CHAPTER III

PRIVACY PRESERVING ASSOCIATION RULE MINING IN CENTRALIZED DATABASE

Now a days many organizations utilize the results obtained from association rule mining process for their analysis purpose to make strategic decisions to take their long range plans in right way. Association rule mining process can be done easily when data providers and users are same. But in most of the organizations, database owners are willing to provide their database and its results to others who may be called as legitimate people or partners to get mutual benefits from them, so these people are also treated as users of the database. Even though the database owner willing to provide the database and mining results to legitimate people for the analysis purpose, the database owner do not wish to provide private data or information to any user as database may possess some private data/information. The leakage of sensitive data or information may degrade the performance and some times the owner may face bad situations in running the organization smoothly. So this makes the issue that how to provide the database to the partners to get mutual benefits and to maintain the goodwill while no single private data or information is revealed when mining is performed on the database. This takes the researchers in new direction of interest to develop techniques that incorporate privacy concerns when data/knowledge is sharing between the users. Many researchers proposed many methodologies to face the privacy issues while data/knowledge is shared between many people.

In centralized database environment, privacy preserving association rule mining can be achieved by determining a database called distorted database from the original database in such a way that when mining is performed on distorted database no sensitive rule related to any sensitive item set is revealed to the parties. In case of distributed database environment global association rules can be generated by finding global frequent item sets, supports from the partitioned databases using cryptographic techniques without disclosing any individuals private data or information.
The different approaches in centralized database environment to find privacy preserving association rule mining are as follows:

Basically there exist three broad categories of approaches for solving the privacy preserving association rule mining in centralized database environment which are heuristic based approach, border based approach, and exact approach. Heuristic approach guarantees to provide solution but also gives some side effects where as border based approach which is guaranteed to provide solution with less number of side effects compared to heuristic approach. No database owner is willing to accept side effects since it causes some damage to its own as well as to the partners as every database owner willing to provide accurate information (non sensitive) to their partners. Exact approach solves the problem of side effects but it takes more execution time to find optimum solution compared to the other two approaches. But some times the solution may not be found since the solution should satisfy all the privacy constraints completely, as correlation existing between items or item sets makes it difficult to provide solution by avoiding side effects completely. Depends on the user's convenience, tolerance of side effects and acceptance of execution time limit, suitable approaches are used. In this chapter, heuristic and exact approaches are considered for providing efficient solutions when data/information is shared between many people satisfying privacy constraints. The process of obtaining a solution is nothing but generating a database called distorted database or sanitized database and it is made by sanitizing the original database by performing some modifications based on some criteria’s over the transactions. This distorted database will be safe to release to their partners who can then use this for generating association rules.

3.1 Privacy Preserving Association Rule Mining

The process of obtaining the solution for association rule mining problem considering the privacy constraints is called association rule hiding problem. Association rule hiding problem can be specified as follows:
Given a database $D$, and a set of sensitive rules $SR$, the aim of Association rule hiding is to prepare a sanitized database $D'$ from $D$ such that when mining is performed on $D'$, all sensitive rules $SR$ will be hidden and only non sensitive rules will be disclosed.

On the whole there exist three types of side effects and which can be expressed in other way as association rule hiding goals. Association rule hiding problem finds solution by obtaining distorted database which can be provided to the partners safely and at the same time the distorted database should satisfy the following goals exactly if not possible or approximately for the sake of owners as well as partners. The goals of association rule hiding problem can be stated as follows:

- No sensitive rule is revealed from distorted database for the user specified minimum support threshold and user specified minimum confidence threshold value.

- No single non sensitive rule is hidden from the partners when they mine the distorted database for the user specified minimum support threshold and user specified minimum confidence threshold value.

- No single wrong or spurious rule is generated from distorted database based on user specified threshold values such as minimum support and minimum confidence thresholds.

Heuristic approaches have been getting focus of attention for majority of the researchers due to their efficiency, scalability and quick responses. However in some circumstances these heuristic based approaches suffer from undesirable side effects. So these side effects may degrade the performance of the hiding process of sensitive association rules. Heuristic approaches can be further subdivided into sensitive transaction identification methods, sensitive association clustering methods and sanitization matrix methods. The border based approaches utilizes the concept of borders to track the impact of altering transactions by greedy selecting those modifications while minimizing the side effects. These approaches focus on preserving the border of non sensitive
frequent item sets rather than considering all non sensitive item sets during sanitization process. Based on border revision concept, researchers proposed many algorithms such as BBA (Border Based Approach), Max–Min1 and Max–Min2 to hide sensitive association rules. Third class of approach is non heuristic called exact, which convert hiding process as constraint satisfaction problem. These problems are solved by integer programming. Compared to heuristic and border based, this guarantees quality for hiding sensitive information.

In this chapter, three different cases are considered in centralized for finding solution for hiding sensitive rules by satisfying association rule hiding goals exactly or approximately. The three cases are stated below:

Case 1: A methodology for finding privacy preserving association rule mining based on heuristic approach in centralized environment.

Case 2: An improved methodology of Inline algorithm for finding privacy preserving association rule mining in centralized environment by satisfying the association rule hiding goals.

Case 3: A partition based hybrid hiding methodology for finding privacy preserving association rule mining in centralized environment by satisfying association rule hiding goals completely for large size database.

The following section discusses the methodology proposed for finding privacy preserving association rule mining by applying heuristic strategies when knowledge is to be shared between many people.

3.2 Case 1

A methodology for finding privacy preserving association rule mining based on heuristic approach is considered as Case 1.

A novel method is proposed in this thesis work related to heuristic approach to hide sensitive association rules specified by the users with minimum side effects. Two criterions are suggested to identify the victim item and selecting suitable supporting transactions efficiently for sanitization
purpose. The functionality of the proposed method is illustrated with sample database by considering two cases related to existence of non overlapping and overlapping sensitive patterns. Especially in case of overlapping patterns, the Criteria1 and Criteria2 are useful to speed up the process of hiding sensitive item sets.

The Criteria1 specifies the efficient selection of victim item and Criteria2 helps to find the suitable supporting transactions for victim item in the sanitization process to minimize the side effects

**Criteria1:**

Suppose the item set \(<A_i, A_j>\) is to be hidden, one can select either \(A_i\) or \(A_j\) as victim item which minimizes the side effects. Victim item can be selected based on the following condition.

If number of times \(<A_i>\) appears in non sensitive frequent item set is greater than number of times \(<A_j>\) appears in non sensitive frequent item set then \(A_j\) be the victim item. If number of times \(<A_i>\) appears in non sensitive frequent item set is less than number of times \(<A_j>\) appears in non sensitive frequent item set then \(A_i\) be the victim item. If number of times \(<A_i>\) appears in non sensitive frequent item set is equal to number of times \(<A_j>\) appears in non sensitive frequent item sets then select \(A_i\) or \(A_j\) randomly as a victim item.

**Criteria2:**

After identifying the victim item to hide item set \(<A_i, A_j>\), the minimum number of suitable transactions has to be selected from all supporting transactions for the item set \(<A_i, A_j>\). The minimum number of transactions required to hide item set is based on the value of \(<A_i, A_j>.supp - MinTrans + 1\). Once the minimum number of transactions is identified then one has to identify suitable transactions in such a way that least number of side effects will occur over non sensitive frequent item sets. For each supporting transactions for item set \(<A_i, A_j>\), weight is computed by using the following:

\[
W(T_g) = \text{Number of dependant items with victim item} - \text{number of infrequent item sets associated with victim item.}
\]
Based on the weights of the transactions, the supporting transactions are sorted in ascending order and stored in MinT. The transactions in MinT are selected orderly for sanitization purpose to hide the sensitive item set \(<A_i, A_j>\).

The proposed method utilizes the above Criterion1 and Criteria2 to hide sensitive item sets with minimum side effects and which is discussed in the next section.

3.2.1. Proposed Algorithm

To find privacy preserving association rule mining for a given database and the set of sensitive item sets, a methodology which adopts the above two criterions 1 & 2 is proposed in this thesis work to obtain a distorted database which hides all sensitive item sets. In this methodology, split pattern technique which is taken from [74] is used. The authors in this thesis work suggested a procedure in which all the sensitive item sets whose length is greater than two are considered to find the pairs sub patterns. From this pair sub patterns only significant pair-sub patterns are considered as sensitive to hide the sensitive patterns. This procedure is very important in a way that it avoids the problem of forward inference attack. In order to avoid forward inference attack problem, at least one sub pattern with length of two of the patterns should be hidden. This split pattern procedure helps to speed up the hiding process. The terminology used in the proposed method is specified in the following Table 3.1.
<table>
<thead>
<tr>
<th>S.No.</th>
<th>Symbols</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DB = {t₁, t₂, ..., tₙ}</td>
<td>A original database consisting of N number of transactions</td>
</tr>
<tr>
<td>2</td>
<td>I = {i₁, i₂, ..., iₘ}</td>
<td>An item set of length M</td>
</tr>
<tr>
<td>3</td>
<td>Lₖ</td>
<td>An item set of length k</td>
</tr>
<tr>
<td>4</td>
<td>Tᵦₙm</td>
<td>The n&lt;sup&gt;th&lt;/sup&gt; transaction of m&lt;sup&gt;th&lt;/sup&gt; item</td>
</tr>
<tr>
<td>5</td>
<td>S = {s₁, s₂, ..., sₙ}</td>
<td>Set of sensitive item sets</td>
</tr>
<tr>
<td>6</td>
<td>MinSupport</td>
<td>User specified Minimum support threshold</td>
</tr>
<tr>
<td>7</td>
<td>Supp(J)</td>
<td>Number of transactions supporting item set J</td>
</tr>
<tr>
<td>8</td>
<td>MinTrans</td>
<td>Based on MinSupport, number of transactions required to support an item set to be frequent</td>
</tr>
<tr>
<td>9</td>
<td>MinConfidence</td>
<td>User specified Minimum confidence threshold</td>
</tr>
<tr>
<td>10</td>
<td>N</td>
<td>Size of original database, DB</td>
</tr>
<tr>
<td>11</td>
<td>F&lt;sub&gt;DB&lt;/sub&gt; = {L₁, L₂, L₃, ..., Lₖ}</td>
<td>A set consists of all frequent item sets</td>
</tr>
<tr>
<td>12</td>
<td>A → B</td>
<td>Association rule between item sets A and B</td>
</tr>
<tr>
<td>13</td>
<td>F₅</td>
<td>The set consisting of sensitive item sets</td>
</tr>
<tr>
<td>14</td>
<td>F&lt;sub&gt;NS&lt;/sub&gt;</td>
<td>The Set consisting of non sensitive frequent item sets</td>
</tr>
<tr>
<td>15</td>
<td>F&lt;sub&gt;2S&lt;/sub&gt;</td>
<td>The set consisting of pairs determined by the procedure split pattern.</td>
</tr>
<tr>
<td>16</td>
<td>&lt;Aᵢ, Aⱼ&gt;</td>
<td>The sensitive item set pair</td>
</tr>
<tr>
<td>17</td>
<td>T&lt;sub&gt;AᵢAⱼ&lt;/sub&gt;</td>
<td>Set of supporting transactions for item set &lt;Aᵢ, Aⱼ&gt;</td>
</tr>
<tr>
<td>18</td>
<td>DB'</td>
<td>Distorted database which hides all sensitive item sets.</td>
</tr>
<tr>
<td>19</td>
<td>Victim item</td>
<td>An item which is selected from the sensitive item pair which produces least side effects or no side effects when modification is done over it.</td>
</tr>
<tr>
<td>20</td>
<td>Victim transactions</td>
<td>Selected transactions to modify the victim item value.</td>
</tr>
<tr>
<td>21</td>
<td>MinT</td>
<td>A set consisting of suitable number transactions, which are to be modified to hide the sensitive item set</td>
</tr>
<tr>
<td>22</td>
<td>Count</td>
<td>Count gives number of times the victim item value has to be modified to hide sensitive item set pair.</td>
</tr>
<tr>
<td>23</td>
<td>W(T&lt;sub&gt;g&lt;/sub&gt;)</td>
<td>Weight for transaction T&lt;sub&gt;g&lt;/sub&gt;</td>
</tr>
</tbody>
</table>
3.2.2 Algorithm

The algorithm for the proposed methodology is specified as follows:

**Step 1** For a given database DB and set of sensitive item sets \( F_s \), generate frequent item sets and store with their support values in \( F_{DB} \).

**Step 2** Let the sensitive item sets are stored in \( F_s \) then the non sensitive frequent item sets are obtained by subtracting \( F_s \) from \( F_{DB} \).

i.e., \( F_{NS} = F_{DB} - F_s \).

**Step 3** If any item sets in \( F_s \) are having more than length of two, call the procedure split pattern to identify the prominent pairs which are to be hidden in order to hide all the item sets whose length is greater than two.

**Step 4** After step 3 a vector \( F_{2S} \) is prepared which consists of all two pair sensitive items.

**Step 5** The generated all pairs sensitive frequent item sets with their support values along with their supporting transactions ID’s are stored in a Table \( T_s \).

**Step 6** All the non sensitive frequent item sets that is \( F - F_{2S} \) are stored along with their support values in a Table \( T_{NS} \).

**Step 7** For each item set in \( F_{2S} \)

- If any non overlapping item set exists go to step 12.
- Else
  - The patterns \(<A_i, A_j><A_j, A_k>\) are chosen
  - Consider \( A_i \) or \( A_j \) as victim item based on Criteria1

**Step 8** Find the intersection of supporting transactions for \( A_iA_j \) and \( A_jA_k \) as follows:

\[ T_{AiAjAk} = T_{AiAj} \cap T_{AjAk} \]

**Step 9** Obtain the value for Count1 and Count2 as follows:

- Count1 for \( A_iA_j = <A_i, A_j>.\text{Supp} - \text{MinTrans} + 1 \)
- Count2 for \( A_jA_k = <A_j, A_k>.\text{Supp} - \text{MinTrans} + 1 \)
**Step 10**  Find minimum number of supporting transactions to be modified by applying Criteria2. From Count1 and Count2 select smaller one and that many transactions are selected from MinT and the victim item $A_j$ values are replaced with zero values. By performing this, item set which has lower count will be hidden. To hide the item set, which is having higher count value, Count1 – Count2 number of transactions which are not yet processed will be selected from MinT for sanitization. To hide this item set, the victim item set can be selected based on their dependencies with item sets in non sensitive item set $F_{NS}$. Accordingly the victim item value will be replaced with zero in the selected transactions. After performing this, the item set which is having higher count value is also hidden.

**Step 11**  Modify $F_{2S}$ by removing the pairs $<A_i,A_j>$ and $<A_j,A_k>$ from it. Go to step18.

**Step 12**  For the sensitive item set pair $<A_i,A_j>$ in $F_{2S}$ find victim item by using criteria 1.

**Step 13**  After identifying the victim item, find the supporting transactions for $<A_i,A_j>$.

**Step 14**  Obtain the value for Count1 and Count2 as follows:

Count1 for $A_i,A_j = <A_i,A_j>_\text{Supp} - \text{MinTrans} + 1$

**Step 15**  Select Count1 number of transactions to be modified from a set MinT which is obtained by Criteria2.

**Step 16**  The value of victim item in the selected transactions is replaced with value zero.

**Step 17**  Update $F_{2S}$ by removing $<A_i,A_j>$ from it.

**Step 18**  Repeat the above steps from step 7 until no more pair in the $F_{2S}$ to hide.

**Step 19**  Finally distorted database, $DB'$ is obtained in which all sensitive item sets in $F_{2S}$ are hidden.

**Step 20**  Stop the process.
3.2.3. Illustration of the Proposed Model with Sample Database

The proposed model is illustrated with sample database which consists of 5 attributes also called items for 8 transactions. Each transaction is represented by its TID value. The following Table 3.2 shows sample database.

Table 3.2: Sample Database

<table>
<thead>
<tr>
<th>TID\Item</th>
<th>A₁</th>
<th>A₂</th>
<th>A₃</th>
<th>A₄</th>
<th>A₅</th>
</tr>
</thead>
<tbody>
<tr>
<td>T₁</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>T₂</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T₃</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>T₄</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>T₅</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>T₆</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T₇</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T₈</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Using apriori algorithm, frequent item sets are generated for the sample database based on user specified minimum support threshold value 40% and the results are as follows:

\{A₁→3,A₂→3,A₃→4,A₄→5,A₅→4,\langle A₁,A₂\rangle→2,\langle A₁,A₄\rangle→2,\langle A₂,A₄\rangle→2,\langle A₂,A₅\rangle→2,\langle A₃,A₄\rangle→3,\langle A₄,A₅\rangle→3,\langle A₂,A₄,A₅\rangle→2\}

**Sub Case I: Non Overlapping Pair Patterns**

Assuming the sensitive item sets\langle A₁,A₄\rangle, \langle A₂,A₅\rangle, which are to be hidden.

Let \( F₅ = \{ \langle A₁,A₄\rangle, \langle A₂,A₅\rangle \} \)

By invoking split procedure, we get \( F_{2S} = \{ \langle A₁,A₄\rangle, \langle A₂,A₅\rangle \} \) Since the pairs in \( F_{2S} \) are non overlapping patterns, each pair has to be hidden individually.

Let us take the pair \langle A₁,A₄\rangle and find the victim item using Criteria1.

For item \langle A₁\rangle, One time appeared in non sensitive frequent item set
For item \(<A_4>\), In three non sensitive frequent item sets, \(A_4\) is appeared.

So victim item is \(A_1\).

Now we have to find suitable supporting transactions to change values of \(A_1\) in order to hide \(<A_1,A_4>\)

Supporting transactions for \(<A_1,A_4> = \{T_1,T_7\}\)

Minimum number of supporting transactions of \(<A_1,A_4>\) are to be determined to update \(A_1\) values

To determine suitable and minimum number of transactions, count value is determined as

\[
\text{Count} = <A_1,A_4>.\text{supp} - \text{MinTrans} + 1
\]

\[
= 2 - 2 + 1 = 1
\]

\[\therefore\] Minimum number of transactions required to update \(A_1\) is = 1

Now Criteria2 is applied to find which transaction is suitable to modify \(A_1\) so that side effects are minimized and shown as follows:

For \(T_1\), weight can be computed as

\[
W(T_1) = -1 \quad \because <A_1,A_4> \text{ is a non sensitive frequent item set.}
\]

For \(T_7\), weight can be computed as

\[
W(T_7) = +1-1=0 \quad \because <A_1,A_2> \text{ is non sensitive frequent item set and } <A_1,A_5> \text{ is infrequent item set.}
\]

Sort the transactions in ascending order based on weight but both has same weight so we can select any transaction to modify.

Sorted transactions is \{\(T_1,T_7\}\).

\(T_1\) is selected for modification to hide \(<A_1,A_4>\).

Replace the value of \(A_1\) i.e., 1 with zero value in transaction \(T_1\).

This modification affects the support value of item sets which are associated with item \(A_1\) in the modified transaction. So, the support of item set \((A_1, A_3)\) is decreased by one. After this modification, all the frequent item sets remains frequent and all the infrequent item sets remains infrequent except \((A_1,A_4)\) which is a sensitive item set is hidden.
Hence \( <A_1, A_4> \) pair is hidden and no side effects occurred.

Let us take the second pair \( <A_2, A_5> \) By using Criterion1, victim item is selected.  
For \( A_2 \), Two non sensitive item sets are associated with \( A_2 \).
For \( A_5 \), Only one non sensitive item set is associated with \( A_5 \).
\[
\therefore \text{victim item is } A_5.
\]
Once victim item is selected, the next task is to find minimum number of transactions required to modify victim item so that \( <A_2, A_5> \) is hidden with minimum side effects.

Supporting transactions for \( <A_2, A_5> = \{T_2, T_7\} \)

Minimum number of supporting transactions of \( <A_2, A_5> \) are to be determined to modify \( A_5 \).

To determine suitable and minimum number of transactions, count value is determined as
\[
\text{Count} = \frac{<A_2, A_5> \cdot \text{supp} \text{-- MinTrans + 1}}{}
\]
\[
= 2 - 2 + 1 = 1
\]
\[
\therefore \text{Minimum number of transactions required to update } A_5 = 1
\]

Now Criteria2 is applied to find which transaction is suitable to modify \( A_5 \) so that side effects are minimized and shown as follows:

For \( T_2 \), weight can be computed as
\[
W(T_2) = +1 \quad \therefore \text{<A_4, A_5> is a non sensitive frequent item set.}
\]
For \( T_7 \), weight can be computed as
\[
W(T_7) = +1-1 = 0 \quad \therefore \text{<A_4, A_5> is non sensitive frequent item set and} \<A_1, A_5> \text{ is infrequent item set}
\]

Sort the transactions in ascending order based on weight

Sorted transactions is \( \{T_7, T_2\} \). From this \( T_7 \) is selected for modification to hide \( <A_2, A_5> \)
Now modify the value of $A_5$ with zero value in transaction $T_7$ to hide $<A_2,A_5>$

In the process of hiding $<A_2,A_5>$ the value of $A_5$ is decreased in $T_7$ and due to the dependency, the item set $<A_4,A_5>$ support value is also decreased but the item set $<A_4,A_5>$ is still frequent.

Hence $<A_2,A_5>$ pair is hidden and no side effects occurred.

From the pairs of sensitive frequent item sets in $F_{2S}$, no side effects occurred.

By doing the above process, distorted database is obtained and shown in Table 3.3.

**Table 3.3: Distorted Database, DB' for DB**

<table>
<thead>
<tr>
<th>TID\Item</th>
<th>$A_1$</th>
<th>$A_2$</th>
<th>$A_3$</th>
<th>$A_4$</th>
<th>$A_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_1$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$T_2$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$T_3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$T_4$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$T_5$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$T_6$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$T_7$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$T_8$</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**Sub Case II: Overlapping Patterns**

Let the sensitive frequent item sets be $<A_2,A_4>$, $<A_4,A_5>$ and which are to be hidden.

Let $F_S = \{ <A_2,A_4>, <A_4,A_5> \}$ and find significant pairs of sensitive patterns by calling the split pattern procedure.

We get $F_{2S} = \{ <A_2, A_4>, <A_4, A_5> \}$ and it is clear that $A_4$ appeared in both these pairs.

$\therefore$ victim item is $<A_4>$
Supporting transactions for \( <A_2, A_4> \) is \( \{T_2, T_7\} \)

Supporting transactions for \( <A_4, A_5> \) is \( \{T_2, T_6, T_7\} \)

Find the common transactions by intersecting these two supporting transactions as

\[
\{T_2, T_7\} \cap \{T_2, T_6, T_7\} = \{T_2, T_7\}
\]

Now find minimum number of transactions required to hide these pairs by determining Count1 Count2 values as shown below:

\[
\text{Count 1} = \langle A_2, A_4 \rangle \text{.Supp } - \text{MinTrans} + 1 = 2 - 2 + 1 = 1
\]

\[
\text{Count 2} = \langle A_4, A_5 \rangle \text{.Supp } - \text{MinTrans} + 1 = 3 - 2 + 1 = 2
\]

\[
\therefore \text{The Count1 value specifies minimum number of transactions required to hide } <A_2, A_4> \text{ and which is one.}
\]

\[
\therefore \text{The Count2 value specifies minimum number of transactions required to hide } <A_4, A_5> \text{ and which is two.}
\]

As Count2 is greater than Count1, Count1 value that is one number of transactions are considered for modifications of \( A_4 \) value. So one suitable supporting transaction is required to modify \( <A_4> \) value and this transaction can be computed as

For \( T_2 \), Weight can be computed as

\[
W(T_2) \text{ for } A_4 \text{ is } 0
\]

For \( T_7 \), Weight can be computed as

\[
W(T_7) \text{ for } A_4 \text{ is } +1
\]

Sort the transactions in ascending order based on weight

Sorted transactions is \( \{T_2, T_7\} \). From this \( T_2 \) is selected for modification to hide \( <A_2, A_4> \) and \( <A_4, A_5> \).

Now modify \( A_4 = 0 \) at \( T_2 \).
This modification changes the support of item pair sets \(<A_2, A_4>\) and \(<A_4, A_5>\). The support is decreased by one and \(<A_2, A_4>\) is hidden but \(<A_4, A_5>\) is not hidden due to its support value is greater than Min Support threshold.

This causes \(<A_2, A_5>\) pair is hidden and no side effects occurred.

Since Count2 - Count1 is not zero, we have to find victim item using Criteria1 to hide item pair \(<A_4, A_5>\).

Since \(A_4 > A_5\) according to dependencies \(2 > 1\)

\[\therefore A_5 \text{ is the victim item.}\]

Supporting transactions = \(\{T_2, T_6, T_7\}\)

Minimum number of suitable transactions can be determined by using Criteria2 as:

\[W(T_2) = +1, W(T_6) = -1, W(T_7) = +1+1-1 = 1\]

Sorted transactions = \(\{T_6, T_2, T_7\}\)

From this set \(T_6\) is selected to modify \(A_5\) value to hide \(<A_4, A_5>\)

Now modify \(A_5 = 0\) in transaction \(T_6\) and causes no side effect.

By doing the above process, distorted database is obtained and shown in Table 3.4.
Hence the proposed methodology efficiently hides sensitive item sets without side effects even in the case of overlapping patterns situations.

### 3.2.4 Performance Analysis of the Proposed Methodology

- The main aim of the proposed methodology is to find a distorted database efficiently in the situations such as when overlapping patterns exist in the sensitive items sets, when non–overlapping patterns exist in the sensitive item sets.

- Generally a sensitive item set may consists of single item or it may consists more than one item.

- In heuristic approach, the efficiency of the algorithm to hide the sensitive item sets can be measured in terms of number of modifications which are required to hide the sensitive item sets with minimum side effects.

- When sensitive item consists of more than one item, the selection of most promising item (victim item) whose value can be considered for distortion is a crucial decision to hide the sensitive item sets. The suggested Criteria in the proposed methodology helps to find the victim item efficiently instead of random selection.
Another important aspect is heuristic approach in the selection of supporting transaction by victim item for modification purpose. The weight concept specified in Criteria2 helps to find the suitable supporting transactions which are to be considered for modification purpose in order to hide sensitive item sets in the process of minimizing side effects.

In the process of minimizing the number of modification to hide the sensitive item sets, in case of overlapping patterns which are exist in the sensitive item set, the suggested procedure efficiently finds the common victim item. This procedure helps to hide the overlapping pattern hidden at one step with minimum of modification over the victim item.

For each modification over the victim item, the related overlapping patterns support will be decreased simultaneously and with this, the overall execution time to hide overlapping patterns with minimum number of changes is minimized by reducing the number of modifications.

The experiments are conducted on synthetic dataset which consists of boolean transactions with 8 attributes. For comparison purpose, the algorithm specified in [60] is considered. Experiments are conducted for both existing and proposed algorithm on synthetic dataset. The variation of execution time for both existing and proposed methodology for various size databases are shown in the following graph.

![Figure 3.1 Existing Vs Heuristic Based Approach](image-url)
The above graph shows that the proposed methodology required less execution time when compared to existing algorithm in the process of hiding sensitive item sets. In the proposed heuristic based methodology, instead of random selection of victim items and supporting transactions, the Criteria1 and Criteria2 are suggested for finding victim item as well as supporting transactions for modification purpose. Because of this reason, the proposed methodology needs less execution time to hide the sensitive frequent item sets.

Hence the proposed heuristic based methodology efficiently hides the sensitive item sets.

The following section discusses the proposed methodology based on Inline algorithm to hide the sensitive item sets.

3.3 Case 2

A Methodology based on Inline algorithm for privacy preserving association rule mining is considered as Case 2.

Inline algorithm is an exact approach which adopts Binary Integer Programming (BIP) problem solving technique for finding optimum solution for hiding sensitive item sets without hiding any single non sensitive frequent item sets. The drawback with this approach is the developed algorithms designed may take several orders of magnitude slower than heuristic ones, especially due to the time that is taken by the implementation of integer programming logic to solve the optimization problem.

In this section, an algorithm which is a modified inline algorithm to hide sensitive frequent item sets to minimize side effects is proposed in the process of finding privacy preserving association rule mining. The Inline algorithm specified by the author Aris Gkoulalas et al, is considered for modification [83]. In this algorithm, the concepts such as border revision and divide and conquer are adopted. When the number of constraints are more, the constraint satisfaction problem (CSP) becomes more complex and the proposed algorithm solve this problem with the application of divide and conquer technique on constraints which reduces the overall time by solving each sub constraint satisfaction problem independently. The following section discusses the proposed methodology.
3.3.1 Proposed Methodology

When compared to heuristics and border based approaches, the inline algorithm provides high quality hiding solutions. However the inline algorithm mainly concentrates on hiding sensitive rules and maintains the frequent items as it is in the process of obtaining sanitized database. But it does not concentrate on preventing the generation of new frequent item sets and consequently there is a possibility of generating new rules. To overcome this problem, a modification of Inline algorithm is proposed in this thesis work which considers both positive and negative border sets and also to speed up the process, a divide and conquer procedure is applied on the constraints. Positive border set is a set consisting of item set which belongs to frequent item set list whose all proper supersets are infrequent. Similarly negative border set consists of item set which belongs to frequent item sets lists whose all proper subsets are frequent. The symbols, used in the proposed algorithm is specified in the Table 3.5.
<table>
<thead>
<tr>
<th>S.No</th>
<th>Symbols</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DB = {t_1, t_2, ..., t_N}</td>
<td>A original database consisting of N number of transactions</td>
</tr>
<tr>
<td>2</td>
<td>I = {i_1, i_2, ..., i_M}</td>
<td>An item set of length M</td>
</tr>
<tr>
<td>3</td>
<td>L_k</td>
<td>An item set of length k</td>
</tr>
<tr>
<td>4</td>
<td>T_{nm}</td>
<td>The n(^{th}) transaction of m(^{th}) item</td>
</tr>
<tr>
<td>5</td>
<td>S = {s_1, s_2, ..., s_r}</td>
<td>Set of sensitive item sets</td>
</tr>
<tr>
<td>6</td>
<td>Sup(J)</td>
<td>Number of transactions supporting item set J</td>
</tr>
<tr>
<td>7</td>
<td>MinSupport</td>
<td>Minimum support</td>
</tr>
<tr>
<td>8</td>
<td>MinConfidence</td>
<td>Minimum confidence</td>
</tr>
<tr>
<td>9</td>
<td>N</td>
<td>Size of original database, DB</td>
</tr>
<tr>
<td>10</td>
<td>F_{DB} = {L_1, L_2, L_3, ..., L_k}</td>
<td>A set consists of all frequent item sets</td>
</tr>
<tr>
<td>11</td>
<td>A \rightarrow B</td>
<td>Association rule between item sets A and B</td>
</tr>
<tr>
<td>12</td>
<td>AR</td>
<td>Set of association rules</td>
</tr>
<tr>
<td>13</td>
<td>S_{min}</td>
<td>Minimal number of sensitive item sets</td>
</tr>
<tr>
<td>14</td>
<td>S_{max}</td>
<td>Maximal number of sensitive item sets</td>
</tr>
<tr>
<td>15</td>
<td>F^{-}_{DB}</td>
<td>A set of frequent item sets in expected distorted database</td>
</tr>
<tr>
<td>16</td>
<td>DB^{-}</td>
<td>Distorted database</td>
</tr>
<tr>
<td>17</td>
<td>P_{BR}</td>
<td>Positive border set consisting of maximally frequent item sets</td>
</tr>
<tr>
<td>18</td>
<td>N_{BR}</td>
<td>Negative border set consisting of minimally infrequent item sets</td>
</tr>
<tr>
<td>19</td>
<td>CSP</td>
<td>Constraints Satisfaction Problem</td>
</tr>
<tr>
<td>20</td>
<td>(BIP) Binary integer programming</td>
<td>Binary integer programming is the problem of finding a binary vector (x) that minimizes a linear function (f^T x) subject to linear constraints: (\min x f^T x) such that (A \cdot x \leq b, Aeq \cdot x = beq, x ) binary.</td>
</tr>
<tr>
<td>21</td>
<td>Border set</td>
<td>It is the union of (P_{BR}) and (N_{BR}), identifies key item sets which separates all frequent patterns from their infrequent patterns</td>
</tr>
<tr>
<td>22</td>
<td>F'_{DBsort}</td>
<td>F'<em>{DBsort} is a set consisting of sorted elements of (F^{-}</em>{DB}), which is sorted by length of the item sets in descending order</td>
</tr>
</tbody>
</table>
This algorithm utilizes various procedures such as generation of frequent item sets using Apriori, generation of Positive Border Set and Negative Border Set, divide and conquer approach to divide the constraints based on the independent attributes and BIP program to solve the each sub optimization problem

3.3.2 Algorithm

**Input:** Database DB, Min-Support, Min-Confidence, S set of sensitive item sets

**Output:** Distorted database DB´

**Step 1** Using Apriori algorithm find frequent item sets $F_{DB}$ based on Min-Support for the database DB

**Step 2** For a given sensitive item set S, Find $S_{min}$ and $S_{max}$ by using the following formulas.

\[
S_{min} = \{ I \in S/ \forall J \subset I, J \not\in S \}
\]

\[
S_{max} = \{ I \in F_{DB}/ \exists J \in S_{min}, J \subset I \}
\]

**Step 3** Find $F^{'DB} = F_{DB} - S_{max}$

**Step 4** Find positive border set $P_{BR}$ by using Positive Border procedure

**Step 5** Find negative border set $N_{BR}$ by using Negative Border procedure

**Step 6** Find minimal number of constraints for positive border elements

**Step 7** Find minimal number of constraints for negative border elements

**Step 8** Combine the constraints computed in step 6 & 7.

**Step 9** Introduce unknown variables $U_{ij}$ in the sensitive item sets supporting transactions in the distorted database. Construct constraints for each positive border element where sum of support value must be greater than or equal to minimum support and also construct constraints based on elements of negative border whose support value must be less than Min-Support.

**Step 10** Apply divide and conquer procedure to divide the constraints based on their dependencies to form sub optimization problems
**Step 11** Using Binary Integer Programming solve each sub CSP to get values to the unknown variables $U_{ij}$.

**Step 12** Update the database based on solution obtained from CSP to get distorted database.

**Step 13** Generate association rules based on user specified minimum confidence by using the distorted database and verify whether the generated rules are related to all the non sensitive frequent item sets and also check the sensitive rules related to sensitive item sets are hidden or not.

**Step 14** Stop the process

**Procedure Positive-Border** // Procedure finds positive border set based on $F^\prime_{\text{DB}}$.

**Input**: $F^\prime_{\text{DB}}, S$

**Output**: $P_{\text{BR}}$ // Positive Border item sets

**Step 1** Call sort procedure for $F^\prime_{\text{DB}}$ to get $F^\prime_{\text{DBsort}}$

**Step 2** Create array count $[\text{size}(F^\prime_{\text{DB}}), 2]$ and initialize with zero

**Step 3** For each $k$ item set $f \in F^\prime_{\text{DBsort}}$ do

    For all $(k-1)$ item set, $q \in F^\prime_{\text{DBsort}}$ do

        If $q \subseteq f$

            $q.$count++

        End for loop

    End for loop

    For each $f \in F^\prime_{\text{DBsort}}$ do

        If $f.$count = 0

            add $f$ to $P_{\text{BR}}$

        End for loop


Step 4 Return Positive Border set $P_{BR}$

Procedure Sort( $F'_DB$) //Sorts the item sets in $F'_DB$ based on item set length

Input : $F'_DB$

Output : $F'_{DBsort}$ // The set consisting of sorted elements

Step 1 Apply insertion sort procedure to get decreased sorted list based on item set length

Step 2 Return the set $F'_{DBsort}$ consisting of sorted item sets

Procedure Negative-border // Finds negative border set based on $F'_DB$

Input : $F'_DB$

Output : $N_{BR}$

Step 1 Initialize $k = 1$

Step 2 For ($K = 1; L_k \neq \emptyset ; k++$) do

If ($k = 1$)

For each item set $x \in L_1$ do

If $x \notin F'_D$

add $x$ to $N_{BR}$

End for loop

Else

if ( $k = 2$ )

For each item set $x \in L_1$ do

For each item set $y \in L_1$ do

$Z = \text{join} (x, y)$

If $((x < y) \text{ and } z \notin F'_D$

add $z$ to $N_{BR}$

End for loop

End for loop

End for loop
Else

For each $x \in L_{k-1}$

For each $y \in L_{k-1}$

If ($(x_1 = y_1) \land ... \land (x_{k-1} < y_{k-1})$

$Z = \text{join}(x, y)$

If ($z \notin F'_DB, \exists r_{k-1} \subset Z; r_{k-1} \notin F'_DB)$

add $Z$ to $N_{BR}$

Endif

Endif

Endfor

Endfor

Endif

Endfor

Step 3 Return Negative Border Set $N_{BR}$

Procedure (GenAssRuls)

//Generates association rules using frequent item set based on users specified minimum confidence threshold

Input : $F'_DB$, Min-Confidence

Output: Association rules (AR)

Step 1 Initialize $j$ with 2;

Step 2 Create a file, Rule-set consisting of three fields rule-id, antdnt, consqnt ;

Step 3 For each item set $L_j$ in $F'_DB$

Generate proper subsets of $L_j$ and store in set $S_{SUB}$

// $S_{SUB} = \{S_1, S_2, \ldots, S_m\}$ where $m$ denotes the number of proper subsets

Initialize $i$ with one
While (i <= m) do

read S_i from S_{SUB}

antdnt = S_i and consqnt = L_j - S_i

confd = sup (antdnt, consqnt) / sup(antdnt)

if (confd >= MinConfidence)

    add (rule-id, antdnt, consqnt, confd) to AR

    i++

End while loop

End for loop

Step 3 Return a set AR consisting of all association rules

3.3.3 Illustration of the Proposed Method

By taking the following sample database consisting of 5 attributes, named A_1, A_2, A_3, A_4, A_5 and 8 transactions. Each transactions has its own identification number called T-ID where T_i denotes i^{th} transaction. If a transaction supports a particular item then the corresponding value will be one otherwise it is zero. The sample database is given in Table 3.6.

Table 3.6: Sample Database

<table>
<thead>
<tr>
<th>Items/Transactions</th>
<th>A_1</th>
<th>A_2</th>
<th>A_3</th>
<th>A_4</th>
<th>A_5</th>
</tr>
</thead>
<tbody>
<tr>
<td>T_1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>T_2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T_3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>T_4</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>T_5</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>T_6</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T_7</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T_8</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SUPPORT</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>4</td>
</tr>
</tbody>
</table>

Let S be the Sensitive item set, S = \{A_3, \{A_2, A_4\}\}
The first step of this algorithm is generating frequent item sets based on Min-Support, 25% and then save these results in FDB which is specified below.

\[ F_{DB} = \{ A_1, A_2, A_3, A_4, A_5, <A_1, A_2>, <A_1, A_4>, <A_2, A_4>, <A_2, A_5>, <A_3, A_4>, <A_4, A_5>, <A_2, A_4, A_5> \} \]

The frequent item sets along with their support specified as follows:

\[ \{ A_1 \rightarrow 3, A_2 \rightarrow 3, A_3 \rightarrow 4, A_4 \rightarrow 5, A_5 \rightarrow 4, <A_1, A_2> \rightarrow 2, <A_1, A_4> \rightarrow 2, <A_2, A_4> \rightarrow 2, <A_2, A_5> \rightarrow 2, <A_3, A_4> \rightarrow 3, <A_4, A_5> \rightarrow 3, <A_2, A_4, A_5> \rightarrow 2 \} \]

Based on sensitive item sets, minimal sensitive item set (S_{min}) and maximal sensitive item sets (S_{max}), the solution can be determined as shown below:

\[ S_{min} = \{ <A_3>, <A_2, A_4> \} \]

\[ F_{DB} = \{ A_1, A_2, A_4, A_5, <A_1, A_2>, <A_1, A_4>, <A_2, A_5>, <A_4, A_5> \} \]

Using these set of values, maximally frequent item sets that is Positive borderset (P_{BR}) and minimally infrequent item set that is negative border set (N_{BR}) are determined by calling corresponding procedures and are shown here.

\[ P_{BR} = \{ <A_4, A_5>, <A_1, A_2>, <A_1, A_4>, <A_2, A_5> \} \]

\[ N_{BR} = \{ A_3, <A_1, A_5>, <A_2, A_4> \} \]

Border set is formed by combining both positive border (P_{BR}) and negative border (N_{BR}) sets and is given here.

Border set = \{ <A_4, A_5>, <A_1, A_2>, <A_1, A_4>, <A_2, A_5>, A_3, <A_1, A_5>, <A_2, A_4> \}

For each sensitive item set, unknown variables are introduced for sensitive item sets of all transactions in the database are specified in the Table 3.7. The problem is to find values for unknown variables such that no sensitive rule is revealed, no non sensitive rule is hidden and no wrong rule is to be generated when competitors mine the database.
Table 3.7: Distorted Database with Unknown Variables

<table>
<thead>
<tr>
<th>Items/Transactions</th>
<th>A₁</th>
<th>A₂</th>
<th>A₃</th>
<th>A₄</th>
<th>A₅</th>
</tr>
</thead>
<tbody>
<tr>
<td>T₁</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>T₂</td>
<td>0</td>
<td>U₂₂</td>
<td>0</td>
<td>U₂₄</td>
<td>1</td>
</tr>
<tr>
<td>T₃</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>T₄</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>T₅</td>
<td>0</td>
<td>0</td>
<td>U₅₃</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>T₆</td>
<td>0</td>
<td>0</td>
<td>U₆₃</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T₇</td>
<td>1</td>
<td>U₇₂</td>
<td>0</td>
<td>U₇₄</td>
<td>1</td>
</tr>
<tr>
<td>T₈</td>
<td>0</td>
<td>0</td>
<td>U₈₃</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SUPPORT</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>4</td>
</tr>
</tbody>
</table>

Using the above Table 3.7, constraints can be found for each element of positive and negative border sets.

Constraints are formed from positive border item sets are as follows:

For item set <A₄, A₅>, the constraint is U₂₄ +1+U₇₄ ≥ 2

For item set <A₁, A₂>, the constraint is 1 + U₇₂ ≥ 2

For item set <A₁, A₄>, the constraint is 1+U₇₄ ≥ 2

For item set <A₂, A₅>, the constraint is U₂₂ +U₇₂ ≥ 2

With respect to negative border set, constraints are formed as follows.

For item set <A₃>, the constraint is 1+U₅₃ + U₆₃ + U₈₃ < 2

For item set <A₁, A₅>, the constraint is 1 < 2

For item set <A₂, A₄>, the constraint is U₂₂ U₂₄ + U₇₂ U₇₄ < 2

The constraints can be simplified by evaluating constants and are shown here.

U₂₄ +1+U₇₄ ≥ 1

U₇₂ ≥ 1 it implies U₇₂ = 1 no need to consider it as a constraint.
U_{74} \geq 1 \text{ implies } U_{72} = 1 \text{ no need to consider it as a constraint.}

U_{22} + U_{72} \geq 2

U_{53} + U_{63} + U_{83} < 1

U_{22} U_{24} + U_{72} U_{74} < 2

Only the constraint related to item <A_3> is independent to other constraints. So the above CSP can be further subdivided in to two sub CSP problems and each problem is solved simultaneously so that overall time complexity can be reduced.

**Sub Problem 1:**

Min \ Z_1 = U_{22} + U_{24} + U_{72} + U_{74}

Subject to the constraints

\[ U_{24} + 1 + U_{74} \geq 1 \]

\[ U_{22} + U_{72} \geq 2 \]

\[ U_{22} U_{24} + U_{72} U_{74} < 2 \]

\[ U_{22}, U_{24}, U_{72}, U_{74} \text{ takes either 0 or 1.} \]

**Sub Problem 2:**

Min \ Z_2 = U_{53} + U_{63} + U_{83}

Subject to the constraints

\[ U_{53} + U_{63} + U_{83} < 1 \]

\[ U_{53}, U_{63}, U_{83} \text{ takes either 0 or 1} \]

The above are simplified constraints however the last constraint in Sub Problem 1 consists of product of two variables and can not be solved by BIP because it accepts linear variables only. Using constraint degree reduction approach this constraint can be converted into a constraint of single variables but introduces 7 additional constraints and illustrated here.

Let \( \Psi_1, \Psi_2 \), be the two new temporary binary variables which replaces product of variables as

\( \Psi_1 = U_{22} U_{24} \) \text{ and } \( \Psi_2 = U_{72} U_{74} \)
By this process the constraint which consist of two product variables become linear. The resulting CSP is presented as follows:

\[
\begin{align*}
\text{Min} & \quad U_{22} + U_{24} + U_{72} + U_{74} \\
\text{Subject to the constraints} & \\
U_{24} + 1 + U_{74} & \geq 2 \\
U_{22} + U_{72} & \geq 2 \\
\Psi_1 & \leq U_{22} \quad \Psi_1 \leq U_{24} \\
\Psi_1 & \geq U_{22} + U_{24} + 2 - 1 \\
\Psi_2 & \leq U_{72} \quad \Psi_2 \leq U_{74} \\
\Psi_2 & \geq U_{72} + U_{74} + 2 - 1 \\
\Psi_1 + \Psi_2 & < 2
\end{align*}
\]
Where \( U_{22}, U_{24}, \Psi_1, \Psi_2 \) takes values either 0 or 1.

The solution of the CSP leads to optimal hiding solution for the Sub Problem 1 is presented as \( U_{22}=1, U_{24}=0, U_{72}=1, U_{74}=1 \)

The solution of the CSP leads to optimal hiding solution for Sub Problem 2 is presented as \( U_{53}=0, U_{63}=0, U_{83}=0, \)

Replace every unknown variable with corresponding value obtained in Sub Problem 1 and Sub Problem 2 to get distorted database and is specified in the following Table 3.8.

**Table 3.8: Distorted Database**

<table>
<thead>
<tr>
<th>Items/Transactions</th>
<th>A_1</th>
<th>A_2</th>
<th>A_3</th>
<th>A_4</th>
<th>A_5</th>
</tr>
</thead>
<tbody>
<tr>
<td>T_1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>T_2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>T_3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>T_4</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>T_5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>T_6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T_7</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T_8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SUPPORT</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>

The list of the rules generated from the distorted database are shown in the following Table 3.9.
Table 3.9: Rules Obtained From Distorted Database

<table>
<thead>
<tr>
<th>Rule Set</th>
<th>Rule</th>
<th>Confidence</th>
</tr>
</thead>
<tbody>
<tr>
<td>(&lt;A_1, A_2&gt;)</td>
<td>(A_1 \rightarrow A_2 = 66%)</td>
<td></td>
</tr>
<tr>
<td>(&lt;A_1, A_4&gt;)</td>
<td>(A_1 \rightarrow A_4 = 66%)</td>
<td></td>
</tr>
<tr>
<td>(&lt;A_2, A_5&gt;)</td>
<td>(A_2 \rightarrow A_5 = 66%)</td>
<td></td>
</tr>
<tr>
<td>(&lt;A_4, A_5&gt;)</td>
<td>(A_4 \rightarrow A_5 = 50%)</td>
<td></td>
</tr>
</tbody>
</table>

Total number of rules = 8

3.3.4 Performance Analysis of the Modified Inline Algorithm

This experiment shows that sensitive association rule hiding problem can be solved efficiently and easily with this methodology. By observation optimum solution hides all the sensitive item sets and in turn hides all the specified sensitive association rules. The proposed algorithm finds easily positive border set and negative border set for any given sensitive item sets. The minimum number of constraints based on these two sets is formulated to transform association rule hiding problem into optimization problem. This optimization problem consists of minimization of objective function and constraints which can be solved by using BIP. This BIP technique is also guaranteed to provide solution for the formulated CSP problem and also provides optimum solution.

From the experimental results, it is observed that all the non sensitive item sets are not hidden and also all the non sensitive rules are accessible to any competitor. The algorithm also solves the problem when the constraint set consisting of product of variables which can be converted in to binary variables easily by using constraints reduction approach. The proposed Inline algorithm which adopts border revision concept finds suitable constraints to find optimum solution. The positive border is used to form constraints which specify the item sets that are not to be hidden and the negative border shows constraints for
sensitive item sets and some infrequent items which are in border to be hidden when mining is performed. By using the positive border concept, the frequent item sets which are in the border (Support value closed to minimum support value) remains frequent as the constraints are formed for each element in the positive borders set. In the same way using the negative border concept the infrequent item sets which are in the border remains infrequent as the constraints are formed for each element in the negative border set. By adopting $P_{BR}$ and $N_{BR}$, the proposed modified Inline algorithm satisfies the goals of association rule hiding in the process of hiding sensitive item sets.

The rules obtained from distorted database is given in Table 3.9 proved that goals are achieved successfully that is all sensitive rules are hidden, all frequent item sets are also frequent in distorted database and no new frequent item set is generated. So any competitor can perform mining process to get association rules on the distorted database which is received from the owner. They can use these rules to make analysis to improve their performance of the system and to make some strategic decisions.

The efficiency of this methodology is further enhanced by adopting divide and conquer technique which is applied on constraints. When the database size is increased, and when the number of sensitive item sets are more, solving the CSP problem becomes more complex. Dividing the CSP problem into sub CSP problems based on the constraints will be helpful to overcome this problem. Each sub problem can be solved independently with ease computations. The solution of each sub problems are combined to get the final solution. In this way database owner can share accurate knowledge to competitors by hiding sensitive item sets.

In the proposed methodology, if all the constraints are linear then solving CSP problem is simple where as if some of the constraints are non linear then constrain degree reduction approach can be applied to convert them into linear and which leads to more number of constraints. Even then the solution can be obtained but it requires more computation time.
Experiments are conducted for both existing Inline and modified Inline algorithm on synthetic dataset which consist of boolean transactions with 8 attributes. For comparison purpose, the existing algorithm introduced in [85] is considered. The following graph shows the variation of execution time for both existing and proposed methodology with various size databases.

![Performance Gain in Synthetic Data](image)

**Figure 3.2 Inline Vs Modified Inline Algorithm**

The graph shows that efficiency of the proposed algorithm is more when compared to existing Inline algorithm. The proposed methodology which utilizes parallel concepts and adopts divide and conquer strategy by dividing the large size constraint satisfaction problem into small sub constraint satisfaction problems and executes these sub constraint satisfaction problem in parallel and individually. Hence the modified Inline algorithm requires less overall execution time when compared to existing considered Inline algorithm.

### 3.4 Case 3

In this thesis work, partition based hybrid hiding methodology is proposed and considered as Case3.

This section proposes a partition based hybrid hiding methodology for large databases which is considered as Case3. Now a days large size database can be organized easily and efficiently by utilizing sophisticated technology like economy based fast and high performance communication facilities, availability
of high configuration hardware with affordable cost, sophisticated tools & algorithms. Association rule mining technique usually employ over large databases since large database may provide efficient results for the analysis. This technique is receiving more attention from researchers since its usage in diverse fields is becoming more. Finding uncovered information from large database is still a challenging problem when knowledge is shared between many legitimate people who are not having privilege to access sensitive information when the sensitive knowledge exists in the database is to be protected. Association rule mining usually employs over large database to do analysis efficiently but protecting the private information without side effects is the major issue. Exact approach can answer this problem by providing an optimum solution which means a solution is obtained without producing any side effects to the owner as well as partners/users.

As the database size is sufficient enough, and then an optimum solution can be determined without causing any damage to any user or owner in knowledge sharing problem. But mostly association rule mining is taking large size database, so finding optimum solution without causing damage to any individual is a difficult task since optimum solution is obtained by exact approach which utilizes Binary Integer programming (BIP) problem solving method. The large size database may create many constraints with many unknown variables which make it difficult to get the optimum solution using BIP and sometimes it may lead to unsolvable problem. To avoid the unsolvable situation for large size database, divide and conquer strategy can be applied to partition the databases to find optimum solution. This method is called partition based hybrid hiding methodology.

The proposed methodology utilizes the concept of hybrid hiding algorithm which is introduced by the other Aris Ghoukulalas et al in [85], and enhances the efficiency by incorporating divide and conquer strategy on the original database.

The first task in divide-and-conquer strategy is to divide the database into two databases of equal size. The original database will be at level $L_0$ and its partitioned databases will be at level $L_2$. If partitioned database size is not small
enough to solve the problems then partition each partitioned database further into two and this task will be repeated until getting the databases whose sizes are sufficient to solve the problems individually & easily.

![Figure 3.3: Binary Tree Structure Represents Partitioned Databases of Height 2](image)

This partition can be viewed as binary tree structure in which the root node represents the original database at level 0. The two partitioned databases can be obtained by partitioning the original database which are at level 1 and these can be called as children of the parent that is original database is parent of these databases. Again when these databases are further partitioned into two, they will become parent of the generated partitions.

The process of finding solution begins from the bottom most level where leaf nodes (databases) exists. At this level each leaf node solution will be computed by applying hybrid hiding algorithm. Once the solutions for each leaf node is obtained, every two leaf nodes which are partitioned from the same parent database, provides the solution for its parent by combining the solution of their databases. In this way, at each level, level i of the tree, the solutions of the two siblings are combined to get the solution of its parent which is at level i – 1. Finally the original database solution can be obtained by combining the solution of its children databases. This final solution of the original database is nothing but a distorted database D’ and it is safe to release D’ to the partners since all the sensitive item sets are hidden completely.
The strategy of divide and conquer will be more helpful to solve large size database problems. The significance of database partitioning method is any large size problem can be solved very easily and quickly. The partitioned databases which are at bottom level finds the solution for their databases individually in parallel. This reduces the overall computation time and moreover finding solution for small databases is very easy when compared to large size database. There will not be a situation where problem is unsolvable and also guarantees optimum solution for every partitioned database. The solution for a whole database for hiding sensitive association rules will be done more easily by simply merging successive sub problems solutions. The algorithm is specified in the next section.

3.4.1 Algorithm for Partition Based Hybrid Hiding Methodology

**Input:** Database (DB), MinSupport, MinConfidence, Sensitive item set  
**Output:** Distorted Database (DB’)

**Step 1** Partition the given database into two equal sub partitions DB$_1$ & DB$_2$ such that the partitioned database can be solved easily and guarantee to provide solution and if it is not small enough to solve, each partitioned database is partitioned further. The partitioning process is repeated until sufficient size databases are obtained.

**Step 2** During the partitioning process, at any stage, if the database size is not even then add dummy record/records so that it can be partitioned equally. Based on the MinSupport and database size, for each partition database a new minimum support value is computed.

**Step 3** For each partitioned databases at bottom level (leaf nodes)

For each item set in sensitive item set

If (support count of sensitive item set < |DB$_i$| * MinSupport)  
Continue

Else

If (support count of sensitive item set ≥ |DB$_i$| * MinSupport then

Call Hybrid-Hiding procedure
The distorted database will be obtained by appending the extended database to the partitioned database

End for loop

//Distorted database is obtained by assigning original partitioned database to it

**Step 4** Count = n-1 // start with level_{n-1}

**Step 5** Repeat the following steps until count = -1

For each node at level_{count}

Combine the solutions obtained from its children nodes to get the solution for the parent node

Endfor

Count --

**Step 6** Distorted database D´ is obtained at the root node which is the final solution for a given large database D.

**Step 7** Call the procedure GenAssRules to generate association rules for the distorted databases(DB´).

**Step 8** Stop the process.

The following procedure finds optimum solution to hide sensitive item sets without side effects.

**Hybrid-Hiding procedure**

**Input**: Database (DB), MinSupport, Sensitive Item set S

**Output**: Association rules AR, Distorted database DB´

**Step 1** Find frequent item sets by using apriori algorithm based on MinSupport threshold

**Step 2** Find \( S_{\text{min}}, S_{\text{max}}, F'_{\text{DB}} \) by using the formula

\[
S_{\text{min}} = \{ i \in S / \forall J \subset i, J \notin S \}
\]

\[
S_{\text{max}} = \{ i \in F_{\text{DB}} / \exists J \in S_{\text{min}}, J \subset i \}
\]

\[
F'_{\text{DB}} = F_{\text{DB}} - S_{\text{max}}
\]
Step 3 Find positive border set \( P_{BR} \) using the Positive-Border procedure

Step 4 Find negative border set \( N_{BR} \) using the Negative-Border procedure

Step 5 Combine revised positive border and negative border to get border item sets.

Step 6 Find size of extended database \( DB_x \), by using this formula

\[
SIZE_x = \left( \frac{\text{Support}(\text{Large size sensitive item set})}{\text{Freq}} \cdot N \right) + 1 \text{ where } \\
N = | DB |, \text{ Freq} = \text{MinSupport} / N
\]

Step 7 If \( SIZE_x \) will not provide sufficient number of transactions to satisfy three goals then few more transactions are required to satisfy goals. For this purpose, size of database must increase to a sufficient level by adding safety margin to \( SIZE_x \)

Step 8 Introduce unknown variables \( U_{ij} \) in each position in the extended database.

Step 9 Construct constraints for each item set in the positive border set using unknown variables based on actual support and MinSupport. Similarly construct constraints for each item set in the negative border set using unknown variables based on actual support and MinSupport. In this step, the hiding process problem is formulated as an optimization problem in which distance is the optimization criterion and is shown below.

The objective of the hiding algorithm can be achieved by appropriately setting the \( U_{qm} \) variables such that all sensitive knowledge is hidden while the distance is minimized. The optimization problem can be stated as follows:

Objective function

\[
\text{Minimize} \left( \sum_{q=1}^{Size_x} \sum_{m=1,M} u_{qm} \right)
\]

Subject to the constraints

\[
\left\{ \left( \sum_{q=1}^{Size_x} \prod_{i \in I} u_{qm} < \text{thr}, \forall I \in P_{BR} \right) \right\}
\]

\[
\left\{ \left( \sum_{q=1}^{Size_x} \prod_{i \in I} u_{qm} \geq \text{thr}, \forall I \in N_{BR} \right) \right\}
\]

Where \( thr = \text{Freq} \times (N + SIZE_x) - \text{Support (I, DB)} \)
Step 10 The optimization may consist of non linear variables in the constraints which can not be solved by binary Integer Programming. So to get the solution, every non linear variable can be converted in to linear variable by using constraints degree reduction approach.

Step 11 Call Binary Integer Programming to solve the formulated CSP to get values for unknown variables.

Step 12 Based on the solution obtained from CSP, the $U_{ij}$ variables are replaced with values of $U_{ij}$ to obtain a modified extended database. This database is called distorted database DB’

Step 13 Return distorted data base DB’

Procedure Positive-Border // The following procedure finds positive border set.

Input: $F^{'DB}$, S

Output: $P_{BR}$ // Positive Border item sets

Step 1 Call sort procedure for $F^{'DB}$ to get $F^{'DB}_{sort}$

Step 2 Create array count [size($F^{'DB}$),2] and initialize with zero

Step 3 For each k item set $f \in F^{'DB}_{sort}$ do

For all (k-1) item set, $q \in F^{'DB}_{sort}$ do

If $q \subset f$

$q$.count++

End for loop

End for loop

For each $f \in F^{'DB}_{sort}$ do

If $f$.count = 0

add $f$ to $P_{BR}$

End for loop

Step 4 Return Positive Border set $P_{BR}$
Procedure Sort \( (F'_{DB}) \)

// The procedure sorts the item sets in \( F'_{DB} \) based on item set length in descending order

**Input**: \( F'_{DB} \)

**Output**: \( F'_{DB\text{sort}} \) // The set consisting of sorted elements

**Step 1** Apply insertion sort procedure to get decreased sorted list based on item set size

**Step 2** Return the set \( F'_{DB\text{sort}} \) consisting of sorted item sets

Procedure Negative-border // This procedure finds negative borders set based on \( F'_{DB} \)

**Input**: \( F'_{DB} \)

**Output**: \( N_{BR} \)

**Step 1** Initialize \( k = 1 \)

**Step 2** For \( (K = 1; \ L_k ≠ \emptyset ; k++) \) do

  If \( (k = 1) \)

    For each item set \( x \in L_1 \) do

    If \( x \notin F'_{D} \)

      add \( x \) to \( N_{BR} \)

      End for loop

  Else

  if \( ( k =2 ) \)

    For each item set \( x \in L_1 \) do

    For each item set \( y \in L_1 \) do

    \( Z = \text{join} ( x, y) \)

    If \( ((x < y) \text{ and } z \notin F'_{D}) \)

    add \( z \) to \( N_{BR} \)

    End for loop
End for loop

Else

For each \( x \in L_{k-1} \)

For each \( y \in L_{k-1} \)

If \((x_1 = y_1) \land \ldots \land (x_{k-1} < y_{k-1})\)

\( Z = \text{join}(x, y) \)

If \((z \notin F'_DB, \exists r_{k-1} \subset Z; r_{k-1} \notin F'_DB)\)

add \( Z \) to \( N_{BR} \)

Endif

Endif

Endfor

Endfor

Endif

Endfor

Step 3  Return Negative Border Set \( N_{BR} \)

The following procedure generates association rules using frequent item set based on users specified minimum confidence threshold

Procedure (GenAssRuls) // Generates association rules

Input : \( F'_DB \), Min-Confidence

Output: Association rules (AR)

Step 1 Initialize \( j \) with 2;

Step 2 Create a file, Rule-set consisting of three fields rule-id, antdnt, consqnt ;

Step 3 For each item set \( L_j \) in \( F'_DB \)

Generate proper subsets of \( L_j \) and store in set \( S_{SUB} \)

// \( S_{SUB} = \{S_1, S_2, ..., S_m \} \) where \( m \) denotes the number of proper subsets
Initialize $i$ with one

While ( $i \leq m$ ) do

read $S_i$ from $S_{SUB}$

$antdnt = S_i$ and $consqnt = L_j - S_i$

$confd = \frac{\text{sup}(antdnt, consqnt)}{\text{sup}(antdnt)}$

if ($confd \geq \text{MinConfidence}$)

    add (rule-id, antdnt, consqnt, confd) to $AR$

    $i++$

End while loop

End for loop

**Step 3** Return a set $AR$ consisting of all association rules

The hybrid algorithm first applies border revision to identify the revised borders for the given original database, and then computes the minimal size for the extended database. By using the elements of border set, constraints are formed and then the given hiding problem is transformed into an optimization problem. By using BIP, optimization problem is solved and finds optimum solution for the extended database.

**3.4.2 Experimental Results for Partition Based Hybrid Hiding Algorithm**

This section illustrates how sensitive item sets are hidden from the competitors without affecting any item sets other than sensitive item sets using hybrid hiding algorithm when database size is large. The main goal of the algorithm is determining an additional database called minimal extension database or extended database by applying hybrid hiding algorithm. Once the extended database is found, one can easily determines distorted database by simply appending the extended database to the original database which can then be provided to the competitors. The competitors can then access the database for generating association rules which do not specify any thing regarding sensitive information.
This experiment takes the database of size 16. This database consists of 16 transactions and 5 attributes and each row designates transaction ID's and each column designates attribute. If the cell $C_{ij}$ consists of value one indicates $i^{th}$ transaction supports $j^{th}$ item otherwise it indicates $i^{th}$ transaction does not support $j^{th}$ item.

**Table 3.10: Sample Database (DB)**

<table>
<thead>
<tr>
<th>Items\T-ID</th>
<th>A1</th>
<th>A2</th>
<th>A3</th>
<th>A4</th>
<th>A5</th>
</tr>
</thead>
<tbody>
<tr>
<td>T₁</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>T₂</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T₃</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>T₄</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>T₅</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>T₆</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T₇</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T₈</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Items\T-ID</th>
<th>A1</th>
<th>A2</th>
<th>A3</th>
<th>A4</th>
<th>A5</th>
</tr>
</thead>
<tbody>
<tr>
<td>T₉</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>T₁₀</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>T₁₁</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>T₁₂</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T₁₃</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>T₁₄</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T₁₅</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T₁₆</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Let us assume that user specified minimum support = 25%, Let S consists of all the sensitive item sets is \{<A₃>,<A₂,A₄>\}

Using divide and conquer strategy, the given sample database of size sixteen is partitioned into two equal databases termed as DB₁ and DB₂ of size 8 which are having sufficient size to find the optimum solution. The following Table 3.11 & Table 3.12 shows partitioned databases termed as DB₁ and DB₂.

<table>
<thead>
<tr>
<th>Items/Transactions</th>
<th>A₁</th>
<th>A₂</th>
<th>A₃</th>
<th>A₄</th>
<th>A₅</th>
</tr>
</thead>
<tbody>
<tr>
<td>T₁</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>T₂</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T₃</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>T₄</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>T₅</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>T₆</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T₇</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T₈</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Sup</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Items/Transactions</th>
<th>A₁</th>
<th>A₂</th>
<th>A₃</th>
<th>A₄</th>
<th>A₅</th>
</tr>
</thead>
<tbody>
<tr>
<td>T₁</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>T₂</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>T₃</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>T₄</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T₅</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>T₆</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T₇</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T₈</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Sup</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>6</td>
</tr>
</tbody>
</table>
The following section explains the problem of finding solution for each Partitioned database.

3.4.2.1 Solution for First Partitioned Database DB₁

Let us find the optimum solution for the database (DB₁). The first step of this experiment is to generate frequent item sets based on user specified minimum support 25% and then saves these results in a set, F_{DB₁} which is specified along with support value as follows:

\[ F_{DB₁} = \{ A₁→3, A₂→3, A₃→4, A₄→5, A₅→4, <A₁,A₂>→2, <A₁,A₄>→2, <A₂,A₄>→2, <A₂,A₅>→2, <A₃,A₄>→2, <A₃,A₅>→3, <A₄,A₅>→3, <A₂,A₄,A₅>→2 \} \]

The next step is to determine the values for S_{min}, S_{max}, F'_{DB} based on set S using following formulae

\[ S_{min} = \{ I \in S/ \forall J \subset I, J \notin S \} \]
\[ S_{max} = \{ I \in F_{DB} / \exists J \in S_{min}, J \subset I \} \]
\[ F'_{DB} = F_{DB₁} - S_{max} \]

\[ \therefore F'_{DB} = \{ A₁, A₂, A₄, A₅, <A₁,A₂>, <A₁,A₄>, <A₂,A₅>, <A₄,A₅> \} \]

After determining the above values, invoke positive border procedure and negative border procedure to get revised positive border and negative border set.

\[ P_{BR} = \{ <A₄,A₅>, <A₁,A₂>, <A₁,A₄>, <A₂,A₅> \} \]
\[ P_{NR} = \{ A₃, <A₁,A₅>, <A₂, A₄> \} \]

Border set can be determined by appending negative border set with positive border set.

\[ \text{Border set} = \{ <A₄,A₅>, <A₁,A₂>, <A₁,A₄>, <A₂,A₅>, A₃, <A₁,A₅>, <A₂,A₄> \} \]

After forming border set, prepare a list which consists of sensitive item set with support values. The sensitive item sets and its supports for DB₁ is \{A₃→4, <A₂,A₄>→2\}. Select the highest support sensitive item set to determine size of extended database as follows:
Once required key sensitive item set is found determine the size of extended database using the following formula.

$\langle A_3 \rangle$ is the highest support item set and support is 4.

\[
\text{SIZE}_x = \left\lfloor \frac{\text{Support}(\text{Large sensitive itemset})}{\text{Freq}} - N \right\rfloor + 1
\]

\[
\text{SIZE}_x = \lceil 4/0.25 - 8 \rceil + 1 = \lceil 16 - 8 \rceil + 1
\]

\[
\text{SIZE}_x = 9
\]

In this case $\text{SIZE}_x$ gives enough value for $D$, so Safety margin value is zero

$\text{SIZE}_x = \text{SIZE}_x + 0$

The extended database of size 9 rows and 5 columns is created and then introduce unknown variables for each position in the extended database.

Table 3.13: Extended Database with Unknown Variables For DB1

<table>
<thead>
<tr>
<th>Items/Transactions</th>
<th>$A_1$</th>
<th>$A_2$</th>
<th>$A_3$</th>
<th>$A_4$</th>
<th>$A_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T'_1$</td>
<td>$U_{11}$</td>
<td>$U_{12}$</td>
<td>$U_{13}$</td>
<td>$U_{14}$</td>
<td>$U_{15}$</td>
</tr>
<tr>
<td>$T'_2$</td>
<td>$U_{21}$</td>
<td>$U_{22}$</td>
<td>$U_{23}$</td>
<td>$U_{24}$</td>
<td>$U_{25}$</td>
</tr>
<tr>
<td>$T'_3$</td>
<td>$U_{31}$</td>
<td>$U_{32}$</td>
<td>$U_{33}$</td>
<td>$U_{34}$</td>
<td>$U_{35}$</td>
</tr>
<tr>
<td>$T'_4$</td>
<td>$U_{41}$</td>
<td>$U_{42}$</td>
<td>$U_{43}$</td>
<td>$U_{44}$</td>
<td>$U_{45}$</td>
</tr>
<tr>
<td>$T'_5$</td>
<td>$U_{51}$</td>
<td>$U_{52}$</td>
<td>$U_{53}$</td>
<td>$U_{54}$</td>
<td>$U_{55}$</td>
</tr>
</tbody>
</table>

The next task is to forming the constraint satisfaction problem by constructing constraints for each element in positive border set as well as negative border set.

Constraints formed from positive border item sets are as follows:

For item set $\langle A_4, A_5 \rangle$, the constraint is $U_{24} + 1 + U_{74} \geq 2$

For item set $\langle A_1, A_2 \rangle$, the constraint is $1 + U_{72} \geq 2$

For item set $\langle A_1, A_4 \rangle$, the constraint is $1 + U_{74} \geq 2$

For item set $\langle A_2, A_5 \rangle$, the constraint is $U_{22} + U_{72} \geq 2$
With respect to negative border set, constraints are

For item set \(< A_3 >\), the constraint is \(1 + U_{53} + U_{63} + U_{83} < 2\)

For item set \(< A_2, A_4 >\), the constraint is \(U_{22} U_{24} + U_{72} U_{74} < 2\)

For item set \(< A_1, A_5 >\), the constraint is \(1 < 2\) which is true, so no need to frame as constraint.

Now the problem is to find values for unknown variables in which database satisfies all three goals accurately that is no sensitive rule is revealed, no non sensitive rule is hidden and no wrong rule is generated from the distorted database.

Now the optimization problem can be defined as

Objective function

Minimize \(Z = U_{11} + U_{12} + U_{13} + U_{14} + U_{15} + U_{21} + \ldots + U_{115}\)

Subject to the constraints

\(U_{24} + 1 + U_{74} \geq 2\)

\(1 + U_{72} \geq 2\)

\(1 + U_{74} \geq 2\)

\(U_{22} + U_{72} \geq 2\)

\(1 + U_{53} + U_{63} + U_{83} < 2\)

\(U_{22} U_{24} + U_{72} U_{74} < 2\)

In the sixth constraint each term consist of product of two variables such as \(U_{22} U_{24}\) and \(U_{72} U_{74}\) which can not be solved with BIP. To solve this problem, Constraint Degree Reduction Approach can be used which converts non linear variable into binary variable.

According to constraint degree Reduction Approach each product variable is replaced with new variable \(\Psi\) also new set of constraints are added for \(U_{22} U_{24} + U_{72} U_{74} < 2\)
Introduce new variable for each product term as follows.

Let \( \Psi_1 = U_{22} U_{24} \) and \( \Psi_1 = U_{72} U_{74} \)

\[
\Psi_1 < U_{22} \quad \Psi_1 < U_{24} \quad \Psi_1 < U_{22} + U_{24} - 1
\]

\[
\Psi_2 < U_{72} \quad \Psi_2 < U_{74} \quad \Psi_2 < U_{72} + U_{74} - 1
\]

\( \Psi_1 + \Psi_2 < 2 \)

Where \( \Psi_1 \) and \( \Psi_2 \) takes values of either 0 or 1

Minimize \( Z = U_{11} + U_{12} + U_{13} + U_{14} + U_{15} + U_{21} + \ldots U_{115} \)

Subject to the constraints

\( U_{24} + U_{74} \geq 2 \)

\( 1 + U_{72} \geq 2 \)

\( 1 + U_{74} \geq 2 \)

\( U_{22} + U_{72} \geq 2 \)

\( 1 + U_{53} + U_{63} + U_{83} < 2 \)

\[
\Psi_1 < U_{22} \quad \Psi_1 < U_{24} \quad \Psi_1 < U_{22} + U_{24} - 1
\]

\[
\Psi_2 < U_{72} \quad \Psi_2 < U_{74} \quad \Psi_2 < U_{72} + U_{74} - 1
\]

\( \Psi_1 + \Psi_2 < 2 \)

where \( U_{22}, U_{24}, U_{72}, U_{74}, U_{53}, U_{63}, U_{83}, \Psi_1, \Psi_2 \) takes either zero or one.

The above CSP consist of 12 constraints, 45 unknown variables and \( \Psi_1, \Psi_2 \). In addition to the above constraints, the following are to be considered as constraints for this problem which are given here.

\[
U_{11} + U_{12} + U_{13} + U_{14} + U_{15} \geq 1 \quad U_{21} + U_{22} + U_{23} + U_{24} + U_{25} \geq 1 \quad U_{31} + U_{32} + U_{33} + U_{34} + U_{35} \geq 1
\]

\[
U_{41} + U_{42} + U_{43} + U_{44} + U_{45} \geq 1 \quad U_{51} + U_{52} + U_{53} + U_{54} + U_{55} \geq 1 \quad U_{61} + U_{62} + U_{63} + U_{64} + U_{65} \geq 1
\]

\[
U_{71} + U_{72} + U_{73} + U_{74} + U_{75} \geq 1 \quad U_{81} + U_{82} + U_{83} + U_{84} + U_{85} \geq 1 \quad U_{91} + U_{92} + U_{93} + U_{94} + U_{95} \geq 1
\]

Now BIP solver is applied to get the optimum solution. The solution is nothing but values for unknown variables in extension database.
The next task is appending partitioned database $DB_1$ with minimal extended database to get distorted database $DB'$.

$$DB'_1 = DB_1$$ is appended with extended database.

The following Table 3.15 shows distorted database for the partitioned database ($DB_1$)

Table 3.15: Distorted Database ($DB'_1$)

<table>
<thead>
<tr>
<th>Items/Transactions</th>
<th>A1</th>
<th>A2</th>
<th>A3</th>
<th>A4</th>
<th>A5</th>
<th>Items/Transactions</th>
<th>A1</th>
<th>A2</th>
<th>A3</th>
<th>A4</th>
<th>A5</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T'_1$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>$T'_6$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$T'_2$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>$T'_7$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$T'_3$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>$T'_8$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$T'_4$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>$T'_9$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$T'_5$</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Support</td>
<td>5</td>
<td>4</td>
<td>0</td>
<td>6</td>
<td>5</td>
</tr>
</tbody>
</table>

Finding optimum solution for second partitioned database using hybrid hiding methodology specified in the following section.
3.4.2.2 Solution for Second Partitioned Database DB2

Let us find the optimum solution for the database DB2 given in Table 3.12. The first step is to generate frequent item sets based on user specified minimum support 25% and then prepares a list \( F_{DB2} \) consisting of frequent item sets with support values as follows:

\[
F_{DB2} = \{ A_1\rightarrow 5, A_2\rightarrow 4, A_3\rightarrow 2, A_4\rightarrow 3, A_5\rightarrow 6, \langle A_1,A_2\rangle\rightarrow 2, \langle A_1,A_4\rangle\rightarrow 2, \langle A_1,A_5\rangle\rightarrow 4, \langle A_2,A_4\rangle\rightarrow 2, \langle A_2,A_5\rangle\rightarrow 4, \langle A_4,A_5\rangle\rightarrow 3, \langle A_1,A_2,A_5\rangle\rightarrow 2, \\
\langle A_1,A_4,A_5\rangle\rightarrow 2, \langle A_2,A_4,A_5\rangle\rightarrow 2 \}
\]

The next step is to determine the values for \( S_{min} \), \( S_{max} \), \( F'_{DB2} \) based on set \( S \) using following formulae:

\[
S_{min} = \{ I \in S/ \forall J \subset I, J \not\in S \}
\]

\[
\therefore S_{min} = \{ \langle A_2,A_4\rangle, \langle A_3\rangle \}
\]

\[
S_{max} = \{ I \in F_{DB}/ \exists J \in S_{min}, J \subset I \}
\]

\[
\therefore S_{max} = \{ A_3, \langle A_2,A_4\rangle, \langle A_2,A_4,A_5\rangle \}
\]

\[
F'_{DB2} = F_{DB2} - S_{max}
\]

\[
\therefore F'_{DB2} = \{ A_1, A_2, A_4, A_5, \langle A_1,A_2\rangle, \langle A_1,A_4\rangle, \langle A_1,A_5\rangle, \langle A_2,A_5\rangle, \langle A_4,A_5\rangle, \langle A_1,A_2,A_5\rangle, \langle A_1,A_4,A_5\rangle \}
\]

After determining the above values invoke positive border procedure and negative border procedure to get revised positive border and negative border set.

\[
P_{BR} = \{ \langle A_1,A_2,A_5\rangle, \langle A_1,A_4,A_5\rangle \}
\]

\[
P_{NR} = \{ A_3 \}
\]

Border set can be determined by appending negative border set with positive border set.

\[
\text{Border set} = \{ \langle A_1,A_2,A_5\rangle, \langle A_1,A_4,A_5\rangle \}, A_3 \}
\]

After forming border set, find a sensitive item set whose support is highest among sensitive item set \( S \) which is specified as \( \{ A_3\rightarrow 2, \langle A_2,A_4\rangle\rightarrow 2 \} \)

Since the above two item sets are having same value so we can choose any one’s support for finding size for minimal extended database using formula

\[
\text{SIZE}_x = [2/0.25 - 8] + 1 = [8 - 8] + 1 = 1
\]
As \( \text{SIZE}_X \) value is one which is insufficient to hide sensitive item set so safety margin is added to \( \text{SIZE}_X \) as

\[
\text{SIZE}_X = \text{SIZE}_X + 1 = 2
\]

The extended database of 2 rows and 5 columns is created and then introduce unknown variables for each position in the extended database.

**Table 3.16: Extended Database with Unknown Variables for DB\textsubscript{2}**

<table>
<thead>
<tr>
<th>Items / Transactions</th>
<th>( A_1 )</th>
<th>( A_2 )</th>
<th>( A_3 )</th>
<th>( A_4 )</th>
<th>( A_5 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( T_1 )</td>
<td>( U_{11} )</td>
<td>( U_{12} )</td>
<td>( U_{13} )</td>
<td>( U_{14} )</td>
<td>( U_{15} )</td>
</tr>
<tr>
<td>( T_2 )</td>
<td>( U_{21} )</td>
<td>( U_{22} )</td>
<td>( U_{23} )</td>
<td>( U_{24} )</td>
<td>( U_{25} )</td>
</tr>
</tbody>
</table>

The next task is to form the constraint satisfaction problem by creating constraints for each element in positive border set as well as negative border set. Constraints are formed from positive border item sets are as follows:

- For item set \( \langle A_1, A_2, A_5 \rangle \), the constraints is \( U_{11} \cdot U_{12} \cdot U_{15} + U_{21} \cdot U_{22} \cdot U_{25} \geq 1 \)
- For item set \( \langle A_1, A_4, A_5 \rangle \), the constraints is \( U_{11} \cdot U_{14} \cdot U_{15} + U_{21} \cdot U_{24} \cdot U_{25} \geq 1 \)

Constraints formed from negative border item sets are as follows:

- For item set \( \langle A_3 \rangle \), the constraints is \( U_{13} + U_{23} < 1 \)

This constraint gives solution directly as \( U_{13} = U_{23} = 0 \).

So optimization problem can be specified as follows:

\[
\text{Min} \ Z = U_{11} + U_{12} + U_{13} + U_{14} + U_{15} + U_{21} + U_{22} + U_{23} + U_{24} + U_{25}
\]

Subject to the constraints

\[
U_{11} \cdot U_{12} \cdot U_{15} + U_{21} \cdot U_{22} \cdot U_{25} \geq 1
\]

\[
U_{11} \cdot U_{14} \cdot U_{15} + U_{21} \cdot U_{24} \cdot U_{25} \geq 1
\]

where \( U_{ij} \) takes binary values only.

The above constraints consists of non linear variables which can not be solved by Binary Integer Programming, so to convert non linear variable into linear using constraints degree reduction approach and conversion process is given below:

According to constraint degree reduction approach each product variable is replaced with new variable \( \Psi \) also new set of constraints are framed.
For the first constraint $U_{11} U_{12} U_{15} + U_{21} U_{22} U_{25} \geq 1$, introduce new variable for each product term as follows:

Let $\Psi_1 = U_{11} U_{12} U_{15}$ and $\Psi_2 = U_{21} U_{22} U_{25}$

$\Psi_1 < U_{11} \quad \Psi_1 < U_{12} \quad \Psi_1 < U_{15} \quad \Psi_1 < U_{11} + U_{12} + U_{15} - 2$

$\Psi_2 < U_{21} \quad \Psi_2 < U_{22} \quad \Psi_2 < U_{25} \quad \Psi_2 < U_{21} + U_{22} + U_{25} - 2$

$\Psi_1 + \Psi_2 \geq 1$

For the second constraint $U_{11} U_{14} U_{15} + U_{21} U_{24} U_{25} \geq 1$, introduce new variable for each product term as follows.

Let $\Psi_3 = U_{11} U_{14} U_{15}$ and $\Psi_4 = U_{21} U_{24} U_{25}$

$\Psi_3 < U_{11} \quad \Psi_3 < U_{14} \quad \Psi_3 < U_{15} \quad \Psi_3 < U_{11} + U_{14} + U_{15} - 2$

$\Psi_4 < U_{21} \quad \Psi_4 < U_{24} \quad \Psi_4 < U_{25} \quad \Psi_4 < U_{21} + U_{24} + U_{25} - 2$

$\Psi_3 + \Psi_4 \geq 1$

Where $\Psi_1, \Psi_2, \Psi_3$ and $\Psi_4$ takes values of either 0 or 1.

Min $Z = U_{11} + U_{12} + U_{13} + \ldots + U_{25} + \Psi_1 + \Psi_2 + \Psi_3 + \Psi_4$

Subject to the constraints

$\Psi_1 < U_{11}; \quad \Psi_1 < U_{12}; \quad \Psi_1 < U_{15}; \quad \Psi_1 < U_{11} + U_{12} + U_{15} - 2$

$\Psi_2 < U_{21}; \quad \Psi_1 < U_{22}; \quad \Psi_1 < U_{25}; \quad \Psi_1 < U_{21} + U_{22} + U_{25} - 2$

$\Psi_1 + \Psi_2 \geq 1$

$\Psi_3 < U_{11}; \quad \Psi_3 < U_{14}; \quad \Psi_3 < U_{15}; \quad \Psi_3 < U_{11} + U_{14} + U_{15} - 2$

$\Psi_4 < U_{21}; \quad \Psi_4 < U_{24}; \quad \Psi_4 < U_{25}; \quad \Psi_4 < U_{21} + U_{24} + U_{25} - 2$

$\Psi_3 + \Psi_4 \geq 1$

where $U_{11}, U_{12}, \ldots U_{25}, \Psi_1, \Psi_2, \Psi_3, \Psi_4$ takes either zero or one.

The CSP consist of 18 constraints, 14 unknown variables and $\Psi_1, \Psi_2, \Psi_3, \Psi_4$ (temporary variables)

Total number of constraints is 14

Now BIP is applied to get the optimum solution. The solution is nothing but values for unknown variables in extension database.
Table 3.17: Extended Databases for Second Partitioned Database (DB₂)

<table>
<thead>
<tr>
<th>Items/Transactions</th>
<th>A₁</th>
<th>A₂</th>
<th>A₃</th>
<th>A₄</th>
<th>A₅</th>
</tr>
</thead>
<tbody>
<tr>
<td>T₁</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>T₂</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

The distorter database (DB₂') is computed by merging transactions from partitioned database (DB₂) with transactions of extended database.

∴ DB₂' = DB₂ is merged with extended database

This distorted database is shown in the following Table 3.18.

Table 3.18: Distorted Database (DB₂') for DB2

<table>
<thead>
<tr>
<th>Items/Transactions</th>
<th>A₁</th>
<th>A₂</th>
<th>A₃</th>
<th>A₄</th>
<th>A₅</th>
</tr>
</thead>
<tbody>
<tr>
<td>T₁</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>T₂</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>T₃</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>T₄</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T₅</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>T₆</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T₇</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T₈</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>T₁'</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>T₂'</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Optimum solution can be found easily for large database because solution is obtained easily by merging operation of partitioned database solutions that is DB₁' and DB₂'. Finally the safest database which can be released to competitors is nothing but distorted database for large database denotes as DB'' will be computed by merging transactions from DB₁' and DB₂'. This distorted database DB'' is shown in Table 3.19.
Table 3.19: Distorted Database for DB

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>T10</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>T19</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>T2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>T11</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>T20</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>T3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>T12</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>T21</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T4</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>T13</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>T22</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>T5</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>T14</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>T23</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T6</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>T15</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>T24</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T7</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>T16</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>T25</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>T8</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>T17</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>T26</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>T9</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>T18</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>T27</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

There are no side effects means all the non sensitive association rules are accessible, no single sensitive rule is accessible and no wrong rule is generated from distorted database.

3.4.3 Hybrid Hiding Methodology without Using Partitioning Concept

The existing method is compared with hybrid hiding methodology which finds solution for the large database. The sample database specified in the Table 3.10 (sample database of size 16) is considered for illustration purpose. For this sample database, frequent item sets are generated using apriori algorithm and these frequent item sets along with support values are given below.

\[ F_{DB} = \{ A_1 \rightarrow 8, A_2 \rightarrow 7, A_3 \rightarrow 6, A_4 \rightarrow 8, A_5 \rightarrow 10, <A_1,A_2> \rightarrow 4, <A_1,A_4> \rightarrow 4, <A_1,A_5> \rightarrow 5, <A_2,A_4> \rightarrow 4, <A_2,A_5> \rightarrow 6, <A_4,A_5> \rightarrow 6, <A_2,A_4,A_5> \rightarrow 4 \} \]

\[ S = \{ <A_3>, <A_2,A_4> \}, S_{min} = \{ <A_3>, <A_2,A_4> \}, \]

\[ S_{max} = \{ A_3, <A_2,A_4>, <A_2,A_4,A_5> \} \]

\[ F'_{DB} = F_{DB} - S_{max} = \{ A_1, A_2, A_4,A_5, <A_1,A_2>, <A_1,A_4>, <A_1,A_5>, <A_2,A_5>, <A_4,A_5> \} \]
By invoking positive border procedure, positive border set can be determined as

\[ P_{BR} = \{ <A_1, A_2>, <A_1, A_4>, <A_1, A_5>, <A_2, A_5>, <A_4, A_5> \} \]

\[ N_{BR} = \{ A_3, <A_1, A_2, A_3>, <A_1, A_4, A_5> \} \]

Border set = \[ \{ <A_1, A_2>, <A_1, A_4>, <A_1, A_5>, <A_2, A_5>, <A_4, A_5>, A_3, <A_1, A_2, A_5>, <A_1, A_4, A_5> \} \]

Now find the value for extended database size using the appropriate computations. Then constraints are formed based on the item sets in \( P_{BR} \) and based on the item sets in \( N_{BR} \) as shown below.

- For item set \( <A_1, A_2> \), the constraint is \( U_{11} U_{12} + U_{21} U_{22} + \ldots + U_{101} U_{102} \geq 1 \)
- For item set \( <A_1, A_4> \), the constraint is \( U_{11} U_{14} + U_{21} U_{24} + \ldots + U_{101} U_{104} \geq 1 \)
- For item set \( <A_1, A_5> \), the constraint is \( U_{11} U_{15} + U_{21} U_{25} + \ldots + U_{101} U_{105} \geq 2 \)
- For item set \( <A_2, A_5> \), the constraint is \( U_{12} U_{15} + U_{22} U_{25} + \ldots + U_{102} U_{105} \geq 3 \)
- For item set \( <A_4, A_5> \), the constraint is \( U_{14} U_{15} + U_{24} U_{25} + \ldots + U_{104} U_{105} \geq 3 \)

For negative border set constraints can be framed as

- For item set \( <A_3> \), the constraint is \( U_{13} + U_{23} + \ldots + U_{103} < 1 \)
- For item set \( <A_1, A_2, A_4>, \) the constraint is \( U_{11} U_{12} U_{15} + U_{21} U_{22} U_{25} + \ldots + U_{101} U_{102} U_{105} < 1 \)
- For item set \( <A_1, A_4, A_5>, \) the constraint is \( U_{11} U_{14} U_{15} + U_{21} U_{24} U_{25} + \ldots + U_{101} U_{104} U_{105} < 1 \)

The above constraints consisting of non linear variables which should be converted into linear by using constraints degree reduction approach. Finally the CSP consist of 238 constraints with 50 unknown variables (excluding temporary variables) and this makes the formed CSP is a complex problem. The solution for this large database with the above constraints can be obtained with more computations and also increases time complexity. The following Table 3.20 shows the distorted database for the given large database.
Table 3.20: Distorted Database (DB') for Large Database (DB)

<table>
<thead>
<tr>
<th>Items/ T-ID</th>
<th>A_1</th>
<th>A_2</th>
<th>A_3</th>
<th>A_4</th>
<th>A_5</th>
<th>Items/ T-ID</th>
<th>A_1</th>
<th>A_2</th>
<th>A_3</th>
<th>A_4</th>
<th>A_5</th>
</tr>
</thead>
<tbody>
<tr>
<td>T_1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>T_10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>T_2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>T_11</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>T_3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>T_12</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T_4</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>T_13</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>T_5</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>T_14</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T_6</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>T_15</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T_7</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>T_16</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>T_8</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>T_17</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>T_9</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>T_18</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Both the method gives same results but solving large database is more difficult. Using partitioned database hybrid hiding methodology one can find solution for any large size database by incorporating divide and conquer strategy.

The following are the frequent item sets obtained from proposed methodology as well as the existing methodology without using partitioned database concept.

Frequent item sets from proposed methodology is \{A_1→14, A_2→11, A_4→15, A_5→16, <A_1,A_2>→7, <A_1, A_4>→8, <A_1,A_5>→8, <A_2,A_5>→9, <A_4,A_5>→10\}

Frequent item sets from existing methodology is \{A_1→14, A_2→11, A_4→15, A_5→16, <A_1,A_2>→7, <A_1,A_4>→8, <A_1,A_5>→8, <A_2,A_5>→9, <A_4,A_5>→10\}

From the results we can say both methods give same results. The performance analysis of the proposed methodology is presented in the following section.
3.4.4 Performance Analysis of Partition Based Hybrid Hiding Methodology

- The proposed methodology efficiently finds the optimal solution to hide the sensitive item for large size database.

- When the database size is large, divided and conquer strategy is adopted to partition the given database into small size databases. This partition can be viewed as a binary tree structure where the original database is at Level\(_0\) and its partitioned databases are at Level\(_1\). Divide and conquer strategy can be applied further for the database which are at Level\(_1\) and this process is repeated until suitable size databases are obtained.

- The leaf node of the binary tree structure are the final partitioned databases. The solution of each database which is at the bottom most level can be obtained individually by applying hybrid hiding algorithm parallel.

- It is a fact the finding a solution for small size database is simple and easy than finding solution for large size database.

- The major task is performed at bottom level only and the processing at other levels of the tree is a simple task. The databases which are at the same level can obtained solution simply merging the solutions of its partitioned databases. In other words the solution of the parent database is nothing but the merged solution of its children database. So, the process of computations is easy in the proposed methodology to find the solution.

- The parallelism concept can be easily applied at each level of the tree to obtained the solution of the databases and thus reduces the overall time complexity.

- Sometimes finding the solution for large database may not be possible since more number of constraints and more number of unknown variables are exist in the CSP which makes this, a complex CSP even to get the solution which is not an optimum solution. The proposed methodology finds the solution for this type of problems by adopting divides & conquers and parallelism concepts.
Experiments are conducted based on synthetic dataset for both existing and proposed methodology for large database. The synthetic dataset consist of Boolean transactions with 8 attributes. For comparison purpose, existing hybrid hiding algorithm which is introduced in [85] is considered. Experimental results are presented in the following graph which shows the variation of execution time for both existing and proposed methodology for various size databases.

![Performance Gain in Synthetic data](image)

**Figure 3.4 Partition Based Hybrid Hiding Vs Hybrid Algorithm**

The graph clearly shows that proposed methodology takes less execution time compared to hybrid hiding algorithm. The proposed methodology adopts partitioning and parallel concept and finds the solution of small size databases which are at the bottom level of the tree executes in parallel and individually. Because of this reason, the proposed methodology takes less time to hide the sensitive item sets compared to hybrid hiding algorithm. The graph also reveals this fact.

The performance of the Partition based hybrid hiding algorithm, the modified Inline algorithm and proposed heuristic based approach are compared in terms of side effects. The experiments are conducted with these three methodologies with synthetic dataset which consist of Boolean transactions with 8-attributes. The following figure indicates the comparison analysis of these three methodologies.
The X-axis, M x N, where M denotes the number of sensitive item sets and N represents the sensitive item set length. The Y-axis indicates the number of side effects occurs in the process of hiding sensitive item sets. The graph clearly specifies that partition based hybrid hiding methodology outperforms than modified Inline and proposed heuristic based methodology. The graph also reveals that modified Inline algorithm is efficient when compared to proposed heuristic based methodology.

Hence the proposed partition based hybrid hiding methodology finds the optimum solution efficiently in order to hide the sensitive item sets for large size database with minimum side effects.