Such a scheme can be converted into a scheme with only good elements in table $\mathcal{B}$ by using twice the amount of space as before. We now have a $(5, m, 12 \times s, 2)$-scheme, where the table sizes are all $4s$, and we have shown that $4s = \Omega(m^{3/4})$.

We summarise our discussion in the following theorem on the lower bound for two-adaptive bitprobes schemes storing five elements.

Theorem 8. $s_A(5, m, 2) = \Omega(m^{3/4})$.

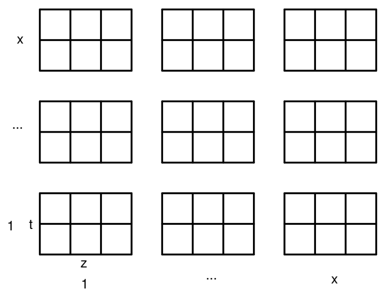

Fig. 2: Figure showing structure of a superblock

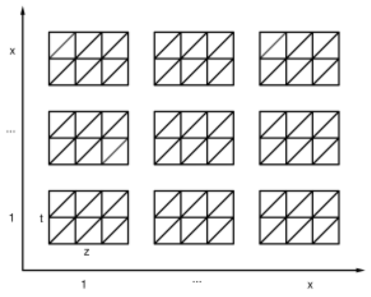

Fig. 3: Lines drawn in the first superblock

3 Our Scheme

In this section, we will present a scheme which stores an arbitrary subset of size at most five from a universe of size m, and answers the membership queries in two adaptive bitprobes. This scheme improves upon the $\mathcal{O}(m^{10/11})$-scheme by the authors [4], and is fundamentally different from that scheme in the way that here block sizes are nonuniform, and any two blocks in table $\mathcal{C}$ share at most one bit. As per the convention of that scheme, we will use the label T to refer to the table $\mathcal{A}$, T_0 to refer to the table $\mathcal{B}$, and T_1 to refer to the table $\mathcal{C}$.

Superblock : In this scheme, we use the idea in Kesh [7] of mapping the elements of the universe on a square grid. Furthermore, we have used the idea of Radhakrishnan *et al.* [10] to divide the universe into blocks and superblocks. Our scheme divides the universe of size m into superblock of size x^2zt. Each superblock is made up of rectangular grids of size $t \times z$, and there are x^2 of them as shown in Figure 2. Further, each integral point on a grid represents a unique element.

Block : For the 1st superblock we draw lines with slope 1 as shown in Figure 3. Each line drawn represents a block. From Figure 3, we can see that some blocks are of equal size and some are of different size. We do this for all the superblocks, and hence partitioning the universe into blocks. For the ith superblock we draw lines with slope $1/i$.

Table T_1 : This table has space equal to that of a single superblock, i.e., x^2zt. All the superblocks can be thought of as superimposed over each other in this table. Structure of this table is shown in Figure 2.

Table T : In this table, we store a single bit for each block. Let there be n superblocks in total. Now let us concentrate on a single grid of Figure 3. The number of lines drawn for the ith superblock is equal to $z + c \cdot it$, where c is a constant. If we sum this for all the superblocks total number of lines drawn for the single grid will be equal to $nz + c \cdot n^2t$. Now, since there are $x \times x$ grids, the total number of lines drawn for all the superblocks will be $(nz + c \cdot n^2t)x^2$. As mentioned earlier, each line represents a block, and for each block, we have one bit of space in table T. So the size of this table T is $(nz + c \cdot n^2t)x^2$ bits.

Table T_0 : In addition to lines drawn in superblocks to divide them into blocks, we also draw dotted lines in all the superblocks, as shown in Figure 4. For the ith superblock we draw dotted lines with slope $1/i$. Further, we store a block of size t in table T_0 for each dotted lines drawn. Now, we can see that for a specific superblock there could be many blocks belonging to that superblock which lies on the same dotted line. All the blocks which lie completely on the same dotted line query the same block in table T_0 kept for the dotted line. Now let us talk about the space taken by table T_0. Using the idea shown in Figure 4 to draw the dotted lines, if we sum the total number of dotted lines drawn for all the superblocks which pass through x-axis, we will get nzx. Further, if we sum the total number of dotted lines drawn for all the superblocks from the y-axis, we get it to be less than or equal to $c_1 \cdot n^2t \times x$, where c_1 is a positive constant. If we sum the total number of dotted lines drawn for all the superblocks from x and y-axis, we get $nzx + c_1 \cdot n^2t \times x$. Since we store a block of size t for each dotted line drawn, total space for table T_0 is $(nzx + c_1 \cdot n^2t \times x) \times t$.

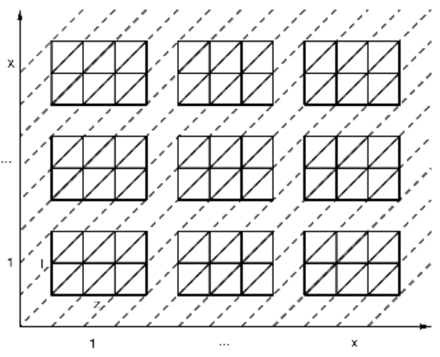

Fig. 4: Dotted lines drawn for the first superblock

Size of data structure : Summing up the space taken by all the tables we get the following equation:

$$s(x,z,t) = x^2zt + (nz + c \cdot n^2t)x^2 + (nzx + c_1 \cdot n^2t \times x) \times t \tag{1}$$

As mentioned earlier size of each superblock is x^2zt, so the total number of superblocks are $n = m/(x^2zt)$. Substituting this in the above equation, we get the following:

$$s(x,z,t) = c_1 \cdot \frac{m^2}{x^3z^2} + c \cdot \frac{m^2}{x^2z^2t} + \frac{m}{x} + \frac{m}{t} + x^2zt \tag{2}$$

Choosing $x = t = m^{1/6}$ and $z = m^{2/6}$, we get the space taken by our data structure to be $\mathcal{O}(m^{5/6})$.

3.1 Query Scheme

Our query scheme has three tables T, T_0 and T_1. Given a query element, we first find out the blocks to which it belongs. Further, we query the bit stored for this block in table T. If the bit returned is zero, we make the next query to table T_0 otherwise to table T_1. We say that query element is part of the set given to be stored if and only if last bit returned is one.

3.2 Storage Scheme

Our storage scheme sets the bits of tables T, T_0 and T_1 to store an arbitrary subset of size at most five in such a way that membership queries can be answered correctly. Storage scheme sets the bit of data structure depending upon the distribution of elements in various superblocks. Distribution of elements into various superblocks leads to various cases of the storage scheme. While generating various cases we consider an arbitrary subset $S = \{n_1, n_2, n_3, n_4, n_5\}$ of size five given to be stored. Each block is either sent to Table T_0 or T_1, and we store its bit-vector there. While sending blocks to either T_0 or T_1, we make sure that no two blocks sharing a bit have conflicting bit common in either of the tables, the correctness of the scheme relies on this fact. Keeping in mind the space constraints, we have discussed a few cases in this section, and for the sake of completeness the rest of the cases which can be handled in a similar fashion are mentioned in the Appendix B. Most of the cases generated and assignment made are similar to those generated in the previous paper on the problem by Baig *et al.* [4] to store an arbitrary subset of size at most five.

Case 1 All the elements belonging to S belongs to the same superblock. In this case, we send all the blocks having elements from the set given to be stored to table T_1. All the empty blocks, i.e., blocks which do not have any elements from S are sent to table T_0.

Case 2 Four elements $S_1 = \{n_1, n_2, n_3, n_4\}$ lies in one superblock and one $S_2 = \{n_5\}$ in other. In this case, we send the block having element n_5 to table T_1 and rest all the blocks belonging to superblock which contains this element to table T_0. All the blocks which are having conflicting bit common with the block having element n_5 are sent to table T_0. Remaining all the blocks of superblocks which contains elements from S_1 are sent to table T_1. Furthermore, rest all the empty blocks of all the superblocks are sent to table T_0.

Case 3 All the elements n_1, n_2, n_3, n_4 and n_5 lies in the different superblocks. In this case, we send all the blocks having elements to table T_0 and all the empty blocks to table T_1.

We conclude this section with the following theorem:

Theorem 9. *There is a fully explicit two adaptive bitprobe scheme, which stores an arbitrary subset of size at most five, and uses $\mathcal{O}(m^{5/6})$ space.*

4 Conclusion

In this paper, we have studied those schemes that store subsets of size at most five and answer membership queries using two adaptive bitprobes. Our first result improves upon the known lower bounds for the problem by generalising the notion of universe of sets in Kesh [8] to what may be referred to as *second order* universe. We hope that suitably defining still higher order universes will help address the lower bounds for subsets whose sizes are larger than five. Though the lower bound of $\Omega(m^{3/4})$ is an improvement, we believe that it is not tight.

We have also presented an improved scheme for the problem. It refines the approach taken by Baig *et al.* [4] and alleviates the need for blocks that overlap completely to save space. This approach helps us achieve an upper bound of $\mathcal{O}(m^{5/6})$ which is a marked improvement over existing schemes.

References

1. Alon, N., Feige, U.: On the power of two, three and four probes. In: Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2009, New York, NY, USA, January 4-6, 2009. pp. 346–354 (2009)
2. Baig, M.G.A.H., Kesh, D.: Two new schemes in the bitprobe model. In: WALCOM: Algorithms and Computation - 12th International Conference, WALCOM 2018, Dhaka, Bangladesh, March 3-5, 2018, Proceedings. pp. 68–79 (2018)
3. Baig, M.G.A.H., Kesh, D., Sodani, C.: An improved scheme in the two query adaptive bitprobe model. In: Combinatorial Algorithms - 30th International Workshop, IWOCA 2019, Pisa, Italy, July 23-25, 2019, Proceedings. pp. 22–34 (2019)
4. Baig, M.G.A.H., Kesh, D., Sodani, C.: A two query adaptive bitprobe scheme storing five elements. In: WALCOM: Algorithms and Computation - 13th International Conference, WALCOM 2019, Guwahati, India, February 27 - March 2, 2019, Proceedings. pp. 317–328 (2019)
5. Buhrman, H., Miltersen, P.B., Radhakrishnan, J., Venkatesh, S.: Are bitvectors optimal? In: Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA. pp. 449–458 (2000)
6. Garg, M.: The Bit-probe Complexity of Set Membership. Ph.D. thesis, School of Technology and Computer Science, Tata Institute of Fundamental Research, Homi Bhabha Road, Navy Nagar, Colaba, Mumbai 400005, India (2016)
7. Kesh, D.: On adaptive bitprobe schemes for storing two elements. In: Combinatorial Optimization and Applications - 11th International Conference, COCOA 2017, Shanghai, China, December 16-18, 2017, Proceedings, Part I. pp. 471–479 (2017)
8. Kesh, D.: Space complexity of two adaptive bitprobe schemes storing three elements. In: 38th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2018, December 11-13, 2018, Ahmedabad, India. pp. 12:1–12:12 (2018)
9. Lewenstein, M., Munro, J.I., Nicholson, P.K., Raman, V.: Improved explicit data structures in the bitprobe model. In: Algorithms - ESA 2014 - 22th Annual European Symposium, Wroclaw, Poland, September 8-10, 2014. Proceedings. pp. 630–641 (2014)
10. Radhakrishnan, J., Raman, V., Rao, S.S.: Explicit deterministic constructions for membership in the bitprobe model. In: Algorithms - ESA 2001, 9th Annual European Symposium, Aarhus, Denmark, August 28-31, 2001, Proceedings. pp. 290–299 (2001)
11. Radhakrishnan, J., Shah, S., Shannigrahi, S.: Data structures for storing small sets in the bitprobe model. In: Algorithms - ESA 2010, 18th Annual European Symposium, Liverpool, UK, September 6-8, 2010. Proceedings, Part II. pp. 159–170 (2010)

Appendix A Lower Bound

In this section, we prove our expression for the space lower bound. We start by proving a simple fact about sum of products.

Claim. Given that $\sum_{i=1}^{n} a_i \geq C_1$ and $\sum_{i=1}^{n} b_i \geq C_2$, then

$$\sum_{i=1}^{n} a_i b_i \geq \frac{C_1 C_2}{n}.$$

Proof. Consider the following sum –

$$\sum_{i=1}^{n} (a_i + b_i)^2.$$

This is minimised when the all of the summands are equal. Thus,

$$\sum_{i=1}^{n} (a_i + b_i)^2 \geq \sum_{i=1}^{n} \left(\frac{C_1 + C_2}{n}\right)^2 = \frac{(C_1 + C_2)^2}{n}.$$

We can now expand the sum to prove the desired inequality.

$$\begin{aligned}
\sum_{i=1}^{n} (a_i + b_i)^2 &= \sum_{i=1}^{n} a_i^2 + \sum_{i=1}^{n} b_i^2 + \sum_{i=1}^{n} 2a_i b_i \\
&\geq n\left(\frac{C_1}{n}\right)^2 + n\left(\frac{C_2}{n}\right)^2 + \sum_{i=1}^{n} 2a_i b_i \\
&\geq \frac{(C_1 + C_2)^2}{n} \\
\implies \sum_{i=1}^{n} 2a_i b_i &\geq \frac{(C_1 + C_2)^2}{n} - n\left(\frac{C_1}{n}\right)^2 - n\left(\frac{C_2}{n}\right)^2 \\
&= 2\frac{C_1 C_2}{n}.
\end{aligned}$$

□

We apply the claim above repeatedly to prove the following lemma. It is important to note that the sum is computed w.r.t. table $\mathcal{B}$, and in table $\mathcal{B}$ all the elements are good.

Lemma 3. $\sum_{e \in \mathcal{U}} \mid \mathcal{U}_{\mathcal{B}}^2(e) \mid \geq \ c \cdot \frac{m^4}{s^3}$.

Proof. We have the following expression for the sum of the sizes of all 2-universes of all elements.

$$\begin{aligned}
\sum_{e \in \mathcal{U}} \mid \mathcal{U}_{\mathcal{B}}^2(e) \mid &= \sum_{e \in \mathcal{U}} \mid \bigcup_{f \in \mathcal{U}_{\mathcal{B}}(e)} \mathcal{C}(f) \setminus \{f\} \mid \\
&= \sum_{e \in \mathcal{U}} \left(\sum_{f \in \mathcal{U}_{\mathcal{B}}(e)} \mid \mathcal{C}(f) \setminus \{f\} \mid \right)
\end{aligned}$$

We could convert the union in the expression above into the summation as no two elements of $\mathcal{U}_{\mathcal{B}}(e)$ share a set. We can similarly expand $\mathcal{U}_{\mathcal{B}}(e)$ from Definition 1.

$$\begin{aligned}\sum_{e\in\mathcal{U}} |\, \mathcal{U}^2_{\mathcal{B}}(e)\, | &= \sum_{e\in\mathcal{U}} \left(\sum_{f\in\mathcal{U}_{\mathcal{B}}(e)} |\, \mathcal{C}(f) \setminus \{f\}\, | \right) \\ &= \sum_{e\in\mathcal{U}} \left(\sum_{g\in\mathcal{B}(e)\setminus\{e\}} \left(\sum_{f\in\mathcal{A}(g)\setminus\{g\}} (|\, \mathcal{C}(f) \setminus \{f\}\, |) \right)\right)\end{aligned}$$

We will first compute the value of the following expression.

$$\begin{aligned}\sum_{e\in\mathcal{U}} |\mathcal{B}(e) \setminus \{e\}| &= \sum_{e\in\mathcal{U}} (|\mathcal{B}(e)| - 1) = \sum_{e\in\mathcal{U}} |\mathcal{B}(e)| - m \\ &= \sum_{X\in\mathcal{B}} c_X |X| - m. && \text{(collecting over the sets of } \mathcal{B})\end{aligned}$$

Here, the sum of the coefficients c_X is m, and the number of terms, which is same as the number of sets of $\mathcal{B}$, is s. Further, $\sum_{X\in\mathcal{B}} |X| = m$. So, applying the above claim,

$$\begin{aligned}\sum_{e\in\mathcal{U}} |\mathcal{B}(e) \setminus \{e\}| &= \sum_{X\in\mathcal{B}} c_X |X| - m \\ &\geq \frac{m \cdot m}{s} - m && \text{(sum of the sizes of the sets of } \mathcal{B} \text{ is } m) \\ &\geq c\frac{m^2}{s} && \text{(for some suitable coefficient c)}\end{aligned}$$

Next, we compute an expression the sum of whose coefficients is the above sum.

$$\begin{aligned}\sum_{e\in\mathcal{U}} \left(\sum_{g\in\mathcal{B}(e)\setminus\{e\}} |\mathcal{A}(g) \setminus \{g\}| \right) &\geq c \sum_{Z\in\mathcal{A}} c_Z |Z|, && \text{(collecting over the blocks of } \mathcal{A}) \\ &\geq c' \cdot \frac{\frac{m^2}{s} m}{s} = c' \frac{m^3}{s^2}. && \text{(for some suitable coefficient c')}\end{aligned}$$

We finally compute the desired expression of which the sum of coefficients is the above expression.

$$\begin{aligned}\sum_{e\in\mathcal{U}} |\, \mathcal{U}^2_{\mathcal{B}}(e)\, | &= \sum_{e\in\mathcal{U}} \left(\sum_{g\in\mathcal{B}(e)\setminus\{e\}} \left(\sum_{f\in\mathcal{A}(g)\setminus\{g\}} (|\, \mathcal{C}(f) \setminus \{f\}\, |) \right)\right) \\ &\geq c \cdot \sum_{Y\in\mathcal{C}} c_Y |Y|, && \text{(collecting over the sets of } \mathcal{C}) \\ &= c' \cdot \frac{\frac{m^3}{s^2} m}{s} = c' \frac{m^4}{s^3}. && \text{(for some suitable coefficient c')}\end{aligned}$$

$\square$

Appendix B Storage Scheme

Rest of the case of Section 3.2 is discussed here.

Case 4 Three elements $S_1 = \{n_1, n_2, n_3\}$ belong to one superblock and two elements $S_2 = \{n_4, n_5\}$ to the other.

Case 4.1 All the blocks to which elements from S_1 belong lies on the same dotted line of their superblock.

Case 4.1.1 Two blocks to which elements from S_2 belong have a conflicting bit common with the blocks corresponding to the elements from S_1 in table T_1. In this case, we send the blocks having elements from S_2 to table T_0. Further, we send empty blocks lying on the dotted lines to which blocks having elements from S_2 belongs to table T_1. We send all the blocks which contain elements from S_1 in table T_1. We send the rest of the empty blocks to table T_0.

Case 4.1.2 Only one block which contains an element from S_2 has a conflicting bit common with the block corresponding to the elements from S_1 in table T_1. In this case, we send all the blocks which contain elements from S_1 to table T_1. We send the block having an element from S_2, and having conflicting bit common with block having an element from S_1, to table T_0, and the rest of the blocks which lies on the dotted line containing this block to table T_1. If after this other nonempty block having an element from S_2 is still unassigned then we send it to table T_1, and all the empty blocks lying on the dotted line containing this block to table T_0 . Rest all the empty blocks are sent to table T_0.

Case 4.1.3 None of the blocks which contain an element from S_2 have a conflicting bit common with the block which includes an element from S_1 in table T_1 . In this case, we send all the nonempty blocks to table T_1 and all the empty blocks to table T_0.

Case 4.2 Two blocks which contain elements say n_1 and n_2 from S_1 lies on the same dotted line and other say n_3 lies on a different dotted line.

Case 4.2.1 All the blocks which contain elements from S_2 have a conflicting bit common with blocks which include elements from S_1 in table T_1.

4.2.1.1 Let us first consider the case where blocks having elements from S_2 have a conflicting bit common with the blocks having elements n_1 and n_2. In this case, we send the blocks having element n_4 and n_5 to table T_0, and all the blocks lying on the dotted lines containing these block to table T_1. Further, we send the blocks having elements n_1 and n_2 to table T_1. Block having element n_3 is sent to table T_0, and all the empty blocks lying on the dotted line containing this block is sent to table T_1. Rest all the empty blocks are sent to table T_0.

4.2.1.2 Without loss of generality let us now consider the case where blocks having an element from S_2 have a conflicting bit common with blocks having element n_1 and n_3. In this case, we send the blocks having elements n_4 and n_5 to table T_1, rest all the blocks lying on the dotted line(lines) containing these blocks to table T_0. Further, we send the blocks having element n_1 and n_3 to table T_0, and all the blocks lying on the dotted lines containing these blocks to table T_1. Rest all the blocks are sent to table T_0.

4.2.1.3 Now we are left with a case where block having an element from S_2 have a conflicting bit common with blocks having elements n_1, n_2 and n_3. In this case, we send the blocks having element n_4 and n_5 to table T_0, and all the empty blocks lying on the dotted lines containing this block to table T_1. Further, we send all the blocks having elements from S_1 to table T_1, and rest all the empty blocks to table T_0.

Case 4.2.2 Only one block having an element from S_2 have a conflicting bit common with the block(blocks) having an element(elements) from S_1.

Case 4.2.2.1 All the blocks having elements from S_2 lies on the same dotted line. Without loss of generality, let us say block having element n_4 from S_2 have a conflicting bit common with a block having an element from S_1. In this case, we send the blocks having element n_4 and n_5 to table T_1, and all the empty blocks lying on the dotted line containing these blocks to table T_0. Further, we send the block(blocks) having an element(elements) from S_1, and having a conflicting bit common with a block having element n_4 to table T_0, and all the blocks lying on the dotted line containing this block to table T_1. We now send the rest of the block(blocks) having an element from S_1 to table T_1. Rest all the empty blocks are sent to table T_0.

Case 4.2.2.2 Now let us consider a case where blocks having an element from S_2 lies on the different dotted line. Without loss of generality lets say block having element n_4 have a conflicting bit common with a block(blocks) having an element(elements) from S_1.

Let us first consider the case where block having element n_4 have a conflicting bit common with either block having element n_1 or n_2. Without loss of generality, let us say block having element n_4 have a conflicting bit common with a block having element n_1. In this case, we send the block having element n_3, n_4 and n_5 to table T_0, and all the empty blocks lying on the dotted line containing these blocks to table T_1. Now, we see the positions of the blocks having elements n_1 and n_2. Let us first consider the case where blocks having element n_1 or n_2 have conflicting bit common with one of the empty blocks lying on the dotted line which contains block having element n_5. Without loss of generality let us say that block having element n_1 have conflicting bit common with one of the empty blocks lying on the dotted line which contains block having element 5. In this case, we send the block having element n_1 to table T_0, and rest all the blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. On the other hand, if the block having element n_1 or n_2 do not have conflicting bit common with empty blocks lying on the dotted line which contains block having element n_5, then we send the blocks having elements n_1 and n_2 to table T_1, and rest all the empty blocks lying on the dotted line containing these blocks to the table T_0. Rest all the empty blocks are sent to the table T_0.

Now let us consider the case where block having element n_4 have a conflicting bit common with a block having element n_3. Now, we see if the block having element n_4 have conflicting bit common with block having element n_1 or n_2. Without loss of generality, let us say that block having element n_4 have conflicting bit common with block having element n_1. In this case, we can use the assignment made in previous paragraph. On the other hand, if the block having element n_4 do not have conflicting bit common with block having element n_1 or n_2, then we send the blocks having elements n_3 and n_5 to table T_0, and all the empty blocks lying on the dotted lines containing these blocks to table T_1. Further, we send the block having element n_4 to table T_1, and all the empty blocks lying on the dotted line containing this block to table T_0. Now we see the position of the blocks having elements n_1 and n_2. Let us first consider the case where on of the blocks having elements n_1 or n_2 has conflicting bit common with one of the empty blocks lying on the dotted line which contains block having element n_5. Without loss of generality, let us say that block having element n_1 has conflicting bit common with one of the empty blocks lying on the dotted line which contains block having element n_5. In this case, we send the block having element n_1 to table T_0, and all the blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. On the other hand, if none of the block having element n_1 or n_2 has conflicting bit common with empty blocks lying on the dotted line which contains block having element n_5, then we send the block having element n_1 and n_2 to table T_1, and rest all the empty blocks to table T_0.

Case 4.2.3 None of the blocks which contain an element from S_2 have a conflicting bit common with the block which includes an element from S_1 in table T_1 . In this case, we send all the nonempty blocks to table T_1 and all the empty blocks to table T_0.

Case 4.3 All the blocks having an element from S_1 lies on the different dotted lines.

Case 4.3.1 Both the blocks having elements n_4 and n_5 have a conflicting bit common with the blocks having elements from S_1 in table T_1. Now we see the positions of the blocks having elements n_4 and n_5. Blocks having elements n_4 and n_5 can either lie on the same dotted line or on the different dotted lines. If the blocks having elements n_4 and n_5 lie on the different dotted lines, then we send all the blocks having elements to table T_0, and all the empty blocks to table T_1. Now we consider the case where blocks having elements n_4 and n_5 lies on the same dotted line. Furthermore, without loss of generality let us consider that blocks having elements n_4 and n_5 conflicts with blocks having elements n_1 and n_2 respectively. In this case, we send the blocks having elements n_1 and n_2 to table T_0, and all the empty blocks lying on the dotted lines containing these blocks to the table T_1. Further, we send the blocks having elements n_4 and n_5 to table T_1, and all the empty blocks lying on the dotted line containing these blocks to table T_1. Now, we see the position of the block having element n_3. If the block having element n_3 has conflicting bit common with blocks having elements n_4 or n_5, then we send the block having element n_3 to table T_0, and all the empty blocks lying on the dotted line which contains this block to table T_1. Rest all the empty blocks to table T_0. On the other hand, if the block having element n_3 do not have conflicting bit common with block having element n_4 or n_5, then we send the block having element n_3 to table T_1, and rest all the empty blocks to table T_0.

Case 4.3.2 Only one of the block having element say n_4 from S_2 have a conflicting bit common with blocks having an element from S_1 in table T_1. Similar to the last case, in this case also we see whether blocks having elements n_4 and n_5 lie on the same dotted line or on the different dotted lines. If the blocks having elements n_4 and n_5 lies on the different dotted lines, then we send the blocks having elements to table T_0, and all the empty blocks to table T_1. Now, we consider the case where blocks having elements n_4 and n_5 lies on the same dotted lines. Furthermore, without loss of generality let us say that block having the element n_1 have a conflicting bit common with the block having the element n_4. In this case, we send the blocks having elements n_1 and n_4 to table T_0, and all the empty blocks lying on the dotted lines containing these blocks to table T_1. Further, we send the blocks having elements n_2, n_3 and n_5 to table T_1. Rest all the empty blocks are sent to table T_0.

Case 4.3.3 None of the blocks which contain an element from S_2 have a conflicting bit common with the block which includes an element from S_1. This case is the same as Case 4.1.3.

Case 5 The elements in $S_1 = \{n_1, n_2, n_3\}$ lies in a superblock and the elements n_4 and n_5 in the different superblocks.

Case 5.1 Blocks having element n_1, n_2 and n_3 lies on a same dotted line.

Case 5.1.1 Blocks having element n_4 and n_5 have a conflicting bit common with the blocks having elements from S_1 in table T_1. In this case, we send the blocks having elements n_4 and n_5 to table T_0 and the rest of the empty blocks which lie on the dotted lines containing these blocks to table T_1. We send the blocks having elements from S_1 to table T_1. Rest all the empty blocks are sent to table T_0.

Case 5.1.2 Only one of the block having element say n_4 have a conflicting bit common with the block having an element from S_1 in table T_1. Without loss of generality let us say block having the element n_1 have a conflicting bit common with the block having the element n_4. In this case, we send the blocks having elements n_4 and n_5 to table T_0 and all the empty blocks lying on the dotted line containing these blocks to table T_1. Now we see whether the dotted line which contains block having the element n_5 passes through the block having element elements from S_1 or not. Let us first consider a case where the dotted line which contains block having the element n_5 passes through one of the blocks having elements from S_1, without loss of generality let us say it passes through block having the element n_2. In this case, we send the block having the element n_2 to table T_0 and rest all the blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. On another hand, if the dotted line which contains block having the element n_5 does not pass through any of the block having element from S_1 then we send the blocks having elements from S_1 to table T_1. Rest all the empty blocks are sent to table T_0.

Case 5.1.3 None of the blocks which contains element n_4 or n_5 have a conflicting bit common with the blocks which include an element from S_1 in table T_1. Now, the blocks having elements n_4 and n_5 can conflict among themselves or it does not. If the block having elements n_4 and n_5 do not conflict among themselves, then we send all the blocks having elements to table T_1, and rest all the empty blocks to table T_0. On the other hand, if the blocks having elements n_4 and n_5 conflict among themselves, then we see whether they conflict on the dotted line which contains block having elements from S_1. If they conflict on the dotted line which contains block having elements from S_1, then we send the blocks having elements n_4 and n_5 to table T_0, and all the empty blocks lying on the dotted line containing this block to table T_1. Further, we send the blocks having elements from S_1 to table T_1, and rest all the empty blocks to table T_0. If the blocks having elements n_4 and n_5 do not conflict on the dotted line which contains blocks having elements from S_1, then we send the block having element n_4 to table T_0, and all the empty blocks lying on the dotted line containing this block to table T_1. Further, we send the block having element n_5 to table T_1, and all the empty blocks lying on the dotted line containing this block to table T_0. Now, we see whether the dotted line which contains block having element n_4 passes through block having element n_1 or n_2. Without loss of generality, let us say that the dotted line which contains block having element n_4 passes through block having element n_1. In this case, we send the block having element n_1 to table T_0, and rest all the blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. If the dotted lines which contains block having elements n_4 or n_5 do not pass through block having elements from S_1, then we send the blocks having elements from S_1 to table T_1, and rest all the empty blocks to table T_0.

Case 5.2 Two elements say n_1 and n_2 lies on the same dotted line and the element n_3 lies on a different dotted line.

Case 5.2.1 Blocks having the elements n_4 and n_5 have a conflicting bit common with the blocks having elements from S_1 in table T_1.

Case 5.2.1.1 Blocks having the elements n_4 and n_5 have a conflicting bit common with the block having elements n_1 and n_2 in table T_1. In this case, we send the blocks having elements n_4 and n_5 to table T_0 and all the empty blocks lying on the dotted lines containing these blocks to table T_1. Also, we send the blocks having elements n_1 and n_2 to table T_1. We send the block which contains the element n_3 to table T_0 and the rest of the empty block lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0.

Case 5.2.1.2 Blocks having elements n_4 and n_5 have a conflicting bit common with the blocks lying on the different dotted lines, say block having elements n_1 and n_3 in table T_1. In this case, we send the blocks having elements n_4 and n_5 to table T_1 and the rest of the empty blocks lying on the dotted line containing these blocks to table T_0 . We send the blocks having elements n_1 and n_3 to table T_0 and the rest of the blocks lying on these dotted lines to table T_1. Rest all the empty blocks are sent to table T_0.

Case 5.2.1.3 Blocks having element n_4 and n_5 have a conflicting bit common in table T_1. If these blocks have a conflicting bit common with the block having the element n_1 or n_2 then we send both the blocks having the element n_4 and n_5 to table T_0 and the rest of the empty block lying on the dotted line containing these blocks to table T_1. Also, we send the blocks having the element n_1, n_2 and the blocks lying on the dotted line containing these blocks to table T_1. We send the block which contains element n_3 to table T_0 and the rest of the empty block which lies on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0.

If the blocks containing elements n_4 and n_5 have a conflicting bit common with the block which contains element n_3 in table T_1 then we send the block containing element n_3 to table T_0 and the rest of the empty block which lie on the dotted line containing this block to table T_1. Also, we send the block containing n_4 to table T_1 and the rest of the empty block lying on the dotted line containing this block to table T_0. We send the block having the element n_5 to table T_0 and the rest of the empty block lying on the dotted line containing this block to table T_1. Now we see whether the dotted line which contains block having the element n_5 passes through the block having the element n_1 or n_2. Without loss of generality let us first consider the case where the dotted line which contains block having the element n_5 passes through the block having the element n_1. In this case, we send the block having the element n_1 to table T_0 and all the blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. If the dotted line which contains block having element n_5 does not pass through the blocks having elements n_1 or n_2 then we send the blocks having elements n_1 and n_2 to table T_1 and rest all the empty blocks to table T_0.

Case 5.2.2 Only one block having an element n_4 or n_5, have a conflicting bit common with a block having the element from S_1 in table T_1. Let us first consider the case where block having element n_4 have a conflicting bit common with the block having element n_1 or n_2. Without loss of generality let us say block having the element n_4 have a conflicting bit common with the block having the element n_1. Now we can have two cases, either the block having element n_4 conflicts with block having the element n_3 or it does not. Let us first consider the case where block having the element n_4 do not conflict with block having the element n_3. In this case, we send the blocks having the elements n_3, n_4 and n_5 to table T_0, and all the empty blocks lying on the dotted line containing these blocks to table T_1. Now we see whether the empty blocks lying on the dotted line which contains block having element n_5 passes through blocks having elements n_1 or n_2. Without loss of generality let us say that empty block lying on the dotted line which contains block having element n_5 passes through block having element n_1. In this case, we send the block having the element n_1 to table T_0 and rest all the blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. On the other hand, if the dotted line which contains block having element n_5 do not pass through blocks having element n_1 or n_2, then we send the block having element n_1 and n_2 to T_1, and rest all the empty blocks to table T_0. Now we consider the case where block having element n_4 have conflicting bit common with block having element n_3. In this case, we send the block having element n_4 to table T_0, and all the empty blocks lying on the dotted line containing this block to table T_1. Now we see the position of the block having element n_5. If the block having element n_5 have conflicting bit common with empty block lying on the dotted line containing blocks having elements n_1 and n_2, then we send the block having element n_1, n_2, n_3 and n_5 to table T_1, and rest all the empty blocks to table T_0. If the block having element n_5 do not have conflicting bit common with empty block lying on the dotted line containing blocks n_1 and n_2, then we send the block having element n_5 to table T_0, and all the empty blocks lying on the dotted line containing this block to table T_1. Now we see whether the blocks having element n_1 or n_2 have conflicting bit common with empty block lying on the dotted line containing block having element n_5. Without loss of generality let us say that block having element n_1 have conflicting bit common with empty block lying on the dotted line containing block having element n_5. In this case, we send the block having element n_1 to table T_0, and rest all the blocks lying on the dotted line containing this block to table T_1. Further, we send the block having n_3 to table T_0, and all the empty blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. If none of the blocks having element n_1 or n_2 have conflicting bit common with empty block lying on the dotted line containing element n_5, then we send the blocks having elements n_3 and n_5 to table T_0, and all the empty blocks lying on the dotted line containing this block to table T_1. Furthermore, we send the blocks having element n_1 and n_2 to table T_1, and rest all the empty blocks to table T_0.

Now let us consider the case where block having the element n_4 have a conflicting bit common only with the block having the element n_3. Now we see whether the blocks having elements n_4 and n_5 conflicts or not. Let us first consider the case where blocks having elements n_4 and n_5 do not conflicts. Now we see the position of the block having element n_5. Let us first consider the case where block having element n_5 conflicts with empty block lying on the dotted line which contains block having element n_1. In this case, we send the block having the element n_1, n_2, n_4 and n_5 to table T_1, and rest all the empty blocks lying on the dotted lines containing these blocks to table T_0. We send the block having the element n_3 to table T_0, and all the empty blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. If the block having the element n_5 have a conflicting bit common with a empty block lying on the dotted line which contains block having the element n_3, then we send the blocks having the elements n_3 and n_5 to table T_1, and all the empty blocks lying on the dotted lines containing these blocks to table T_0. We send the block having element n_4 to table T_0, and rest all the empty blocks lying on the dotted line containing this block to table T_1. If the block having the element n_1 have a conflicting bit common with block lying on the dotted line which contains block having the element n_4, then we send the block having the element n_1 to table T_0, and rest all the blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. Similar is the case if the block having the element n_2 have a conflicting bit common with block lying on the dotted line which contains block having the element n_4. On the other hand, if the dotted line which contains block having element n_4 do not pass through blocks having elements n_1 or n_2, then we send the blocks having elements n_1 and n_2 to table T_1, and rest all the empty blocks to table T_0. If none of the above case occurs, and the block having the element n_5 does not have a conflicting bit common with a block lying on the dotted line which contains block having the element n_3, then we send the block having the element n_1, n_2, n_4 and n_5 to table T_1, and rest all the empty blocks lying on the dotted lines containing these blocks to table T_0. Further, we send the block having element n_3 to table T_0, and all the empty blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. Now, we see the case where block having element n_4 and n_5 conflicts. Now we can have several cases depending upon whether blocks having element n_4 and n_5 conflicts with empty block lying on the dotted line which contains block having element n_1. Let us first consider the case where block having element n_5 conflicts with empty block lying on the dotted line which contains block having element n_1. In this case, we send the blocks having elements n_3, n_4 and n_5 to table T_0, and rest all the empty blocks lying on the dotted lines containing these blocks to table T_1. Now we see whether the dotted line which contains block having element n_4 passes through block having element n_1 or n_2. Without loss generality let us say that dotted line which contains block having element n_4 passes through block having element n_1. In this case, we send the block having the element n_1 to table T_0, and rest all the blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. On the other hand, if the dotted line which contains block having element n_4 do not pass through blocks having elements n_1 or n_2, then we send the blocks having elements n_1 and n_2 to table T_1, and rest all the empty blocks to table T_0. Similar is the case when block having element n_4 conflicts with empty block lying on the dotted line which contains block having element n_1. If none of the above occurs, then we send the blocks having elements n_3 and n_5 to table T_0, and rest all the empty blocks lying on the dotted lines containing these blocks to table T_1. Further, we send the block having element n_4 to table T_1, rest all the empty blocks lying on the dotted line containing this block to table T_0. Now we see whether the dotted line which contains block having element n_5 passes through block having element n_1 or n_2. Without loss generality let us say that dotted line which contains block having element n_5 passes through block having element n_1. In this case, we send the block having the element n_1 to table T_0, and rest all the blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. On the other hand, if the dotted line which contains block having element n_4 do not pass through blocks having elements n_1 or n_2, then we send the blocks having elements n_1 and n_2 to table T_1, and rest all the empty blocks to table T_0.

Case 5.2.3 None of the blocks having the elements n_4 or n_5 have a conflicting bit common with blocks having elements from S_1 in table T_1. We can have several cases depending upon whether the blocks having element n_4 and n_5 conflicts or not. Let us first consider the case where blocks having elements n_4 and n_5 do not conflict. In this case we send all the blocks having elements to table T_1, and all the empty blocks to table T_0. Now let us consider the case where blocks having elements n_4 and n_5 conflicts. Now, we see the position of the block having elements n_4 and n_5. Let us first consider the case where blocks having elements n_4 and n_5 conflicts on the dotted line which contains block having element n_1. In this case, we send the blocks having elements n_1 and n_2 to table T_1, and rest all the empty blocks lying on the dotted line containing these blocks to table T_0. Further, we send the blocks having elements n_3, n_4 and n_5 to table T_0, rest all the empty blocks lying on the dotted lines containing these blocks to the table T_1. Rest all the empty blocks are sent to table T_0. Now let us consider the case where block having elements n_4 and n_5 conflicts outside the dotted line which contains block having element n_1. Now we can have a case where either block having elements n_4 or n_5 conflicts with empty block lying on the dotted line which contains block having element n_1. Without loss of generality let us say that block having element n_4 conflicts with empty block lying on the dotted line which contains block having element n_1. In this case, we send the blocks having elements n_3, n_4 and n_5 to table T_0, and rest all the empty blocks lying on the dotted line containing these blocks to table T_1. Now we see whether the dotted line which contains block having element n_5 passes through block having element n_1 or n_2. Without loss of generality, let us say that the dotted line which contains block having element n_5 passes through block having element n_1. In this case, we send the block having element n_1 to table T_0, and rest all the blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. On the other hand, if the dotted line which contains block having element n_5 do not pass through block having element n_1 or n_2, then we send the block having element n_1 and $n2$ to table T_1, and rest all the empty blocks to table T_0. Now we consider the case where block having element n_4 or n_5 do not conflict with empty block lying on the dotted line which contains block having element n_1. In this case, we send the blocks having elements n_3 and n_5 to table T_1, and rest all the empty blocks lying on the dotted lines containing these blocks to table T_0. Further, we send the block having element n_4 to table T_0, and rest all the empty blocks lying on the dotted line containing this block to table T_1. Now we see if the dotted line which contains block having element n_4 passes through block having element n_1 or n_2. Without loss of generality let us say that dotted line which contains block having element n_4 passes through block having element n_1. In this case, we send the block having element n_1 to T_0, and rest all the blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. If the dotted line which contains block having element n_4 do not pass through block having element n_1 or n_2, then we send the blocks having elements n_1 and n_2 to table T_1, and rest all the empty blocks to table T_0.

Case 5.3 All the elements belonging to S_1 lies on the different dotted lines. In this case, we send all the blocks having elements to table T_0 and all the empty blocks to table T_1.

Case 6 Two elements $S_1 = \{n_1, n_2\}$ lies in a superblock other two elements $S_2 = \{n_3, n_4\}$ lies in other superblock and an element $S_3 = \{n_5\}$ in a different superblock.

Case 6.1 Blocks having elements belonging to S_1 lies on the same dotted line, blocks having elements belonging to S_2 lies on the same dotted line.

Case 6.1.1 Two blocks having elements from S_2 and S_3 have a conflicting bit common with the blocks having elements from S_1 in table T_1. Without loss of generality, let us say that block having element n_3 have a conflicting bit common with the block having the element n_1 and the block having the element n_5 have a conflicting bit common with the block having the element n_2. In this case, we send the block having the element n_4 to table T_0 and the rest of the block lying on the dotted line containing this block to table T_1. We send the block having the element n_1 to table T_0, and the rest of the block lying on this dotted line to table T_1. Also, we send the block having the element n_5 to table T_0, and the rest of the empty block lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0.

If the blocks having element from S_2 and S_3 have a conflicting bit common with the same block having the element from S_1, then we send the blocks having a conflicting bit common from S_2 and S_3 to table T_0, and the rest of the blocks lying on these dotted lines to table T_1. Also, we send the blocks having elements from S_1 to table T_1. Rest all the empty blocks are sent to table T_0.

Case 6.1.2 Only one block having the element from S_2 or S_3 have a conflicting bit common with the block having the element from S_1.

Case 6.1.2.1 Block having an element from S_2 have a conflicting bit common with the block having an element from S_1. Without loss of generality let us say that the block having the element n_3 have a conflicting bit common with the block having the element n_1. Now, we can have two cases, either the block containing element n_5 have a conflicting bit common with the block containing element n_4 or it does not.

If the block containing n_5 have a conflicting bit common with the block containing n_4, then we send the block containing n_3 to table T_0, and the rest of the block lying on the dotted line containing this block to table T_1. We send the block containing n_5 to table T_0, and the rest of the block lying on the dotted line containing this block to table T_1. Now we see whether the dotted line which contains block having element n_5 passes through block having element n_2 or not. Let us first consider the case where dotted line which contains block having element n_5 passes through block having element n_2. In this case, we send the block having the element n_2 to table T_0, and the rest of the block lying on this dotted line to table T_1. Rest all the empty block are sent to table T_0. On the other hand, if the dotted line which contains block having element n_5 do not pass through block having element n_2, then we send the blocks having elements n_1 and n_2 to table T_1, and rest all the empty blocks to table T_0.

If the block containing n_5 do not have a conflicting bit common with the block having the element n_4, then we send the block having the element n_3 to table T_0 and the rest of the block lying on this dotted line to table T_1. Now we see the position of the block which contains the element n_5 to make the assignment. If the block which contains element n_5 have conflicting bit common with empty block lying on the dotted line which contains block having elements from S_1, then we send the block having the element n_5 to table T_1, and rest all the empty blocks lying on the dotted line which contains this block to table T_0. Also, we send blocks having elements from S_1 to table T_1, and the rest of the empty blocks lying on the dotted line which contains this block to table T_0. Rest of the empty blocks we send to table T_0.

Rest for all other positions of n_5, we send blocks having elements n_3 to table T_0, and all the blocks lying on the dotted line containing this block to table T_1. Further, we send the block having the element n_5 to table T_0, and all other blocks lying on the dotted line containing this block to table T_1. Now we see whether the dotted line which contains block having element n_5 passes through block having element n_1 or n_2. Without loss of generality, let us say that dotted line which contains block having element n_5 passes through block having element n_2, in this case we send the block having element n_2 to table T_0, and rest all the blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. On the other hand if the dotted line which contains block having element n_5 do not pass through block having element n_1 or n_2, then we send the blocks having elements n_1 and n_2 to table T_1, and rest all the empty blocks to table T_0.

Case 6.1.2.2 Block having an element from S_3 have a conflicting bit common with the block having an element from S_1. Without loss of generality let us say that block having element n_5 have a conflicting bit common with the block having the element n_1. In this case, we send the block having the element n_5 to table T_0 and the rest of the block lying on the dotted line which contains this block to table T_1. Now we see the position of the block having the element from S_2.

One of the block having an element from S_2 have a conflicting bit common with the empty block on the dotted line which contains block having the element n_5. Without loss of generality let us say that block having element n_3 have a conflicting bit common with an empty block on the dotted line which contains block having the element n_5. In this case, we send the block having an element n_3 to table T_0, and the rest of the block lying on the dotted line containing this block to table T_1. Now we see the position of the block having element n_4, if it has a conflicting bit common with a empty block lying on the dotted line which contains block having element from S_1, then we send the block having element n_1 and n_2 to table T_1, and the rest of the empty block lying on the dotted line containing these blocks to table T_0. Rest all the empty blocks are sent to table T_0. On another hand, if the block having the element n_4 does not have a conflicting bit common with a block lying on the dotted line which contains block having the element from S_1 then we send the block having the element n_2 to table T_0, and the rest of the block lying on the dotted line containing this block to table T_1. We send the rest of the empty block to table T_0.

If the block having the element from S_2 do not have a conflicting bit common with a block lying on the dotted line which contains block having the element n_5 then we send the blocks having an element from S_2 to table T_1, and the rest of the empty block lying on the dotted line which contains these blocks to table T_0. Also, we send the blocks having the element n_1 and n_2 to table T_1. We send rest of the empty block to table T_0.

Case 6.1.3 None of the block having elements from S_2 or S_3 has a conflicting bit common with the blocks having an element from S_1 in table T_1. Now, here, we can have two cases. Either the block having the element from S_3 have a conflicting bit common with the block having the element from S_2 or it does not.

Let us first consider the case where one of the blocks having the element from S_3 have a conflicting bit common with one of the blocks having an element from S_2. Without loss of generality, we can say that block having the element n_5 have a conflicting bit common with the block having the element n_4. Now, we see whether the dotted lines containing blocks having elements from S_2 or S_3 passes through block having elements from S_1 or not. Without loss of generality, let us say that dotted line which contains block having element n_5 passes through block having element n_1. In this case, we send the block having the element n_5 to table T_1, and the rest of the empty blocks lying on the dotted line which contains this block to table T_0. We send the block having the element n_4 to table T_0, and rest of the block lying on the dotted line which contains this block to table T_1. Now we see the positions of the blocks having an element from S_1. Let us first consider the case where a block having an element from S_1 have a conflicting bit common with a block lying on the dotted line which contains blocks having elements from S_2. Without loss of generality let us say block having the element n_2 have a conflicting bit common with block lying on the dotted line which contains blocks having elements from S_2. In this case, we send the block having the element n_2 to table T_0, and all other block lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. On the other hand, if the blocks having elements from S_1 do not have a conflicting bit common with block lying on the dotted line which contains blocks having elements from S_2, then we send the block having elements from S_1 to table T_1, and rest all the empty blocks to table T_0. If none of the dotted lines which contains blocks having elements from S_2 or S_3 passes through block having elements from S_1, then we send the blocks having elements n_4 and n_5 to table T_0, and all the blocks lying on the dotted lines containing these blocks to table T_1. Further, we send the blocks having elements from S_1 to table T_1, and rest all the empty blocks to table T_0.

If the block having an element from S_3 do not have a conflicting bit common with block having an element from S_2, then we send all the block having elements to table T_1, and all the empty block to table T_0.

Case 6.2 Now we consider the case where one of the sets having elements lie on a dotted line and other set having element lie on the different dotted lines. Without loss of generality consider the case where blocks having elements belonging to S_1 lies on a dotted line and blocks having elements belonging to S_2 lies on the different dotted lines.

Case 6.2.1 All the blocks having elements from S_2 and S_3 have a conflicting bit common with the blocks having elements from S_1. In this case, we send the block having an element from S_2 and S_3 to table T_0, and rest of the empty blocks lying on the dotted lines containing these blocks to table T_1. Also, we send the blocks having elements from S_1 to table T_1 and the rest of the empty blocks lying on the dotted line which contains these blocks to table T_0. We send the rest of the empty blocks to table T_0.

Case 6.2.2 Two blocks having elements from S_2 and S_3 have a conflicting bit common with the block having an element from S_1. Now, here we can have two cases, either those two blocks have elements belonging to S_2, or we can have one element belonging to S_2, and other to S_3.

Let us first consider the case where two blocks having elements from S_2 have a conflicting bit common with blocks having an element from S_1. Without loss of generality, let us say that block having element n_3 have a conflicting bit common with the block having the element n_1 and the block having the element n_4 have a conflicting bit common with the block having n_2. In this case, we send the block having elements n_3 and n_4 to table T_0 and rest of the empty block lying on the dotted lines containing these blocks to table T_1. Now we see the position of the block having the element n_5. Here we can have two cases either the dotted line which contains the block having the element n_5 passes through one of the block having an element from S_1 or it does not. Let us first consider the case where it passes through one of the blocks having elements from S_1. Without loss of generality let us say that the dotted line which contains block having the element n_5 passes through the block having the element n_1. In this case, we send the block having the element n_5 to table T_0, and rest of the block lying on the dotted line which contains this block to table T_1. Also, we send the block having the element n_1 to table T_0, and rest of the block lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. On another hand, if the dotted line which contains the block having the element n_5 do not pass through blocks having elements from S_1, then we send the block having the element n_5 to table T_0, and rest of the empty blocks on the dotted line containing this block to table T_1. Also, we send the blocks having elements from S_1 to table T_1, and rest of the empty blocks to table T_0. If the block having elements n_3 and n_4 conflicts with the same block having elements from S_1, say n_1, then assignment made above will work in this case.

Now we consider the case where one of the blocks having an element from S_2 and the block having an element from S_3 have a conflicting bit common with the block(blocks) having an element from S_1. Without loss of generality say blocks having the element n_3 and n_5 have a conflicting bit common with the block having elements from S_1. Now here we can have two cases either blocks having elements n_3 and n_5 have a conflicting bit common with the same block having an element from S_1 or it have a conflicting bit common with different blocks having elements from S_1. Let us first consider the case where blocks having element n_3 and n_5 have a conflicting bit common with the same block having an element from S_1 say n_1. In this case, we send the blocks having elements n_3, n_4 and n_5 to table T_0, and rest of the blocks lying on the dotted lines containing these blocks to table T_1. Now we see whether the dotted line which contains block having element n_4 passes through block having element from S_1 or not. Without loss of generality let us say that dotted line which contains block having element n_4 passes through block having element n_2. In this case, we send the block having the element n_2 to table T_0, and rest of the block which lies on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. If the dotted line which contain block having element n_4 do not pass through block having element from S_1, then we send the blocks having elements from S_1 to table T_1, rest all the empty blocks to table T_0. Now we consider the case where blocks having elements n_3 and n_5 have a conflicting bit common with different blocks having elements from S_1. Without loss of generality let us say that block having element n_3 have a conflicting bit common with the block having the element n_1 and the block having the element n_5 have a conflicting bit common with the block having the element n_2. In this case also assignment made above will work.

Case 6.2.3 Only one block having an element from S_2 or S_3 have a conflicting bit common with the block having an element from S_1 in table T_1. Now here we can have two cases either the block having an element from S_2 have a conflicting bit common or the block having an element from S_3 have a conflicting bit common with the block having an element from S_1.

Case 6.2.3.1 The block having an element from S_2 have a conflicting bit common with a block having an element from S_1 in table T_1. Without loss of generality let us say that the block having the element n_3 conflicts with the block having the element n_1. Now we see the position of the blocks having the element n_4 and n_5. Let us first consider the case where blocks having elements n_4 and n_5 have a conflicting bit common.

If the blocks having elements n_4 and n_5 have a conflicting bit common on the dotted line which contains blocks having elements from S_1, then we send the blocks having elements n_3, n_4 and n_5 to table T_0, and all the blocks lying on the dotted lines containing these blocks to table T_1. We send the block having element n_1 and n_2 to table T_1. Rest all the empty blocks are sent to table T_0.

If blocks having elements n_4 and n_5 have a conflicting bit common outside the dotted line which contains blocks having elements from S_1, then we see the positions of the dotted lines which contains block having elements n_4 and n_5. Without loss of generality let us say that the dotted line which contains block having element n_5 passes through block having element n_2, and the dotted line which contains block having element n_4 passes through block having element n_1. In this case, we send the blocks having elements n_2, n_3 and n_5 to table T_0, and all the blocks lying on the dotted lines containing these blocks to table T_1. Further, we send the block having element n_4 to table T_1. Rest all the empty blocks are sent to table T_0. Now we consider the case where only one dotted line which contains block having element n_4 or n_5 passes through block having element from S_1. Without loss of generality let us say that block having element n_5 passes through block having element n_2. In this case, we send the blocks having elements n_2, n_3, n_4 and n_5 to table T_0, and rest all the blocks lying on the dotted lines containing these block to table T_1. Further, we send rest all the empty blocks to table T_0. If none of the dotted lines, which contains blocks having element n_4 or n_5 passes through block having elements from S_1, then we send the blocks having elements n_3, n_4 and n_5 to table T_0, and all the empty blocks lying on the dotted lines containing these blocks to table T_1. Further, we send the blocks having elements from S_1 to table T_1, and rest all the empty blocks are sent to table T_0.

Now we are left with the case where blocks having elements n_4 and n_5 do not have a conflicting bit common. This can have several sub-cases. Let us first consider the case where both the blocks having elements n_4 and n_5 passes through the dotted line which contains blocks having elements from S_1. In this case, we send the blocks having elements n_3, n_4 and n_5 to table T_0, and all the blocks lying on the dotted lines containing these blocks to table T_1. We send the blocks having elements n_1 and n_2 to table T_1. Rest all the empty blocks are sent to table T_1.

Now let us consider the case where n_4 and n_5 do not have a conflicting bit common with block lying on the dotted line which contains block having elements from S_1. In this case, we send the blocks having elements n_3, n_4 and n_5 to table T_1, and all the blocks lying on the dotted lines containing these blocks to table T_0. Block having the element n_1 is sent to table T_0, and rest all the blocks lying on the dotted line containing these blocks to table T_1. Rest all the empty blocks are sent to table T_0.

Now, we can also have a case where only one block having the element n_4 or n_5 have a conflicting bit common with block lying on the dotted line which contains blocks having elements from S_1. Let us first consider the case where block having the element n_4 have a conflicting bit common with the block lying on the dotted line which contains block having elements from S_1. In this case, we send the blocks having elements n_3, n_4 and n_5 to table T_0, and all other blocks lying on the dotted line containing these blocks to table T_1. Now, we see whether the block having elements from S_1 have a conflicting bit common with block lying on the dotted line which contains block having the element n_5. Without loss of generality let us say block having the element n_1 have a conflicting bit common with block lying on the dotted line which contains block having the element n_5. In this case, we send the block having the element n_1 to table T_0, and all other blocks lying on the dotted line which contains this block to table T_1. Rest all the empty blocks are sent to table T_1. On another hand, if the dotted line which contains block having the element n_5 do not pass through block having element from S_1, then we send the block having element from S_1 to table T_1, and all the blocks lying on the dotted line containing this block to table T_0. Rest all the empty blocks are sent to table T_0. Now let us consider the case where block having the element n_5 have a conflicting bit common with block lying on the dotted line which contains block having elements from S_1. In this case, we send the blocks having elements n_3, n_4 and n_5 to table T_0 and all other blocks lying on the dotted line containing these blocks to table T_1. Now we see whether the dotted line which contains block having element n_4 passes through block having element n_1 or n_2. Without loss of generality let us say that dotted line which contains block having element n_4 passes through block having element n_2. In this case, we send the block having element n_2 to table T_0, rest all the blocks lying on the dotted line which contains this block to table T_1. Rest all the empty blocks are sent to table T_0. If the dotted line which contains block having element n_4 do not pass through block having element from S_1, then we send the blocks having elements from S_1 to table T_1, and rest all the empty blocks to table T_0.

Now let us consider the case where none of the blocks having elements n_4 or n_5 conflict with the dotted line which contains block having elements from S_1. In this case, we send the blocks having elements n_3, n_4 and n_5 to table T_0, and all the blocks lying on the dotted lines containing these blocks to table T_1. Further, blocks having elements from S_1 are sent to table T_1, and rest all the empty blocks are sent to table T_0.

Case 6.2.3.2 The block having an element from S_3 have a conflicting bit common with a block having an element from S_1 in table T_1. Without loss of generality let us say that block having element n_5 have a conflicting bit common with the block having the element n_1. Now we see the position of the blocks having elements from S_2.

Both the blocks having an element from S_2 have a conflicting bit common with block lying on the dotted line which contains block having an element from S_1 in table T_1. In this case, we send the blocks having the elements n_3, n_4 and n_5 to table T_0 and all the blocks lying on the dotted lines containing these blocks to table T_1. Blocks having elements from S_1 are sent to table T_1 and rest all the empty blocks are sent to table T_0.

Both the blocks having an element from S_2 do not have a conflicting bit common with block lying on the dotted line which contains blocks having an element from S_1. Now we see whether the blocks having elements from S_2 conflicts with block having element n_5. Let us first consider the case where none of the blocks having elements from S_2 conflicts with block having element n_5. In this case, we send the blocks having elements n_3, n_4 and n_5 to table T_1, and all the empty blocks lying on the dotted lines containing these blocks to table T_0. Further, we send the block having the element n_1 to table T_0, and all the blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. Now let us consider the case where only one block having element from S_2 conflicts with block having element n_5. Without loss of generality let us say that block having element n_4 conflicts with block having element n_5. In this case, we send the blocks having elements n_3, n_4 and n_5 to table T_0, and all the blocks lying on the dotted lines containing these blocks to table T_1.Now we see whether the dotted line which contains block having element n_3 passes through block having element n_1 or n_2. Without loss of generality let us say that the dotted line which contains block having element n_3 passes through block having element n_2. In this case, we send the block having element n_2 to table T_0, all the blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. If the dotted line which contains block having element n_3 do not pass through block having element from S_1, then we send the block having element from S_1 to table T_1, rest all the empty blocks to table

T_0. Now we consider the case where both the blocks having elements from S_2 conflicts with block having element n_5. In this case, we send the blocks having elements n_3, n_4 and n_5 to table T_0, all the blocks lying on the dotted lines containing these blocks to table T_1. Further, we send the block having element n_2 to table T_0, all the blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. Only one block having the element from S_2 have a conflicting bit common with block lying on the dotted line which contains blocks having elements from S_1 in table T_1. Without loss of generality let us say that block having element n_3 have a conflicting bit common with block lying on the dotted line which contains block having an element from S_1. Now we see whether the block having element n_4 conflicts with block having element n_5. Let us first consider the case where blocks having element n_4 and n_5 conflicts. In this case, we send the blocks having elements n_1, n_2, n_3 and n_4 to table T_1, and all the empty blocks lying on the dotted lines which contain these blocks to table T_0. We send the blocks having the element n_5 to table T_0, and rest all the blocks lying on the dotted lines which contain these blocks to table T_1. Rest all the empty blocks are sent to table T_0. If the blocks having elements n_4 and n_5 do not conflict, then we send the blocks having elements n_3, n_4 and n_5 to table T_0, and all the empty blocks lying on the dotted line containing this block to table T_1. Now we see whether a block having element from S_1 conflicts with empty block lying on the dotted line which contains block having element n_4. Without loss of generality, let us say that block having element n_2 conflicts with empty block lying on the dotted line which contains block having element n_4. In this case, we send the block having element n_2 to table T_0, and rest all the blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. If none of the block having element from S_1 conflicts with empty block lying on the dotted line which contains block having element n_4, then we send the blocks having elements from S_1 to table T_1, and rest all the empty blocks to table T_0.

Case 6.2.4 None of the blocks having an element from S_2 or S_3 have a conflicting bit common with the block having elements from S_1 in table T_1. In this case, we can have either the block having the element n_5 have a conflicting bit common with the block having elements from S_2 or it does not.

Let us first consider the case where the block having the element n_5 have a conflicting bit common with one of the blocks having the element from S_2. Without loss of generality let us say that the block having the element n_5 have a conflicting bit common with the block having the element n_3. Now we see whether the block having the element n_4 have conflicting bit common with a block lying on the dotted line which contains block having elements from S_1 or not. Let us first consider the case where block having the element n_4 have conflicting bit common with block lying on the dotted line which contains block having an element from S_1. In this case, we send the block having the element n_3 to table T_1, and all the blocks lying on the dotted line containing this block to table T_0. We send the block having the element n_5 to table T_0, and all the blocks lying on the dotted line which contains this block to table T_1. Now we see whether any of the block having elements from S_1 have a conflicting bit common with block lying on the dotted line which contains element n_5 or not. Without loss of generality let us say that the block having the element n_1 have conflicting bit common with block lying on the dotted line which contains block having the element n_5. In this case, we send the blocks having the elements n_1 and n_4 to table T_0, and all other blocks lying on the dotted lines which contain these blocks to table T_1. Rest all the empty blocks are sent to table T_0. If none of the blocks having elements from S_1 have conflicting bit common with block lying on the dotted line which contains block having the element n_5, then we send the blocks having elements n_1, n_2 and n_4 to table T_1, and rest all the empty blocks to table T_0. Now let us consider the case where block having the element n_4 do not have conflicting bit common with block lying on the dotted line which contains block having elements from S_1. In this case, we see whether a block having element n_5 have conflicting bit common with a block having element n_4. Let us first consider the case where a block having element n_5 do not have conflicting bit common with a block having element n_4. In this case, we send the blocks having elements n_4 and n_5 to table T_1, and all the blocks lying on the dotted lines containing these blocks to table T_0. Further, we send the block having element n_3 to table T_0, and all the empty blocks lying on the dotted line containing this block to table T_1. Now, we see if either of the blocks having element n_1 or n_2 have conflicting bit common with block lying on the dotted line which contains block having element n_3. Without loss of generality, let us say block having element n_1 have conflicting bit common with block lying on the dotted line which contains block having element n_3. In this case, we send the block having element n_1 to table T_0, and all the blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. On the contrary, if neither of the block having element n_1 or n_2 has conflicting bit common with block lying on the dotted line which contains block having element n_3, then we send the block having an element from S_1 to table T_1, and rest all the empty to table T_0. We are now left with the case where block having element n_4 have conflicting bit common with a block having element n_5. In this case, we send the block having n_3 and n_4 to table T_1, and all the blocks lying on the dotted line containing this block to table T_0. Further, we send the block having element n_5 to table T_0, and all the empty blocks lying on the dotted line containing this block to table T_1. Now we see whether any of the block having elements from S_1 have a conflicting bit common with block lying on the dotted line which contains element n_5 or not. Without loss of generality let us say that the block having the element n_1 have conflicting bit common with block lying on the dotted line which contains block having the element n_5. In this case, we send the block having element n_1 to table T_0, and all the blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. If none of the blocks having elements from S_1 have conflicting bit common with block lying on the dotted line which contains block having the element n_5, then we send the blocks having elements from S_1 to table T_1 and rest all the empty blocks to table T_0.

Now we are left with the case where block having the element n_5 do not have a conflicting bit common with the block having an element from S_2. In this case, we send the blocks having elements to table T_1 and all the empty blocks to table T_0.

Case 6.3 All the elements belonging S_1, S_2 and S_3 lies on the different dotted line. In this case, we send the blocks having elements to table T_0, and all the empty blocks to table T_1.

Case 7 Two blocks having elements belonging to $S_1 = \{n_1, n_2\}$ lies in a same superblock and the element n_3, n_4 and n_5 to the different superblocks. If the blocks having elements from S_1 lies on the different dotted line, then assignment made in Case 5.3 can be used. So let us consider the case where blocks having the element from S_1 lies on the same dotted line.

Case 7.1 Only one block having element from n_3, n_4 and n_5 have conflicting bit common with the block having element from S_1. Without loss of generality let us say block having the element n_3 have conflicting bit common with the block having the element n_1.

Case 7.1.1 Blocks having elements n_4 and n_5 have conflicting bit common.

Case 7.1.1.1 Blocks having elements n_4 and n_5 have conflicting bit common with block lying on the dotted line which contains block having an element from S_1. In this case, we send the block having elements n_3, n_4 and n_5 to table T_0, and rest of the empty blocks lying on the dotted lines containing these blocks to table T_1. Blocks having the elements n_1 and n_2 are sent to table T_1 and rest all the empty blocks are sent to table T_0.

Case 7.1.1.2 Blocks having elements n_4 and n_5 does not have conflicting bit common with block lying on the dotted line, which contains block having an element from S_1. Now here we can have several cases depending upon whether the dotted lines which contain blocks having elements n_4 and n_5 passes through blocks having elements from S_1 or not.

Without loss of generality let us say that dotted line which contains block having the element n_4 passes through the block having the element n_1, and the dotted line which contain block having the element n_5 passes through the block having the element n_2. Now in this case we send the blocks having elements n_4 and n_1 to table T_1. Also, we send the blocks having elements n_3, n_5 and n_2 to table T_0, and rest of the blocks lying on the dotted lines containing these blocks to table T_1. We send rest of the empty blocks to table T_0.

Now consider a case where only one dotted line which contains a block having element n_4 or n_5 passes through the block having an element from S_1. Without loss of generality let us say that block having the element n_4 passes through the block having an element from S_1. Let us first consider the case where the dotted line which contains block having the element n_4 passes through the block having the element n_2. In this case, we send the blocks having elements n_2, n_3, n_4 and n_5 to table T_0, and rest of the blocks lying on the dotted lines containing these blocks to table T_1. Rest all the empty blocks are sent to table T_0. Now we can also have a case where block having the element n_4 passes through the block having the element n_1. In this case, we send the block having the elements n_1, n_3, n_4 and n_5 to table T_0, and rest of the blocks lying on the dotted lines containing these blocks to table T_1. Rest all the empty blocks are sent to table T_0. If none of the dotted lines which contains blocks having elements n_4 and n_5 passes through blocks having elements from S_1 then we send the blocks having elements n_3, n_4 and n_5 to table T_0, and all the blocks lying on the dotted lines containing these blocks to table T_1. Also, we send the blocks having elements from S_1 to table T_1. Rest all the empty blocks are sent to table T_0.

Case 7.1.2 Blocks having the elements n_4 and n_5 do not have conflicting bit common in table T_1 .

Case 7.1.2.1 Both the blocks having element n_4 and n_5 have conflicting bit common with block lying on the dotted line which contains block having an element from S_1. In this case, we send the blocks having elements n_3, n_4 and n_5 to table T_0, and rest of the empty blocks lying on the dotted line containing these blocks to table T_1. Also, we send the blocks having elements n_1 and n_2 to table T_1, and rest of the empty blocks to table T_0.

Case 7.1.2.2 One of the blocks having an element n_4 or n_5 have conflicting bit common with block lying on the dotted line which contains blocks having elements from S_1 and other do not have conflicting bit common. Without loss of generality let us say block having the element n_4 have conflicting bit common with block lying on the dotted line which contains blocks having elements from S_1, and block having the element n_5 lies outside it. In this case, we send the blocks having elements n_3, n_4 and n_5 to table T_0, and all the blocks lying on the dotted lines containing these blocks to table T_1. If any block having an element from S_1 have conflicting bit common with block lying on the dotted line which contains block having elements n_5, then we send that block to table T_0 all other blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. On another hand, if none of the blocks having elements from S_1 have conflicting bit common with block lying on the dotted line which contains block having elements n_5 then we send the block having elements from S_1 to table T_1, and rest all the empty blocks to table T_0.

Case 7.1.2.3 None of the blocks having elements n_4 or n_5 have conflicting bit common with block lying on the dotted line which contains blocks having elements from S_1. Now we see position of the block having element n_4 and n_5.

Let us first consider the case where blocks having elements n_4 and n_5 do not conflicts with block having element n_3. In this case, we send the blocks having elements n_2, n_3, n_4 and n_5 to table T_1. Further, we send the block having the element n_1 to table T_0 and the rest of the empty blocks lying on the dotted line which contains this block to table T_1. Rest all the empty blocks are sent to table T_0. If both the blocks having elements n_4 and n_5 conflicts with block having element n_3. In this case, we send the blocks having elements n_3, n_4 and n_5 to table T_0, and all the blocks lying on the dotted lines which contain these blocks to table T_1. Further, we send the block having element n_2 to table T_0, and all the blocks lying on the dotted line which contain this block to table T_1. Rest all the empty blocks are sent to table T_0. Now we are left with the case, where only one block having element n_4 or n_5 conflicts with block having element n_3. Without loss of generality let us say that block having element n_4 conflicts with block having element n_3. In this case, we send the block having element n_3 and n_5 to table T_0, and rest all the empty blocks lying on the dotted lines containing theses blocks to table T_1. We send the block having element n_4 to table T_1, and rest all the empty blocks lying on the dotted line containing this block to table T_0. Now we see whether the dotted line which contains block having element n_5 passes through block having element n_1 or n_2. Without loss of generality let us say that dotted line which contains block having element n_5 passes through block having element n_1. In this case, we send the block having element n_1 to table T_0, and rest all the blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. On the other hand, if the dotted line which contains block having element n_5 do not pass through block having element n_1 or n_2, then we send the blocks having elements from S_1 to table T_1, and rest all the empty blocks to table T_0.

Case 7.2 Two blocks having elements from n_3, n_4 or n_5 have conflicting bit common with the block having elements from S_1. Without loss of generality let us say that blocks having elements n_3 and n_4 have conflicting bit common with the block having elements from S_1. Now, here we can have two cases, either the blocks having elements n_3 and n_4 have conflicting bit common with the same block having an element from S_1 or it has conflicting bit common with the different blocks having elements from S_1.

Let us first consider the case where blocks having the element n_3 and n_4 have conflicting bit common with the different blocks having elements from S_1. Without loss of generality, let us say that block having element n_3 have conflicting bit common with the block having the element n_1 and the block having the element n_4 have conflicting bit common with the block having the element n_2. Now let us consider the intersection of the dotted line which contains block having the element n_5 from the blocks having elements from S_1 . Without loss of generality, let us say that the dotted line which contains block having element n_5 passes through the block having element n_1, in this case, we send the blocks having elements n_1, n_3, n_4 and n_5 to table T_0 and all other blocks lying on the dotted lines which contains these blocks to table T_1. Rest all the empty blocks are sent to table T_1. On the another hand, if the block having element n_5 do not pass through the blocks having elements from S_1, then we send the blocks having n_3, n_4 and n_5 to table T_0, and all other blocks lying on the dotted lines which contains these blocks to table T_1. Further, we send the blocks having elements from S_1 to table T_1 and rest all the empty blocks to table T_0.

Now we are left with the case where blocks having the element n_3 and n_4 have conflicting bit common with only one block having an element from S_1. Without loss of generality let us say that blocks having elements n_3 and n_4 have conflicting bit common with the block having the element n_1. In this case, we see the position of the block having the element n_5. If the block having the element n_5 have conflicting bit common with empty block lying on the dotted line having elements from S_1, then we send the block having the element n_3, n_4 and n_5 to table T_0, and all the empty blocks lying on the dotted lines containing these blocks to table T_1. We send the blocks having elements from S_1 to table T_1. Rest all the empty blocks are sent to table T_0. Further, if the block having the element n_5 do not have conflicting bit common with empty block lying on the dotted line which contains blocks having element from S_1 then we send the block having element n_3, n_4 and n_5 to table T_0, and all the empty blocks lying on the dotted lines containing these blocks to table T_1. Now we see whether the dotted line which contains block having the element n_5 passes through a block having an element from S_1 or not. Let us first consider the case where the dotted line which contains block having the element n_5 passes through a block having an element from S_1. Without loss of generality, let us say that the dotted line which contains block having the element n_5 passes through the block having the element n_1. In this case, we send the block having the element n_1 to table T_0, and all other blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. On another hand, if the dotted line which contains block having the element n_5 do not pass through the block having an element from S_1, then we send the blocks having the elements from S_1 to table T_1. Rest all the empty blocks are sent to table T_0.

Case 7.3 All the three blocks having elements n_3, n_4 and n_5 have conflicting bit common with the blocks having elements from S_1. In this case, we send the blocks having elements n_3, n_4 and n_5 to table T_0, and all the blocks lying on the dotted lines containing these blocks to table T_1. Blocks having elements from S_1 are sent to table T_1. Rest all the empty blocks are sent to table T_0.

Case 7.4 None of the blocks having elements from n_3, n_4 and n_5 have conflicting bit common with the blocks having elements from S_1.

Let us first consider the case where all the blocks having elements n_3, n_4 and n_5 have conflicting bit common. If all of them have conflicting bit common on the dotted line which contains blocks having elements from S_1 then we send the blocks having elements n_3, n_4 and n_5 to table T_0, and rest of the empty blocks lying on the dotted lines containing these blocks to table T_1. Also we send the blocks having elements n_1 and n_2 to table T_1. Rest all the empty blocks are sent to table T_0.

If the blocks having elements n_3, n_4 and n_5 have conflicting bit common outside the dotted line which contains blocks having elements from S_1, then we see the intersection of the dotted lines containing these blocks with the blocks having elements from S_1. Now since all the blocks having elements from n_3, n_4 and n_5 have conflicting bit common at most two dotted lines having blocks containing these elements can have conflicting bit common with the blocks having elements from S_1. Without loss of generality let us say that the dotted line which contains block having the element n_3 passes through the block having the element n_1 and the dotted line which contains block having the element n_4 passes through the block having the element n_2. In this case, we send the blocks having element n_1 and n_3 to table T_1. Also, we send the blocks having elements n_2, n_4 and n_5 to table T_0, and rest of the empty blocks lying on the dotted line containing these blocks to table T_1. Rest all the empty blocks are sent to table T_0.

Without loss of generality let us now consider a case where only one dotted line having element say n_3 passes through the block having element say n_1, then we send the blocks having elements n_1, n_2 and n_3 to table T_1. Also, we send the blocks having elements n_4 and n_5 to table T_0, and rest of the empty blocks lying on the dotted lines containing these blocks to table T_1.Rest all the empty blocks are sent to table T_0.

If none of the dotted lines containing blocks having elements n_3, n_4 and n_5 passes through the blocks having element from S_1 then we send the blocks having elements n_3, n_4 and n_5 to table T_0, and rest of the empty blocks lying on the dotted line containing these blocks to table T_1. Further, we send the blocks having element n_1 and n_2 to table T_1, and rest of the empty blocks to table T_0.

Now let us consider the case where only two blocks having elements from n_3, n_4 and n_5 have conflicting bit common. Without loss of generality let us say the blocks having elements n_3 and n_4 have conflicting bit common. Now we see the intersection of dotted lines containing the blocks n_3 and n_4. Let us first consider the case where both the dotted lines containing blocks n_3 and n_4 passes through the blocks having elements from S_1. Without loss of generality let us say the dotted line which contains block having the element n_3 passes through the block having the element n_1 and the dotted line which contains block having the element n_4 passes through the block having the element n_2. Now, we see the position of the block having element n_5. Let us first consider the case where block having n_5 have conflicting bit common with the block lying on the dotted line which contains block having an element from S_1. Further, let us consider that the dotted line which contains block having element n_5 intersects the block having element n_3 and n_4. Now, it is interesting to note that if two nonempty and an empty blocks intersect, then we can either send both nonempty blocks to table T_1, or we can send a nonempty block and an empty to table T_1. Let us first consider the case where we can send both the blocks having element n_3 and n_4 to table T_1. In this case, we send all the blocks having elements to table T_1, and all the empty blocks to table T_0. Now, without loss of generality let us consider that we can send the blocks having element n_3 and the empty block lying on the dotted line which contains block having element n_5 to table T_1. In this case, we send the block having element n_2, n_4 and n_5 to table T_0, and all the blocks lying on the dotted line containing this block to table T_1. Further, we send the block having element n_3 to table T_1. Rest all the empty blocks are sent to table T_0. Let us now consider the case where dotted line which contains block having element n_5 do not pass through both the block having element n_3 and n_4. Now since dotted line which contains block having element n_5 do not intersect with both the blocks having elements n_4 and n_5 it can conflict with at most one block having element. Without loss of generality, let us say that empty block lying on the dotted line which contains block having element n_5 either conflict with a block having element n_3 or it does not. In this case, we send the blocks having elements n_1, n_3 and n_5 to table T_0, and all the blocks lying on the dotted lines containing these blocks to table T_1. Further, we send the blocks having element n_2 and n_4 to table T_1. Rest all the empty blocks are sent to table T_0. Let us now consider the case where block having element n_5 do not intersect with the dotted line which contains block having elements from S_1. Now let us consider the case where block having element n_5 do not conflict with block lying on the dotted line having elements from S_1. In this case, we send the block having element n_5 to table T_1 and all the empty blocks lying on the dotted line containing this block to table T_0. Now we see if the block having element n_5 have conflicting bit common either with block lying on the dotted line which contains block having element n_3 or n_4. Without loss of generality, let us say block having element n_5 have conflicting bit common with empty block lying on the dotted line which contains block having element n_3. In this case, we send the block having element n_1 and n_3 to table T_1. Further, we send the block having element n_2 and n_4 to table T_0, and all the blocks lying on the dotted lines containing these blocks to table T_1. Rest all the empty blocks are sent to table T_0. If the block having element n_5, do not have conflicting bit common with block lying on the dotted line which contains block having element n_3 or n_4 then previous assignment works.

Let us consider the case where only one dotted line which has block having the element n_3 or n_4 passes through the block having elements from S_1. Without loss of generality let us say that dotted line which contains block having the element n_3 passes through the block having the element n_1. In this case, we see the position of the block having the element n_5. Let us first consider the case where block having the element n_5 have conflicting bit common with block lying on the dotted line which contains blocks having elements from S_1. In this case, we send the block having element n_1, n_3, n_4 and n_5 to table T_0, and all the blocks lying on the dotted lines containing these blocks to table T_1. Further, we send rest all the empty blocks to table T_0.

Now let us consider the case where the block having the element n_5 do not have conflicting bit common with block lying on the dotted line which contains blocks having elements from S_1. Now we see whether the block having element n_5 conflicts with the dotted line which contain block having element n_3 or n_4. If the block having element n_5 conflicts with block lying on the dotted line which contains block having element n_3, then we send the blocks having elements n_2, n_4 and n_5 to table T_0, rest all the blocks lying on the dotted lines containing these blocks to table T_1. Further, we send the block having element n_3 to table T_1, and rest all the empty blocks to table T_0. If the block having element n_5 conflicts with block lying on the dotted line which contains block having element n_4, then we send the blocks having elements n_4 and n_5 to table T_1, and rest all the blocks lying on the dotted lines containing these blocks to table T_1. Further, we send the blocks having elements n_1 and n_3 to table T_0, and all the blocks lying on the dotted line containing these blocks to table T_1. Rest all the empty blocks are sent to table T_0. If the block having element n_5 do not conflict with block lying on the dotted line which contains block having element n_3 or n_4, then we send the blocks having elements n_1, n_2, n_3 and n_5 to table T_1, and all the blocks lying on the dotted lines containing these blocks to table T_0. Further, we send the block having element n_4 to table T_0, and all the empty blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0.

If none of the dotted lines which contains block having elements n_3 or n_4 passes through block having element from S_1, then we send the blocks having elements n_3, n_4 and n_5 to table T_0, and all the empty blocks lying on the dotted lines containing these blocks to table T_1. Now either the block having an element from S_1 have conflicting bit common with block lying on the dotted line which contains block having the element n_5 or it does not. Let us first consider the case where block having elements from S_1 have conflicting bit common with block lying on the dotted line which contains block having the element n_5. Without loss of generality let us say that block having element n_1 have conflicting bit common with block lying on the dotted line which contains block having the element n_5. In this case, we send the block having the element n_1 to table T_0, and all the blocks lying on the dotted line containing this block to table T_1. Rest all the empty blocks are sent to table T_0. If none of the block having an element from S_1 has conflicting bit common with block lying on the dotted line which contains block having the element n_5, then we send the blocks having an element from S_1 to table T_1. Rest all the empty blocks are sent to table T_0.

Now we are left with the case where none of the blocks having elements n_3, n_4 and n_5 have conflicting bit common. In this case, we send the blocks having elements to table T_1, and all the empty blocks to table T_0.