On a Rough Sets based Data Mining Tool in Prolog: An Overview

著者	Sakai Hiroshi
journal or publication title	Lecture Notes in Artificial Intelligence
volume	4369
page range	48-65
year	2005-10-22
URL	http://hdl.handle.net/10228/00006373

doi: info:doi/10.1007/11963578_5

On a Rough Sets Based Data Mining Tool in Prolog: An Overview

Hiroshi Sakai

Department of Mathematics and Computer Aided Science
Faculty of Engineering, Kyushu Institute of Technology
Tobata, Kitakyushu 804, Japan
`sakai@mns.kyutech.ac.jp`

Abstract. *Rough Non-deterministic Information Analysis* (*RNIA*) is a framework for handling rough sets based concepts, which are defined in not only *DISs* (*Deterministic Information Systems*) but also *NISs* (*Non-deterministic Information Systems*), on computers. *RNIA* is also recognized as a framework of data mining from uncertain tables. This paper focuses on programs in prolog, and briefly surveys a software tool for *RNIA*.

1 Introduction

Rough set theory offers a new mathematical approach to vagueness and uncertainty, and the rough sets based concepts have been recognized to be very useful [1,2,3,4]. This theory usually handles tables with deterministic information, which we call *Deterministic Information Systems* (*DISs*). Many applications of this theory to information analysis, data mining, rule generation, machine learning and knowledge discovery have been investigated [5,6,7,8,9].

Non-deterministic Information Systems (*NISs*) and *Incomplete Information Systems* have been proposed for handling information incompleteness in *DISs* [10,11,12,13,14,15,16,17]. In [10,11,12], the necessity of non-deterministic information is shown. In [13,14], Lipski showed a question-answering system besides an axiomatization of logic. The relation between rough logic and incomplete information is clarified in [15], and relational algebra with null values is also discussed in [17].

We have also proposed a framework *RNIA* depending upon not only *DISs* but also *NISs*, and we have realized a software tool. For realizing this tool, we developed several effective algorithms, and we employed prolog for implementation. Because, it is necessary to handle non-deterministic cases in *NISs*, it may be difficult to apply a procedural language like C for realizing this tool. As far as the author knows, very little work deals with *NISs* nor incomplete information on computers. Throughout this paper, we show several examples, which clarify the role of this software tool for *RNIA*.

2 Basic Definitions and Information Analysis in Deterministic Information Systems

This section surveys basic definitions on rough sets and rough sets based information analysis according to [1,2,3,4].

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*.

For $ATR=\{A_1, \cdots, A_n\} \subset AT$, we call $(f(x, A_1), \cdots, f(x, A_n))$ a *tuple* (for ATR) of $x \in OB$. If $f(x, A)=f(y, A)$ holds 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. Let $[x]_{ATR}$ denote an equivalence class $\{y \in OB | f(y, A)=f(x, A)$ for every $A \in ATR\}$, and let $eq(ATR)$ denote the family of all equivalence classes for ATR. We identify an equivalence relation for ATR with $eq(ATR)$. A formula $[A, f(x, A)]$ implies that $f(x, A)$ is the value of the attribute A. This is called a *descriptor*.

Now, let us show some rough sets based concepts defined in $DISs$ [1,3].

(i) The Definability of a Set: If a set $X \subset OB$ is the union of some equivalence classes in $eq(ATR)$, we say X is *definable* (for ATR) in DIS. Otherwise, we say X is *rough* (for ATR) in DIS. For example, if $X=[x_1]_{\{A\}} \cup [x_2]_{\{A\}}$ holds, X is characterized by a formula $[A, f(x_1, A)] \vee [A, f(x_2, A)]$. If X is not definable for any ATR, it is impossible to characterize X by means of conditions on descriptors.

(ii) The Consistency of an Object: Let us consider two disjoint sets $CON \subset AT$ which we call *condition attributes* and $DEC \subset AT$ which we call *decision attributes*. An object $x \in OB$ is *consistent* (with any other object $y \in OB$ in the relation from CON to DEC), if $f(x, A)=f(y, A)$ holds for every $A \in CON$ implies $f(x, A)=f(y, A)$ holds for every $A \in DEC$.

(iii) Dependencies among Attributes: We call a ratio $deg(CON, DEC)= |\{x \in OB|$ x is consistent in the relation from CON to DEC $\}|/|OB|$ the *degree of dependency* from CON to DEC. Clearly, $deg(CON, DEC)=1$ holds if and only if every object $x \in OB$ is consistent.

(iv) Rules and Criteria (Support, Accuracy and Coverage): For any object $x \in OB$, let $imp(x, CON, DEC)$ denote a formula called an *implication*: $\wedge_{A \in CON}[A, f(x, A)] \Rightarrow \wedge_{A \in DEC}[A, f(x, A)]$. In most of work on rule generation, a rule is defined by an implication $\tau : imp(x, CON, DEC)$ satisfying some constraints. A constraint, such that $deg(CON, DEC)=1$, has been proposed in [1]. Another familiar constraint is defined by three values in the following:

$support(\tau)= |[x]_{CON} \cap [x]_{DEC}|/|OB|$,
$accuracy(\tau)=|[x]_{CON} \cap [x]_{DEC}|/|[x]_{CON}|$,
$coverage(\tau)=|[x]_{CON} \cap [x]_{DEC}|/|[x]_{DEC}|$.

(v) Reduction of Condition Attributes in Rules: Let us consider such an implication $imp(x, CON, DEC)$ that x is consistent in the relation from CON to DEC. An attribute $A \in CON$ is *dispensable* in CON, if x is consistent in the relation from $CON - \{A\}$ to DEC.

Rough set theory makes use of equivalence classes for ATR. Every definition from (i) to (v) is examined by means of applying equivalence classes. As for the definability of a set $X \subset OB$, X is definable (for ATR) in a DIS, if and only if $\cup_{x \in X}[x]_{ATR}{=}X$ holds.

Now, let us show the most important proposition, which connects two equivalence classes $[x]_{CON}$ and $[x]_{DEC}$ with the consistency of x.

Proposition 1 [1]. For every DIS, (1) and (2) in the following are equivalent.
(1) An object $x \in OB$ is consistent in the relation from CON to DEC.
(2) $[x]_{CON} \subset [x]_{DEC}$.

According to Proposition 1, the degree of dependency from CON to DEC is equal to $|\{x \in OB | [x]_{CON} \subset [x]_{DEC}\}|/|OB|$. As for criteria *support*, *accuracy* and *coverage*, they are defined by equivalence classes $[x]_{CON}$ and $[x]_{DEC}$. As for the reduction of attributes values in rules, let us consider such an implication $imp(x, CON, DEC)$ that x is consistent in the relation from CON to DEC. Then, an attribute $A \in CON$ is dispensable, if $[x]_{CON-\{A\}} \subset [x]_{DEC}$ holds.

In this way, definitions from (i) to (v) are uniformly computed by means of applying equivalence classes in DISs.

2.2 Rough Sets Based Information Analysis in Deterministic Information Systems

Let us see an outline of rough sets based information analysis according to Table 1, which shows a relation between attributes $Head(ache)$, $Temp(erature)$ and Flu over a set $Patient$ of objects. This table may be too small, but it will be sufficient to know rough sets based concepts.

Table 1. A deterministic information system

$Patient$	$Head(ache)$	$Temp(erature)$	Flu
$p1$	no	$very\_high$	yes
$p2$	yes	$very\_high$	yes
$p3$	no	$normal$	no

We identify a tuple with a set of implications, for example,
`imp1:[Head,no]`$\Rightarrow$`[Flu,yes]`,
`imp2:[Head,no]`$\wedge$`[Temp,very_high]`$\Rightarrow$`[Flu,yes]`
are extracted from patient $p1$, and
`imp3:[Head,no]`$\Rightarrow$`[Flu,no]`
is extracted from $p3$. Implication $imp1$ contradicts $imp3$, because the same condition $[Head, no]$ concludes the different decisions $[Flu, yes]$ and $[Flu, no]$. How-

ever, $imp2$ is consistent with implications from any other tuple. Most of rough sets based rules are defined by means of this concept of '*consistency*' [1,2,3,4]. We usually define rules in a DIS by consistent implications.

Three measures, *support*, *accuracy* and *coverage* are also applied to defining rules in $DISs$ [1,2,4,8]. Each value of every measure is between 0 and 1. Implication $imp1$ occurs once in Table 1, so $support(imp1)$=1/3. This means $imp1$ represents about 33% data in Table 1. A formula $[Head, no]$ occurs twice and $[Flu, yes]$ occurs once under the condition of $[Head, no]$, so $accuracy(imp1)$=1/2. This ratio 1/2 means the degree of the consistency of $imp1$. Similarly, a formula $[Flu, yes]$ occurs twice and $[Head, no]$ occurs once under the condition of $[Flu, yes]$, so $coverage(imp1)$=1/2.

Equivalence classes in $DISs$ are usually employed to examine every concept [1,2,3,4,5,6,7]. In Table 1, both $p1$ and $p3$ satisfy $[Head, no]$, so $p1$ and $p3$ belong to the same class. Patient $p2$ only satisfies $[Head, yes]$, so $p2$ belongs to another class. In this way, we have equivalence classes $h1=\{p1, p3\}$ and $h2=\{p2\}$ on an attribute $Head$. We similarly have equivalence classes $t1=\{p1, p2\}$ and $t2=\{p3\}$ on $Temp$, and $f1=\{p1, p2\}$ and $f2=\{p3\}$ on Flu.

According to Proposition 1, the concept of the consistency is examined by the inclusion of equivalence classes. The relation $h1 \not\subset f1$ implies that $p1, p3 \in h1$ are inconsistent for attributes $Head$ and Flu, and the relation $t1 \subset f1$ implies $p1, p2 \in t1$ are consistent for attributes $Temp$ and Flu.

Data dependency between attributes is also examined by equivalence classes. For attributes $CON=\{Head, Temp\}$ and $DEC=\{Flu\}$, we have two families of all equivalence classes. $eq(CON)=\{\{p1\}, \{p2\}, \{p3\}\}$ and $eq(DEC)=\{\{p1, p2\}, \{p3\}\}$, respectively. For every $X \in eq(CON)$, there exists $Y \in eq(DEC)$ such that $X \subset Y$. Therefore, every object is consistent with other object. In this case, the degree of dependency from CON to DEC is 1.

3 Rough Non-deterministic Information Analysis

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 a real value in this set but this value is not known. Especially if the real value is not known at all, $g(x, A)$ is equal to VAL_A.

$NISs$ were proposed by Pawlak, Orłowska and Lipski in order to handle information incompleteness in $DISs$ [10,11,12,13,14].

Table 2. A non-deterministic information system

$Patient$	$Head(ache)$	$Temp(erature)$	Flu
$p1$	$\{no\}$	$\{very\_high\}$	$\{yes\}$
$p2$	$\{yes, no\}$	$\{high, very\_high\}$	$\{yes\}$
$p3$	$\{no\}$	$\{normal, high\}$	$\{yes, no\}$

In Table 2, it is possible to obtain a *DIS* by replacing every set with a value in every set. There are 16 possible *DISs*, which we name *derived DISs*. Table 1 is a derived *DIS* from *NIS* in Table 2. According to the interpretation to *NISs*, there exists a derived *DIS* with real information in these 16 derived *DISs*. Two modalities *certainty* and *possibility*, which are defined by means of all derived *DISs*, are introduced into *NISs*.
(Certainty) If a formula α holds in every derived *DIS* from a *NIS*, α also holds in the unknown real *DIS*.
(Possibility) If a formula α holds in some derived *DISs* from a *NIS*, there exists such a possibility that α holds in the unknown real *DIS*.

We have coped with several issues related to these two modalities, for example, the definability of a set in *NISs* [18,19], the consistency of an object in *NISs*, data dependency in *NISs* [20], rules in *NISs* [21], reduction of attributes in *NISs* [22], etc. An important problem is how to compute two modalities depending upon all derived *DISs* from a *NIS*. A simple method, such that every definition is sequentially computed in all derived *DISs* from a *NIS*, is not suitable, because the number of derived *DISs* from a *NIS* increases in exponential order. We have solved this problem by means of applying either *inf* and *sup* information or *possible equivalence relations* [19,20,21].

4 An Overview of a Tool for RNIA

Now, we sequentially refer to a software tool handling *NISs*. This tool mainly consists of the following:
(1) *Programs for checking the definability of a set*
(2) *Programs for equivalence relations*
(3) *Programs for data dependency*
(4) *Programs for rule generation*
Programs are implemented in prolog and C, and they are realized on a workstation with 450 MHz UltraSparc CPU.

4.1 An Exemplary Non-deterministic Information System

Table 3 is an artificial database, which is automatically generated by using a random number generation program. The following is the real prolog data expressing Table 3. According to this syntax, it is possible to handle any *NISs*.

```
% more data.pl
object(10,8).
data(1,[3,[1,3,4],3,2,5,5,[2,4],3]).
data(2,[2,[3,4],[1,3,4],4,[1,2],[2,4,5],2,2]).
data(3,[[4,5],5,[1,5],5,2,5,[1,2,5],1]).
   :        :        :
data(9,[2,3,5,3,[1,3,5],4,2,3]).
data(10,[4,2,1,5,2,[4,5],3,1]).
```

Table 3. An exemplary NIS. Here, OB=$\{1, 2, \cdots, 10\}$ and AT=$\{A, B, \cdots, H\}$. For object 1 and attribute A, the attribute value is definite, and it is 3. For object 1 and attribute B, there exists a set {1,3,4}. We interpret that either 1, 3 or 4 is the real attribute value, but it is impossible to decide the real value due to the information incompleteness.

OB	A	B	C	D	E	F	G	H
1	{3}	{1,3,4}	{3}	{2}	{5}	{5}	{2,4}	{3}
2	{2}	{3,4}	{1,3,4}	{4}	{1,2}	{2,4,5}	{2}	{2}
3	{4,5}	{5}	{1,5}	{5}	{2}	{5}	{1,2,5}	{1}
4	{1}	{3}	{4}	{3}	{1,2,3}	{1}	{2,5}	{1,2}
5	{4}	{1}	{2,3,5}	{5}	{2,3,4}	{1,5}	{4}	{1}
6	{4}	{1}	{5}	{1}	{4}	{2,4,5}	{2}	{1,2,3}
7	{2}	{4}	{3}	{4}	{3}	{2,4,5}	{4}	{1,2,3}
8	{4}	{5}	{4}	{2,3,5}	{5}	{3}	{1,2,3}	{1,2,3}
9	{2}	{3}	{5}	{3}	{1,3,5}	{4}	{2}	{3}
10	{4}	{2}	{1}	{5}	{2}	{4,5}	{3}	{1}

In Table 3, there are $12(= 2^2 \times 3)$ derived $DISs$ for attributes $\{A, B\}$, and we see there exists a DIS, which contains real information, in 12 derived $DISs$. For attributes $\{A, B, C, D, E, F, G, H\}$, there are (more than 7 billion) derived $DISs$. It will be hard to enumerate 7346640384 derived $DISs$ sequentially.

4.2 Definability of a Set in NISs

In Table 3, there are two derived $DISs$ for attribute A. If the attribute value is 4 in object 3, the equivalence relation (or the family of all equivalence classes) is $\{\{1\}, \{2, 7, 9\}, \{3, 5, 6, 8, 10\}, \{4\}\}$. Otherwise, the equivalence relation is $\{\{1\}, \{2, 7, 9\}, \{3\}, \{5, 6, 8, 10\}, \{4\}\}$. We name such equivalence relation a *possible equivalence relation* (*pe*-relation), and name every element in a *pe*-relation a *possible equivalence class* (*pe*-class). In a DIS, there exists an equivalence relation for $ATR \subset AT$, however there may exist some possible equivalence relations for ATR in a NIS.

In a NIS, the definability of a set depends upon every derived DIS, and two modalities are introduced into the definability of a set. In programs, we identify an attribute with the ordinal number of this attribute, for example, we identify attributes B and C with 2 and 3, respectively. As for descriptors, we identify [B,2] and [C,3] with [2,2] and [3,3], respectively. Let us show the real execution of programs.

```
% more attrib_atr.pl
atr([1,2,3]).
% prolog
?-consult(tool.pl).
yes
```

```
?-translate_atr. [Operation 1]
File Name for Read Open:'data.pl'.
Attribute Definition File:'attrib_atr.pl'.
EXEC_TIME=0.076(sec)
yes
?-class([6,7]). [Operation 2]
[1] (EQUIVALENCE)RELATION:[[6],[7]] for ATR=[1,2,3]
    POSITIVE SELECTION: CONDITION OF 6:[4,1,5], CONDITION OF 7:[2,4,3]
    NEGATIVE SELECTION: CONDITION OF 2:[2,4,3], CONDITION OF 5:[4,1,5]
Possibly definable !!
EXEC_TIME=0.002(sec)
yes
?-class([3,4]).
[1] (EQUIVALENCE)RELATION:[[3],[4]] for ATR=[1,2,3]
    POSITIVE SELECTION: CONDITION OF 3:[4,5,1], CONDITION OF 4:[1,3,4]
    NEGATIVE SELECTION: NO
[2] (EQUIVALENCE)RELATION:[[3],[4]] for ATR=[1,2,3]
      :       :       :
[4] (EQUIVALENCE)RELATION:[[3],[4]] for ATR=[1,2,3]
    POSITIVE SELECTION: CONDITION OF 3:[5,5,5], CONDITION OF 4:[1,3,4]
    NEGATIVE SELECTION: NO
Certainly definable !!
EXEC_TIME=0.006(sec)
yes
```

According to this execution, a set $\{6,7\}$ is *possibly definable*, namely there exist some $DISs$ which make a set $\{6,7\}$ definable. On the other hand, a set $\{3,4\}$ is *certainly definable*, namely this set is definable in all derived $DISs$.

In Operation 1, *data.pl* is translated to inf and *sup* information according to the attribute definition file *attrib_atr.pl*. Intuitively, inf is a set of objects with certain information and *sup* is a set of objects with possible information, for example, $inf(6,\{A,B,C\},(4,1,5))$=$\{6\}$, $sup(6,\{A,B,C\},(4,1,5))$=$\{5,6\}$, $inf(7,$ $\{A,B,C\},(2,4,3))$=$\{7\}$ and $sup(7,\{A,B,C\},(2,4,3))$=$\{2,7\}$. For such inf and *sup*, every equivalence class CL, which depends upon an object x, attributes ATR and its tuple, satisfies $inf(x,ATR,tuple) \subset CL \subset sup(x,ATR,tuple)$.

In Operation 2, the definability of a set $\{6,7\}$ is examined. Three lists are initialized to EQ=$\{\}$(Equivalence Relation), $PLIST$=$\{\}$(Positive Selection List) and $NLIST$=$\{\}$(Negative Selection List). In $\{6,7\}$, the first object 6 is picked up. Here, the applicable equivalence classes of object 6 are $\{6\}$(=inf) and $\{5,6\}$(=*sup*). Since $\{5,6\} \not\subset \{6,7\}$, $\{6\}$ is selected, and lists are revised to EQ=$\{\{6\}\}$ and $PLIST$=$\{[6,(4,1,5)]\}$. At the same time, $\{5,6\}$ is rejected, and object 5 must have the different tuple from (4,1,5). Since there exist other tuples except (4,1,5) in object 5, [5,(4,1,5)] is added to the list of the negative selection, namely $NLIST$=$\{[5,(4,1,5)]\}$. The same procedure is repeated for a new set $\{7\}$(=$\{6,7\}-\{6\}$). Similarly, just an equivalence class $\{7\}$(=inf) is applicable to this new set. The tuple (2,4,3) does not violate the current selections, so [7,(2,4,3)] is added to the list of the positive selection, and [2,(2,4,3)] is added to the list of the negative selection. Namely, we have EQ=$\{\{6\},\{7\}\}$, $PLIST$=$\{[6,(4,1,5)],[7,(2,4,3)]\}$ and $NLIST$=$\{[5,(4,1,5)],[2,(2,4,3)]\}$. Since we have an empty set in the next step, we know a set $\{6,7\}$ is definable according to selections. The order of the translation program depends upon $|derived\ DISs| \times |OB|^2$, and the order of program $class(SET)$ depends upon the number of derived $DISs$, which make SET definable. We show program $class0$, which is the main part of program $class$.

```
class0(ATT,SET,EQ,EQ_Ans,PLIST,PLIST_Ans,NLIST,NLIST_Ans)
  :-SET==[],EQ_Ans=EQ,PLIST_Ans=PLIST,NLIST_Ans=NLIST.
class0(ATT,[X|X1],EQ,EQ_Ans,PLIST,PLIST_Ans,NLIST,NLIST_Ans)
  :-candidate(ATT,[X|X1],CAN,PLIST,PLIST1,NLIST,NLIST1),
    minus([X|X1],CAN,REST),
    class0(ATT,REST,[CAN|EQ],EQ_Ans,PLIST1,PLIST_Ans,NLIST1,NLIST_Ans).
```
`ATT`:Attributes, `SET`:A set of objects, `EQ,PLIST,NLIST`:Temporary lists,
`EQ_Ans,PLIST_Ans,NLIST_Ans`:Obtained lists for *pe*-classes, PLIST and NLIST,
`CAN`:A candidate of *pe*-class including object X, `REST:REST=[X|X1]-CAN`.

In program $class0$, `candidate(ATT,SET,CAN,PLIST,PLIST1,NLIST,NLIST1)` finds a *pe*-class CAN which satisfies the next two conditions.
(1) $CAN \subset SET$, and $inf \subset CAN \subset sup$.
(2) This CAN makes no contradiction for $PLIST$ and $NLIST$.
The details of this algorithm are in [18,19].

4.3 Possible Equivalence Relations in NISs

A set of all *pe*-classes is a kind of reduced information from databases, and these *pe*-classes contain enough information to calculate most of rough set concepts.

Let us consider methods to obtain all kinds of *pe*-relations for any set of attributes. The first method is as follows;
(Method 1) Because OB is definable in all derived $DISs$, we solve the definability of a set OB, and we pick up a *pe*-relation from the variable EQ.

However, Method 1 depends upon $|derived\ DISs|$, and the number of derived $DISs$ increases in exponential order. So, we propose the second method.
(Method 2) Let $peq(A)$ be a *pe*-relation for a set A of attributes and $peq(B)$ be a *pe*-relation for a set B of attributes. Then, $\{M \subset OB | M = CL_A \cap CL_B, CL_A \in peq(A), CL_B \in peq(B)\}$ be a *pe*-relation for a set $A \cup B$. According to this property, we first generate all *pe*-relations for each attribute, and we execute program $merge$ for obtaining all kinds of *pe*-relations.

For handling equivalence relations in C language, we introduced two arrays $head[]$ and $succ[]$. For example, we express a class $\{1,2,3\}$ by $head[1]$=$head[2]$=$head[3]$=1, $succ[1]$=2, $succ[2]$=3 and $succ[3]$=0. Program $merge$ generates new arrays $head_{A\cup B}[]$ and $succ_{A\cup B}[]$ from $head_A[]$, $succ_A[]$, $head_B[]$ and $succ_B[]$. The order of $merge$ is $o(|OB|)$ in the best case, and the order is $o(|OB|^2)$ in the worst case [20]. In Method 2, it also seems necessary to apply program *merge* $|derived\ DISs|$ times. However in reality, lots of *pe*-relations become the same *pe*-relation, and the cases of applying *merge* are drastically reduced.

Let us show the real execution according to Method 2.

```
?-translate.
File Name for Read Open:'data.pl'.
EXEC_TIME=0.242(sec)
yes
?-pe. [Operation 3]
<< Attribute 1 >>
  [1] [[1],[2,7,9],[3,5,6,8,10],[4]] 1
  [2] [[1],[2,7,9],[3],[4],[5,6,8,10]] 1
  POSSIBLE CASES 2
<< Attribute 2 >>
  [1] [[1,5,6],[2,4,9],[3,8],[7],[10]] 1
  [2] [[1,5,6],[2,7],[3,8],[4,9],[10]] 1
      :       :       :
<< Attribute 8 >>
  [1] [[1,6,7,8,9],[2,4],[3,5,10]] 1
  [2] [[1,6,7,8,9],[2],[3,4,5,10]] 1
      :       :       :
  [54] [[1,9],[2],[3,4,5,6,7,8,10]] 1
  POSSIBLE CASES 54
EXEC_TIME=1.520(sec)
yes
```

In Operation 3, all *pe*-relations of each attribute are generated, and *pe*-relations are stored in files from 1.*pe* to 8.*pe*. For attribute 1 which means attribute A, there are two *pe*-relations $\{\{1\},\{2,7,9\},\{3,5,6,8,10\},\{4\}\}$ and $\{\{1\},\{2,7,9\},\{3\},\{4\},\{5,6,8,10\}\}$. There are 54 derived $DISs$ and 54 kinds of *pe*-relations for attribute 8.

```
% more 1.rs [Operation 4]
object(10).
attrib(1).
cond(1,1,1,3).
pos(1,1,1).
cond(2,1,1,2).
pos(2,1,1).
cond(3,1,1,4).
   :       :       :
inf([1,1,1],[1,1,1],[[1],[1]]).
sup([1,1,1],[1,1,1],[[1],[1]]).
inf([2,1,1],[2,1,1],[[2,7,9],[1,1,1]]).
inf([7,1,1],[2,1,1],[[],[]]).
inf([9,1,1],[2,1,1],[[],[]]).
   :       :       :
inf([8,1,1],[5,1,1],[[],[]]).
inf([10,1,1],[5,1,1],[[],[]]).
% more 1.pe
10 1 2 2 1 0 2 7 9 0 3 5 6 8 10 0 4 0 -1 1 1 0 2 7 9 0 3 0
4 0 5 6 8 10 0 -1 1
% more merge.dat
123.pe
3
1.pe
2.pe
3.pe
% merge [Operation 5]
Merging
1.pe...
2.pe...
3.pe...
EXEC_TIME=0.010(sec)
% more 123.pe
10 3 216 4 1 0 2 0 3 0 4 0 5 0 6 0 7 0 8 0 9 0 10 0 -1 120
1 0 2 0 3 0 4 0 5 6 0 7 0 8 0 9 0 10 0 -1 60 1 0 2 7 0 3 0
4 0 5 0 6 0 8 0 9 0 10 0 -1 24 1 0 2 7 0 3 0 4 0 5 6 0 8 0
9 0 10 0 -1 12
```

In Operation 4, inf and *sup* information is displayed. The contents in the file 1.*pe* are also displayed. Every number 0 discriminates each *pe*-class. In Operation 5, all *pe*-relations for attributes $\{A, B, C\}$ are generated. Program $merge$ generates new *pe*-relations based on a set of *pe*-relations, which are defined in a file named $merge.dat$. In the generated file 123.*pe*, there are 216 derived $DISs$ and 4 kinds of *pe*-relations, i.e., 120 *pe*-relations of $\{\{1\}, \{2\}, \cdots, \{10\}\}$, 60 *pe*-relations of $\{\{1\}, \{2\}, \cdots, \{5, 6\}, \cdots, \{10\}\}$, 24 *pe*-relations of $\{\{1\}, \{2, 7\}, \{3\}, \cdots, \{10\}\}$ and 12 *pe*-relations of $\{\{1\}, \{2, 7\}, \{3\}, \cdots, \{5, 6\}, \cdots, \{10\}\}$.

Let us show execution time for other $NISs$ in Table 4. In Table 5, $N1$ denotes the number of derived $DISs$ for ATR=$\{A, B, C\}$, and $N2$ denotes the number of distinct *pe*-relations.

4.4 Degrees of Dependency in NISs

In a DIS, the degree of dependency $deg(CON, DEC)$ from CON to DEC is an important criterion for measuring the relation from CON to DEC. The concept of the consistency can be characterized by the inclusion relation of equivalence classes according to Proposition 1, i.e., object x is consistent if and only if $[x]_{CON} \subset [x]_{DEC}$ for $[x]_{CON} \in eq(CON)$ and $[x]_{DEC} \in eq(DEC)$. Thus, the numerator in the degree is $|\cup\{[x]_{CON} | [x]_{CON} \subset [x]_{DEC}\}|$, which is easily calcu-

Table 4. Definitions of $NISs$

NIS	$\|OB\|$	$\|AT\|$	$Derived\_DISs$
NIS_1	30	5	$7558272(= 2^7 \times 3^{10})$
NIS_2	50	5	$120932352(= 2^{11} \times 3^{10})$
NIS_3	100	5	$1451188224(= 2^{13} \times 3^{11})$

Table 5. Execution time (sec). M1 means Method 1 and M2 means Method 2. If there exist lots of pe-relations, Method 2 seems more effective than Method 1.

NIS	$translate(in\_prolog)$	$pe(in\_prolog)$	$merge(in\_C)$	$N1$	$N2$
NIS_1	$M1 : 0.134/M2 : 0.308$	$M1 : 13.351/M2 : 1.415$	$M1 : 0/M2 : 0.690$	5832	120
NIS_2	$M1 : 0.200/M2 : 0.548$	$M1 : 7.489/M2 : 8.157$	$M1 : 0/M2 : 0.110$	5184	2
NIS_3	$M1 : 0.483/M2 : 1.032$	$M1 : 56.300/M2 : 16.950$	$M1 : 0/M2 : 2.270$	20736	8

lated by using $eq(CON)$ and $eq(DEC)$. The order of this calculation depends upon the size of object $|OB|$.

In a NIS, there exist some derived $DISs$, so there exist the minimum and the maximum degree of dependency. Predicate *depratio* means 'dependency with consistent ratio for every object'.

```
% depratio [Operation 6]
File Name for Condition:123.pe
File Name for Decision:8.pe
------ Dependency Check ---------------------------
CRITERION 1(Num_of_Consistent_DISs/Num_of_All_DISs)
  Number of Derived DISs:11664
  Number of Derived Consistent DISs:8064
  Degree of Consistent DISs:0.691
CRITERION 2(Total_Min_and_Max_Degree)
  Minimum Degree of Dependency:0.600
  Maximum Degree of Dependency:1.000
------ Consistency Ratio for Every Object ---------
  Object 1:1.000(=11664/11664)
  Object 2:0.889(=10368/11664)
  Object 3:1.000(=11664/11664)
      :       :       :
  Object 9:1.000(=11664/11664)
  Object 10:1.000(=11664/11664)
EXEC_TIME=0.040(sec)
yes
```

In Operation 6, the degree of dependency from attributes $\{A, B, C\}$ to $\{H\}$ is examined. For a set of attributes $\{A, B, C, H\}$, there are 11664 derived $DISs$. Therefore, it is necessary to obtain each degree of dependency in 11664 derived

$DISs$. For solving this problem, we apply *pe*-relations. In reality, there exist only 4 *pe*-relations for a set of attributes $\{A, B, C\}$. and 54 *pe*-relations for $\{H\}$. It is possible to know all degree of dependency by means of checking the combinations of 4 and 54 *pe*-relations. The number of combinations is 216(=4×54).

Table 6. Execution time(sec). $N3$ denotes the number of derived $DISs$ for $\{A, B, C, E\}$, and $N4$ denotes the number of combined pairs of *pe*-relations in $\{A, B, C\}$ and $\{E\}$.

NIS	*depratio*	$N3$	$N4$
NIS_1	0.080	104976	2160
NIS_2	0.060	279936	108
NIS_3	0.130	4478976	1728

According to these two criterion values, we define the data dependency from CON to DEC in $NISs$. In Operation 6, 69% of all derived $DISs$ are consistent, and the minimum degree of dependency is 0.6. We may agree the dependency from $\{A, B, C\}$ to $\{H\}$. In Table 6, let us show execution time of *depratio* from $\{A, B, C\}$ to $\{E\}$ in other $NISs$.

4.5 Support, Accuracy and Coverage of Rules in NISs

Three measures *support*, *accuracy* and *coverage* in $DISs$ are extended to *minimum* and *maximum* of them, for example, *minacc* and *maxacc*.

Let us consider an implication *imp*4:[2,5]∧[5,2]⇒[8,1] from object 3 in Table 3. This implication *imp*4 appears in all 17946 derived $DISs$ for attributes $\{B, E, H\}$, so the minimum and the maximum values of three measures are definable for attributes $\{B, E, H\}$. The calculation of values depends upon all 17946 derived $DISs$, however it is possible to obtain these values due to the following results.

For a NIS, let us consider an implication τ:$[CON, \zeta] \Rightarrow [DEC, \eta]$. Let INA denote a set $[sup(x, CON, \zeta) - inf(x, CON, \zeta)] \cap sup(x, DEC, \eta)$, and let $OUTA$ denote a set $[sup(x, CON, \zeta) - inf(x, CON, \zeta)] - inf(x, DEC, \eta)$. Let INC denote a set $[sup(x, DEC, \eta) - inf(x, DEC, \eta)] \cap sup(x, CON, \zeta)$, and let $OUTC$ denote a set $[sup(x, DEC, \eta) - inf(x, DEC, \eta)] - inf(x, CON, \zeta)$. Then, the following holds [21]. Including the definitions of $inf(object, attributes, tuple)$ and $sup(object, attributes, tuple)$, some definitions are in [21].

(1) $minsup(\tau)=|inf(x, CON, \zeta) \cap inf(x, DEC, \eta)|/|OB|$.
(2) $maxsup(\tau)=|sup(x, CON, \zeta) \cap sup(x, DEC, \eta)|/|OB|$.
(3) $minacc(\tau)=\frac{(|inf(x,CON,\zeta)\cap inf(x,DEC,\eta)|)}{(|inf(x,CON,\zeta)|+|OUTA|)}$.
(4) $maxacc(\tau)=\frac{(|inf(x,CON,\zeta)\cap sup(x,DEC,\eta)|+|INA|)}{(|inf(x,CON,\zeta)|+|INA|)}$.
(5) $mincov(\tau)=\frac{(|inf(x,CON,\zeta)\cap inf(x,DEC,\eta)|)}{(|inf(x,DEC,\eta)|+|OUTC|)}$.
(6) $maxcov(\tau)=\frac{(|sup(x,CON,\zeta)\cap inf(x,DEC,\eta)|+|INC|)}{(|inf(x,DEC,\eta)|+|INC|)}$.

In this way, it is possible to obtain these values by using inf and sup information.

```
?-threevalues(3,[[2,5],[5,2]]). [Operation 7]
[(0.1,0.1),(1.0,1.0),(0.142,0.333)]
EXEC_TIME=0.001(sec)
yes
```

In Operation 7, the minimum and the maximum values of support, accuracy and coverage for $imp4$ are sequentially (0.1,0.1), (1.0,1.0) and (0.142,0.333). Since the minimum value of $accuracy$ is 1.0, $imp4$ is consistent in all derived $DISs$.

4.6 Certain and Possible Rules in NISs

In Table 3, let us consider $imp4$ in the previous subsection and $imp5$:[1,4]∧[2,5]⇒[8,1] from object 3. Implication $imp4$ is definite, and $imp4$ is consistent in all derived $DISs$. In this case, we say $imp4$ is *globally consistent* (GC). On the other hand $imp5$ is indefinite, since [1,4] is selected from [1,4]∨[1,5]. Implication $imp5$ is consistent in some derived $DISs$, and we say $imp5$ is *marginal* (MA). According to this consideration, we define 6 classes of implications in Table 7.

Table 7. Six classes of implications in NISs

	$GC(Globally\_Consistent)$	$MA(Marginal)$	$GI(Globally\_Inconsistent)$
$Definite$	DGC	DMA	DGI
$Indefinite$	IGC	IMA	IGI

In $DISs$, there exist only two classes DGC and DGI. These two classes are extended to 6 classes in $NISs$. In Table 7, implications in DGC class are not influenced by the information incompleteness, therefore we name implications in DGC class *certain rules*. We also name implications in either IGC, DMA or IMA classes *possible rules*.

For an implication imp, we may sequentially examine the consistency of imp and we know the class which imp belongs to. However, this method depends upon all derived $DISs$. There exists another method, which depends upon inf and sup information, to examine the class [21].

For $imp4$, $sup(3,\{B,E\},(5,2))$=$sup(3,\{B\},(5)) \cap sup(3,\{E\},(2))$=$\{3,8\} \cap \{2,3,4,5,10\}$=$\{3\}$, and $inf(3,\{H\},(1))$=$\{3,5,10\}$ holds. In this case, the inclusion relation $sup(3,\{B,E\},(5,2)) \subset inf(3,\{H\},(1))$ also holds, and this implies $imp4$ is GC. Similarly for $imp5$, $sup(3,\{A,B\},(4,5))$=$\{3,8\}$ holds. In this case, the inclusion relation $sup(3,\{A,B\},(4,5)) \subset inf(3,\{H\},(1))$ does not hold, and this implies $imp5$ is not GC. However, $inf(3,\{A,B\},(4,5))$=$\{3,8\}$ and $sup(3,\{H\},(1))$=$\{3,4,5,6,7,8,10\}$ holds, and the inclusion relation $inf(3,\{A,B\},(4,5)) \subset sup(3,\{H\},(1))$ holds. This implies $imp5$ is MA.

4.7 Minimal Certain Rules in NISs

Let us consider two implications *imp*6:[2,2]⇒[8,1] and *imp*7: [2,2]∧[3,1]⇒[8,1] from object 10. Both implications are certain rules, and *imp*6 is simpler than *imp*7, because [3,1] is added to the condition part of *imp*6. A *minimal* certain rule is a certain rule whose condition part is simpler than any other certain rules.

Now, we focus on minimal certain rule generation. Implication *imp*7 from object 10 belongs to DGC class, so it is possible to generate minimal certain rules from object 10. In this case, we employ a *discernibility function* $DF_{DGC}(10)$ of object 10. We have extended a discernibility function in $DISs$ [23] to a discernibility function in $NISs$. For the decision attribute $\{H\}$, we employ $inf(10,\{H\},(1))=\{3,5,10\}$. In DGC class, it is necessary to discriminate each object in $\{1,2,4,6,7,8,9\}=\{1,\cdots,10\}-inf(10,\{H\},(1))$ from $inf(10,\{H\},(1))$. Since $sup(10,\{A\},(4))=\{3,5,6,8,10\}$ and $1 \notin sup(10,\{A\},(4))$, the condition [$A$,4] can discriminate object 1 from object 10. Similarly, each condition [B,2], [C,1], [D,5], [E,2] and [G,3] can discriminate object 1 from object 10. In this way, a disjunction ([A,4]∨[B,2]∨[C,1]∨[D,5]∨ [E,2]∨[G,3]) becomes a condition, which discriminate object 1 from object 10. Let $DISC(10,1)$ denote this disjunction. A *discernibility function* $DF_{DGC}(10)$ is $\wedge_{i=1,2,4,6,7,8,9}DISC(10,i)$.
Theorem 2. [**22**] Let us suppose that an implication $[CON,\zeta] \Rightarrow [DEC,\eta]$ from object x belongs to DGC class. For a minimal solution SOL of $DF_{DGC}(x)$, $\wedge_{[A,\zeta_A]\in SOL}[A,\zeta_A] \Rightarrow [DEC,\eta]$ is a minimal certain rule from x.

4.8 Minimal Certain Rule Generation in NISs

We have proposed some algorithms to obtain a minimal solution of $DF_{DGC}(x)$. The details are in [22]. Let us show real execution.

```
% more attrib_rule.pl
decision([8]).
decval([1]).
condition([1,2,3,4,5,6,7]).
```

File *attrib_rule.pl* defines the implication: *condition* ⇒ [8, 1].

```
?-translate_rule. [Operation 8]
File Name for Read Open:'data.pl'.
Attribute Definition File:'attrib_rule.pl'.
EXEC_TIME=0.076(sec)
yes
?-init.
DECLIST:<inf=[3,5,10]>
Certain Rules come from [3,5,10]
EXEC_TIME=0.003(sec)
yes
```

In Operation 8, *inf* and *sup* information is created. Then, program *init* examines objects, which a certain rule can be generated from. In this case, we know that certain rules are generated from objects 3, 5 and 10.

```
?-minimal. [Operation 9]
<<Minimal Certain Rules from object 3>>
  DF:[[1,[2,5],[4,5],[5,2]], ···,[9,[2,5],[4,5],[5,2],[6,5]]]
<<Minimal Certain Rules from object 5>>
  DF:[[1,[1,4],[4,5]], ···,[8,[2,1],[7,4]],[9,[1,4],[2,1],[4,5],[7,4]]]
<<Minimal Certain Rules from object 10>>
  [2,2]=>[8,1][324/324(=6/6,54/54),DGC:Common]
  Rule covers objects [10], [(0.1,0.1),(1.0,1.0),(0.142,0.333)]
EXEC_TIME=0.015(sec)
yes
```

In Operation 9, program *minimal* tries to generate minimal certain rules, whose condition part consists of only core or common descriptors. As for objects 3 and 5, there is no such minimal certain rule, and every discernibility function in each object is displayed. For object 10, there exists such a minimal certain rule, which is *imp*6. For objects 3 and 5, we apply interactive method.

```
?-solall(5). [Operation 10]
Input Descriptors to Start Exhaustive Search:5.
Exhaustive Search for less than 32 Cases !!
<<Minimal Certain Rules from object 5>>
    Core Descriptors:[]
    DF without Core:[[1,[1,4],[4,5]],[2,[1,4],[2,1],[4,5],[7,4]],
    [4,[1,4],[2,1],[4,5],[7,4]],[6,[4,5],[7,4]],[7,[1,4],[2,1],[4,5]],
    [8,[2,1],[7,4]],[9,[1,4],[2,1],[4,5],[7,4]]]
    Currently Selected Descriptors:[]
    [Loop:1]
     Descriptors in DF:[[1,4],[2,1],[4,5],[7,4]]
     Exhaustive Search for [[1,4],[2,1],[4,5],[7,4]]
     Finally Selected Descriptors:[]
     [4,5]&[7,4]=>[8,1][5832/5832(=108/108,54/54),DGC]
        This rule covers objects [5],Coverage=0.333
        [(0.1,0.1),(1.0,1.0),(0.142),(0.333)]
     [2,1]&[4,5]=>[8,1][972/972(=18/18,54/54),DGC]
        This rule covers objects [5],Coverage=0.333
        [(0.1,0.1),(1.0,1.0),(0.142),(0.333)]
     [1,4]&[7,4]=>[8,1][3888/3888(=72/72,54/54),DGC]
        This rule covers objects [5],Coverage=0.333
        [(0.1,0.1),(1.0,1.0),(0.142,0.333)]
EXEC_TIME(for Exhaustive Search)=0.014(sec)
yes
```

Table 8. Definitions of NISs

NIS	$\|OB\|$	$\|AT\|$	$\|VAL_A\|$	$derived\_DISs$
NIS_4	50	10	10	1.57×10^{18}
NIS_5	100	10	10	7.01×10^{35}
NIS_6	300	10	10	6.74×10^{86}

Table 9. Execution time(sec) of programs. The *object* column implies the number of objects, in which some minimal certain rules are generated. The execution time of *minimal* depends upon the number of objects.

NIS	$translate\_rule$	$minimal$	$object$	$solall$
NIS_4	0.896	0.723	7	0.764
NIS_5	6.503	3.589	16	1.370
NIS_6	49.892	35.345	21	2.943

In Operation 10, minimal certain rules from object 5 are handled. Predicate $solall(x)$ means 'Solve all solutions from object x'. In Loop 1, there are four descriptors in this discernibility function, and this value is less than 5. Therefore, exhaustive search begins for all subsets of four descriptors. Three minimal certain rules are generated, and these rules are all minimal certain rules from object 5. If the condition of the threshold value is not satisfied, we select another descriptor and the absorption law is applied to reducing the discernibility function. Then, the next loop is invoked.

Let us show execution time for other $NISs$ in Table 8. In Table 9, the *object* column implies the number of objects, in which some minimal certain rules are generated. The execution time of *minimal* depends upon the number of objects. Program *solall* are also applied to an object in each NIS. In this execution, the threshold value was fixed to 10. Since $|AT|$=10, this program began to enumerate $1024(= 2^{10})$ subsets without specifying any descriptors, and generated all minimal certain rules. According to Table 9, we may employ a threshold value 10 for $NISs$, which consists of more than 10 attributes.

5 Concluding Remarks

An overview of a tool in prolog and a framework of Rough Non-deterministic Information Analysis ($RNIA$) are surveyed according to [19,20,21,22]. We follow rough sets based concepts in $DISs$ and propose a framework of $RNIA$. $NISs$, which were proposed by Pawlak, Orłowska and Lipski, have been recognized to be one of the most important framework for handling incomplete information. Therefore, $RNIA$ will also be an important framework for rough sets based information analysis under incomplete information.

Acknowledgment

The author gratefully acknowledges the helpful comments and suggestions of the reviewers. This work is partly supported by the Grant-in-Aid for Scientific Research (C) (No.16500176, No.18500214) from Japanese Society for the Promotion of Science.

References

1. Z.Pawlak: *Rough Sets: Theoretical Aspects of Reasoning about Data*, Kluwer Academic Publishers, Dordrecht, 1991.
2. Z.Pawlak: Some Issues on Rough Sets, *Transactions on Rough Sets*, Int'l. Rough Set Society, vol.1, pp.1-58, 2004.
3. J.Komorowski, Z.Pawlak, L.Polkowski and A.Skowron: Rough Sets: a tutorial, *Rough Fuzzy Hybridization*, Springer, pp.3-98, 1999.
4. A.Nakamura, S.Tsumoto, H.Tanaka and S.Kobayashi: Rough Set Theory and Its Applications, *Journal of Japanese Society for AI*, vol.11, no.2, pp.209-215, 1996.
5. L.Polkowski and A.Skowron (eds.): *Rough Sets in Knowledge Discovery 1, Studies in Fuzziness and Soft Computing*, vol.18, Physica-Verlag, 1998.
6. L.Polkowski and A.Skowron (eds.): *Rough Sets in Knowledge Discovery 2, Studies in Fuzziness and Soft Computing*, vol.19, Physica-Verlag, 1998.
7. J.Grzymala-Busse: A New Version of the Rule Induction System LERS, *Fundamenta Informaticae*, vol.31, pp.27-39, 1997.
8. S.Tsumoto: Knowledge Discovery in Clinical Databases and Evaluation of Discovered Knowledge in Outpatient Clinic, *Information Sciences*, vol.124, pp.125-137, 2000.
9. Rough Set Software, *Bulletin of Int'l. Rough Set Society*, vol.2, pp.15–46, 1998.
10. E.Orłowska and Z.Pawlak: Representation of Nondeterministic Information, *Theoretical Computer Science*, vol.29, pp.27-39, 1984.
11. E.Orłowska (Ed.): *Incomplete Information: Rough Set Analysis*, Physica-Verlag, 1998.
12. S.Demri and E.Orłowska: *Incomplete Information: Structure, Inference, Complexity, Monographs in Theoretical Computer Science*, Springer, 2002.
13. W.Lipski: On Semantic Issues Connected with Incomplete Information Data Base, *ACM Trans. DBS*, vol.4, pp.269-296, 1979.
14. W.Lipski: On Databases with Incomplete Information, *Journal of the ACM*, vol.28, pp.41-70, 1981.
15. A.Nakamura: A Rough Logic based on Incomplete Information and Its Application, *Int'l. Journal of Approximate Reasoning*, vol.15, pp.367-378, 1996.
16. M.Kryszkiewicz: Rules in Incomplete Information Systems, *Information Sciences*, vol.113, pp.271-292, 1999.
17. M.Nakata and S.Miyamoto: Databases with Non-deterministic Information, *Bulletin of Int'l. Rough Set Society*, vol.7, pp.15-21, 2003.
18. H.Sakai and A.Okuma: An Algorithm for Finding Equivalence Relations from Tables with Non-deterministic Information, *Lecture Notes in AI*, Springer-Verlag, vol.1711, pp.64–72, 1999.
19. H.Sakai: Effective Procedures for Handling Possible Equivalence Relations in Non-deterministic Information Systems, *Fundamenta Informaticae*, vol.48, pp.343-362, 2001.

20. H.Sakai: Effective Procedures for Data Dependencies in Information Systems, *Rough Set Theory and Granular Computing, Studies in Fuzziness and Soft Computing*, Springer, vol.125, pp.167–176, 2003.
21. H.Sakai and A.Okuma: Basic Algorithms and Tools for Rough Non-deterministic Information Analysis, *Transactions on Rough Sets*, Int'l. Rough Set Society, vol.1, pp.209-231, 2004.
22. H.Sakai and M.Nakata: Discernibility Functions and Minimal Rules in Non-deterministic Information Systems, *Lecture Notes in Artificial Intelligence*, Springer-Verlag, Vol.3641, pp.254-264, 2005.
23. A.Skowron and C.Rauszer: The Discernibility Matrices and Functions in Information Systems, *Intelligent Decision Support - Handbook of Advances and Applications of the Rough Set Theory*, Kluwer Academic Publishers, pp. 331-362, 1992.