Concept of Entire Boolean Values Recalculation From Aggregates in the Preprocessed Category of Incomplete Soft Sets

MUHAMMAD SADIQ KHAN1, TUTUT HERAWAN1,2, AINUDDIN WAHID ABDUL WAHAB1, GHULAM MUJTABA1,3, AND MOHAMMED ALI AL-GARADI1
1Faculty of Computer Science and Information Technology, University of Malaya, Kuala Lumpur 50603, Malaysia
2AMCS Research Center, Yogyakarta 55581, Indonesia
3Department of Computer Science, Sukkur Institute of Business Administration, Sukkur 65200, Pakistan
Corresponding author: Tutut Herawan (tutut@um.edu.my)

This work was supported by the University of Malaya Research Grant under Grant RP036-15AET.

ABSTRACT Soft set is a mathematical tool for dealing with vague and imprecise data. It is used in many applications and decision-making after representing the uncertain data in the Boolean-valued information system (BIS). BISs become incomplete because of various reasons, such as security, viral attack, and errors. Several soft set-based approaches exist to handle incomplete BISs for decision-making. These approaches are categorized into two categories: preprocessed (PP) and unprocessed (UP). UP approaches cannot be used for the recalculation of overall missing values. Meanwhile, PP approaches can be extended to calculate the entire missing values. This paper presents the basic concept of actual technique and initially applies it to the PP incomplete soft set. This novel concept will open a new chapter for researchers in the development of applications in the fields of mathematics, especially in Boolean data, discrete mathematics, and computer science.

INDEX TERMS Aggregate values, data compression, data filling, data prediction, incomplete information systems, soft sets, vague data.

I. INTRODUCTION Vague or uncertain data cannot be processed using conventional mathematical tools of crisp and clear data. Special models and theories, such as fuzzy set, probability, interval mathematics, rough set, grey set, and soft set, are used for the precise handling of the uncertainties in vague data to process it before use in any application and decision. In the soft set theory [1], an application is usually based on a standard soft set with all its values represented in a binary table known as Boolean-valued information system (BIS). Ordinary arithmetic operations and processing, such as crisp data, can be performed with BIS for use in any application. BISs are mainly used for decision-making and finding optimal choices by arithmetically adding the weights of all objects, and the parameter with the maximum value is considered as the best choice [2]. The reduct set for the soft set BIS is defined as the subset of all parameter sets that has the same decision values of optimal choice as those of the original set [3]. In a modified definition, a reduct set must be able to maintain the integrity of the decision values as the original set for optimal and suboptimal choices [4], and this parameterization reduction is more efficient if the method used for its calculation is easy to understand, implement, and has less computational complexity during execution [5]. Apart from these main applications of decision-making with parameterization reduction, soft set and BIS are used in several daily life applications [6]–[17].

These applications become worthless and may yield incorrect results if several values are lost in a given BIS. Values in a soft set can be lost because of communicational errors, virus attacks, improper entry, intentional and unintentional mistakes, security, or any other probable reasons. In cases where no equivalency information of aggregates or parity bits can be found, researchers have attempted to fill and predict them from other available set of values using weighted average [18], association between parameters [19]–[21], and probability [22] techniques. Meanwhile, the following recalculation techniques are presented from available aggregates and parity bits [23], [24].

In this study, we present the concept of recalculating entire Boolean values from aggregates in the preprocessed category...
of incomplete soft sets. Our contributions are described as follows:

We classify an incomplete soft set into preprocessed (PP) and unprocessed (UP) categories based on the nature of the input for recalculations and data filling, respectively, and present a heuristic method of recalculating the entire values from available aggregates in the former category.

Our approach uses the concept of solving simultaneous linear equations for identifying unknown variables. The proposed approach bypasses the restrictions of simultaneous linear equations, such that, the number of equations must be equal to or more than unknown variables. Unlike solving simultaneous linear equations, our approach has the capacity to calculate more variables than that of the given number of relations.

We take advantage of the binary nature and limited domain of the standard soft set. This new concept can be used by researchers to develop good applications in binary-ranged data regardless of the soft set.

The rest of our paper is organized as follows. The definition of soft set theory and its application in BIS and incomplete soft set are provided with examples in Section 2. Section 3 describes the existing approaches and their categorization to PP and UP incomplete soft sets. In Section 4, the proposed approach is explained in detail with definitions, examples, and the algorithm. Finally, we conclude our work in Section 5.

II. SOFT SET THEORY

This section presents the definition of soft set and its application in the BIS and incomplete soft set, which are accompanied with respective examples.

A. SOFT SET THEORY

Let \( U \) be an initial universal set and \( E \) be a set of parameters. A pair \((F, E)\) is called a soft set (over \( U \)) if and only if \( F \) is a mapping of \( E \) into the set of all subsets of the set \( U \) [1]. The following provides us an example of a soft set.

**Example 1:** Let \( U = \{h_1, h_2, h_3, h_4, h_5, h_6\} \) be a universal set of houses and \( E = \{e_1, e_2, e_3, e_4, e_5, e_6\} \) be a set of parameters representing “new houses,” “cheap,” “expensive,” “wooden,” “in green surrounding,” and “beautiful,” respectively. A soft set \((F, E)\) describes the attractiveness of houses for Mr. Gul who intends to buy.

Supposing the new houses are \( h_1, h_2, h_3, \) and \( h_6\); the cheap house is \( h_2\); the expensive houses are \( h_1, h_3, h_4, \) and \( h_5\); the wooden houses are \( h_2, h_3, h_5, \) and \( h_6\); the houses with green surroundings are \( h_3, h_4, h_5, \) and \( h_6\); and the beautiful houses are \( h_1, h_2, h_3, h_4, h_5, \) and \( h_6\), then

\[
(F, E) = \left\{ (e_1, \{h_1, h_2, h_5, h_6\}), (e_2, \{h_2\}), (e_3, \{h_1, h_3, h_4, h_5\}), (e_4, \{h_2, h_3, h_5, h_6\}), (e_5, \{h_3, h_4, h_5, h_6\}), (e_6, \{h_1, h_2, h_3, h_4, h_5, h_6\}) \right\}
\]

B. REPRESENTATION OF A SOFT SET AS A BOOLEAN-VALUED INFORMATION SYSTEM

If \( U \) is a non-empty finite set of objects, \( AT \) is a nonempty finite set of attributes, \( V = \bigcup \mathcal{V}_r \), such that \( \mathcal{V}_r \) is the value domain of attribute \( r \), and \( f \) is an information function given by \( f : U \times A \rightarrow V \). Then, the quaternion \( S = (U, AT, V, f) \) is called an information system. The soft set \((F, E)\) of Example 1 is represented in Table 1.

<table>
<thead>
<tr>
<th>( U / E )</th>
<th>( e_1 )</th>
<th>( e_2 )</th>
<th>( e_3 )</th>
<th>( e_4 )</th>
<th>( e_5 )</th>
<th>( e_6 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( h_1 )</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>( h_2 )</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>( h_3 )</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( h_4 )</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( h_5 )</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( h_6 )</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 1 above can be referred to as a BIS, thereby representing a standard soft set.

TABLE 1. Representation of soft set \((F, E)\) in tabular form BIS.

<table>
<thead>
<tr>
<th>( U / E )</th>
<th>( e_1 )</th>
<th>( e_2 )</th>
<th>( e_3 )</th>
<th>( e_4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( h_1 )</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>( h_2 )</td>
<td>0</td>
<td>1</td>
<td>*</td>
<td>1</td>
</tr>
<tr>
<td>( h_3 )</td>
<td>0</td>
<td>*</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>( h_4 )</td>
<td>*</td>
<td>0</td>
<td>1</td>
<td>*</td>
</tr>
</tbody>
</table>

Table 2 above can be referred to as a BIS, thereby representing an incomplete soft set.

TABLE 2. Representation of incomplete soft set, where * represents unknown values.

C. INCOMPLETE SOFT SET

An information system \( S = (U, AT, V, f) \) is called incomplete if \( f(x_i, a_i) \) is not known, where \( U = \{x_1, x_2, \ldots, x_n\}, A = \{a_1, a_2, \ldots, a_m\}, x_i \in U, i = 1, 2, \ldots, n, \) and \( a_j \in A \) for \( j = 1, 2, \ldots, m \). Any unknown value is represented by * in Table 2.

**Example 2:** An incomplete soft set is represented in Table 2. Unknown values are represented by *.

The incomplete information system in Table 2 shows that \( h_1 \in F(e_1) \), \( h_2 \cap h_3 \notin F(e_1). \) However, whether \( h_4 \) belongs to \( F(e_1) \) or not is unknown. Therefore, \((h_4, e_1) = u_{41}\) is unknown and represented by *. Similarly, \( u_{32}, u_{23}, \) and \( u_{44}\) are unknown and also represented by *.

III. RELATED WORKS

In this section, we present existing approaches of handling incomplete soft sets into two categories: UP and PP.

A. UNPROCESSED CATEGORY TECHNIQUES

Existing approaches that fall in the UP category are totally dependent on other available values. Finding aggregates or
other equivalencies require an extra processing, and these techniques do not consider any equivalency values of aggregates or decision values when filling or predicting missing values. Therefore, we consider them as unprocessed in terms of having no equivalency values and place them in the UP category.

The initial attempt in this category predicts only the decision values using the weighted average technique while the missing values remain missing [18]. Another shortcoming of the weighted average technique is that it is too complex and difficult to understand and implement. Subsequently, a less complex and easy to understand technique was recently presented using the simple probability of 1 and 0 s, which also overcomes the main shortcoming of the previous technique by assigning several values to the original missing values [22]. Both methods have the same decision results, but the missing values calculated in the latter technique are still non-compatible because, instead of binary values, it predicts values in a compact unit interval [0, 1] range razing in the integrity of the standard soft set and BIS. Meanwhile, another technique known as DFIS calculates missing values within the binary range, thereby maintaining the integrity of the standard soft set using association between parameters for strong associations and using probability for less or no association between parameters [20], [21]. Our most recent work, ADFIS, recognizes that DFIS is the most suitable among all the previous approaches of this category, and ADFIS further improves the accuracy of DFIS by revising its association calculation method [19]. Association between soft set parameters [19]–[21] has been successfully applied to link prediction problem in online social networks and detection of a new type of network community [25]. All approaches of this category are discussed and compared in detail in ADFIS.

B. PREPROCESSED CATEGORY TECHNIQUES

This category has two techniques. The techniques of this category rely on equivalency values, from which missing values are recalculated [23], [24]. Certain operations are required to find these equivalencies in the form of aggregates and/or decision values. Thus, we name the techniques of this group as PP category techniques. In subsequent sections, these techniques have been discussed.

1) USING PARITY BITS AND SUPPORTED SETS

In this approach, soft set is represented in BIS. The supported sets from all objects, and even parity bits for each row and column, are calculated for a complete table (no missing information at the time of these calculations). If few values are missing after these equivalencies, they can be recalculated using the available supported sets and parity bits [24]. Supported set is simply the arithmetic sum of the values of an object or number of 1 s in a row. Mathematically, for object $u$,

$$\text{supp}(u) = \text{card}_u (e \in E : f(u, e) = 1)$$  \hfill (1)

and the set of $\text{supp}(u)$ for all objects is the supported set.

A bit column is to make the parity bits of each object even. “0” is placed in the parity bit column if the object already has an even number of 1 s; otherwise, 1 is placed. Mathematically, for an object $u$

$$P_{bit} = \text{supp}(u) \mod 2.$$  

Similarly, an attribute or column parity bit is defined as follows:

$$C_{bit} = \left( \sum_{i=1}^{n} f(u, e_i) \right) \mod 2$$

a: USING AGGREGATE VALUES

In this approach, parity bits are not considered unlike the previous approach. In addition to the supported values of the rows, all aggregates of the diagonals and columns are used to recalculate the missing values. The diagonal aggregates of two directions, for example, left to right (LR) and right to left (RL), yield two sets of aggregates, while rows and columns aggregates yield another set of two aggregates [23]. In our proposed work, this technique is used as a base technique. Moreover, we also use its mathematical relation in our proposed work. We briefly discuss this technique below to elucidate several preliminary concepts.

b: ATTRIBUTE AGGREGATE VALUES

The arithmetic sum of attribute values is

$$C_{agg} = \sum_{i=1}^{n} f(u, e)$$  \hfill (2)

c: DIAGONAL AGGREGATE VALUES

For the soft set table that has $u_i$ objects and parameter set $E$, a tuple or diagonal can be expressed mathematically as follows:

$$t_i = (f(u_i, a_1), f(u_i, a_2), f(u_i, a_3), \ldots, f(u_i, a_A)),$$

where $i = 1, 2, 3, \ldots, |U|$  \hfill (3)

If $D$ is the number of unidirectional diagonals in a table, then

$$D = |U| + |A| - 1$$  \hfill (4)

Diagonals can be dealt through LR and RL manners for obtaining 2D accumulated values because rows and columns are treated horizontally and vertically. The number of diagonals (D) is more than that of the column or rows. Therefore, we have two different cases for both LR and RL.

Case 1: For $1 \leq k \leq |A|$,

$$\text{Diag}_{LR}(k) = \sum_{i=1}^{k} f(u_i, a_j), \quad \text{where } j = k - i + 1$$  \hfill (5)

$$\text{Diag}_{RL}(k) = \sum_{i=1}^{k} f(u_i, a_j), \quad \text{where } j = |A| - k + i$$  \hfill (6)
Algorithm 1 Calculating Partial Missing Values From Aggregates

Input: Incomplete BIS and aggregate values.
Output: Complete BIS.

1. Calculate supported values of rows, aggregate values of columns, and diagonals.
2. Find every single value first by applying horizontal, vertical, or diagonal summations.
3. Repeat Step 2 until no single value remains.
4. Find other missing values, thereby applying supported, column, and/or diagonal aggregate.

TABLE 3. Complete soft set represented in tabular form.

<table>
<thead>
<tr>
<th>U / E</th>
<th>e1</th>
<th>e2</th>
<th>e3</th>
<th>e4</th>
<th>e5</th>
</tr>
</thead>
<tbody>
<tr>
<td>h1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>h2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>h3</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>h4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>h5</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>h6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Case 2: For $|A| < k < D$.

$$
\text{Diag}_{RLw} (k) = \sum_{j=|A|+1}^{U} f(u_i, a_j), \quad \text{where } j = k - i + 1, \\
\text{for } i \leq k \text{ and } j \leq |U|
$$

(7)

$$
\text{Diag}_{LRw} (k) = \sum_{j=|A|+1}^{U} f(u_i, a_j), \quad \text{where } j = |A| - k + i, \\
\text{for } i \leq k \text{ and } j \leq |U|
$$

(8)

The following algorithm presents the process of calculating partial missing values from aggregates.

Example 3: Considering the complete soft set given in Table 3, we calculate its aggregates of rows and columns in Table 4. In addition, its LR and RL diagonal aggregate values are provided in Tables 5 and 6, respectively.

The soft set with several values missing is shown in Table 7.

In Table 7, the missing values are $u_{11}, u_{42}, u_{43}, u_{52}, u_{53},$ and $u_{54}$. From Table 6, the column aggregate of $e_1$ is 2, therefore $u_{11} = 0$. For $u_{42}$, RL is equal to 2 in Table 6, therefore $u_{42} = 0$. Similarly, using the available equivalency information in Tables 4, 5, and 6, we can identify all other missing values and easily obtain our original Table 3.

IV. PROPOSED APPROACH FOR CALCULATING ENTIRE MISSING VALUES FROM AGGREGATES

In this section, we present the concept of recalculating the entire BIS values from available aggregates. First, we answer the question, “Is finding more variables than available relations through linear equations possible?” After answering this for a special case of BIS, we present the proposed method with several important definitions and examples.
TABLE 7. Soft set Table 3 with few missing values.

<table>
<thead>
<tr>
<th>$U/E$</th>
<th>$e_1$</th>
<th>$e_2$</th>
<th>$e_3$</th>
<th>$e_4$</th>
<th>$e_5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$h_1$</td>
<td>*</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$h_2$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$h_3$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>$h_4$</td>
<td>1</td>
<td>*</td>
<td>*</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$h_5$</td>
<td>0</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>0</td>
</tr>
<tr>
<td>$h_6$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

A. SOLVING NON-SIMULTANEOUS LINEAR EQUATIONS IN REAL DOMAIN

Simultaneous linear equations are defined as, “The set of two or more than two equations is called the set of simultaneous linear equations or simply simultaneous linear equations, if each equation contains two or more variables, such that the number of variables is less than equal to the number of equations, and the values of variables can satisfy both or all equations simultaneously.”

Suppose we have a set of linear equations as follows:

$$x + y + z = 2 \quad (9)$$

$$z = 1 \quad (10)$$

According to the above definition, this set is not the set of simultaneous linear equations because the number of relations is less than the number of variables and an exact solution of unknowns cannot be found. If we place $z = 1$ in Equation (9) we obtain the following:

$$x + y = 1 \quad (11)$$

We can identify infinite number of values for $x$ and $y$ of relation (11) in the real domain. Thus, the sum of both will be equal to 1. In the case of real numbers, finding exact values through non-simultaneous linear equations is impossible.

B. SOLVING NON-SIMULTANEOUS LINEAR EQUATIONS IN BOOLEAN DOMAIN

We reconsider the set of linear equations given above. If we know that the domain of these variables is of Boolean values, then we can easily identify two possible solutions for the relations (11) as given below.

$$x = 1 \text{ and } y = 0$$
$$x = 0 \text{ and } y = 1$$

Two steps are involved in finding the above solutions. First, we suppose $x = 0$ and place it in (11) to obtain $y = 1$. Then, supposing $x = 1$ yields $y = 0$. Hence, unlike the previous case of real domain, obtaining the finite number of possible values for such non-simultaneous relations by supposition in the binary domain is possible. If we have a clue of cross confirmation to select either one of the possible result or the other, then we can identify the exact one solution among all possible solutions.

C. POSSIBILITY OF FINDING ENTIRE MISSING VALUES IN BOOLEAN-VALUED INFORMATION SYSTEM FROM AGGREGATES

From the above discussion, the following points can be concluded as follows.

1. If we have a finite domain of values, obtaining all possible values of unknowns is possible even through the non-simultaneous linear equation by supposition.
2. If we have a clue of cross confirmation, then we can select the one exact set of values for unknowns among the set of all possible values calculated in Step 1.

Accordingly, BIS has the following:

a. A finite domain of binary values, and either 0 or 1 can be supposed as the possible value to obtain all possible sets of values.

b. Four sets of aggregates, where one is selected as the linear equation for the supposition of Step 1, and the other three sets function as the clue of cross confirmation for selecting one set of values as the exact solution.

Hence, recalculating all missing values from the aggregates in BIS is possible.

D. PROPOSED METHOD

The main idea of the proposed method is concluded in the above Points A and B. To formalize the concept, we present several important definitions and algorithm, and then solve an example using the proposed algorithm as a proof of concept. Each LR and RL diagonals have two cases as discussed in Equations (5) to (8). We define one general case for those cases as follows.

**Definition 1:** Let $(F, E)$ be a soft set and the diagonal be defined as $Diag_l = f(u_j, a_j)$, where $l = 1, 2, \cdots, D$, $m = |U|$, and $n = |A|$, such that $D = m + n - 1$, $m = |U|$, and $n = |A|$ are number of rows and columns, respectively.

From Definition 1, we introduce the concept of empty, universal, and hybrid (EUH) diagonals.

**Definition 2:** Let $(F, E)$ be a soft set. A diagonal is called empty if its aggregate is equal to zero, i.e.

$$\sum f(u_i, a_j) = 0.$$  

**Definition 3:** Let $(F, E)$ be a soft set. A diagonal is called universal if its aggregate is equal to the number of its cells, i.e.

$$\sum f(u_i, a_j) = |\{ f(u_i, a_j) \}|.$$  

**Definition 4:** Let $(F, E)$ be a soft set. A diagonal is called hybrid if it is neither empty nor universal, i.e.

$$0 < \sum f(u_i, a_j) < |\{ f(u_i, a_j) \}|.$$  

In several special cases, only empty and universal diagonals are used to calculate missing data without going to any supposition from hybrid diagonals. This makes our approach more efficient, and our proposed algorithm successfully ends on Step 6. In most cases of large tables, we are unable to
accomplish this task on the bases of empty and universal diagonals only. Thus, we need to suppose binary values for hybrid diagonals.

Let $\sum f(u_i, a_j) = H_l$ be the aggregate value and $|f(u_i, a_j)| = M_l$, $\forall |a_{ij}| = 1$ be the cardinality or maximum value or size of a hybrid diagonal $Diag_l$.

**Definition 5:** Let $(F, E)$ be a soft set. If $S_l$ is the number of suppositions for diagonal $Diag_l$, then

$$S_l = \prod M_l, \quad l = 1, 2, \ldots, D, \ldots, 2D - 1, 2D.$$  

**Definition 6:** Let $(F, E)$ be a soft set. The total number of $1s$ in $S_l$ for a $Diag_l$ must be $H_l$ while the number of $0$s will be automatically $M_l - H_l$.

In our approach, we first construct the $m \times n$ table from the given number of rows and columns. We fill all empty and universal diagonals according to Definitions 2 and 3 and 0s and 1s, respectively. Then, all columns, rows, and diagonals are checked and filled in if possible according to its aggregate values. Second, data is temporarily filled in the shortest diagonals first by supersing diagonal cells as 0 or 1 according to Definition 5. Suppositions are crossexamined with related aggregate values, where possible. Initially supposed values are permanently assigned to specific cells only if other aggregates verify it. Otherwise, the supposition order is changed. The process is repeated again until the original values are identified. These values are assigned permanently after confirmation of having no contradiction with any of the related aggregate. We propose Algorithm 2 for recalculating the entire BIS from aggregate values.

The following example describes how our proposed algorithm handles missing data.

**Example 4:** Supposing we are provided with four non-empty sets,
We construct another table (Table 9) and assign all unknown values to temporary variables for identification, such that \( O_i = \{s_i, t_i, v_i, w_i, x_i, y_i, z_i\} \) for \( i = 1, 2, \ldots, 7 \). The row and column aggregates are also shown in the same table.

In Tables 10 and 11, we show this unknown table with LR and RL diagonal aggregates.

Tables 10 and 11 show that LR\(_1\), LR\(_2\), LR\(_3\), LR\(_13\), RL\(_1\), RL\(_3\), and RL\(_12\) are universal while LR\(_3\) and RL\(_12\) are null. According to Definitions 2 and 3, we place 1 for universal and 0 for empty diagonals and obtain some missing information as provided in Table 12.

In Table 12, we can complete the 1st column by placing \( w_1 = 0 \), thus obtain Table 13 because we know that \( C_1 = 4 \).

Considering Table 13, we start the supposition from the shortest incomplete diagonals, which are LR\(_12\) and RL\(_2\). Both have two cells and aggregate values that are equal to 1. In both diagonals, one value must be 0 and the other must be 1. Supposing \( y_7 = 0 = t_7 \Rightarrow z_6 = s_6 = 1 \), we still cannot proceed.
without further supposition for the next shortest diagonals, which are LR11 and RL3. These diagonals have three cells and aggregate values that are equal to 1 and 2, respectively. Supposing $x_7 = y_6 = v_7 = 0 \Rightarrow z_5 = t_6 = s_5 = 1$, Table 14 is obtained after placing these values.

$C_7$ disproves the supposition in Table 14. It cannot be obtained by placing $w_7 = 1$ only because we know that its aggregate is equal to 4. Reconsidering Table 13, all suppositions are disproved through cross-checking except $y_7 = s_6 = z_5 = v_7 = t_6 = 1$, which implies that
TABLE 14. Placing non-contradicting supposed values for $LR_{12}$, $RL_{2}$, $LR_{11}$, and $RL_{3}$.

<table>
<thead>
<tr>
<th>$U/E$</th>
<th>$e_1$</th>
<th>$e_2$</th>
<th>$e_3$</th>
<th>$e_4$</th>
<th>$e_5$</th>
<th>$e_6$</th>
<th>$e_7$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$o_1$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>$s_4$</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$o_2$</td>
<td>1</td>
<td>0</td>
<td>$t_3$</td>
<td>$t_4$</td>
<td>$t_5$</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$o_3$</td>
<td>0</td>
<td>$v_2$</td>
<td>$v_3$</td>
<td>$v_4$</td>
<td>$v_5$</td>
<td>$v_6$</td>
<td>0</td>
</tr>
<tr>
<td>$o_4$</td>
<td>0</td>
<td>$w_2$</td>
<td>$w_3$</td>
<td>$w_4$</td>
<td>$w_5$</td>
<td>$w_6$</td>
<td>$w_7$</td>
</tr>
<tr>
<td>$o_5$</td>
<td>1</td>
<td>$x_2$</td>
<td>$x_3$</td>
<td>$x_4$</td>
<td>$x_5$</td>
<td>$x_6$</td>
<td>0</td>
</tr>
<tr>
<td>$o_6$</td>
<td>0</td>
<td>1</td>
<td>$y_3$</td>
<td>$y_4$</td>
<td>$y_5$</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>$o_7$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>$z_4$</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

TABLE 15. Placing values of non-contradictive supposition.

<table>
<thead>
<tr>
<th>$U/E$</th>
<th>$e_1$</th>
<th>$e_2$</th>
<th>$e_3$</th>
<th>$e_4$</th>
<th>$e_5$</th>
<th>$e_6$</th>
<th>$e_7$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$o_1$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>$s_4$</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$o_2$</td>
<td>1</td>
<td>0</td>
<td>$t_3$</td>
<td>$t_4$</td>
<td>$t_5$</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$o_3$</td>
<td>0</td>
<td>$v_2$</td>
<td>$v_3$</td>
<td>$v_4$</td>
<td>$v_5$</td>
<td>$v_6$</td>
<td>1</td>
</tr>
<tr>
<td>$o_4$</td>
<td>0</td>
<td>$w_2$</td>
<td>$w_3$</td>
<td>$w_4$</td>
<td>$w_5$</td>
<td>$w_6$</td>
<td>$w_7$</td>
</tr>
<tr>
<td>$o_5$</td>
<td>1</td>
<td>$x_2$</td>
<td>$x_3$</td>
<td>$x_4$</td>
<td>$x_5$</td>
<td>$x_6$</td>
<td>0</td>
</tr>
<tr>
<td>$o_6$</td>
<td>0</td>
<td>1</td>
<td>$y_3$</td>
<td>$y_4$</td>
<td>$y_5$</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$o_7$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>$z_4$</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

TABLE 16. Placing values of $s_4$, $z_4$, $v_6$, $w_6$, $x_6$, and $w_7$.

<table>
<thead>
<tr>
<th>$U/E$</th>
<th>$e_1$</th>
<th>$e_2$</th>
<th>$e_3$</th>
<th>$e_4$</th>
<th>$e_5$</th>
<th>$e_6$</th>
<th>$e_7$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$o_1$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$o_2$</td>
<td>1</td>
<td>0</td>
<td>$t_3$</td>
<td>$t_4$</td>
<td>$t_5$</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$o_3$</td>
<td>0</td>
<td>$v_2$</td>
<td>$v_3$</td>
<td>$v_4$</td>
<td>$v_5$</td>
<td>$v_6$</td>
<td>1</td>
</tr>
<tr>
<td>$o_4$</td>
<td>0</td>
<td>$w_2$</td>
<td>$w_3$</td>
<td>$w_4$</td>
<td>$w_5$</td>
<td>$w_6$</td>
<td>$w_7$</td>
</tr>
<tr>
<td>$o_5$</td>
<td>1</td>
<td>$x_2$</td>
<td>$x_3$</td>
<td>$x_4$</td>
<td>$x_5$</td>
<td>$x_6$</td>
<td>1</td>
</tr>
<tr>
<td>$o_6$</td>
<td>0</td>
<td>1</td>
<td>$y_3$</td>
<td>$y_4$</td>
<td>$y_5$</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>$o_7$</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

$z_6 = t_1 = x_7 = y_6 = s_5 = 0$, by supposing different possible combinations. Meanwhile, Table 15 is obtained from placing these values.

In Table 15, we can easily place 1 for $s_4$, $z_4$, $v_6$, $w_6$, $x_6$, and 0 for $w_7$ using $R_1$, $R_7$, $C_6$, and $C_7$, thereby obtaining Table 16.

Substituting Table 16 into Table 17, we can place $t_5 = 0$ and $y_5 = 1$ form $LR_{10} = 3$ and $LR_4 = 2$, respectively.

In Table 17, from $RL_8 = 5$, we can place $v_2 = w_3 = x_4 = 1$, thereby obtaining Table 18.

Considering Table 18, given that $LR_4 = 2$, we can place $t_3 = 0 \Rightarrow t_4 = 1$ because $R_3 = 3$. We also consider $C_2 = 3$, which implies that $w_2 = x_2 = 0$, thereby obtaining Table 19.

In Table 19, given that $LR_5 = 3$, $RL_4 = 2$, and $RL_{10} = 1$, we can place $v_3 = 1$, $v_4 = 0$, and $y_3 = 0$, respectively, in Table 20.

In Table 20, we can fill $v_4 = x_3 = x_5 = 0$ from $R_3$ and $R_5$. We can also place $w_5 = 1$ from $LR_8$. We calculate the remaining values for $w_4$ and $y_4$, thereby obtaining a complete Table 21.
Finite field of data compression at the binary level.

In the field of binary data in mathematics, computer science, and in the future, this new idea can be used in many applications and OSNs. His areas of expertise are data mining, data filling algorithms analysis, information retrieval, data compression, data prediction, and online social networks.

We presented the algorithm of our technique able rows, columns, and diagonals by supposition and cross recalculates all missing values from the aggregates of available values from aggregates in the PP category. Our approach new concept for the recalculation of the entire BIS missing only be predicted in the UP category. We also presented a prediction and re-calculation in incomplete soft sets as BIS.

In this study, we discussed the existing approaches to data prediction and re-calculation in incomplete soft sets as BIS.

Therefore, we successfully calculated all unknowns through our proposed approach in Table 21. Supposing that $P_i$ are the parameters functions for $i = 1, 2, \ldots, 7$, then we have

$$
(F, E) = \begin{cases}
P_1 = \{o_1, o_2, o_5, o_7\} \\
P_2 = \{o_1, o_3, o_6\} \\
P_3 = \{o_4, o_7\} \\
P_4 = \{o_1, o_2, o_5, o_6, o_7\} \\
P_5 = \{o_4, o_6, o_7\} \\
P_6 = \{o_1, o_2, o_3, o_4, o_5\} \\
P_7 = \{o_1, o_3, o_6, o_7\}
\end{cases}
$$

as our required soft set.

V. CONCLUSION

In this study, we discussed the existing approaches to data prediction and re-calculation in incomplete soft sets as BIS. We categorized the previous approaches to PP and UP categories and showed that only preprocessed incomplete soft sets can be used for re-calculation, and missing values can only be predicted in the UP category. We also presented a new concept for the re-calculation of the entire BIS missing values from aggregates in the PP category. Our approach recalculates all missing values from the aggregates of available columns, and diagonals by supposition and cross confirmation. We presented the algorithm of our technique and explained it with an example as a proof of concept. In the future, this new idea can be used in many applications of binary data in mathematics, computer science, and in the field of data compression at the binary level.

REFERENCES


TUTUT HERAWAN received the Ph.D. degree in information technology from Universiti Tun Hussein Onn Malaysia in 2010. He is currently an Associate Professor with the Department of Information Systems, University of Malaya. He is also a Principal Researcher with the Soft Computing and Data Mining Research Center, Universiti Tun Hussein Onn Malaysia, and a Visiting Researcher with the Parallel and Distributed Computing Research Laboratory, Deakin University, Australia. He is also a part-time Lecturer in many universities in Indonesia. He has over 12 years experiences as academic and also successfully supervised five Ph.D. students. He presently supervises 15 Master and Ph.D. students and has examined Master and Ph.D. Theses. His research area includes information retrieval and decision support in information systems. He delivered over 20 keynote addresses, invited workshop, and seminars and has been actively served as the Chair, a Co-Chair, a Program Committee Member, and a Co-Organizer for numerous international conferences/workshops. He is the Founding Chair of SCDM and DaEng conferences with are supported by the renowned publisher Advances in intelligent system and computing and Lecture Notes in electrical engineering of Springer Verlag, respectively. He is the Executive Editor of the MALAYSIAN JOURNAL OF COMPUTER SCIENCE (ISI JCR with IF 0.405). He is a Guest Editor of many special issues in many reputable international journals. He is an Editor of five Springer-series books. He has authored over 200 articles in various book chapters, international journals, and conference proceedings which are indexed in Scopus database (with H-index 14). He is an Active Reviewer of various journals, including the IEEE TRANSACTIONS ON FUZZY SYSTEMS, the Knowledge-Based Systems, the Information Sciences, and the IEEE ACCESS.

AINUDDIN WAHID ABDUL WAHAB received the Ph.D. degree in computing from the University of Surrey, U.K. He is currently a Senior Lecturer with the Department of Computer System and Technology, Faculty of Computer Science and Information technology, University of Malaya, Kuala Lumpur, Malaysia, where he is also the Deputy Dean Research with the Faculty of Computer Science and Information technology. He has over 10 years of experience as Academician and also successfully supervised Ph.D. students. He presently supervises 14 master’s and Ph.D. degree students and has examined Master & Ph.D. Theses. He has authored many articles in academic journals indexed in well reputed databases, such as ISI-indexed and Scopus-indexed.

GHULAM MUJTABA received the master’s degree in computer science from National University, FAST, Karachi, Pakistan. He is also a gold medalist in his master’s degree. He is currently pursuing the Ph.D. degree with the Faculty of Computer Science and Information Technology, University of Malaya, Kuala Lumpur, Malaysia. He was involved in well-known software house in Karachi for four years. He is also an Assistant Professor with the Sukkur Institute of Business Administration, Sukkur, Pakistan, since 2006. He has vast experience in teaching and research. He has authored several articles in academic journals indexed in well reputed databases, such as ISI-indexed and Scopus-indexed.

SANI DANJUMA received the B.Sc. degree from the Kano University of Science and Technology, Wudil, and the M.Sc. degree from the Liaoning University of Technology, China. He is currently pursuing the Ph.D. degree in computer science with the University of Malaya. He has several publications. His research areas are rough set soft set and data mining.

MOHAMMED ALI AL-GARADI received the M.Tech. degree in electronic and communication engineering from JNTU India. He is currently pursuing the Ph.D. degree with the Faculty of Computer Science and Information Technology, University of Malaya, Kuala Lumpur, Malaysia. He has authored several articles in academic journals indexed in well reputed databases, such as ISI-indexed and Scopus-indexed.

* * *