Rules and Apriori Algorithm in Non-deterministic Information Systems

Hiroshi Sakai[1], Ryuji Ishibashi[1], Kazuhiro Koba[1] and Michinori Nakata[2]

[1] Mathematical Sciences Section, Department of Basic Sciences,
Faculty of Engineering, Kyushu Institute of Technology,
Tobata, Kitakyushu 804, Japan
sakai@mns.kyutech.ac.jp
[2] Faculty of Management and Information Science,
Josai International University,
Gumyo, Togane, Chiba 283, Japan
nakatam@ieee.org

Abstract. This paper presents a framework of rule generation in *Non-deterministic Information Systems* (*NISs*), which follows rough sets based rule generation in *Deterministic Information Systems* (*DISs*). Our previous work about *NISs* coped with *certain rules*, *minimal certain rules* and *possible rules*. These rules are characterized by the concept of *consistency*. This paper relates possible rules to *rules* by the criteria *support* and *accuracy* in *NISs*. On the basis of the information incompleteness in *NISs*, it is possible to define new criteria, i.e., *minimum support*, *maximum support*, *minimum accuracy* and *maximum accuracy*. Then, two strategies of rule generation are proposed based on these criteria. The first strategy is *Lower Approximation strategy*, which defines rule generation under the worst condition. The second strategy is *Upper Approximation strategy*, which defines rule generation under the best condition. To implement these strategies, we extend *Apriori* algorithm in *DISs* to *Apriori* algorithm in *NISs*. A prototype system is implemented, and this system is applied to some data sets with incomplete information.

Keywords: Rough sets, Non-deterministic information, Incomplete information, Rule generation, Lower and upper approximations, Apriori algorithm.

1 Introduction

Rough set theory has been used as a mathematical tool of soft computing for approximate two decades. This theory usually handles tables with deterministic information. Many applications of this theory, such as rule generation, machine learning and knowledge discovery, have been presented [5, 9, 15, 21–25, 36, 38].

We follow rule generation in *Deterministic Information Systems* (*DISs*) [21–24, 33], and we describe rule generation in *Non-deterministic Information Systems* (*NISs*). *NISs* were proposed by Pawlak [21], Orłowska [19, 20] and

Lipski [13, 14] to handle information incompleteness in $DISs$, like null values, unknown values, missing values. Since the emergence of incomplete information research, $NISs$ have been playing an important role. Therefore, rule generation in $NISs$ will also be an important framework for rule generation from incomplete information.

The following shows some important researches on rule generation from incomplete information. In [13, 14], Lipski showed a question-answering system besides an axiomatization of logic, and Orłowska established rough set analysis for non-deterministic information [3, 19, 20]. Grzymala-Busse developed a system named $LERS$ which depends upon $LEM1$ and $LEM2$ algorithms [5–7], and recently proposed four interpretations of missing attribute values [8]. Stefanowski and Tsoukias defined non symmetric similarity relations and valued tolerance relations for analyzing incomplete information [34, 35]. Kryszkiewicz proposed a framework of rules in incomplete information systems [10–12]. According to authors' knowledge, these are the most important researches on incomplete information. We have also discussed several issues related to non-deterministic information and incomplete information [16–18], and proposed a framework named *Rough Non-deterministic Information Analysis* ($RNIA$) [26–32].

In this paper, we briefly review $RNIA$ including certain and possible rules, then develop rule generation by the criteria *support* and *accuracy* in $NISs$. In this rule generation, we extend *Apriori* algorithm in $DISs$ to a new algorithm in $NISs$. The computational complexity of this new algorithm is almost the same as *Apriori* algorithm. Finally, we investigate a prototype system, and apply it to some data sets with incomplete information.

2 Basic Definitions and Background of the Research

This section summarizes basic definitions, and reviews the background of this research in [28, 31, 32].

2.1 Basic Definitions

A *Deterministic Information System* (DIS) is a quadruplet $(OB, AT, \{VAL_A | A \in AT\}, f)$, where OB is a finite set whose elements are called *objects*, AT is a finite set whose elements are called *attributes*, VAL_A is a finite set whose elements are called *attribute values* and f is such a mapping that $f : OB \times AT \rightarrow \cup_{A \in AT} VAL_A$ which is called a *classification function*. If $f(x, A)=f(y, A)$ for every $A \in ATR \subset AT$, we see there is a relation between x and y for ATR. This relation is an equivalence relation over OB, and this is called an *indiscernibility relation*.

We usually define two sets $CON \subseteq AT$ which we call *condition attributes* and $DEC \subseteq AT$ which we call *decision attributes*. An object $x \in OB$ is *consistent* (with any distinct object $y \in OB$), if $f(x, A)=f(y, A)$ for every $A \in CON$ implies $f(x, A)=f(y, A)$ for every $A \in DEC$.

A *Non-deterministic Information System* (NIS) is also a quadruplet $(OB, AT, \{VAL_A | A \in AT\}, g)$, where $g : OB \times AT \rightarrow P(\cup_{A \in AT} VAL_A)$ (a power set of $\cup_{A \in AT} VAL_A$). Every set $g(x, A)$ is interpreted as that there is an actual value in this set, but this value is not known. For a NIS=$(OB, AT, \{VAL_A | A \in AT\}, g)$ and a set $ATR \subseteq AT$, we name a DIS=$(OB, ATR, \{VAL_A | A \in ATR\}, h)$ satisfying $h(x, A) \in g(x, A)$ a *derived* DIS (*for* ATR) *from* NIS. For a set ATR=$\{A_1, \cdots, A_n\} \subseteq AT$ and any $x \in OB$, let $PT(x, ATR)$ denote the Cartesian product $g(x, A_1) \times \cdots \times g(x, A_n)$. We name every element a *possible tuple* (*for* ATR) *of* x. For a possible tuple ζ=$(\zeta_1, \cdots, \zeta_n) \in PT(x, ATR)$, let $[ATR, \zeta]$ denote a formula $\bigwedge_{1 \leq i \leq n} [A_i, \zeta_i]$. Every $[A_i, \zeta_i]$ is called a *descriptor*. Let $PI(x, CON, DEC)$ ($x \in OB$) denote a set $\{[CON, \zeta] \Rightarrow [DEC, \eta] | \zeta \in PT(x, CON), \eta \in PT(x, DEC)\}$. We name an element of $PI(x, CON, DEC)$ a *possible implication* (*from* CON *to* DEC) *of* x. In the following, τ denotes a possible implication, and τ^x denotes a possible implication obtained from an object x.

Now, we define six classes of possible implications, certain rules and possible rules. For any $\tau^x \in PI(x, CON, DEC)$, let $DD(\tau^x, x, CON, DEC)$ denote a set $\{\varphi | \ \varphi$ is such a derived DIS for $CON \cup DEC$ that an implication from x in φ is equal to $\tau^x\}$. If $PI(x, CON, DEC)$ is a singleton set $\{\tau^x\}$, we say τ^x is *definite*. Otherwise we say τ^x is *indefinite*. If a set $\{\varphi \in DD(\tau^x, x, CON, DEC) | \ x$ is consistent in $\varphi\}$ is equal to $DD(\tau^x, x, CON, DEC)$, we say τ^x is *globally consistent* (GC). If this set is equal to $\{\}$, we say τ^x is *globally inconsistent* (GI). Otherwise, we say τ^x is *marginal* (MA). By combining two cases, i.e., 'D(*efinite*) or I(*ndefinite*)' and 'GC, MA or GI', we define six classes, DGC, DMA, DGI, IGC, IMA, IGI in Table 1. A possible implication τ^x belonging to DGC class is consistent in all derived $DISs$, and this τ^x is not influenced by the information incompleteness, therefore we name τ^x a *certain rule* or more correctly a *candidate of* a *certain rule*. A possible implication τ^x belonging to either DGC, IGC, DMA or IMA class is consistent in some $\varphi \in DD(\tau^x, x, CON, DEC)$. Therefore, we name τ^x a *possible rule* or more correctly a *candidate of* a *possible rule*.

Table 1. Six classes of possible implications in NISs.

	GC	MA	GI
Definite	DGC	DMA	DGI
Indefinite	IGC	IMA	IGI

Now, we give necessary and sufficient conditions for characterizing GC, MA and GI classes. For any $\zeta \in PT(x, ATR)$, we define two sets

$$inf(x, ATR, \zeta) = \{y \in OB | PT(y, ATR) = \{\zeta\}\} \cup \{x\},$$
$$sup(x, ATR, \zeta) = \{y \in OB | \zeta \in PT(y, ATR)\}.$$

Intuitively, $inf(x, ATR, \zeta)$ implies a set of objects whose tuples are ζ and definite. If a tuple $\zeta \in PT(x, ATR)$ is not definite, this object x does not satisfy

$PT(x, ATR)$=$\{\zeta\}$. Therefore, we added a set $\{x\}$ in the definition of inf. A set $sup(x, ATR, \zeta)$ implies a set of objects whose tuples may be ζ. Even though x does not appear in the right hand side of sup, we employ the $sup(x, ATR, \zeta)$ notation due to the $inf(x, ATR, \zeta)$ notation. Generally, $\{x\} \subseteq inf(x, ATR, \zeta)$= $sup(x, ATR, \zeta)$ holds in $DISs$, and $\{x\} \subseteq inf(x, ATR, \zeta) \subseteq sup(x, ATR, \zeta)$ holds in $NISs$.

Theorem 1. [**28, 29**] For a NIS, let us consider a possible implication τ^x:$[CON, \zeta] \Rightarrow [DEC, \eta] \in PI(x, CON, DEC)$. Then, the following holds.
(1) τ^x belongs to GC class, if and only if $sup(x, CON, \zeta) \subseteq inf(x, DEC, \eta)$.
(2) τ^x belongs to MA class, if and only if $inf(x, CON, \zeta) \subseteq sup(x, DEC, \eta)$.
(3) τ^x belongs to GI class, if and only if $inf(x, CON, \zeta) \not\subseteq sup(x, DEC, \eta)$.

Proposition 2. [**28, 29**] For any NIS, let $ATR \subseteq AT$ be $\{A_1, \cdots, A_n\}$, and let a possible tuple $\zeta \in PT(x, ATR)$ be $(\zeta_1, \cdots, \zeta_n)$. Then, the following holds.
(1) $inf(x, ATR, \zeta)$=$\cap_i inf(x, \{A_i\}, (\zeta_i))$.
(2) $sup(x, ATR, \zeta)$=$\cap_i sup(x, \{A_i\}, (\zeta_i))$.

2.2 An Illustrative Example

Let us consider NIS_1 in Table 2. There are four derived $DISs$ in Table 3.

Table 2. A table of NIS_1.

OB	$Color$	$Size$
1	$\{red, green\}$	$\{small\}$
2	$\{red, blue\}$	$\{big\}$
3	$\{blue\}$	$\{big\}$

Table 3. Four derived $DISs$ from NIS_1. Tables are φ_1, φ_2, φ_3, φ_4 to the right.

OB	$Color$	$Size$
1	red	$small$
2	red	big
3	$blue$	big

OB	$Color$	$Size$
1	red	$small$
2	$blue$	big
3	$blue$	big

OB	$Color$	$Size$
1	$green$	$small$
2	red	big
3	$blue$	big

OB	$Color$	$Size$
1	$green$	$small$
2	$blue$	big
3	$blue$	big

Let us focus on a possible implication
$\tau_1^3 : [Color, blue] \Rightarrow [Size, big] \in PI(3, \{Color\}, \{Size\})$.
This τ_1^3 means the first implication from object 3, and τ_1^3 appears in four derived $DISs$. Since the following holds,
$\{2, 3\} = sup(3, \{Color\}, (blue)) \subseteq inf(3, \{Size\}, (big)) = \{2, 3\}$,
τ_1^3 belongs to DGC class according to Theorem 1. Namely, τ_1^3 is consistent in each derived DIS. As for the second possible implication,
$\tau_2^1 : [Color, red] \Rightarrow [Size, small] \in PI(1, \{Color\}, \{Size\})$,

the following holds:

$\{1,2\} = sup(1,\{Color\},(red)) \not\subseteq inf(1,\{Size\},(small)) = \{1\}$,

$\{1\} = inf(1,\{Color\},(red)) \subseteq sup(1,\{Size\},(small)) = \{1\}$.

According to Theorem 1, τ_2^1 belongs to IMA class, namely τ_2^1 appears in φ_1 and φ_2, and τ_2^1 is consistent just in φ_2.

2.3 Certain Rule Generation in Non-deterministic Information Systems

This subsection briefly reviews the previous research on certain rule generation in $NISs$ [28, 29]. We have named possible implications in DGC class certain rules. For certain rule generation, we dealt with the following problem.

Problem 1. [**29**] For a NIS, let DEC be decision attributes and let η be a tuple of decision attributes values for DEC. Then, find minimal certain rules in the form of $[CON,\zeta] \Rightarrow [DEC,\eta]$.

According to Theorem 1, Problem 1 is reduced to find some minimal sets of descriptors $[CON,\zeta]$ satisfying $sup(x,CON,\zeta) \subseteq inf(x,DEC,\eta)$. For solving this problem, we employed a discernibility function in $DISs$ [33]. We adjusted the discernibility function to $NISs$, and implemented utility programs [29].

Example 1. Let us focus on a possible implication $\tau_1^3 : [Color,blue] \Rightarrow [Size,big]$ in Table 2, again. Since $inf(3,\{Size\},(big))=\{2,3\}$, it is necessary to discriminate object $1 \notin \{2,3\}$ from object 3. The descriptor $[Color,blue]$ discriminates object 1 from object 3, because $sup(3,\{Color\},(blue))=\{2,3\}$ and $1 \notin sup(3,\{Color\},(blue))$ hold. In this way, the discernibility function $DF(3)$ becomes $[Color,blue]$, and we obtain minimal certain rule τ_1^3. The following is a real execution.

```
% ./plc
?-consult(dgc_rule.pl).
yes
?-trans.
File Name for Read Open:'data.pl'.
Decision Definition File:'attrib.pl'.
File Name for Write Open:'data.rs'.
EXEC_TIME=0.01796603203(sec)
yes
?-minimal.        /* [1,blue](=[Color,blue]),[2,big](=[Size,big]) */
<<Minimal Certain Rules from object 3>>
 Descriptor [1,blue] is a core for object 1
 [1,blue]=>[2,big] [4/4(=4/4,1/1),Definite,GC: Only Core Descriptors]
EXEC_TIME=0.01397013664(sec)
yes
```

This program is implemented in prolog [28–30]. Each attribute is identified with its ordinal number, namely $Color$ and $Size$ are identified with 1 and 2, respectively. The underlined parts are specified by a user.

2.4 Non-deterministic Information and Incomplete Information

This subsection clarifies the semantic difference of non-deterministic information and incomplete information.

Table 4. A table of DIS with incomplete information.

OB	$Color$	$Size$
1	$*$	$small$
2	$*$	big
3	$blue$	big

Let us consider Table 4. The symbol "$*$" is often employed for indicating incomplete information. Table 4 is generated by replacing non-deterministic information in Table 2 with $*$. There are some interpretations of this $*$ symbol [4, 7, 8, 10, 34, 17]. In the most simple interpretation of incomplete information, the symbol $*$ may be each attribute value. Namely, $*$ may be either red, $blue$ or $green$, and there are 9 (=3×3) possible tables in Table 4. In such a possible table, the implication from object 1 may be $[Color, blue] \Rightarrow [Size, small]$, and this contradicts $\tau_1^3 : [Color, blue] \Rightarrow [Size, big]$. On the other hand in Table 2, the function is $g(1, \{Color\})=\{red, green\} \subsetneq \{red, blue, green\}$, and we dealt with four derived $DISs$. In Table 2, we did not handle $[Color, blue] \Rightarrow [Size, small]$ from object 1. Like this, τ_1^3 is globally consistent in Table 2, but τ_1^3 is inconsistent in Table 4.

The function $g(x, A)$ and a set $sup(x, ATR, \zeta)$ are employed for handling information incompleteness, and cause the semantic difference of non-deterministic information and incomplete information. In $RNIA$, the interpretation of the information incompleteness comes from the meaning of the function $g(x, A)$. There is no other assumption on this interpretation.

2.5 A Problem of Possible Rule Generation in Non-deterministic Information Systems

We have defined possible rules by possible implications which belong to either DGC, DMA, IGC or IMA classes. In this case, there may be a large number of possible implications satisfying condition (2) in Theorem 1. For example in Table 2, there are four possible implications including τ_1^3 and τ_2^1, and every possible implication is consistent in at least a derived DIS. Thus, every possible implication is a possible rule. This implies the definition of possible rules may be too weak. Therefore, we need to employ other criteria for defining rules except certain rules.

In the subsequent sections, we follow the framework of rule generation [1, 2, 22, 36, 38], and employ criteria, *support* and *accuracy* for defining rules including possible rules.

3 New Criteria: Minimum Support, Minimum Accuracy, Maximum Support and Maximum Accuracy

This section proposes new criteria in $NISs$, and investigates the calculation of criteria. These new criteria depend upon each element in $DD(\tau^x, x, CON, DEC)$, but the complexity of the calculation does not depend upon the number of elements in $DD(\tau^x, x, CON, DEC)$.

3.1 Definition of New Criteria

In a DIS, criteria *support* and *accuracy* are usually applied to defining rules [1, 2, 36]. In a NIS, we define the following four criteria, i.e., minimum *support*: $minsupp(\tau^x)$, maximum *support*: $maxsupp(\tau^x)$, minimum *accuracy*: $minacc(\tau^x)$ and maximum *accuracy*: $maxacc(\tau^x)$ in the following:

(1) $minsupp(\tau^x) = Minimum_{\varphi \in DD(\tau^x, x, CON, DEC)}\{support(\tau^x) \text{ in } \varphi\}$,
(2) $maxsupp(\tau^x) = Maximum_{\varphi \in DD(\tau^x, x, CON, DEC)}\{support(\tau^x) \text{ in } \varphi\}$,
(3) $minacc(\tau^x) = Minimum_{\varphi \in DD(\tau^x, x, CON, DEC)}\{accuracy(\tau^x) \text{ in } \varphi\}$,
(4) $maxacc(\tau^x) = Maximum_{\varphi \in DD(\tau^x, x, CON, DEC)}\{accuracy(\tau^x) \text{ in } \varphi\}$.

If τ^x is definite, $DD(\tau^x, x, CON, DEC)$ is equal to all derived $DISs$. If τ^x is indefinite, $DD(\tau^x, x, CON, DEC)$ is a subset of all derived $DISs$. If we employ all derived $DISs$ instead of $DD(\tau^x, x, CON, DEC)$ in the above definition, $minsupp(\tau^x)$ and $minacc(\tau^x)$ are 0, respectively. Because, there exist some derived $DISs$ where τ^x does not appear. This property for each indefinite τ^x is trivial, so we define $minsupp(\tau^x)$ and $minacc(\tau^x)$ over $DD(\tau^x, x, CON, DEC)$.

Example 2. In Table 2, let us focus on a possible implication
$\tau_1^3 : [Color, blue] \Rightarrow [Size, big] \in PI(3, \{Color\}, \{Size\})$.
In $DD(\tau_1^3, 3, \{Color\}, \{Size\})$=$\{\varphi_1, \varphi_2, \varphi_3, \varphi_4\}$, the following holds:
$1/3 = minsupp(\tau_1^3) \leq maxsupp(\tau_1^3) = 2/3$,
$1 = minacc(\tau_1^3) \leq maxacc(\tau_1^3) = 1$.
As for the second possible implication,
$\tau_2^1 : [Color, red] \Rightarrow [Size, small] \in PI(1, \{Color\}, \{Size\})$,
in $DD(\tau_2^1, 1, \{Color\}, \{Size\})$=$\{\varphi_1, \varphi_2\}$, the following holds:
$1/3 = minsupp(\tau_2^1) \leq maxsupp(\tau_2^1) = 1/3$,
$1/2 = minacc(\tau_2^1) \leq maxacc(\tau_2^1) = 1$.

3.2 A Simple Method for Calculating Criteria

In order to obtain $minsupp(\tau^x)$, $minacc(\tau^x)$, $maxsupp(\tau^x)$ and $maxacc(\tau^x)$, the most simple method is to examine each $support(\tau^x)$ and $accuracy(\tau^x)$ in

every $\varphi \in DD(\tau^x, x, CON, DEC)$. This method is simple, however the number of elements in $DD(\tau^x, x, CON, DEC)$ is $\prod_{A \in CON, B \in DEC, x \neq y} |g(y, A)||g(y, B)|$, and the number of elements increases in exponential order. Therefore, this simple method will not be applicable to $NISs$ with a large number of derived $DISs$.

3.3 Effective Calculation of Minimum Support and Minimum Accuracy

Let us consider how to calculate $minsupp(\tau^x)$ and $minacc(\tau^x)$ for $\tau^x : [CON, \zeta] \Rightarrow [DEC, \eta]$ from object x. Each object y with descriptors $[CON, \zeta]$ or $[DEC, \eta]$ influences $minsupp(\tau^x)$ and $minacc(\tau^x)$. Table 5 shows all possible implications with descriptors $[CON, \zeta]$ or $[DEC, \eta]$. For example in $CASE$ 1, we can obtain just an implication. However in $CASE$ 2, we can obtain either (C2.1) or (C2.2). Every possible implication depends upon the selection of a value in $g(y, DEC)$. This selection of attribute values specifies some derived $DISs$ from a NIS.

Table 5. Seven cases of possible implications (related to $[CON, \zeta] \Rightarrow [DEC, \eta]$ from object x, $\eta \neq \eta'$, $\zeta \neq \zeta'$) in $NISs$.

	$Condition : CON$	$Decision : DEC$	$Possible\_Implications$
$CASE1$	$g(y, CON) = \{\zeta\}$	$g(y, DEC) = \{\eta\}$	$[CON, \zeta] \Rightarrow [DEC, \eta](C1.1)$
$CASE2$	$g(y, CON) = \{\zeta\}$	$\eta \in g(y, DEC)$	$[CON, \zeta] \Rightarrow [DEC, \eta](C2.1)$ $[CON, \zeta] \Rightarrow [DEC, \eta'](C2.2)$
$CASE3$	$g(y, CON) = \{\zeta\}$	$\eta \notin g(y, DEC)$	$[CON, \zeta] \Rightarrow [DEC, \eta'](C3.1)$
$CASE4$	$\zeta \in g(y, CON)$	$g(y, DEC) = \{\eta\}$	$[CON, \zeta] \Rightarrow [DEC, \eta](C4.1)$ $[CON, \zeta'] \Rightarrow [DEC, \eta](C4.2)$
$CASE5$	$\zeta \in g(y, CON)$	$\eta \in g(y, DEC)$	$[CON, \zeta] \Rightarrow [DEC, \eta](C5.1)$ $[CON, \zeta] \Rightarrow [DEC, \eta'](C5.2)$ $[CON, \zeta'] \Rightarrow [DEC, \eta](C5.3)$ $[CON, \zeta'] \Rightarrow [DEC, \eta'](C5.4)$
$CASE6$	$\zeta \in g(y, CON)$	$\eta \notin g(y, DEC)$	$[CON, \zeta] \Rightarrow [DEC, \eta'](C6.1)$ $[CON, \zeta'] \Rightarrow [DEC, \eta'](C6.2)$
$CASE7$	$\zeta \notin g(y, CON)$	Any	$[CON, \zeta'] \Rightarrow Decision(C7.1)$

Now, we revise the definition of inf and sup information in the previous section. We handled both inf and sup information for every object x. However, in the subsequent sections it is enough to handle minimum and maximum sets of an equivalence class defined by a descriptor $[ATR, val]$. This revision is very simple, and this revision reduces the manipulation of each calculation.

Definition 1. For each descriptor $[ATR, val](= [\{A_1, \cdots, A_k\}, (\zeta_1, \cdots, \zeta_k)], (k \geq 1)$) in a NIS, $Descinf$ and $Descsup$ are defined as follows:
(1) $Descinf([A_i, \zeta_i])=\{x \in OB | PT(x, \{A_i\})=\{\zeta_i\}\}=\{x \in OB | g(x, \{A_i\})=\{\zeta_i\}\}$.
(2) $Descinf([ATR, val])=Descinf(\wedge_i[A_i, \zeta_i])=\cap_i Descinf([A_i, \zeta_i])$.
(3) $Descsup([A_i, \zeta_i])=\{x \in OB | \zeta_i \in PT(x, \{A_i\})\}=\{x \in OB | \zeta_i \in g(x, \{A_i\})\}$.
(4) $Descsup([ATR, val])=Descsup(\wedge_i[A_i, \zeta_i])=\cap_i Descsup([A_i, \zeta_i])$.

The definition of $Descinf$ requires that every element in this set is definite. Even though the definition of $Descsup$ is the same as sup, we employ the $Descsup([ATR, \zeta])$ notation due to the $Descinf([ATR, \zeta])$ notation. Clearly, $Descinf([CON, \zeta])$ is a set of objects belonging to either $CASE$ 1, 2 or 3 in Table 5, and $Descsup([CON, \zeta])$ is a set of objects belonging to either $CASE$ 1 to $CASE$ 6. $Descsup([CON, \zeta]) - Descinf([CON, \zeta])$ is a set of objects belonging to either $CASE$ 4, 5 or 6.

Proposition 3. Let $|X|$ denote the cardinality of a set X. In Table 6, the *support* value of $\tau^x : [CON, \zeta] \Rightarrow [DEC, \eta]$ from x is minimum.
If τ^x is definite, namely τ^x belongs to $CASE$ 1,
$minsupp(\tau^x)$=$|Descinf([CON, \zeta]) \cap Descinf([DEC, \eta])|/|OB|$.
If τ^x is indefinite, namely τ^x does not belong to $CASE$ 1,
$minsupp(\tau^x)$=$(|Descinf([CON, \zeta]) \cap Descinf([DEC, \eta])| + 1)/|OB|$.
Proof. This selection of attribute values in a NIS excludes every $[CON, \zeta] \Rightarrow [DEC, \eta]$ from object $y \neq x$. In reality, we remove (C2.1), (C4.1) and (C5.1) from Table 5. Therefore, the *support* value of τ^x is minimum in a derived DIS with such selections of attribute values. If τ^x is definite, object x is in a set $Descinf([CON, \zeta]) \cap Descinf([DEC, \eta])$. Otherwise, τ^x belongs to either (C2.1), (C4.1) or (C5.1). Thus, it is necessary to add 1 to the numerator.

Table 6. Selections from Table 5. These selections make the *support* value of $[CON, \zeta] \Rightarrow [DEC, \eta]$ minimum.

	$Condition : CON$	$Decision : DEC$	$Selection$
$CASE1$	$g(y, CON) = \{\zeta\}$	$g(y, DEC) = \{\eta\}$	$[CON, \zeta] \Rightarrow [DEC, \eta](C1.1)$
$CASE2$	$g(y, CON) = \{\zeta\}$	$\eta \in g(y, DEC)$	$[CON, \zeta] \Rightarrow [DEC, \eta'](C2.2)$
$CASE3$	$g(y, CON) = \{\zeta\}$	$\eta \notin g(y, DEC)$	$[CON, \zeta] \Rightarrow [DEC, \eta'](C3.1)$
$CASE4$	$\zeta \in g(y, CON)$	$g(y, DEC) = \{\eta\}$	$[CON, \zeta'] \Rightarrow [DEC, \eta](C4.2)$
$CASE5$	$\zeta \in g(y, CON)$	$\eta \in g(y, DEC)$	$[CON, \zeta] \Rightarrow [DEC, \eta'](C5.2)$ $[CON, \zeta'] \Rightarrow [DEC, \eta](C5.3)$ $[CON, \zeta'] \Rightarrow [DEC, \eta'](C5.4)$
$CASE6$	$\zeta \in g(y, CON)$	$\eta \notin g(y, DEC)$	$[CON, \zeta] \Rightarrow [DEC, \eta'](C6.1)$ $[CON, \zeta'] \Rightarrow [DEC, \eta'](C6.2)$
$CASE7$	$\zeta \notin g(y, CON)$	Any	$[CON, \zeta'] \Rightarrow Decision(C7.1)$

Proposition 4. Table 7 is a part of Table 5. In Table 7, the *accuracy* value of $\tau^x : [CON, \zeta] \Rightarrow [DEC, \eta]$ from x is minimum. Let $OUTACC$ denote $[Descsup([CON, \zeta]) - Descinf([CON, \zeta])] - Descinf([DEC, \eta])$.
If τ^x is definite,
$minacc(\tau^x)$=$\frac{|Descinf([CON,\zeta]) \cap Descinf([DEC,\eta])|}{|Descinf([CON,\zeta])|+|OUTACC|}$.
If τ^x is indefinite,
$minacc(\tau^x)$=$\frac{|Descinf([CON,\zeta]) \cap Descinf([DEC,\eta])|+1}{|Descinf([CON,\zeta]) \cup \{x\}|+|OUTACC-\{x\}|}$.
Proof. Since $m/n \leq (m + k)/(n + k)$ $(0 \leq m \leq n,\ n \neq 0,\ k > 0)$ holds, we excludes every $[CON, \zeta] \Rightarrow [DEC, \eta]$ from object $y \neq x$. We select possi-

Table 7. Selections from Table 5. These selections make the *accuracy* value of $[CON, \zeta] \Rightarrow [DEC, \eta]$ minimum.

	$Condition : CON$	$Decision : DEC$	$Selection$
$CASE1$	$g(y, CON) = \{\zeta\}$	$g(y, DEC) = \{\eta\}$	$[CON, \zeta] \Rightarrow [DEC, \eta](C1.1)$
$CASE2$	$g(y, CON) = \{\zeta\}$	$\eta \in g(y, DEC)$	$[CON, \zeta] \Rightarrow [DEC, \eta'](C2.2)$
$CASE3$	$g(y, CON) = \{\zeta\}$	$\eta \notin g(y, DEC)$	$[CON, \zeta] \Rightarrow [DEC, \eta'](C3.1)$
$CASE4$	$\zeta \in g(y, CON)$	$g(y, DEC) = \{\eta\}$	$[CON, \zeta'] \Rightarrow [DEC, \eta](C4.2)$
$CASE5$	$\zeta \in g(y, CON)$	$\eta \in g(y, DEC)$	$[CON, \zeta] \Rightarrow [DEC, \eta'](C5.2)$
$CASE6$	$\zeta \in g(y, CON)$	$\eta \notin g(y, DEC)$	$[CON, \zeta] \Rightarrow [DEC, \eta'](C6.1)$
$CASE7$	$\zeta \notin g(y, CON)$	Any	$[CON, \zeta'] \Rightarrow Decision(C7.1)$

ble implications $[CON, \zeta] \Rightarrow [DEC, \eta']$, which increase the denominator. The *accuracy* value of τ^x is minimum in a derived DIS with such selection of attribute values. The set $OUTACC$ defines objects in either $CASE$ 5 or $CASE$ 6. As for $CASE$ 4 and $CASE$ 7, the condition part is not $[CON, \zeta]$. Therefore, we can omit such implications for calculating $minacc(\tau^x)$. If τ^x is definite, the numerator is $|Descinf([CON, \zeta]) \cap Descinf([DEC, \eta])|$ and the denominator is $|Descinf([CON, \zeta])| + |OUTACC|$. If τ^x is indefinite, τ^x belongs to either (C2.1), (C4.1) or (C5.1). The denominator is $|Descinf([CON, \zeta]) \cup \{x\}| + |OUTACC - \{x\}|$ in every case, and the numerator is $|Descinf([CON, \zeta]) \cap Descinf\ ([DEC, \eta])| + 1$.

Theorem 5. For a NIS, let us consider a possible implication τ^x:$[CON, \zeta] \Rightarrow [DEC, \eta] \in PI(x, CON, DEC)$. Let $SUPP_{min}$=$\{\varphi | \varphi$ is a derived DIS from NIS, and $support(\tau^x)$ is minimum in $\varphi\}$. Then, $accuracy(\tau^x)$ is minimum in some $\varphi \in SUPP_{min}$.

Proof. Table 7 is a special case of Table 6. Namely, in $CASE$ 5 of Table 6, either (C5.2), (C5.3) or (C5.4) may hold. In $CASE$ 6 of Table 6, either (C6.1) or (C6.2) may hold. In every selection, the minimum *support* value is the same. In Table 7, (C5.2) in $CASE$ 5 and (C6.1) in $CASE$ 6 are selected.

Theorem 5 assures that there exists a derived DIS, where both $support(\tau^x)$ and $accuracy(\tau^x)$ are minimum. DIS_{worst} denotes such a derived DIS, and we name DIS_{worst} a *derived DIS* with the *worst condition* for τ^x. This is an important property for Problem 3 in the subsequent section.

3.4 Effective Calculation of Maximum Support and Maximum Accuracy

In this subsection, we show an effective method to calculate $maxsupp(\tau^x)$ and $maxacc(\tau^x)$ based on $Descinf$ and $Descsup$. The following can be proved according the same manner as Proposition 3, 4 and Theorem 5. A derived DIS

Table 8. Selections from Table 5. These selections make the *support* and *accuracy* values of $[CON, \zeta] \Rightarrow [DEC, \eta]$ maximum.

	$Condition(CON)$	$Decision(DEC)$	$Selection$
$CASE1$	$g(y, CON) = \{\zeta\}$	$g(y, DEC) = \{\eta\}$	$[CON, \zeta] \Rightarrow [DEC, \eta](C1.1)$
$CASE2$	$g(y, CON) = \{\zeta\}$	$\eta \in g(y, DEC)$	$[CON, \zeta] \Rightarrow [DEC, \eta](C2.1)$
$CASE3$	$g(y, CON) = \{\zeta\}$	$\eta \notin g(y, DEC)$	$[CON, \zeta] \Rightarrow [DEC, \eta'](C3.1)$
$CASE4$	$\zeta \in g(y, CON)$	$g(y, DEC) = \{\eta\}$	$[CON, \zeta] \Rightarrow [DEC, \eta](C4.1)$
$CASE5$	$\zeta \in g(y, CON)$	$\eta \in g(y, DEC)$	$[CON, \zeta] \Rightarrow [DEC, \eta](C5.1)$
$CASE6$	$\zeta \in g(y, CON)$	$\eta \notin g(y, DEC)$	$[CON, \zeta'] \Rightarrow [DEC, \eta'](C6.2)$
$CASE7$	$\zeta \notin g(y, CON)$	Any	$[CON, \zeta'] \Rightarrow Decision(C7.1)$

defined in Table 8 makes both *support* and *accuracy* maximum.

Proposition 6. For $\tau^x : [CON, \zeta] \Rightarrow [DEC, \eta]$ from x, the following holds.
$maxsupp(\tau^x) = |Descsup([CON, \zeta]) \cap Descsup([DEC, \eta])|/|OB|$.

Proposition 7. For $\tau^x : [CON, \zeta] \Rightarrow [DEC, \eta]$ from x, let $INACC$ denote $[Descsup([CON, \zeta]) - Descinf([CON, \zeta])] \cap Descsup\ ([DEC, \eta])$.
If τ^x is definite,
$maxacc(\tau^x) = \frac{|Descinf([CON,\zeta]) \cap Descsup([DEC,\eta])| + |INACC|}{|Descinf([CON,\zeta])| + |INACC|}$.
If τ^x is indefinite,
$maxacc(\tau^x) = \frac{|Descinf([CON,\zeta]) \cap Descsup([DEC,\eta]) - \{x\}| + |INACC - \{x\}| + 1}{|Descinf([CON,\zeta]) \cup \{x\}| + |INACC - \{x\}|}$.

Theorem 8. For a NIS, let us consider a possible implication τ^x:$[CON, \zeta] \Rightarrow [DEC, \eta] \in PI(x, CON, DEC)$. Let $SUPP_{max} = \{\varphi | \varphi$ is a derived DIS from NIS, and $support(\tau^x)$ is maximum in $\varphi\}$. Then, $accuracy(\tau^x)$ is maximum in some $\varphi \in SUPP_{max}$.

Theorem 8 assures that there exists a derived DIS, where both $support(\tau^x)$ and $accuracy(\tau^x)$ are maximum. DIS_{best} denotes such a derived DIS, and we name DIS_{best} a *derived DIS* with the *best condition* for τ^x. This is also an important property for Problem 4 in the subsequent section.

4 Rule Generation by New Criteria in Non-deterministic Information Systems

This section applies Proposition 3, 4, 6, 7 and Theorem 5, 8 to rule generation in $NISs$.

4.1 Rules by the Criteria in Deterministic Information Systems

In $DISs$, rule generation by the criteria is often defined as the following.

Problem 2. In a table or a DIS, find every implication τ that $support(\tau) \geq \alpha$ and $accuracy(\tau) \geq \beta$ for given α and β $(0 < \alpha, \beta \leq 1)$.

For solving this problem, *Apriori* algorithm was proposed by Agrawal [1, 2]. In this framework, *association rules* in *transaction data* are obtained. The application of the *large item set* is the key point in *Apriori* algorithm. This Problem 2 has also been considered in [22, 36, 38].

4.2 Rules by New Criteria and Two Strategies in Non-deterministic Information Systems

Now, we extend Problem 2 to Problem 3 and Problem 4 in the following.

Problem 3. (Rule Generation by Lower Approximation Strategy) For a NIS, let $CON \subseteq AT$ and $DEC \subseteq AT$ be condition attributes and the decision attribute, respectively. Find every possible implication $\tau^x : [CON, \zeta] \Rightarrow [DEC, \eta]$ satisfying $minsupp(\tau^x) \geq \alpha$ and $minacc(\tau^x) \geq \beta$ for given α and β $(0 < \alpha, \beta \leq 1)$.

Problem 4. (Rule Generation by Upper Approximation Strategy) For a NIS, let $CON \subseteq AT$ and $DEC \subseteq AT$ be condition attributes and the decision attribute, respectively. Find every possible implication $\tau^x : [CON, \zeta] \Rightarrow [DEC, \eta]$ satisfying $maxsupp(\tau^x) \geq \alpha$ and $maxacc(\tau^x) \geq \beta$ for given α and β $(0 < \alpha, \beta \leq 1)$.

It is necessary to remark that both $minsupp(\tau^x)$ and $minacc(\tau^x)$ are defined over $DD(\tau^x, x, CON, DEC)$. For definite τ^x, $DD(\tau^x, x, CON, DEC)$ is equal to all derived $DISs$. However for indefinite τ^x, $DD(\tau^x, x, CON, DEC)$ is not equal to all derived $DISs$, and $minsupp(\tau^x)$=0 and $minacc(\tau^x)$=0 may hold. This may be an important issue in *lower approximation strategy*. However in this paper, we employ a set $DD(\tau^x, x, CON, DEC)$ instead of all derived $DISs$. As for *upper approximation strategy*, $maxsupp(\tau^x)$ and $maxacc(\tau^x)$ over $DD(\tau^x, x, CON,\ DEC)$ are the same as $maxsupp(\tau^x)$ and $maxacc(\tau^x)$ over all derived $DISs$. We employed terms Min-Max and Max-Max strategies in [31, 32]. According to rough sets based concept, we rename these terms *lower approximation strategy* and *upper approximation strategy*, respectively.

Next Proposition 9 clarifies the relation between certain rules, possible rules and rules by new criteria.

Proposition 9. For a possible implication τ^x, the following holds.
(1) τ^x is a certain rule in Section 2.1, if and only if τ^x is definite and $minacc(\tau^x)$=1.
(2) τ^x is a possible rule in Section 2.1, if and only if $maxacc(\tau^x)$=1.

The concept of consistency defines certain and possible rules, therefore there is no definition about *support*. In certain rule generation, we often have a possible

implication whose $minacc(\tau^x)$=1 and $minsupp(\tau^x)$ is quite small. Proposition 10, 11 and 12 clarify the properties of rule generation.

Proposition 10. For a given α and β ($0 < \alpha, \beta \leq 1$), let $Rule(\alpha, \beta, LA)$ denote a set of rules defined by *lower approximation strategy* with α and β, and let $Rule(\alpha, \beta, UA)$ denote a set of rules defined by *upper approximation strategy* with α and β. Then, $Rule(\alpha, \beta, LA) \subseteq Rule(\alpha, \beta, UA)$ holds.

Proposition 11. The following, which are related to a possible implication $\tau^x : [CON, \zeta] \Rightarrow [DEC, \eta]$, are equivalent.
(1) τ^x is obtained according to *lower approximation strategy*, namely $minsupp(\tau^x) \geq \alpha$ and $minacc(\tau^x) \geq \beta$.
(2) $support(\tau^x) \geq \alpha$ and $accuracy(\tau^x) \geq \beta$ in each $\varphi \in DD(\tau^x, x, CON, DEC)$.
(3) In a derived DIS_{worst} defined in Table 7, $support(\tau^x) \geq \alpha$ and $accuracy(\tau^x) \geq \beta$ hold.
Proof: For each $\varphi \in DD(\tau^x, x, CON, DEC)$, $support(\tau^x) \geq minsupp(\tau^x)$ and $accuracy(\tau^x) \geq minacc(\tau^x)$ hold, therefore (1) and (2) are equivalent. According to Theorem 5, a derived DIS_{worst} (depending upon τ^x) defined in Table 7 assigns minimum values to both $support(\tau^x)$ and $accuracy(\tau^x)$. Thus, (1) and (3) are equivalent.

Proposition 12. The following, which are related to a possible implication $\tau^x : [CON, \zeta] \Rightarrow [DEC, \eta]$, are equivalent.
(1) τ^x is obtained according to *upper approximation strategy*, namely $maxsupp(\tau^x) \geq \alpha$ and $maxacc(\tau^x) \geq \beta$.
(2) $support(\tau^x) \geq \alpha$ and $accuracy(\tau^x) \geq \beta$ in a $\varphi \in DD(\tau^x, x, CON, DEC)$.
(3) In a derived DIS_{best} defined in Table 8, $support(\tau^x) \geq \alpha$ and $accuracy(\tau^x) \geq \beta$ hold.
Proof: For each $\varphi \in DD(\tau^x, x, CON, DEC)$, $support(\tau^x) \leq maxsupp(\tau^x)$ and $accuracy(\tau^x) \leq maxacc(\tau^x)$ hold. According to Theorem 8, a derived DIS_{best} (depending upon τ^x) defined in Table 8 assigns maximum values to both $support(\tau^x)$ and $accuracy(\tau^x)$. In this DIS_{best}, $maxsupp(\tau^x)$=$support(\tau^x)$ and $maxacc(\tau^x)$=$accuracy(\tau^x)$ hold. Thus, (1), (2) and (3) are equivalent.

Due to Proposition 10, 11 and 12, $Rule(\alpha, \beta, LA)$ defines a set of possible implications in a DIS_{worst}, and $Rule(\alpha, \beta, UA)$ defines a set of possible implications in a DIS_{best}. This implies that we do not have to examine each derived DIS in $DD(\tau^x, x, CON, DEC)$, but we have only to examine a DIS_{worst} for the *lower approximation strategy* and a DIS_{best} for the *upper approximation strategy*.

4.3 Extended Apriori Algorithms for Two Strategies and A Simulation

This subsection proposes two extended *Apriori* algorithms in Algorithm 1 and 2. In $DISs$, $Descinf([A, \zeta])$=$Descsup([A, \zeta])$ holds, however $Descinf([A, \zeta]) \subseteq$

$Descsup([A, \zeta])$ holds in $NISs$. *Apriori* algorithm handles transaction data, and employs the sequential search for obtaining large item sets [1, 2]. In $DISs$, we employ the manipulation of $Descinf$ and $Descsup$ instead of the sequential search. According to this manipulation, we obtain the minimum set and maximum set of an equivalence class. Then, we calculate $minsupp(\tau^x)$ and $minacc(\tau^x)$ by using $Descinf$ and $Descsup$. The rest is almost the same as *Apriori* algorithm.

Algorithm 1: Extended Apriori Algorithm for Lower Approximation Strategy

---

Input : A NIS, a decision attribute DEC, threshold value α and β.
Output: Every rule defined by lower approximation strategy.
for ($every\ A \in AT$) **do**
  Generate $Descinf([A, \zeta])$ and $Descsup([A, \zeta])$;
end
For the condition $minsupp(\tau^x)$=$|SET|/|OB| \geq \alpha$, obtain the number NUM of elements in SET;
Generate a set $CANDIDATE(1)$, which consists of descriptors $[A, \zeta_A]$ satisfying either ($CASE$ A) or ($CASE$ B) in the following;
($CASE$ A) $|Descinf([A, \zeta_A])| \geq NUM$,
($CASE$ B) $|Descinf([A, \zeta_A])|$=$(NUM - 1)$ and
$(Descsup([A, \zeta_A]) - Descinf([A, \zeta_A])) \neq \{\}$.
Generate a set $CANDIDATE(2)$ according to the following procedures;
(Proc 2-1) For every $[A, \zeta_A]$ and $[DEC, \zeta_{DEC}]$ $(A \neq DEC)$ in $CANDIDATE(1)$, generate a new descriptor $[\{A, DEC\}, (\zeta_A, \zeta_{DEC})]$;
(Proc 2-2) Examine condition ($CASE$ A) and ($CASE$ B) for each $[\{A, DEC\}, (\zeta_A, \zeta_{DEC})]$;
If either ($CASE$ A) or ($CASE$ B) holds and $minacc(\tau) \geq \beta$ display $\tau : [A, \zeta_A] \Rightarrow [DEC, \zeta_{DEC}]$ as a rule;
If either ($CASE$ A) or ($CASE$ B) holds and $minacc(\tau) < \beta$, add this descriptor to $CANDIDATE(2)$;
Assign 2 to n;
while $CANDIDATE(n) \neq \{\}$ **do**
  Generate $CANDIDATE(n + 1)$ according to the following procedures;
  (Proc 3-1) For $DESC_1$ and $DESC_2$ ($[DEC, \zeta_{DEC}] \in DESC_1 \cap DESC_2$) in $CANDIDATE(n)$, generate a new descriptor by using a conjunction of $DESC_1 \wedge DESC_2$;
  (Proc 3-2) Examine the same procedure as (Proc 2-2).
  Assign $n + 1$ to n;
end

Now, we show an example, which simulates Algorithm 1.

Example 3. Let us consider $Descinf$ and $Descsup$, which are obtained from NIS_2 in Table 9, and let us consider Problem 3. We set α=0.3, β=0.8, condition attribute $CON \subseteq \{P, Q, R, S\}$ and decision attribute DEC=$\{T\}$. Since $|OB|$=5 and $minsupp(\tau)$=$|SET|/5 \geq 0.3$, $|SET| \geq 2$ must hold. According to Table 10,

Algorithm 2: Extended Apriori Algorithm for Upper Approximation Strategy

Input : A NIS, a decision attribute DEC, threshold value α and β.
Output: Every rule defined by upper approximation strategy.
Algorithm 2 is proposed as Algorithm 1 with the following two revisions :

1. ($CASE$ A) and ($CASE$ B) in Algorithm 1 are replaced with ($CASE$ C).
   ($CASE$ C) $|Descsup([A, \zeta_A])| \geq NUM$.
2. $minacc(\tau)$ in Algorithm 1 is replaced with $maxacc(\tau)$.

we generate Table 11 satisfying either ($CASE$ A) or ($CASE$ B) in the following:
($CASE$ A) $|Descinf([A, \zeta_A] \wedge [T, \eta])| \geq 2$ ($A \in \{P, Q, R, S\}$).
($CASE$ B) $|Descinf([A, \zeta_A] \wedge [T, \eta])|$=1 and $Descsup([A, \zeta_A] \wedge [T, \eta]) - Descinf([A, \zeta_A] \wedge [T, \eta]) \neq \{\}$ ($A \in \{P, Q, R, S\}$).
The conjunction $[P, 3] \wedge [T, 3]$ in Table 11 means an implication $\tau_3^1, \tau_3^5 : [P, 3] \Rightarrow [T, 3]$. Because $Descsup([P, 3] \wedge [T, 3])$=$\{1, 5\}$ holds, τ_3^1 and τ_3^5 come from object 1 and 5, respectively. Since $1, 5 \in Descinf([P, 3] \wedge [T, 3])$ holds, $minsupp(\tau_3^1)$= $minsupp(\tau_3^5)$=$|\{1, 5\}|/5$=0.4 holds. Then, the conjunction $[Q, 1] \wedge [T, 3]$ in Table 11 means an implication $\tau_4^1, \tau_4^5 : [Q, 1] \Rightarrow [T, 3]$. Since $5 \in Descinf([Q, 1] \wedge [T, 3])$

Table 9. A Table of NIS_2.

OB	P	Q	R	S	T
1	$\{3\}$	$\{1, 3\}$	$\{3\}$	$\{2\}$	$\{3\}$
2	$\{2\}$	$\{2, 3\}$	$\{1, 3\}$	$\{1, 3\}$	$\{2\}$
3	$\{1, 2\}$	$\{2\}$	$\{1, 2\}$	$\{3\}$	$\{1\}$
4	$\{1\}$	$\{3\}$	$\{3\}$	$\{2, 3\}$	$\{1, 2, 3\}$
5	$\{3\}$	$\{1\}$	$\{1, 2\}$	$\{3\}$	$\{3\}$

Table 10. $Descinf$ and $Descsup$ information in Table 9.

	$[P, 1]$	$[P, 2]$	$[P, 3]$	$[Q, 1]$	$[Q, 2]$	$[Q, 3]$	$[R, 1]$	$[R, 2]$	$[R, 3]$
$Descinf$	$\{4\}$	$\{2\}$	$\{1, 5\}$	$\{5\}$	$\{3\}$	$\{4\}$	$\{\}$	$\{\}$	$\{1, 4\}$
$Descsup$	$\{3, 4\}$	$\{2, 3\}$	$\{1, 5\}$	$\{1, 5\}$	$\{2, 3\}$	$\{1, 2, 4\}$	$\{2, 3, 5\}$	$\{3, 5\}$	$\{1, 2, 4\}$

	$[S, 1]$	$[S, 2]$	$[S, 3]$	$[T, 1]$	$[T, 2]$	$[T, 3]$
$Descinf$	$\{\}$	$\{1\}$	$\{3, 5\}$	$\{3\}$	$\{2\}$	$\{1, 5\}$
$Descsup$	$\{2\}$	$\{1, 4\}$	$\{2, 3, 4, 5\}$	$\{3, 4\}$	$\{2, 4\}$	$\{1, 4, 5\}$

Table 11. Conjunctions of descriptors satisfying either ($CASE$ A) or ($CASE$ B) in Table 10.

	$[P, 3] \wedge [T, 3]$	$[Q, 1] \wedge [T, 3]$	$[R, 3] \wedge [T, 3]$	$[S, 2] \wedge [T, 3]$	$[S, 3] \wedge [T, 1]$	$[S, 3] \wedge [T, 3]$
$Descinf$	$\{1, 5\}$	$\{5\}$	$\{1\}$	$\{1\}$	$\{3\}$	$\{5\}$
$Descsup$	$\{1, 5\}$	$\{1, 5\}$	$\{1, 4\}$	$\{1, 4\}$	$\{3, 4\}$	$\{4, 5\}$

holds, $minsupp(\tau_4^5)$=$|\{5\}|/5$=0.2 holds. On the other hand, $1 \in Descsup([Q,1] \wedge [T,3]) - Descinf([Q,1] \wedge [T,3])$ holds, so $minsupp(\tau_4^1)$=$(|\{5\}|+1)/5$=0.4 holds in object 1. According to this consideration, we obtain the candidates of rules, which satisfy $minsupp(\tau^x) \geq 0.3$, as follows:

$\tau_3^1, \tau_3^5 : [P,3] \Rightarrow [T,3]$, $\tau_4^1 : [Q,1] \Rightarrow [T,3]$, $\tau_5^4 : [R,3] \Rightarrow [T,3]$,
$\tau_6^4 : [S,2] \Rightarrow [T,3]$, $\tau_7^4 : [S,3] \Rightarrow [T,1]$, $\tau_8^4 : [S,3] \Rightarrow [T,3]$.

For these candidates, we examine each $minacc(\tau^x)$ according to Proposition 4. For τ_3^1 and τ_3^5, $Descsup([P,3])$=$\{1,5\}$, $Descinf([P,3])$=$\{1,5\}$, $Descinf([P,3] \wedge [T,3])$=$\{1,5\}$ and $OUTACC$=$[\{1,5\}-\{1,5\}]-\{1,5\}$=$\{\}$. Since $1,5 \in Descinf([P,3] \wedge [T,3])$ holds, $minacc(\tau_3^1)$= $minacc(\tau_3^5)$=$|\{1,5\}|/(|\{1,5\}|+|\{\}|)$=1 is derived. For $\tau_7^4 : [S,3] \Rightarrow [T,1]$, $Descsup([S,3])$=$\{2,3,4,5\}$, $Descinf([S,3])$=$\{3,5\}$, $Descinf([S,3] \wedge [T,1])$=$\{3\}$, $Descsup([S,3] \wedge [T,1])$= $\{3,4\}$ and $OUTACC$=$[\{2,3,4,5\}-\{3,5\}]-\{3\}$=$\{2,4\}$ holds, so $minacc(\tau_7^4)$=$(|\{3\}|+1)/(|\{3,5\} \cup \{4\}|+|\{2,4\}-\{4\}|)$=0.5 is derived. In this way, we obtain three rules satisfying $minsupp(\tau^x) \geq 0.3$ and $minacc(\tau^x) \geq 0.8$ in the following:

$\tau_3^1, \tau_3^5 : [P,3] \Rightarrow [T,3]$ ($minsupp$=0.4, $minacc$=1),
$\tau_4^1 : [Q,1] \Rightarrow [T,3]$ ($minsupp$=0.4, $minacc$=1),
$\tau_6^4 : [S,2] \Rightarrow [T,3]$ ($minsupp$=0.4, $minacc$=1).

Any possible implication including $[R,3] \wedge [T,3]$ does not satisfy $minsupp(\tau^x) \geq 0.3$. As for $[S,3] \wedge [T,1]$ and $[S,3] \wedge [T,3]$, the same results hold.

The following shows a real execution on Example 3.

```
% ./nis_apriori
version 1.2.8
File Name:'nis2.dat'
========================================
Lower Approximation Strategy
========================================
CAN(1)=[P,1],[P,2],[P,3],[Q,1],[Q,2],[Q,3],[R,3],[S,2],[S,3],[T,1],
[T,2],[T,3](12)
CAN(2)=[S,3][T,1](<DEF>0.250,<INDEF>0.500),[P,3][T,3](<DEF>1.000,
<INDEF>1.000),[Q,1][T,3](<DEF>1.000,<INDEF>1.000),[R,3][T,3](<DEF>0.333,
<INDEF>0.667),[S,2][T,3](<DEF>0.500,<INDEF>1.000),[S,3][T,3](<DEF>0.250,
<INDEF>0.500)(6)
========== OBTAINED RULE ==========
[P,3]=>[T,3](minsupp<DEF>=0.400,minsupp<INDEF>=0.400,minacc<DEF>=1.000,
minacc<INDEF>=1.000) (<DEF>from 1,5) (<INDEF>from )
[Q,1]=>[T,3](minsupp<DEF>=0.200,minsupp<INDEF>=0.400,minacc<DEF>=1.000,
minacc<INDEF>=1.000) (<DEF>from ) (<INDEF>from 1)
[S,2]=>[T,3](minsupp<DEF>=0.200,minsupp<INDEF>=0.400,minacc<DEF>=0.500,
minacc<INDEF>=1.000) (<DEF>from ) (<INDEF>from 4)
EXEC_TIME=0.0000000000(sec)

========================================
Upper Approximation Strategy
========================================
CAN(1)=[P,1],[P,2],[P,3],[Q,1],[Q,2],[Q,3],[R,3],[S,2],[S,3],[T,1],
```

```
[T,2],[T,3](12)
CAN(2)=[S,3][T,1](<DEF>0.667,<INDEF>0.667),[P,3][T,3](<DEF>1.000,
<INDEF>1.000),[Q,1][T,3](<DEF>1.000,<INDEF>1.000),[R,3][T,3](<DEF>1.000,
<INDEF>1.000),[S,2][T,3](<DEF>1.000,<INDEF>1.000),[S,3][T,3](<DEF>0.667,
<INDEF>0.667)(6)
========== OBTAINED RULE ==========
[P,3]=>[T,3](maxsupp<DEF>=0.400,maxsupp<INDEF>=0.400,maxacc<DEF>=1.000,
maxacc<INDEF>=1.000) (<DEF>from 1,5) (<INDEF>from )
[Q,1]=>[T,3](maxsupp<DEF>=0.400,maxsupp<INDEF>=0.400,maxacc<DEF>=1.000,
maxacc<INDEF>=1.000) (<DEF>from 5) (<INDEF>from 1)
[R,3]=>[T,3](maxsupp<DEF>=0.400,maxsupp<INDEF>=0.400,maxacc<DEF>=1.000,
maxacc<INDEF>=1.000) (<DEF>from 1) (<INDEF>from 4)
[S,2]=>[T,3](maxsupp<DEF>=0.400,maxsupp<INDEF>=0.400,maxacc<DEF>=1.000,
maxacc<INDEF>=1.000) (<DEF>from 1) (<INDEF>from 4)
EXEC_TIME=0.0000000000(sec)
```

According to this execution, we know

$Rule(0.3, 0.8, LA)=\{[P,3] \Rightarrow [T,3], [Q,1] \Rightarrow [T,3], [S,2] \Rightarrow [T,3]\}$,

$Rule(0.3, 0.8, UA)=\{[P,3] \Rightarrow [T,3], [Q,1] \Rightarrow [T,3], [S,2] \Rightarrow [T,3], [R,3] \Rightarrow [T,3]\}$.

The possible implication $[R,3] \Rightarrow [T,3] \in Rule(0.3, 0.8, UA) - Rule(0.3, 0.8, LA)$ depends upon the information incompleteness. This can not be obtained by the *lower approximation strategy*, but this can be obtained by the *upper approximation strategy*.

4.4 Main program for Lower Approximation Strategy

A program *nis_apriori* is implemented on a Windows PC with Pentium 4 (3.40 GHz), and it consists of about 1700 lines in C. This *nis_apriori* mainly consists of two parts, i.e., a part for *lower approximation strategy* and a part for *upper approximation strategy*.

As for *lower approximation strategy*, a function $GenRuleByLA()$ (Generate Rules By LA strategy) is coded.

```
GenRuleByLA(table.obj,table.att,table.kosuval,table.con_num,
  table.dec_num,table.con,table.dec,thresh,minacc_thresh);
```

In $GenRuleByLA()$, a function $GenCandByLA()$ is called, and generates a candidate $CANDIDATE(n)$.

```
GenCandByLA(desc,cand,conj_num_max,ob,at,desc_num,c_num,
  d_num,co,de,thr,minacc_thr);
```

At the same time, $minsupp(\tau)$ and $minacc(\tau)$ are calculated according to Proposition 3 and 4. As for *upper approximation strategy*, the similar functions are implemented.

5 Computational Issues in Algorithm 1

This section focuses on the computational complexity of Algorithm 1. As for Algorithm 2, the result is almost the same as Algorithm 1.

5.1 A Simple Method for Lower Approximation Strategy

Generally, a possible implication τ^x depends upon the number of derived $DISs$, i.e., $\prod_{x \in OB, A \in AT} |g(x, A)|$, and condition attributes CON ($CON \subseteq 2^{AT-DEC}$). Furthermore, $minsupp(\tau^x)$ and $minacc(\tau^x)$ depend on $DD(\tau^x, x, CON, DEC)$, whose number of elements is $\prod_{A \in CON, B \in DEC, x \neq y} |g(y, A)||g(y, B)|$. Therefore, it will be impossible to employ a simple method that we sequentially pick up every possible implication τ^x and sequentially examine $minsupp(\tau^x)$ and $minacc(\tau^x)$.

5.2 Complexity on Extended Apriori Algorithm for Lower Approximation Strategy

In order to solve this computational issue, we focus on descriptors $[A, \zeta]$ ($A \in AT$, $\zeta \in VAL_A$). The number of all descriptors is usually very small. Furthermore, Proposition 3 and 4 show us the methods to calculate $minsupp(\tau^x)$ and $minacc(\tau^x)$. These methods do not depend upon the number of element in $DD(\tau^x, x, CON, DEC)$.

Now, we analyze each step in Algorithm 1.

(STEP 1) (Generation of $Descinf$, $Descsup$ and $CANDIDATE(1)$)
We first prepare two arrays $Descinf_{A,val}[]$ and $Descsup_{A,val}[]$ for each $val \in VAL_A$ ($A \in AT$). For each object $x \in OB$, we apply (1) and (2) in the following:
(1) If $g(x, A)$=$\{val\}$, add x to $Descinf_{A,val}[]$ and $Descsup_{A,val}[]$.
(2) If $g(x, A) \neq \{val\}$ and $val \in g(x, A)$, add x to $Descsup_{A,val}[]$.
Then, all descriptors satisfying either ($CASE$ A) or ($CASE$ B) in Algorithm 1 are added to $CANDIDATE(1)$. For each $A \in AT$, this procedure is applied, and the complexity depends upon $|OB| \times |AT|$.

(STEP 2) (Generation of $CANDIDATE(2)$)
For each $[A, val_A], [DEC, val_{DEC}] \in CANDIDATE(1)$, we produce $[A, val_A] \wedge [DEC, val_{DEC}]$, and generate

$$Descinf([A, val_A] \wedge [DEC, val_{DEC}]) = Descinf([A, val_A]) \cap Descinf([DEC, val_{DEC}]),$$

$$Descsup([A, val_A] \wedge [DEC, val_{DEC}]) = Descsup([A, val_A]) \cap Descsup([DEC, val_{DEC}]).$$

If $[A, val_A] \wedge [DEC, val_{DEC}]$ satisfies either ($CASE$ A) or ($CASE$ B) in Algorithm 1, this descriptor is added to $CANDIDATE(2)$. Furthermore, we examine $minacc([A, val_A] \wedge [DEC, val_{DEC}])$ in (Proc 2-2) according to Proposition 4. The complexity of (STEP 2) depends upon the number of combined descriptors $[A, val_A] \wedge [DEC, val_{DEC}]$.

(STEP 3) (Repetition of STEP 2 on $CANDIDATE(n)$)
For each $DESC_1$ and $DESC_2$ in $CANDIDATE(n)$, we generate a conjunction $DESC_1 \wedge DESC_2$. For such conjunctions, we apply the same procedure as (STEP 2).

In the execution, two sets $Descinf([CON, \zeta])$ and $Descsup([CON, \zeta])$ are stored in arrays, and we can obtain $Descinf([CON, \zeta] \wedge [DEC, \eta])$ by using

the intersection operation $Descinf([CON, \zeta]) \cap Descinf([DEC, \eta])$. The same property holds for $Descsup([CON, \zeta] \wedge [DEC, \eta])$. Therefore, it is easy to obtain $CANDIDATE(n+1)$ from $CANDIDATE(n)$. This is a merit of employing equivalence classes, and this is the characteristics of rough set theory. In *Apriori* algorithm, such $Descinf$ and $Descsup([CON, \zeta])$ are not employed, and the total search of a database is executed for generating every combination of descriptors. It will be necessary to consider the merit and demerit of handling two sets $Descinf([CON, \zeta])$ and $Descsup([CON, \zeta])$ in the next research.

Apriori algorithm employs an equivalence class for each descriptors, and handles only deterministic information. On the other hand, Algorithm 1 employs the minimum and the maximum sets of an equivalence class, i.e., $Descinf$ and $Descsup$, and handles non-deterministic information as well as deterministic information. In Algorithm 1, it takes twice steps of *Apriori* algorithm for manipulating equivalence classes. The rest is almost the same as *Apriori* algorithm, therefore the complexity of Algorithm 1 will be almost the same as *Apriori* algorithm.

6 Concluding Remarks and Future Work

We proposed rule generation based on *lower approximation strategy* and *upper approximation strategy* in $NISs$. We employed $Descinf$, $Descsup$ and the concept of large item set in *Apriori* algorithm, and proposed two extended *Apriori* algorithms in $NISs$. These extended algorithms do not depend upon the number of derived $DISs$, and the complexity of these extended algorithms is almost the same as *Apriori* algorithm. We implemented the extended algorithms, and applied them to some data sets. According to these utility programs, we can explicitly handle not only deterministic information but also non-deterministic information.

Now, we briefly show the application to Hepatitis data in UCI Machine Learning Repository [37]. In reality, we applied our programs to Hepatitis data. This data consists of 155 objects, 20 attributes. There are 167 missing values, which are about 5.4% of total data. The number of objects without missing values is 80, namely the number is about the half of total data. In usual analyzing tools, it may be difficult to handle total 155 objects.

We employ a list for expressing non-deterministic information, for example, [red,green], [red,blue] for $\{red, green\}$ and $\{red, blue\}$ in Table 2. This syntax is so simple that we can easily generate data of $NISs$ by using Excel. As for Hepatitis data, we loaded this data into Excel, and replaced each missing value (? symbol) with a list of all possible attribute values. For some numerical values, the discretized attribute values are also given in the data set. For example, in the 15th attribute BILIRUBIN, attribute values are discretized to the six attribute values, i.e., 0.39, 0.80, 1.20, 2.00, 3.00, 4.00. We employed these discretized values in some attributes. The following is a part of the real revised Hepatitis data in Excel. There are 78732 ($=2 \times 6 \times 9^4$) derived $DISs$ for these six objects. Prob-

ably, it seems hard to handle all derived $DISs$ for total 155 objects sequentially.

```
155     //Number of objects
20      //Number of Attributes
2 30 2 1 2 2 2 2 1 2 2 2 2 2 0.8 80 13 3.8 [10,20,30,40,50,60,70,80,90] 1
2 50 1 1 2 1 2 2 1 2 2 2 2 2 0.8 120 13 3.8 [10,20,30,40,50,60,70,80,90] 1
2 70 1 2 2 1 2 2 2 2 2 2 2 2 0.8 80 13 3.8 [10,20,30,40,50,60,70,80,90] 1
2 30 1 [1,2] 1 2 2 2 2 2 2 2 2 2 0.8 33 13 3.8 80 1
2 30 1 2 2 2 2 2 2 2 2 2 2 2 0.8 [33,80,120,160,200,250] 200 3.8 -
       [10,20,30,40,50,60,70,80,90] 1
2 30 1 2 2 2 2 2 2 2 2 2 2 2 0.8 80 13 3.8 70 1
                :           :           :
```

The decision attribute is the first attribute CLASS (1:die, 2:live), and we fixed α=0.25 and β=0.85. Let us show the results of two cases.

(CASE 1) Obtained Rules from 80 Objects without Missing Values
It is possible to apply our programs to the standard $DISs$. For 80 objects, it took 0.015(sec), and 14 rules including the following are generated.

```
[AGE,30]=>[Class,live] (support=0.287,accuracy=0.958),
[ASCITES,yes]=>[CLASS,live] (support=0.775,accuracy=0.912),
[ALBUMIN,4.5]=>[CLASS,live] (support=0.287,accuracy=0.958).
```

(CASE 2) Obtained Rules from 155 Objects with 167 Missing Values
Due to two strategies, 22 rules and 25 rules are generated, respectively. It took 0.064(sec). Let us show every rule, which is obtained by *upper approximation strategy* but is not obtained by *lower approximation strategy*. Namely, every rule is in boundary set $Rule(0.25, 0.85, UA) - Rule(0.25, 0.85, LA)$. There are three such rules.

```
[Alk_PHOSPHATE,80]=>[CLASS,live]
  (minsupp=0.25,minacc=0.841,maxsupp=0.348,maxacc=0.857)
[ANOREXIA,yes]&[SGOT,13]=>[CLASS,live]
  (minsupp=0.25,minacc=0.829,maxsupp=0.381,maxacc=0.855)
[SPLEEN_PALPABLE,yes]&[SGOT,13]=>[CLASS,live]
  (minsupp=0.25,minacc=0.848,maxsupp=0.368,maxacc=0.877)
```

In the 17th attribute SGOT, there are four missing values. The above two rules with descriptor [SGOT,13] depend upon these four missing values. These rules show us the difference between *lower approximation strategy* and *upper approximation strategy*.

We are also focusing on the difference between rule generation in $DISs$ and $NISs$. Let us suppose a NIS. We remove every object with non-deterministic information from the NIS, and we obtain a DIS. We are interested in rules, which are not obtained from the DIS but obtained from the NIS.

According to some experiments including Hepatitis data and Mammographic data in UCI repository, we verified our utility programs work well, even if there are huge number of derived $DISs$. However, we have not analyzed the meaning

of the obtained rules. Because, the main issue of this paper is to establish the framework and to implement algorithms. From now on, we will apply our utility programs to real data with missing values, and we want to obtain meaningful rules from $NISs$. Our research is not toward rule generation from data with a large number of objects, but it is toward rule generation from incomplete data with a large number of derived $DISs$.

This paper is a revised and extended version of papers [31, 32].

Acknowledgment The authors would be grateful to anonymous referees for their useful comments. This work is partly supported by the Grant-in-Aid for Scientific Research (C) (No.16500176, No.18500214), Japan Society for the Promotion of Science.

References

1. Agrawal, R. and Srikant, R.: Fast Algorithms for Mining Association Rules. In : *Proceedings of the 20th Very Large Data Base*, (1994) 487-499.
2. Agrawal, R., Mannila, H., Srikant, R., Toivonen, H. and Verkamo, A.: Fast Discovery of Association Rules. *Advances in Knowledge Discovery and Data Mining*, AAAI/MIT Press, (1996) 307-328.
3. Demri, S. and Orłowska, E.: Incomplete Information: Structure, Inference, Complexity, Monographs in Theoretical Computer Science. Springer (2002).
4. Grzymala-Busse, J.: On the Unknown Attribute Values in Learning from Examples. In : *Proceedings of the 6th International Symposium on Methodologies for Intelligent Systems (LNAI 542)*, Charlotte, North Carolina (1991) 368-377.
5. Grzymala-Busse, J.: A New Version of the Rule Induction System LERS. *Fundamenta Informaticae* 31 (1997) 27-39.
6. Grzymala-Busse, J. and Werbrouck, P.: On the Best Search Method in the LEM1 and LEM2 Algorithms. *Incomplete Information: Rough Set Analysis* 13, Phisica-Verlag (1998) 75-91.
7. Grzymala-Busse, J.: Data with Missing Attribute Values: Generalization of Indiscernibility Relation and Rule Induction. *Transactions on Rough Sets* 1 (2004) 78-95.
8. Grzymala-Busse, J.: Incomplete data and generalization of indiscernibility relation, definability, and approximations. In : *Proceedings of the 10th International Conference on Rough Sets, Fuzzy Sets, data Mining, and Granular Computing (LNAI 3641)*, Regina, Canada (2005) 244-253.
9. Komorowski, J., Pawlak, Z., Polkowski, L. and Skowron, A.: Rough Sets: a tutorial. In: Pal, S. and Skowron, A. (Eds.), *Rough Fuzzy Hybridization*. Springer (1999) 3-98.
10. Kryszkiewicz, M.: Rules in Incomplete Information Systems. *Information Sciences* 113 (1999) 271-292.
11. Kryszkiewicz, M. and Rybinski, H.: Computation of Reducts of Composed Information Systems. *Fundamenta Informaticae* 27 (1996) 183-195.
12. Kryszkiewicz, M.: *Maintenance of Reducts in the Variable Precision Rough Sets Model*. ICS Research Report 31/94, Warsaw University of Technology (1994).
13. Lipski, W.: On Semantic Issues Connected with Incomplete Information Data Base. *ACM Trans. DBS* 4 (1979) 269-296.

14. Lipski, W.: On Databases with Incomplete Information. *Journal of the ACM* 28 (1981) 41-70.
15. Nakamura, A., Tsumoto, S., Tanaka, H. and Kobayashi, S.: Rough Set Theory and Its Applications. *Journal of Japanese Society for AI* 11 (1996) 209-215.
16. Nakamura, A.: A Rough Logic based on Incomplete Information and Its Application. *International Journal of Approximate Reasoning* 15 (1996) 367-378.
17. Nakata, M. and Sakai, H.: Rough-set-based Approaches to Data Containing Incomplete Information: Possibility-based Cases. In: Nakamatsu, K. and Abe, J. (Eds.), *Advances in Logic Based Intelligent Systems*, *Frontiers in Artificial Intelligence and Applications* 132. IOS Press, (2005) 234-241.
18. Nakata, M. and Sakai, H.: Lower and Upper Approximations in Data Tables Containing Possibilistic Information. *Transactions on Rough Sets* 7 (2007) 170-189.
19. Orłowska, E.: What You Always Wanted to Know about Rough Sets. *Incomplete Information: Rough Set Analysis* 13, Physica-Verlag (1998) 1-20.
20. Orłowska, E. and Pawlak, Z.: Representation of Nondeterministic Information. *Theoretical Computer Science* 29 (1984) 27-39.
21. Pawlak, Z.: Rough Sets, Kluwer Academic Publisher (1991).
22. Pawlak, Z.: Some Issues on Rough Sets. *Transactions on Rough Sets* 1 (2004) 1-58.
23. Polkowski, L. and Skowron, A. (Eds.): *Rough Sets in Knowledge Discovery 1*, *Studies in Fuzziness and Soft Computing* 18, Physica-Verlag (1998).
24. Polkowski, L. and Skowron, A. (Eds.): *Rough Sets in Knowledge Discovery 2*, *Studies in Fuzziness and Soft Computing* 19, Physica-Verlag (1998).
25. Rough Set Software. *Bulletin of Int'l. Rough Set Society* 2 (1998) 15-46.
26. Sakai, H.: Effective Procedures for Handling Possible Equivalence Relations in Nondeterministic Information Systems. *Fundamenta Informaticae* 48 (2001) 343-362.
27. Sakai, H.: Effective Procedures for Data Dependencies in Information Systems. *Rough Set Theory and Granular Computing*, *Studies in Fuzziness and Soft Computing* 125. Springer-Verlag, (2003) 167-176.
28. Sakai, H. and Okuma, A.: Basic Algorithms and Tools for Rough Non-deterministic Information Analysis. *Transactions on Rough Sets* 1 (2004) 209-231.
29. Sakai, H. and Nakata, M.: An Application of Discernibility Functions to Generating Minimal Rules in Non-deterministic Information Systems. *Journal of Advanced Computational Intelligence and Intelligent Informatics* 10 (2006) 695-702.
30. Sakai, H.: On a Rough Sets Based Data Mining Tool in Prolog: An Overview. In : *Declarative Programming for Knowledge Management, Revised Selected Papers (LNAI 4369)* (2006) 48-65.
31. Sakai, H. and Nakata, M.: On Possible Rules and Apriori Algorithm in Nondeterministic Information Systems. In : *Proceedings of the 5th International Conference on Rough Sets and Current Trends in Computing (LNAI 4259)*, Kobe, Japan (2006) 264-273.
32. Sakai, H., Ishibashi, R., Koba, K. and Nakata, M.: On Possible Rules and Apriori Algorithm in Non-deterministic Information Systems 2. In : *Proceedings of the 11th International Conference on Rough Sets, Fuzzy Sets, Data Mining and Granular Computing (LNAI 4482)*, Toronto, Canada (2007) 280-288.
33. Skowron, A. and Rauszer, C.: The Discernibility Matrices and Functions in Information Systems. In: *Intelligent Decision Support - Handbook of Advances and Applications of the Rough Set Theory*, Kluwer Academic Publishers, (1992) 331-362.
34. Stefanowski, J. and Tsoukias, A.: On the Extension of Rough Sets under Incomplete Information. In: *Proceedings of the 7th International Workshop on Rough*

Sets, Data Mining and Granular-Soft Computing (LNAI 1711), Yamaguchi, Japan (1999) 73-81.
35. Stefanowski, J. and Tsoukias, A.: Incomplete Information Tables and Rough Classification. *Computational Intelligence* 7 (2001) 212-219.
36. Tsumoto, S.: Knowledge Discovery in Clinical Databases and Evaluation of Discovered Knowledge in Outpatient Clinic. *Information Sciences* 124 (2000) 125-137.
37. UCI Machine Learning Repository: `http://mlearn.ics.uci.edu/MLRepository.html`
38. Ziarko, W.: Variable Precision Rough Set Model. *Journal of Computer and System Sciences* 46 (1993) 39-59.