An Adjusted Apriori Algorithm to Itemsets Defined by Tables and an Improved Rule Generator with Three-Way Decisions

Zhiwen Jian[1], Hiroshi Sakai[1](✉), Takuya Ohwa[1], Kao-Yi Shen[2], and Michinori Nakata[3]

[1] Graduate School of Engineering, Kyushu Institute of Technology, Tobata, Kitakyushu 804-8550, Japan
sakai@mns.kyutech.ac.jp

[2] Department of Banking and Finance, Chinese Culture University (SCE), Taipei, Taiwan
kyshen@sce.pccu.edu.tw

[3] Faculty of Management and Information Science, Josai International University, Gumyo, Togane, Chiba 283-0002, Japan
nakatam@ieee.org

Abstract. The NIS-Apriori algorithm, which is extended from the Apriori algorithm, was proposed for rule generation from non-deterministic information systems and implemented in SQL. The realized system handles the concept of certainty, possibility, and three-way decisions. This paper newly focuses on such a characteristic of table data sets that there is usually a fixed decision attribute. Therefore, it is enough for us to handle itemsets with one decision attribute, and we can see that one frequent itemset defines one implication. We make use of these characteristics and reduce the unnecessary itemsets for improving the performance of execution. Some experiments by the implemented software tool in Python clarify the improved performance.

Keywords: Rule generation · The Apriori algorithm · Frequent itemset · Incomplete information · Three-way decisions

1 Introduction

We are following rough set based rule generation from table data sets [10,14,22] and Apriori based rule generation from transaction data sets [1,2,9], and we are investigating a new framework of rule generation from table data sets with information incompleteness [17–21].

Table 1 is a standard table. We term such a table as a *Deterministic Information System* (DIS). In DISs, several rough set based rule generation methods are proposed [3,5,10,14,16,22,23]. Furthermore, *missing values* '?' [6,7,11] (Table 2) and a *Non-deterministic Information System* (NIS) [12,13,15] (Table 3) were also

Table 1. An exemplary DIS ψ.

Object	P	Q	R	S	Dec
$x1$	3	1	2	2	a
$x2$	2	2	2	1	a
$x3$	1	2	2	1	b
$x4$	1	3	3	2	b
$x5$	3	2	3	1	c

Table 2. An exemplary NIS Φ with missing value '?', whose value is one of 1, 2, 3.

Object	P	Q	R	S	Dec
$x1$	3	?	2	2	a
$x2$	2	$\{2, 3\}$	2	?	a
$x3$	?	2	2	$\{1, 2\}$	b
$x4$	1	3	3	2	b
$x5$	3	2	3	?	c

Table 3. An exemplary NIS Φ. Each '?' is replaced with a set $\{1, 2, 3\}$ of possible attribute values.

Object	P	Q	R	S	Dec
$x1$	3	$\{1, 2, 3\}$	2	2	a
$x2$	2	$\{2, 3\}$	2	$\{1, 2, 3\}$	a
$x3$	$\{1, 2, 3\}$	2	2	$\{1, 2\}$	b
$x4$	1	3	3	2	b
$x5$	3	2	3	$\{1, 2, 3\}$	c

investigated to cope with information incompleteness. In [12], question-answering based on possible world semantics was investigated, and an axiom system was given for query translation to one equivalent normal form [12].

In NIS, some attribute values are given as a set of possible attribute values due to information incompleteness. In Tables 2, $\{2, 3\}$ in $x2$ implies '*either 2 or 3 is the actual value, but there is no information to decide it*', and '?' does there is no information. We replace each '?' with all possible attribute values and have Table 3. Thus, we can handle '?' in NIS (some discretization may be necessary for continuous attribute values). Formerly in NISs, question-answering and information retrieval were investigated, and we are coping with rule generation from NISs.

The Apriori algorithm [1] was proposed by Agrawal for handling transaction data sets. We adjust this algorithm to DIS and NIS by using the characteristics of table data sets. The highlight of this paper is the following.

(1) A brief survey of Apriori based rule generation and a rule generator,
(2) Some improvements of the Apriori based algorithm and a rule generator,
(3) Experiment by the improved rule generator in Python.

This paper is organized as follows: Sect. 2 surveys our framework on NISs and the Apriori algorithm [1,2,9]. Section 3 connects table data sets to transaction data sets and copes with the manipulation of candidates of rules. Then, more effective manipulation is proposed in DISs and NISs. Section 4 describes a new NIS-Apriori based system in Python and presents the improved results. Section 5 concludes this paper.

2 Preliminary: An Overview of Rule Generation and Examples

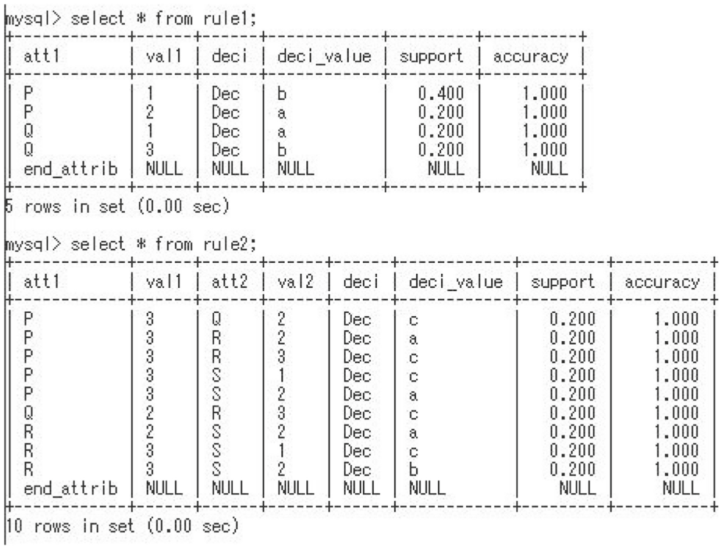
This section briefly reviews rule generation from DISs and NISs.

2.1 Rules and Rule Generation from DISs

In Table 1, we consider implications like $[P, 3] \Rightarrow [Dec, a]$ from $x1$ and $[R, 2] \wedge [S, 1] \Rightarrow [Dec, b]$ from $x3$. Generally, a *rule* is defined as an implication satisfying some constraint. The following is one standard definition of rules [1,2,9,14,22]. We follow this definition and consider the following rule generation from DIS.

(A rule from DIS). A *rule* is an implication τ satisfying $support(\tau) \geq \alpha$ and $accuracy(\tau) \geq \beta$ $(0 < \alpha,\ \beta \leq 1.0)$ for given threshold values α and β.
(Rule generation from DIS). If we fix α and β in DIS, the set of all rules is also fixed, but we generally do not know them. Rule generation is to generate all minimal rules (we term a rule with minimal condition part a *minimal rule*).

Here, $support(\tau)$ is an occurrence ratio of an implication τ for the total objects and $accuracy(\tau)$ is a consistency ratio of τ for the condition part of τ. For example, let us consider $\tau : [R, 2] \wedge [S, 1] \Rightarrow [Dec, b]$ from $x3$. Since τ occurs one time for five objects, we have $support(\tau) = 1/5$. Since $[R, 2] \wedge [S, 1]$ occurs two times, we have $accuracy(\tau) = 1/2$. Fig. 1 shows all minimal rules (redundant rules are not generated) from Table 1.


```
mysql> select * from rule1;
+------------+------+------+------------+---------+----------+
| att1       | val1 | deci | deci_value | support | accuracy |
+------------+------+------+------------+---------+----------+
| P          | 1    | Dec  | b          |   0.400 |    1.000 |
| P          | 2    | Dec  | a          |   0.200 |    1.000 |
| Q          | 1    | Dec  | a          |   0.200 |    1.000 |
| Q          | 3    | Dec  | b          |   0.200 |    1.000 |
| end_attrib | NULL | NULL | NULL       |    NULL |     NULL |
+------------+------+------+------------+---------+----------+
5 rows in set (0.00 sec)

mysql> select * from rule2;
+------------+------+------+------+------+------------+---------+----------+
| att1       | val1 | att2 | val2 | deci | deci_value | support | accuracy |
+------------+------+------+------+------+------------+---------+----------+
| P          | 3    | Q    | 2    | Dec  | c          |   0.200 |    1.000 |
| P          | 3    | R    | 2    | Dec  | a          |   0.200 |    1.000 |
| P          | 3    | R    | 3    | Dec  | c          |   0.200 |    1.000 |
| P          | 3    | S    | 1    | Dec  | c          |   0.200 |    1.000 |
| P          | 3    | S    | 2    | Dec  | a          |   0.200 |    1.000 |
| Q          | 2    | R    | 3    | Dec  | c          |   0.200 |    1.000 |
| R          | 2    | S    | 2    | Dec  | a          |   0.200 |    1.000 |
| R          | 3    | S    | 1    | Dec  | c          |   0.200 |    1.000 |
| R          | 3    | S    | 2    | Dec  | b          |   0.200 |    1.000 |
| end_attrib | NULL | NULL | NULL | NULL | NULL       |    NULL |     NULL |
+------------+------+------+------+------+------------+---------+----------+
10 rows in set (0.00 sec)
```

Fig. 1. The obtained all minimal rules ($support(\tau) \geq 0.2$, $accuracy(\tau) \geq 0.9$) from Table 1. Our system ensures that there is no other rule except them. In the table *rule*1, the first rule is $\tau : [P, 1] \Rightarrow [Dec, b]$. Even though $\tau' : [P, 1] \wedge [Q, 2] \Rightarrow [Dec, b]$ satisfies the constraint of rules, τ' is a redundant implication of τ and τ' is not minimal.

2.2 Rules and Rule Generation from NISs

From now, we employ the symbols Φ and ψ for expressing NIS and DIS, respectively. In NIS Φ, we replace a set of all possible values with an element of this set, and then we have one DIS. We term such a DIS a *derived DIS* from NIS, and let $DD(\Phi)$ denote a set of all derived DISs from NIS. Table 1 is a derived DIS from Table 3. In NISs like Table 3, we consider the following two types of rules,

(1) A rule which we certainly conclude from NIS (a certain rule),
(2) A rule which we may conclude from NIS (a possible rule).

These two types of rules seem to be natural for rule generation with information incompleteness. Yao recalls three-valued logic in rough sets and proposes *three-way decisions* [23,24]. These types of rules concerning missing values were also investigated in [6,11], and we coped with the following two types of rules based on possible world semantics [18,20]. The definition in [6,11] and the following definition are semantically different [18].

(A certain rule from NIS). An implication τ is a *certain rule*, if τ is a rule in each of derived DIS from NIS,
(A possible rule from NIS). An implication τ is a *possible rule*, if τ is a rule in at least one derived DIS from NIS.
(Rule generation from NIS). If we fix α and β in NIS, the set of all certain rules and the set of all possible rules are also fixed. Rule generation is to generate all minimal certain rules and all minimal possible rules.

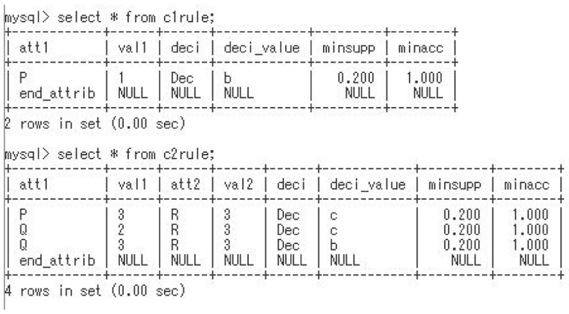
Two types of rules depend on all derived DISs from NIS, and the number of them increases exponentially. For Table 3, the number is 324 ($=2^2 \times 3^4$), and the number is more than 10^{100} for the Mammographic data set [4]. Thus, the realization of a system to handle two types of rules was seemed to be hard, however, we gave one solution to this problem.

(Proved Property). For each implication τ, we developed some formulas to calculate the following,

(1) $minsupp(\tau) = \min_{\psi \in DD(\Phi)}\{support(\tau) \text{ in } \psi\}$,
(2) $minacc(\tau) = \min_{\psi \in DD(\Phi)}\{accuracy(\tau) \text{ in } \psi\}$,
(3) $maxsupp(\tau) = \max_{\psi \in DD(\Phi)}\{support(\tau) \text{ in } \psi\}$,
(4) $maxacc(\tau) = \max_{\psi \in DD(\Phi)}\{accuracy(\tau) \text{ in } \psi\}$.

This calculation employs the rough sets based concept and is independent of the number of derived DISs [18,20,21]. By using these formulas, we proved a method to examine 'τ is a certain rule or not' and 'τ is a possible rule or not'. This method is also independent of the number of all derived DISs [18,20,21].

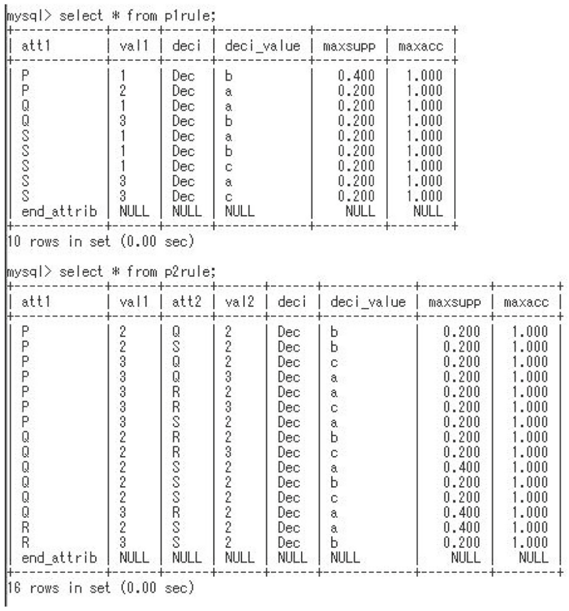
We apply this property to the Apriori algorithm for realizing a rule generation system. The Apriori algorithm effectively enumerates itemsets (candidates of rules), and the *support* and *accuracy* values of every candidate are calculated by the Proved Property. Figures 2 and 3 show the obtained minimal certain rules and minimal possible rules from Table 3. As for the execution time, we discuss it in Sect. 4.



```
mysql> select * from c1rule;
+-------------+------+------+------------+---------+--------+
| att1        | val1 | deci | deci_value | minsupp | minacc |
+-------------+------+------+------------+---------+--------+
| P           | 1    | Dec  | b          |   0.200 |  1.000 |
| end_attrib  | NULL | NULL | NULL       |    NULL |   NULL |
+-------------+------+------+------------+---------+--------+
2 rows in set (0.00 sec)

mysql> select * from c2rule;
+-------------+------+------+------+------+------------+---------+--------+
| att1        | val1 | att2 | val2 | deci | deci_value | minsupp | minacc |
+-------------+------+------+------+------+------------+---------+--------+
| P           | 3    | R    | 3    | Dec  | c          |   0.200 |  1.000 |
| Q           | 2    | R    | 3    | Dec  | c          |   0.200 |  1.000 |
| Q           | 3    | R    | 3    | Dec  | b          |   0.200 |  1.000 |
| end_attrib  | NULL | NULL | NULL | NULL | NULL       |    NULL |   NULL |
+-------------+------+------+------+------+------------+---------+--------+
4 rows in set (0.00 sec)
```

Fig. 2. The obtained all minimal certain rules ($support(\tau) \geq 0.2$, $accuracy(\tau) \geq 0.9$) from Table 3. There is no rule except them.



```
mysql> select * from p1rule;
+-------------+------+------+------------+---------+--------+
| att1        | val1 | deci | deci_value | maxsupp | maxacc |
+-------------+------+------+------------+---------+--------+
| P           | 1    | Dec  | b          |   0.400 |  1.000 |
| P           | 2    | Dec  | a          |   0.200 |  1.000 |
| Q           | 1    | Dec  | a          |   0.200 |  1.000 |
| Q           | 3    | Dec  | b          |   0.200 |  1.000 |
| S           | 1    | Dec  | a          |   0.200 |  1.000 |
| S           | 1    | Dec  | b          |   0.200 |  1.000 |
| S           | 1    | Dec  | c          |   0.200 |  1.000 |
| S           | 3    | Dec  | a          |   0.200 |  1.000 |
| S           | 3    | Dec  | c          |   0.200 |  1.000 |
| end_attrib  | NULL | NULL | NULL       |    NULL |   NULL |
+-------------+------+------+------------+---------+--------+
10 rows in set (0.00 sec)

mysql> select * from p2rule;
+-------------+------+------+------+------+------------+---------+--------+
| att1        | val1 | att2 | val2 | deci | deci_value | maxsupp | maxacc |
+-------------+------+------+------+------+------------+---------+--------+
| P           | 2    | Q    | 2    | Dec  | b          |   0.200 |  1.000 |
| P           | 2    | S    | 2    | Dec  | b          |   0.200 |  1.000 |
| P           | 3    | Q    | 2    | Dec  | c          |   0.200 |  1.000 |
| P           | 3    | Q    | 3    | Dec  | a          |   0.200 |  1.000 |
| P           | 3    | R    | 2    | Dec  | a          |   0.200 |  1.000 |
| P           | 3    | R    | 3    | Dec  | c          |   0.200 |  1.000 |
| P           | 3    | S    | 2    | Dec  | a          |   0.200 |  1.000 |
| Q           | 2    | R    | 2    | Dec  | b          |   0.200 |  1.000 |
| Q           | 2    | R    | 3    | Dec  | c          |   0.200 |  1.000 |
| Q           | 2    | S    | 2    | Dec  | a          |   0.400 |  1.000 |
| Q           | 2    | S    | 2    | Dec  | b          |   0.200 |  1.000 |
| Q           | 2    | S    | 2    | Dec  | c          |   0.200 |  1.000 |
| Q           | 3    | R    | 2    | Dec  | a          |   0.400 |  1.000 |
| R           | 2    | S    | 2    | Dec  | a          |   0.400 |  1.000 |
| R           | 3    | S    | 2    | Dec  | b          |   0.200 |  1.000 |
| end_attrib  | NULL | NULL | NULL | NULL | NULL       |    NULL |   NULL |
+-------------+------+------+------+------+------------+---------+--------+
16 rows in set (0.00 sec)
```

Fig. 3. The obtained all minimal possible rules ($support(\tau) \geq 0.2$, $accuracy(\tau) \geq 0.9$) from Table 3.There is no rule except them.

2.3 A Relation Between Rules in DISs and Rules in NISs

Let ψ^{actual} be a derived DIS with actual information from NIS Φ (we cannot decide ψ^{actual} from Φ, but we suppose there is an actual ψ^{actual} for Φ), then we can easily have the next inclusion relation.

$$\{\tau \mid \tau \text{ is a certain rule in } \Phi\} \subseteq \{\tau \mid \tau \text{ is a rule in } \psi^{actual}\}$$
$$\subseteq \{\tau \mid \tau \text{ is a possible rule in } \Phi\}$$

Due to information incompleteness, we know lower and upper approximations of a set of rules in ψ^{actual}. This property follows the concept of rough sets based approximations.

2.4 The Apriori Algorithm for Transaction Data Sets

Let us consider Table 4, which shows four persons' purchase of items. Such structured data is termed a *transaction data set*. In this data set, let us focus on a set $\{ham, beer\}$. Such a set is generally termed an *itemset*. For this itemset, we consider two implications $\tau_1 : ham \Rightarrow beer$ and $\tau_2 : beer \Rightarrow ham$. In τ_1, $support(\tau_1) = 3/4$ and $accuracy(\tau_1) = 3/3$. In τ_2, $support(\tau_2) = 3/4$ and $accuracy(\tau_2) = 3/4$. For an itemset $\{ham, beer, corn\}$, we consider six implications, $ham \wedge beer \Rightarrow corn, \cdots, beer \Rightarrow corn \wedge ham$. Like this, Agrawal proposed a method to obtain rules from transaction data sets, which is known as the Apriori algorithm [1,2,9]. This algorithm makes use of the following.

Table 4. An exemplary transaction data set

Transaction	Items
1	bread, milk, ham, beer, corn
2	cheese, ham, beer
3	ham, beer, apple, potato, corn
4	cheese, cake, beer

(Monotonicity of *support*). For two itemsets P and Q, if $P \subseteq Q$, $support(Q) \leq support(P)$ holds.

By using this property, the Apriori algorithm enumerates all itemsets, which satisfy $support \geq \alpha$. Each of such itemsets is termed a *frequent itemset*. Let us consider the manipulation of itemsets in Table 4 under $support \geq 0.5$. Since there are four transactions, each itemset must occur more than two times. Let CAN_i and FI_i $(i \geq 0)$ denote a set of all candidates of itemsets and a set of all frequent itemsets consisting of $(i+1)$-items, respectively. We have the following.

$$\begin{aligned}
CAN_0 &= \{\{bread\}(\text{Occurrence=1}), \{milk\}(1), \{ham\}(3), \{beer\}(4), \{corn\}(2),\\
&\quad \{cheese\}(2), \{apple\}(1), \{potato\}(1), \{cake\}(1)\},\\
FI_0 &= \{\{ham\}(3), \{beer\}(4), \{corn\}(2), \{cheese\}(2)\},\\
CAN_1 &= \{\{ham, beer\}, \{ham, corn\}, \{ham, cheese\}, \{beer, corn\},\\
&\quad \{beer, cheese\}, \{corn, cheese\}\},\\
FI_1 &= \{\{ham, beer\}(3), \{ham, corn\}(2), \{beer, corn\}(2), \{beer, cheese\}(2)\},\\
CAN_2 &= \{\{ham, beer, corn\}, \{ham, beer, cheese\}, \{ham, corn, cheese\},\\
&\quad \{beer, corn, cheese\}\},\\
FI_2 &= \{\{ham, beer, corn\}(2)\}.
\end{aligned}$$

Each element in CAN_i $(i \geq 1)$ is generated by the combination of two itemsets in FI_{i-1} [1,2]. Then, every itemset satisfying the *support* condition becomes the element of FI_i. For example, for $A : \{ham, corn\}$, $B : \{beer, cheese\} \in FI_1$, we add one element of B to A and have $\{ham, corn, beer\}, \{ham, corn, cheese\} \in CAN_2$. We also do the converse and have $\{beer, cheese, ham\}, \{beer, cheese, corn\} \in CAN_2$. Only one itemset $\{ham, corn, beer\}$ satisfies the *support* condition and becomes an element of FI_2. Like this, FI_1, FI_2, $\cdots$, FI_n are obtained at first, then the *accuracy* value of each implication defined by a frequent itemset is evaluated. In the subsequent sections, we change the above manipulation by using the characteristics of table data sets.

3 Some Improvements of the NIS-Apriori Based Rule Generator

We describe the improvements in our framework based on Sect. 2.

3.1 From Transaction Data Sets to Table Data Sets

We translate Table 1 to Table 5 and identify each descriptor with an item. Then, we can see that Table 5 is a transaction data set. Thus, we can apply the Apriori algorithm to rule generation.

Table 5. A transaction data set for DIS ψ in Table 1.

Object	Descriptors as items
$x1$	[P,3], [Q,1], [R,2], [S,2], [Dec,a]
$x2$	[P,2], [Q,2], [R,2], [S,1], [Dec,a]
$x3$	[P,1], [Q,2], [R,2], [S,1], [Dec,b]
$x4$	[P,1], [Q,3], [R,3], [S,2], [Dec,b]
$x5$	[P,3], [Q,2], [R,3], [S,1], [Dec,c]

We define the next sets IMP_1, IMP_2, $\cdots$, IMP_n.

$$IMP_1 = \{[A, val_A] \Rightarrow [Dec, val]\},$$
$$IMP_2 = \{[A, val_A] \wedge [B, val_B] \Rightarrow [Dec, val]\},$$
$$IMP_3 = \{[A, val_A] \wedge [B, val_B] \wedge [C, val_C] \Rightarrow [Dec, val]\},$$

Here, IMP_i means a set of implications which consist of i-condition attributes. A minimal rule is an implication $\tau \in \cup_i IMP_i$, and we may examine each $\tau \in \cup_i IMP_i$. However, in the subsequent sections, we consider some effective manipulations to generate minimal rules in IMP_1, IMP_2, $\cdots$, sequentially.

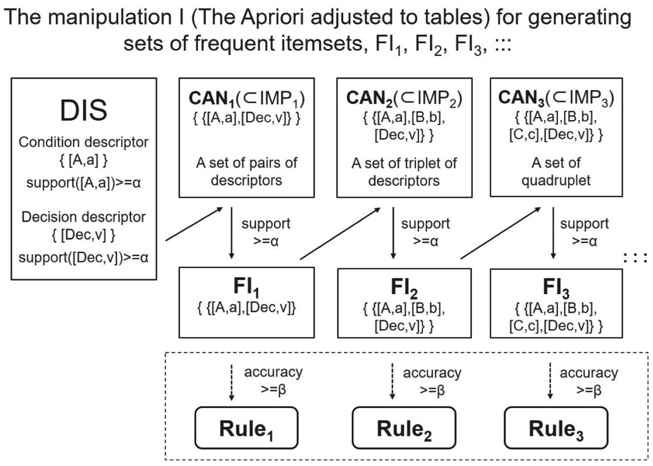

The manipulation I (The Apriori adjusted to tables) for generating sets of frequent itemsets, FI1, FI2, FI3, :::
DIS
Condition descriptor { [A,a] } support([A,a])>=α
Decision descriptor { [Dec,v] } support([Dec,v])>=α
CAN1(⊂IMP1) { {[A,a],[Dec,v]} } A set of pairs of descriptors
CAN2(⊂IMP2) { {[A,a],[B,b], [Dec,v]} } A set of triplet of descriptors
CAN3(⊂IMP3) { {[A,a],[B,b], [C,c],[Dec,v]} } A set of quadruplet
support >=α
FI1 { {[A,a],[Dec,v]}
FI2 { {[A,a],[B,b], [Dec,v]} }
FI3 { {[A,a],[B,b], [C,c],[Dec,v]} }
accuracy >=β
Rule1
Rule2
Rule3

Fig. 4. The manipulation I for itemsets.

3.2 The Manipulation I for Frequent Itemsets by the Characteristics of Table Data Sets

Here, we make use of the characteristics of table data sets below.

(TA1). The decision attribute *Dec* is fixed. So, it is enough to consider each itemset including one descriptor whose attribute is *Dec*. For example, we do not handle any itemset like $\{[P,3],[Q,2]\}$ nor $\{[P,3],[Dec,a],[Dec,b]\}$ in Table 5.
(TA2). An attribute is related to each descriptor. So, we handle itemsets with different attributes. For example, we do not handle any itemset like $\{[P,3],[P,1],[Q,2],[Dec,b]\}$ in Table 5.
(TA3). To consider implications, we handle CAN_1, FI_1 ($\subseteq IMP_1$), CAN_2, FI_2 ($\subseteq IMP_2$), $\cdots$, which are defined in Sect. 2.4.

Based on the above characteristics, we can consider Fig. 4. In Fig. 4, itemsets satisfying (TA1) and (TA2) are enumerated. Generally, in the Apriori algorithm, the *accuracy* value is examined after obtaining all FI_i, because the decision attribute is not fixed. For each set in FI_i, there are plural implications. However, in a table data set, one implication corresponds to a frequent itemset. We employed this property and proposed the Apriori algorithm adjusted to table data sets [20,21] in Fig. 5. We term this algorithm the *DIS-Apriori algorithm*. Here, we calculate the *accuracy* value of every frequent itemset in each while loop (the rectangle area circled by the dotted line in Fig. 4 and lines 5-7 in Fig. 5). We can easily handle certain rules and possible rules in NISs by extending the DIS-Apriori algorithm.

```
Input: Table data set DIS $\psi$, decision attribute $Dec$, threshold values $\alpha$, $\beta$.
Output: A set $Rule(\psi)$ of minimal rules.
 1: $Rule(\psi) \leftarrow \{\}$; $i \leftarrow 1$;
 2: create $FI_1$=$\{\{[A,a],[Dec,v]\} | support([A,a] \Rightarrow [Dec,v]) \geq \alpha\}$ from $CAN_1$;
 3: while $(|FI_i| \geq 1)$ do
 4:     $Rest_i \leftarrow \{\}$; $Rule_i \leftarrow \{\}$;
 5:     for all $\tau_{i,j} \in FI_i$ do
 6:         if $accuracy(\tau_{i,j}) \geq \beta$ then $\underline{\text{add } \tau_{i,j} \text{ to } Rule_i;}$ else $\underline{\text{add } \tau_{i,j} \text{ to } Rest_i;}$
 7:         end if
 8:     end for
 9:     remove redundant implications from $Rule_i$;
10:     $i \leftarrow i+1$; create $FI_i$;
11: end while
12: return $Rule(\psi)$=$\cup_{k<i} Rule_k$
```

Fig. 5. The Apriori algorithm adjusted to table data set DIS ψ. We can examine the *accuracy* value in each while loop (the rectangle area circled by the dotted line in Fig. 4). This examination is not done in the Apriori algorithm for transaction data sets.

Proposition 1. *[20, 21]*

(1) We replace DIS ψ with NIS Φ, support and accuracy with minsupp and minacc, respectively. Then, this algorithm generates all minimal certain rules.
(2) We replace DIS ψ with NIS Φ, support and accuracy with maxsupp and maxacc, respectively. Then, this algorithm generates all minimal possible rules.
(3) We term the algorithm consisting of (1) and (2) the NIS-Apriori algorithm.

Both DIS-Apriori and NIS-Apriori algorithms are logically sound and complete for rules. They generate rules without excess and deficiency.

Figures 1, 2 and 3 by the rule generator in SQL are based on the algorithm in Fig. 5 and Proposition 1.

3.3 The Manipulation II for Frequent Itemsets by the Characteristics of Table Data Sets

Now, we advance the manipulation I to the manipulation II. We focus on the statement 'create FI_i' in lines 2 and 10 in Fig. 5. In every while loop, we examine each $\tau \in FI_i \subseteq CAN_i \subseteq IMP_i$, so to reduce sets CAN_i and FI_i will influence the performance of execution. In Fig. 5, we at first need to remark the following.

(Rule generation). The purpose of rule generation is to generate each minimal implication $\tau \in \cup_i IMP_i$ satisfying $support(\tau) \geq \alpha$ and $accuracy(\tau) \geq \beta$. We obtain $Rule_1, Rest_1 \subseteq IMP_1$ in the 1st while loop, $Rule_2, Rest_2 \subseteq IMP_2$ in the 2nd while loop, and $Rule_3, Rest_3$ in the 3rd while loop, $\cdots$.

(Relation between sets in Fig. 5). We clarify the relation and the definition of $NOrule_i$ below.

(1) $Rule_i = \{\tau \in IMP_i \mid support(\tau) \geq \alpha,\ accuracy(\tau) \geq \beta\}$,
(2) $Rest_i = \{\tau \in IMP_i \mid support(\tau) \geq \alpha,\ accuracy(\tau) < \beta\}$,
(3) $FI_i = \{\tau \in IMP_i \mid support(\tau) \geq \alpha\}$,
(4) $NOrule_i = \{\tau \in IMP_i \mid support(\tau) < \alpha\}$,
(5) $IMP_i = FI_i \cup NOrule_i = (Rule_i \cup Rest_i) \cup NOrule_i$.

(A case of $\tau \in Rule_i$). If $\tau : \wedge_j [A_j, val_j] \Rightarrow [Dec, val] \in Rule_i$, we do not deal with any redundant implication $\tau' : (\wedge_j [A_j, val_j]) \wedge [B, b] \Rightarrow [Dec, val] \in IMP_{i+1}$, because τ' cannot be a minimal rule.
(A case of $\tau \in NOrule_i$). If $\tau : \wedge_j [A_j, val_j] \Rightarrow [Dec, val] \in NOrule_i$, any redundant implication $\tau' : (\wedge_j [A_j, val_j]) \wedge [B, b] \Rightarrow [Dec, val]$ satisfies $support(\tau') < \alpha$. So, $\tau' \in IMP_{i+1}$ cannot be a rule. Thus, we do not deal with any redundant implication τ'.
(A case of $\tau \in Rest_i$). In the *accuracy* value, the monotonicity like *support* does not hold (an example is in [20]). Thus, if $\tau : \wedge_j [A_j, val_j] \Rightarrow [Dec, val] \in Rest_i$, $accuracy(\tau') \geq \beta$ may hold for a redundant implication $\tau' : (\wedge_j [A_j, val_j]) \wedge [B, b] \Rightarrow [Dec, val] \in FI_{i+1}$.

Proposition 2. *Let us suppose that we had $Rule_i$ and $Rest_i$ (IMP_i=$Rule_i \cup Rest_i \cup NOrule_i$) in the i-th while loop in Fig. 5. Every candidate of a minimal rule in IMP_{i+1} is a redundant implication of $\tau \in Rest_i$.*

($Proof$)
For every implication $\tau \notin FI_i \subseteq IMP_i$, its redundant implication τ' satisfies $support(\tau') \leq support(\tau) < \alpha$. Thus, τ' cannot be a minimal rule in IMP_{i+1}. Based on the Apriori algorithm, we need to combine two frequent itemsets in FI_i=$Rule_i \cup Rest_i$ (an example of this combination is described in Sect. 2.4). However, for the minimality condition of rules, we do not handle any redundant implication of $\tau \in Rule_i$. Thus, we conclude that every candidate of a minimal rule in IMP_{i+1} is a redundant implication of $\tau \in Rest_i$.

Definition 1. *We define a set $RCAN_i$ ($\subseteq CAN_i$), whose element is a candidate of a minimal rule in IMP_i w.r.t. rules $\cup_{j=1,\cdots,(i-1)} Rule_j$ and a set $RFI_i = \{\tau \in RCAN_i \mid support(\tau) \geq \alpha\}$ ($\subseteq FI_i \subseteq IMP_i$).*

In the Apriori algorithm, the concept of redundancy is not introduced, so that some redundant rules may be generated. The sets CAN_i and FI_i in Fig. 4 are generated from FI_{i-1} (=$Rule_{i-1} \cup Rest_{i-1}$). However, we can generate $RCAN_i (\subseteq CAN_i)$ and $RFI_i (\subseteq FI_i)$ from $Rest_{i-1}$. Furthermore, we previously generated itemsets $\{[A, a], [B, b], [Dec, v1]\}, \{[A, a], [B, b], [Dec, v2]\} \in RCAN_2$ from $\{[A, a], [Dec, v1]\}, \{[B, b], [Dec, v2]\} \in Rest_1$, and we removed this combination, because there is no object satisfying both $[Dec, v1]$ and $[Dec, v2]$. This combination formerly generated meaningless itemsets. This revision is another improvement in the manipulation of itemsets.

Proposition 3. *The set $RCAN_i$ and RFI_i are given as follows:*

$$\begin{aligned}
&(i=1) RCAN_1 = CAN_1 \text{ and } RFI_1 = FI_1,\\
&(i \geq 2) RCAN_i = \{\tau : (\wedge_j[A_j, val_j]) \wedge [B,b] \Rightarrow [Dec, val] \mid\\
&\quad \wedge_j[A_j, val_j] \Rightarrow [Dec, val] \in Rest_{i-1}, [B,b] \Rightarrow [Dec, val] \in Rest_1\},\\
&\quad RFI_i = \{\tau \in RCAN_i \mid support(\tau) \geq \alpha\}.
\end{aligned}$$

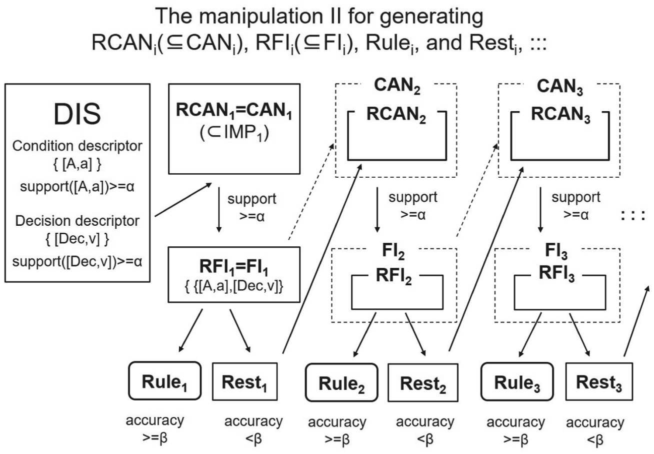


Fig. 6. New manipulation II of itemsets. We can handle $RCAN_i \subseteq CAN_i$ and $RFI_i \subseteq FI_i$ for generating minimal rules. In the Apriori algorithm, CAN_i and FI_i are employed, so redundant rules may be generated. By using $RCAN_i$ and RFI_i, the candidates of rules are reduced, and the performance of execution is improved.

(Proof)
(In case of $i=1$) $RCAN_1 = CAN_1$ and $RFI_1 = FI_1$ hold, because redundant rules occur after 2nd while loop.
(In case of $i \geq 2$) We add one descriptor $[B,b]$ to $\wedge_j[A_j, val_j] \Rightarrow [Dec, val] \in Rest_{i-1}$ and have a redundant implication $\tau : (\wedge_j[A_j, val_j]) \wedge [B,b] \Rightarrow [Dec, val] \in IMP_i$ due to Proposition 2.

(1) In order to handle the same decision, $[B,b]$ must be the condition part of $\tau' : [B,b] \Rightarrow [Dec, val] \in RFI_1 = FI_1$. (If $\tau' \notin FI_1$, $support(\tau) < \alpha$ holds and τ cannot be a rule, because τ is a redundant implication of τ').
(2) $FI_1 = Rule_1 \cup Rest_1$ holds. If $\tau' \in Rule_1$, τ cannot be a minimal rule, because τ' is a minimal rule.

Based on the above discussion, we conclude $\tau' \in Rest_1$.

We propose the manipulation II in Fig. 6 due to the above propositions. In the Apriori algorithm, CAN_i is generated by FI_{i-1}, but we can remove redundant implications of $\tau \in Rule_{i-1}$. Thus, we can handle $RCAN_i$, which is a subset of CAN_i. If the number of elements in $Rule_{i-1}$ is large, the number of elements in $RCAN_i$ will be much smaller than that of CAN_i.

Proposition 4. *The DIS-Apriori algorithm with the manipulation II is sound and complete for minimal rules in DIS, and the NIS-Apriori algorithm with the manipulation II is also sound and complete for minimal certain rules and minimal possible rules in NIS. They do not miss any rule defined in DIS ψ or NIS Φ.*

(*Sketch of Proof*). *We have proved that the DIS-Apriori and NIS-Apriori algorithms are sound and complete [20,21]. We newly introduced sets $RCAN_i \subseteq CAN_i$ and $RFI_i \subseteq FI_i$ by using the redundancy of rules, and we extended the previous two algorithms to those with the manipulation II. The proposed algorithm does not examine each $\tau \in \cup_j IMP_j$, but examines each $\tau \in \cup_j RCAN_j$. As a result, this algorithm generates the same rules defined by the procedure 'to examine each $\tau \in \cup_j IMP_j$'.*

4 An Improved Apriori Based Rule Generator and Some Experiments

This section compares the NIS-Apriori algorithm and the NIS-Apriori algorithm with the manipulation II. Of course, two algorithms generate the same rules due to Propositions 1 and 4, and the latter algorithm makes use of the redundancy concept. We newly implemented two systems in Python (Windows PC, CPU: Intel i7-4600U, 2.7 z). Table 6 shows the results on the Car Evaluation data set [4], and Table 7 does the results on the Phishing data set [4]. They are the cases of DISs, and the characteristic of $RCAN_i \subseteq CAN_i$ is effectively employed.

Now, we show two examples by the NIS-Apriori algorithm. The one is the Congressional Voting data set [4], and the other is the Lithology data set [8].

Table 6. The Car Evaluation data set (Objects: 1728, condition attributes: 6). A:$|Rule_1|$, B:$|CAN_2|$ or $|RCAN_2|$, C:$|Rule_2|$, D:$|CAN_3|$ or $|RCAN_3|$, E:$|Rule_3|$, F:$|CAN_4|$ or $|RCAN_4|$, G:$|Rule_4|$.

CASE	Manipulation	Time (sec)	A	B	C	D	E	F	G
$support \geq 0.2$	I	0.037	5	24	0	0	0	0	0
$accuracy \geq 0.7$	II	0.027	5	2	0	0	0	0	0
$support \geq 0.1$	I	0.096	8	366	0	27	0	0	0
$accuracy \geq 0.7$	II	0.059	8	74	0	0	0	0	0
$support \geq 0.05$	I	0.189	8	366	0	1694	0	0	0
$accuracy \geq 0.7$	II	0.123	8	176	0	572	0	0	0
$support \geq 0.01$	I	0.621	8	732	0	3388	1	6588	0
$accuracy \geq 0.7$	II	0.329	8	349	0	1172	1	1840	0

Table 7. The Phishing data set (Objects: 1353, condition attributes: 9). Here, A, B, $\cdots$, G are the same as Table 6.

CASE	Manipulation	Time (sec)	A	B	C	D	E	F	G
$support \geq 0.2$	I	0.139	3	148	2	276	0	15	0
$accuracy \geq 0.7$	II	0.083	3	25	2	30	0	0	0
$support \geq 0.1$	I	0.847	6	426	13	2380	1	5774	0
$accuracy \geq 0.7$	II	0.291	6	167	13	552	1	1101	0
$support \geq 0.05$	I	1.409	7	831	23	5355	9	12438	2
$accuracy \geq 0.7$	II	0.647	7	285	23	1259	9	3508	2
$support \geq 0.01$	I	2.532	7	831	30	5355	25	22113	11
$accuracy \geq 0.7$	II	1.522	7	583	30	3118	25	10611	11

Table 8. The Congressional Voting data set (Objects: 435, condition attributes: 16). There are 392 missing values, thus $|DD(\Phi)| = 2^{392} \geq 10^{100}$ (the number of derived DISs exceeds 10^{100}). A certain rule is a rule in each of more than 10^{100} derived DISs. A possible rule is a rule in at least one derived DISs. Here, A, B, $\cdots$, G are the same as Table 6.

CASE	Manipulation	Time (sec)	A	B	C	D	E	F	G
$support \geq 0.2$ $accuracy \geq 0.6$	I (certain rule)	23.73	23	900	6	8120	0	50960	0
	II (certain rule)	0.12	23	50	6	77	0	0	0
	I (possible rule)	23.56	28	960	3	8120	0	50960	0
	II (possible rule)	0.12	28	41	3	30	0	0	0
$support \geq 0.1$ $accuracy \geq 0.6$	I (certain rule)	26.35	23	960	6	8960	0	58240	0
	II (certain rule)	0.81	23	132	6	448	0	1064	0
	I (possible rule)	26.72	29	960	7	8960	2	58240	0
	II (possible rule)	0.52	29	100	7	290	2	580	0
$support \geq 0.05$ $accuracy \geq 0.6$	I (certain rule)	26.59	23	960	6	8960	0	58240	0
	II (certain rule)	1.79	23	220	6	949	0	2788	0
	I (possible rule)	27.29	29	960	7	8960	2	58240	0
	II (possible rule)	1.84	29	223	7	984	2	2967	0
$support \geq 0.01$ $accuracy \geq 0.6$	I (certain rule)	27.46	23	960	6	8960	0	58240	0
	II (certain rule)	4.28	23	354	6	1981	0	7630	0
	I (possible rule)	28.71	29	960	7	8960	2	58240	0
	II (possible rule)	3.59	29	296	7	1599	2	6141	0

As we described in Proposition 1, the NIS-Apriori algorithm (certain rule generation) is the DIS-Apriori algorithm with criterion values *minsupp* and *minacc*. Thus, the number of candidates of itemsets is also reduced by the manipulation II. The experiments easily examine the advancement of the manipulation II (Tables 8 and 9).

Table 9. The Lithology data set (Objects: 1923, condition attributes: 10). There are 519 missing values, therefore there are more than 10^{100} ($2^{519} \fallingdotseq (2^{10})^{50} > (10^3)^{50} > 10^{100}$) derived DISs. Here, A, B, $\cdots$, G are the same as Table 6.

CASE	Manipulation	Time (sec)	A	B	C	D	E	F	G
$support \geq 0.2$ $accuracy \geq 0.5$	I (certain rule)	0.18	11	54	0	120	0	210	0
	II (certain rule)	0.06	11	0	0	0	0	0	0
	I (possible rule)	0.2	11	54	0	156	0	210	0
	II (possible rule)	0.07	11	0	0	0	0	0	0
$support \geq 0.1$ $accuracy \geq 0.5$	I (certain rule)	0.43	17	127	0	464	0	985	0
	II (certain rule)	0.06	17	0	0	0	0	0	0
	I (possible rule)	0.51	17	127	0	549	0	1521	0
	II (possible rule)	0.06	17	0	0	0	0	0	0
$support \geq 0.05$ $accuracy \geq 0.5$	I (certain rule)	0.84	18	900	0	1228	0	3657	0
	II (certain rule)	0.06	18	36	0	4	0	0	0
	I (possible rule)	1.26	19	1122	0	4128	0	4535	0
	II (possible rule)	0.08	19	76	0	97	0	0	0
$support \geq 0.01$ $accuracy \geq 0.5$	I (certain rule)	17.05	23	6055	7	44940	21	222420	14
	II (certain rule)	4.18	23	1185	7	7772	21	36799	14
	I (possible rule)	48.87	39	8806	27	116466	37	755202	34
	II (possible rule)	6.45	39	1413	27	9804	37	48932	34

5 Concluding Remarks

We recently adjusted the Apriori algorithm to table data sets and proposed the DIS-Apriori and NIS-Apriori algorithms. This paper makes use of the characteristics of table data sets (one decision attribute Dec is fixed) and improved these algorithms. If we do not handle table data sets, there was no necessity for considering Fig. 6. The framework of the manipulation II (Fig. 6) is an improvement of Apriori based rule generation by using the characteristics of table data sets. We can generate minimal rules by using $RCAN_i \subseteq CAN_i$ and $RFI_i \subseteq FI_i$. This reduction causes to reduce the candidates of itemsets. We newly implemented the proposed algorithm in Python and examined the improvement of the performance of execution by experiments.

Acknowledgment. The authors would be grateful to the anonymous referees for their useful comments. This work is supported by JSPS (Japan Society for the Promotion of Science) KAKENHI Grant Number JP20K11954.

References

1. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: Proceedings of VLDB 1994, pp. 487–499. Morgan Kaufmann (1994)
2. Agrawal, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A.I.: Fast discovery of association rules. In: Advances in Knowledge Discovery and Data Mining, pp. 307–328. AAAI/MIT Press (1996)

3. Ciucci, D., Flaminio, T.: Generalized rough approximations in PI 1/2. Int. J. Approx. Reason. **48**(2), 544–558 (2008)
4. Frank, A., Asuncion, A.: UCI machine learning repository. School of Information and Computer Science, University of California, Irvine (2010). http://mlearn.ics.uci.edu/MLRepository.html. Accessed 10 July 2019
5. Greco, S., Matarazzo, B., Słowiński, R.: Granular computing and data mining for ordered data: the dominance-based rough set approach. In: Meyers, R.A. (ed.) Encyclopedia of Complexity and Systems Science, pp. 4283–4305. Springer, New York (2009). https://doi.org/10.1007/978-0-387-30440-3
6. Grzymała-Busse, J.W., Werbrouck, P.: On the best search method in the LEM1 and LEM2 algorithms. In: Orłowska, E. (ed.) Incomplete Information: Rough Set Analysis. Studies in Fuzziness and Soft Computing, vol. 13, pp. 75–91. Springer, Heidelberg (1998). https://doi.org/10.1007/978-3-7908-1888-8_4
7. Grzymala-Busse, J.W.: Data with missing attribute values: generalization of indiscernibility relation and rule induction. In: Peters, J.F., Skowron, A., Grzymała-Busse, J.W., Kostek, B., Świniarski, R.W., Szczuka, M.S. (eds.) Transactions on Rough Sets I. LNCS, vol. 3100, pp. 78–95. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-27794-1_3
8. Hossain, T.M., Watada, J., Hermana, M., Shukri, S.R., Sakai, H.: A rough set based rule induction approach to geoscience data. In: Proceedings of UMSO 2018. IEEE (2018). https://doi.org/10.1109/UMSO.2018.8637237
9. Jovanoski, V., Lavrač, N.: Classification rule learning with APRIORI-C. In: Brazdil, P., Jorge, A. (eds.) EPIA 2001. LNCS (LNAI), vol. 2258, pp. 44–51. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45329-6_8
10. Komorowski, J., Pawlak, Z., Polkowski, L., Skowron, A.: Rough sets: a tutorial. In: Pal, S.K., Skowron, A. (eds.) Rough Fuzzy Hybridization: A New Method for Decision Making, pp. 3–98. Springer, Heidelberg (1999)
11. Kryszkiewicz, M.: Rules in incomplete information systems. Inf. Sci. **113**(3–4), 271–292 (1999)
12. Lipski, W.: On databases with incomplete information. J. ACM **28**(1), 41–70 (1981)
13. Orłowska, E., Pawlak, Z.: Representation of nondeterministic information. Theoret. Comput. Sci. **29**(1–2), 27–39 (1984)
14. Pawlak, Z.: Rough sets. Int. J. Comput. Inf. Sci. **11**(5), 341–356 (1982)
15. Pawlak, Z.: Systemy Informacyjne: Podstawy Teoretyczne. WNT (1983). (in Polish)
16. Riza, L.S., et al.: Implementing algorithms of rough set theory and fuzzy rough set theory in the R package RoughSets. Inf. Sci. **287**(10), 68–89 (2014)
17. Sakai, H., Ishibashi, R., Koba, K., Nakata, M.: Rules and apriori algorithm in non-deterministic information systems. Trans. Rough Sets **9**, 328–350 (2008)
18. Sakai, H., Wu, M., Nakata, M.: Apriori-based rule generation in incomplete information databases and non-deterministic information systems. Fundam. Inf. **130**(3), 343–376 (2014)
19. Sakai, H.: Execution logs by RNIA software tools. http://www.mns.kyutech.ac.jp/~sakai/RNIA. Accessed 10 July 2019
20. Sakai, H., Nakata, M.: Rough set-based rule generation and Apriori-based rule generation from table data sets: a survey and a combination. CAAI Trans. Intell. Technol. **4**(4), 203–213 (2019)
21. Sakai, H., Nakata, M., Watada, J.: NIS-Apriori-based rule generation with three-way decisions and its application system in SQL. Inf. Sci. **507**, 755–771 (2020)

22. Skowron, A., Rauszer, C.: The discernibility matrices and functions in information systems. In: Słowiński, R. (ed.) Intelligent Decision Support - Handbook of Advances and Applications of the Rough Set Theory, pp. 331–362. Kluwer Academic Publishers, Berlin (1992)
23. Yao, Y.Y.: Three-way decisions with probabilistic rough sets. Inf. Sci. **180**, 314–353 (2010)
24. Hu, M., Yao, Y.: Structured approximations as a basis for three-way decisions in rough set theory. Knowl.-Based Syst. **165**, 92–109 (2019)