Rough Set Approaches for Discovery of Rules and Attribute Dependencies

Wojciech Ziarko
Department of Computer Science
University of Regina
Regina, SK, S4S 0A2 Canada

Abstract

The article presents an elementary overview of techniques for data analysis and predictive modeling from data using the rough set approach. The specific knowledge discovery-related data analysis problems discussed in the article are the discovery of functional and partial functional dependencies and the discovery of rules in data. The presentation is focused on the application of the basic rough set model to knowledge discovery and does not include the discussion of the application of the extended probabilistic model of rough set for that purpose.

Keywords: rough sets, knowledge discovery, data mining, data dependencies, rules.

1 Introduction

The theory of rough sets was introduced by Pawlak [1,7,16,17,19-21,32,43] in early eighties. The intuition behind this idea is derived from the simple fact that in real life, when dealing with sets we often have no means of distinguishing individual set elements. The elements may posses some measurable characteristics but in principle, due to limited resolution of our perception mechanism, we can distinguish only classes of elements rather than individuals. Elements within classes are indistinguishable.

For example, assume that we have a large database of hospital patient records and that all what we know about each patient is whether patient’s body temperature (BT) is LOW, NORMAL or HIGH, whether patients blood pressure (BP) is LOW, NORMAL or HIGH and whether the heart rate (HR) is NORMAL or ABNORMAL. If that’s all what we know, then regardless how many records we would have in our database, we will have maximum eighteen classes of patients corresponding to different combinations of values BP, BT and HR, such as for instance the combination (BT=NORMAL, BP=High, HR=ABNORMAL). Each combination potentially corresponds to thousands of patient records, each of which is not different, as far as blood pressure and body temperature are concerned, from any other record from this group. We say that the patients have been classified into a certain number of categories, or elementary sets in the rough set terminology. Such a classification has several advantages:

1. Reduction in the complexity of information representation (rather than analyzing huge information files about individual patients we deal with relatively small tables corresponding to observed combinations of generalized features).

2. The ability to discover repetitive patterns in data since each combination potentially corresponds to a large number of patients.

3. The ability to reason about relations among features (attributes) of objects occurring in a large data collection based on results of analysis of relatively small and size-constrained tables, referred to as decision tables, representing classifications of the domain of interest.

The theory of rough sets provides a collection of methods which can be applied to the analysis and reduction of such tables. The methods are in particular applicable to data mining or knowledge discovery problems but also have other applications such as trainable control and pattern recognition[30,33-44].

For the purpose of illustration, we will use the decision table given in Table 2, which was derived from data collection partially shown in Table 1 by replacing the original attribute values with some discrete ranges. In this table, the column headings S,H,E,C,T are abbreviations for attribute names SIZE, HEIGHT, ENERGY, CURRENT and TEMPERATURE, respectively. The attributes are generalized properties of objects, identified by sequence numbers in the column OBJ of Table 1. The classes of objects with the same combinations of attribute values are identified in Table 2 in the column CLASS.

### Table 1: Example raw data representing 9558 objects

<table>
<thead>
<tr>
<th>OBJ</th>
<th>S</th>
<th>H</th>
<th>E</th>
<th>C</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>12.0</td>
<td>132.2</td>
<td>1715</td>
<td>7.0</td>
<td>75</td>
</tr>
<tr>
<td>2</td>
<td>18.2</td>
<td>148.0</td>
<td>3015</td>
<td>32.3</td>
<td>130</td>
</tr>
<tr>
<td>3</td>
<td>19.0</td>
<td>175.8</td>
<td>1826</td>
<td>11.2</td>
<td>60</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>9555</td>
<td>17.5</td>
<td>199.1</td>
<td>1917</td>
<td>4.0</td>
<td>143</td>
</tr>
<tr>
<td>9556</td>
<td>18.0</td>
<td>111.0</td>
<td>2001</td>
<td>17.1</td>
<td>95</td>
</tr>
<tr>
<td>9557</td>
<td>19.6</td>
<td>186.6</td>
<td>4222</td>
<td>29.9</td>
<td>152</td>
</tr>
<tr>
<td>9558</td>
<td>15.7</td>
<td>103.2</td>
<td>3832</td>
<td>41.1</td>
<td>161</td>
</tr>
</tbody>
</table>

### Table 2: The decision table derived from Table 1

<table>
<thead>
<tr>
<th>CLASS</th>
<th>S</th>
<th>H</th>
<th>E</th>
<th>C</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>E₁</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>E₂</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>E₃</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>E₄</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>E₅</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>E₆</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>E₇</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>E₈</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

The theory of rough sets provides a collection of methods which can be applied to the analysis and reduction of such tables. The methods are in particular applicable to data mining or knowledge discovery problems but also have other applications such as trainable control and pattern recognition[30,33-44].

For the purpose of illustration, we will use the decision table given in Table 2, which was derived from data collection partially shown in Table 1 by replacing the original attribute values with some discrete ranges. In this table, the column headings S,H,E,C,T are abbreviations for attribute names SIZE, HEIGHT, ENERGY, CURRENT and TEMPERATURE, respectively. The attributes are generalized properties of objects, identified by sequence numbers in the column OBJ of Table 1. The classes of objects with the same combinations of attribute values are identified in Table 2 in the column CLASS.

## 2 Discovering Attribute Dependencies

An important aspect of KDD is the discovery, analysis, and characterization of dependencies among attributes. The problem of discovery of attribute dependencies was studied independently by many researchers[2,4,12]. In this article, we specifically focus on discovering functional and partial functional dependencies using rough set theory. The probabilistic extension of this theory, called variable precision rough set model, allows for discovery of probabilistic dependencies among attributes [39,40] using essentially the same techniques as presented in the rest of this section.

In the rough set approach to discovery of dependencies, the subject of the analysis are decision tables derived from the original data rather than the data itself. As demonstrated in the example shown in the Introduction, the decision table is obtained by performing a suitably selected multidimensional mapping converting the original data into finite-valued secondary attribute-value vectors . The definition of the mapping is domain-dependent and often requires significant domain expertise for proper definition. Typically, the values of the secondary attributes correspond to some generalized qualitative categories, such LOW, MEDIUM, HIGH etc. The key issue is to have sufficient amount of representative data points so that classification of the universe of discourse OBJ (i.e. domain of interest containing objects represented by data points accumulated in the database) is complete. The complete classification covers all feasible combinations of values of the generalized secondary attributes. For example, we will assume that the decision table in Table 2 exhausts all feasible combinations of discretized attributes S,H,E,C,T. In the absence of the complete classification of the
domain, any conclusions derived from the decision table should be constrained to the accumulated data only. They can be false in the whole domain from which the data were collected. In particular, the dependencies discovered in the decision table will hold in the data on hand, but may not hold in the whole domain, if the decision table is incomplete.

In the search for dependencies, we are interested not only in functional dependencies but also in a spectra of partial functional dependencies. In more detail, in the context of the data-derived decision table, we ask the question as to whether there is any functional or partial functional dependency between groups of attributes \( P \subseteq A \) and \( Q \subseteq A \), where \( A \) is a set of all attributes of the decision table.

To precisely define the notions of partial functional dependency, deterministic and non-deterministic decision rules, we need to use some elementary ideas of rough sets theory, as presented below\cite{1}.

Let \( R(P) \) be the equivalence relation among objects in the universe \( OBJ \). The pair \((OBJ, R(P))\) will be called an approximation space\cite{1}. Also, let \( R^*(P) \) be the collection of equivalence classes (also called elementary sets) of \( R(P) \). That is, elements of \( R^*(P) \) are groups of objects having the same values of attributes belonging to \( P \). The elementary sets reflect our basic knowledge about the domain, the knowledge in the sense of knowing which categories of objects occur in the domain, and being able to classify each object into one of the categories. With this kind of knowledge it is not possible, in general, to construct discriminating description of any arbitrary subset of the domain. What it means in practice is that some concepts can never be learned precisely with the available information. Instead, rough approximations of these concepts can be learned. These approximations are defined below. They are fundamental notions of rough set theory which are necessary to evaluate the quality of information describing objects of the domain of interest, relative to the target set of objects \( Y \). The quality of the information is expressed in these definitions in terms of the ability to form tight lower and upper approximations of the target set \( Y \).

The lower approximation in the approximation space \((OBJ, R(P))\), or alternatively the interior \( INT(Y) \) of an arbitrary subset \( Y \subseteq OBJ \), is defined as the union of those equivalence classes of \( R^*(P) \) which are completely contained by \( Y \), i.e.,

\[
INT(Y) = \bigcup \{ E \in R^*(P) : E \subseteq Y \}
\]

The lower approximation characterizes objects which can be classified into \( Y \) without any uncertainty, based on the available information. In other words, the lower approximation is a largest set of objects which has discriminating description and is contained in the target set \( Y \). The upper approximation defines objects which possibly belong to the target set. It is the smallest set of objects having discriminating description and containing the target set. The upper approximation of \( Y \), denoted here as \( UPP(Y) \), is a union of those elementary classes which have some overlap with \( Y \), i.e.,

\[
UPP(Y) = \bigcup \{ E \in R^*(P) : E \cap Y \neq \emptyset \}.
\]

Because the set of the attributes \( Q \) corresponds to the partitioning \( R^*(Q) \) of the universe \( OBJ \), the degree \( K(P, Q) \) of the deterministic, or functional dependency in the relationship between attribute collections \( P \) and \( Q \) can be defined as the total relative size of lower approximations of classes of the partition \( R^*(Q) \) in the approximation space \((OBJ, R(P))\).

That is, if \( R^*(Q) = \{Y_1, Y_2, ..., Y_m\} \) then

\[
K(P, Q) = \frac{\sum_{i=1}^{m} \text{card}(INT(Y_i))}{\text{card}(OBJ)}
\]

where \( \text{card} \) is a set cardinality. \( K(P, Q) \) assumes values in the range \([0,1]\) with \( K(P, Q) = 1 \) for functional dependency and \( K(P, Q) = 0 \) when no value of \( Q \) attributes can be uniquely determined by values of \( P \) attributes. \( K(P, Q) \) can be interpreted as a proportion of such objects in the domain for which it suffices to know the values of attributes in \( P \) to determine the values of attributes in \( Q \). In the practice, we are most often interested in analyzing dependencies where \( Q \) contains only a single attribute.

For example, it is easy to verify, based on the Table 2, that for \( P = \{\text{SIZE, HEIGHT, ENERGY, CURRENT}\} \) and \( Q = \{\text{TEMPERATURE}\} \) the degree of dependency of \( K(P, Q) \) is 1. This means that this dependency is functional whereas the dependency between \( P = \{\text{ENERGY, CURRENT}\} \)
and $Q = \{TEMPERATURE\}$ is only partially functional with $K(P,Q) = 0.5$. The methodology for the dependency computation can be summarized in the following algorithm. In the algorithm, it is assumed that there is only one finite-valued target attribute. In the case of several target attributes, any subset of them can be treated as a compound single attribute and processed in the same way.

**Algorithm DEP: Attribute Dependency Computation**

1. Construct decision table from the original data according to pre-defined mapping of data records into secondary attribute vectors.
2. For each value $V_i$ of the target attribute identify the set of objects (represented by data records) $Y_i$ containing that value.
3. Compute the cardinality of the lower approximation of each set $Y_i$.
4. Output the degree of dependency as the ratio of the total size of all lower approximations of the sets $Y_i$ to the total number of objects (data records here).

In the rough set approach, any full or partial function reflecting dependency occurring in data is represented by the decision table. The decision table plays the same role as the analytical formula for numeric dependencies (functions). The tabular representation can be simplified, to eliminate any redundant attributes, and analyzed to determine the relative significance of the attributes involved in such a dependency. This step would be equivalent to finding a simpler formula representing a numeric function. The simplification of table dependencies is based on the concept of relative reduct of rough sets theory[1], as presented below. The relative reduct of the attribute collection $P$, with respect to the dependency $K(P,Q)$, is defined as a subset $RED(P,Q) \subseteq P$ such that:

1. $K(RED(P,Q),Q) = K(P,Q)$, i.e., relative reduct preserves the degree of inter-attribute dependency, and
2. For any attribute $a \in RED(P,Q)$, $K(RED(P,Q) – \{a\},Q) \neq K(P,Q)$, that is the relative reduct is a minimal subset with respect to the property 1.

A single relative reduct can be computed in linear time with the following attribute elimination procedure. The outcome of the process depends on the pre-determined priority ordering of attributes.

**Algorithm RRED: Computation of Relative Reduct**

1. Consider all attributes of the initial collection of attributes $P$, one by one, in the reverse order of priority. For each attribute $a$ do the following:
   1.1 Remove the attribute $a$ from $P$ temporarily.
   1.2 Check if the dependency $K(P,Q)$ changed. If yes, return $a$ back to $P$, else remove the attribute $a$ from $P$ permanently.
2. Output the set of attributes remaining in $P$ at the end of the process. It is a relative reduct of the initial set of attributes $P$.

For example, one possible relative reduct with respect to the dependency between $P = \{SIZE, HEIGHT, ENERGY, CURRENT\}$ and $Q = \{TEMPERATURE\}$ is $RED(P,Q) = \{HEIGHT, ENERGY\}$. This means that the discovered dependency can be characterized by fewer attributes, leading to possible savings in information representation, better understanding of the nature of the dependency and stronger patterns. In general, a number of alternative reducts can be computed for each analyzed dependency and the one of the lowest total cost can be selected to represent the discovered dependency. The dependency represented in the reduced form is usually more regular as it reflects stronger data patterns. This is illustrated in Table 3. For computing all attribute reducts, the decision matrix methodology, described in the next section, can be used. The details are provided in [29]. More research results and algorithms related to finding reducts can be also found in [14].
To find fundamental factors contributing to the discovered dependency the idea of relative core can be used. The relative core set of attributes with respect to the dependency \( K(P, Q) \) is a subset \( \text{CORE} \subseteq P \) such that for all \( a \in \text{CORE}, K(P - \{a\}, Q) \neq K(P, Q) \). In other words, \( \text{CORE} \) is a set of the essential attributes which cannot be eliminated from \( P \) without affecting the dependency between \( P \) and \( Q \).

For example, the core of the dependency between \( P = \{\text{SIZE, HEIGHT, ENERGY, CURRENT}\} \) and \( Q = \{\text{TEMPERATURE}\} \) is \( \{\text{HEIGHT}\} \). That is, \( \text{HEIGHT} \) is the fundamental dependency factor and it is included in every relative reduct representing this dependency[1]. The relative core is a context sensitive notion (depends on other attributes in the decision table) and can be empty. This can be interpreted as a case of highly over specified system with superfluous attributes. In a reduct, all attributes are core attributes (relative to the reduct attributes only) since the elimination of any attribute from the reduct results in a drop of the dependency. The percentage drop of the dependency can be used as a measure of relative significance of an attribute in the reduct: the higher the drop the more significant is the attribute with respect to the dependency \( K(\text{RED}(P, Q), Q) \). For instance, in the dependency between reduct attributes \( \text{RED}(P, Q) = \{\text{HEIGHT, ENERGY}\} \) and the attribute \( \{\text{TEMPERATURE}\} \), as represented in the Table 3, the relative significance of \( \text{HEIGHT} \) is 60 percent whereas the significance of \( \text{ENERGY} \) is 40 percent.

### Table 3: Reduced representation of the dependency between \( P \) and \( Q \)

<table>
<thead>
<tr>
<th>( \text{HEIGHT} )</th>
<th>( \text{ENERGY} )</th>
<th>( \text{TEMPERATURE} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

3 Rule Discovery

Discovering rules from data is one of the most important tasks of knowledge discovery in databases. Many systems and approaches for rule computation have been used for rule or decision tree discovery[3-6,9-11,13,24,26,28-29,30-32,40-44]. Rules can be perceived as data patterns which represent relationships between attribute values. In the rough sets approach, we distinguish certain, or deterministic rules, and possible, or non-deterministic rules. In addition to the above, in the variable precision rough set model, the probabilistic rules can be computed[8,15,18,27]. Since the rule computation methodology developed within the original rough set framework is directly applicable to computation of probabilistic rules by using probabilistic definitions of set approximations, in this section we will focus on rule discovery with the original rough set model only. In data mining context, the most useful are the maximally general rules, the ones which minimize the number of rule conditions. We will refer to such rules as minimal rules. They correspond to value reducts known in the rough set theory, or to prime implicants of the digital circuit design theory. The rough set-based algorithms for rule discovery can either compute rules forming minimal cover or approximation of minimal cover of the target class, or they can be used to find all minimal rules for the target class. Representative algorithms of both kinds are presented in the rest of this section.

To precisely define the concept of rules in the rough set approach some additional notational conventions are needed. Let \( V \) denote a selected value of the decision (target) attribute \( \text{d} \in A \) and let \( \text{supp}(\text{d} = V) \) denote the set of objects matching this value, referred to as target set of objects. The rules can be computed either with respect to lower approximation \( \text{INT}(\text{supp}(\text{d} = V)) \) or upper approximation \( \text{UPP}(\text{supp}(\text{d} = V)) \) of the target set. When the lower approximation is selected then deterministic rules are obtained. The non-deterministic rules are obtained from upper approximation of the target set. In either case, the computational procedure is the same. Therefore, to describe the methods we can assume, without loss of generality, that \( \text{supp}(\text{d} = V) \) is an exact set, or not rough, i. e., \( \text{supp}(\text{d} = V) = \text{INT}(\text{supp}(\text{d} = V)) = \text{UPP}(\text{supp}(\text{d} = V)) \), as otherwise \( \text{supp}(\text{d} = V) \) would be substituted by either \( \text{INT}(\text{supp}(\text{d} = V)) \) or \( \text{UPP}(\text{supp}(\text{d} = V)) \).

In our notation, the rule derived from data is a logical implication formula corresponding to the combination of values of some attributes such that the set of all objects matching this combination
is contained in the target set of objects. The rule \( r \) can be expressed as
\[
    r : (a_{i1} = V_{i1}) \land (a_{i2} = V_{i2}) \land \ldots \land (a_{in} = V_{in}) \rightarrow (d = V)
\]
where \( a_{ij}, V_{ij} \) denote condition attributes and attribute values, respectively. The set of attribute-value pairs occurring on the left hand side of the rule \( r \) is referred to as the rule condition part, denoted \( cond(r) \), and the right hand side is the decision part, \( dec(r) \), so the rule can be expressed as \( cond(r) \rightarrow dec(r) \). The set of all objects \( supp(cond(r)) \) in the universe \( OBJ \) whose attribute values match the rule conditions is called the rule support. To define minimal rules, the notion of value reduct is introduced first. The value reduct of the set of attribute-value pairs \( cond(r) \) is a subset \( red(r) \subseteq cond(r) \) such that

1. \( supp(red(r)) \subseteq supp(d = V) \) i.e., value reduct preserves the relation of inclusion of the rule support set in the target set, and
2. For any attribute-value pair \( (a_{ij} = V_{ij}) \in red(r) \), \( supp(red(r) - \{(a_{ij} = V_{ij})\}) \not\subseteq supp(d = V) \) that is, the value reduct is a minimal subset with respect to the property 1.

Similar to relative reduct, a single value reduct can be computed in linear time with the following value elimination procedure. The outcome of the process depends on the pre-determined priority ordering of attributes.

**Algorithm VRED: Computation of Value Reduct**

1. Consider all attribute-value pairs of the set \( cond(r) \), one by one, in the reverse order of priority. 
   For each pair \( x \) do the following:
   1.1 Remove the pair \( x \) from \( cond(r) \) temporarily.
   1.2 Check if the support set of \( cond(r) \), \( supp(cond(r)) \) is included in \( supp(d = V) \). If no, return \( x \) back to \( cond(r) \), else remove the pair \( x \) from \( cond(r) \) permanently.
2. Output the set of attribute-value pairs remaining in \( cond(r) \) at the end of the process. It is a value reduct of the original contents of \( cond(r) \).

Every value reduct corresponds to a minimal rule \( red(r) \rightarrow dec(r) \) derived from the initial rule \( cond(r) \rightarrow dec(r) \). For example, some of the value reducts of the set of attribute-value pairs of the rule corresponding to the first row of the Table 2, with respect to target value \( T = 0 \), are \{\( (H = 0) \}, \{ (E = 1) \} \) and \{\( (S = 0) \}, \{ (C = 0) \} \). These value reducts translate into rules such as, for example the following:

\[
    \begin{align*}
    \text{(HEIGHT} = 0) \land \text{(ENERGY} = 1) & \rightarrow \text{(TEMPERATURE} = \text{'0'}) \\
    \text{(SIZE} = 0) \land \text{(CURRENT} = 0) & \rightarrow \text{(TEMPERATURE} = \text{'0'})
    \end{align*}
\]

As it follows from the above, a single minimal rule can be computed in linear time in the number of attributes. Some minimal rules, selected according to a predefined heuristic and covering the target set, or rough approximation of it, can be computed relatively easily in time proportional to number of attributes and number of elementary sets in lower(or upper) approximation of the target class. Algorithms for computing such rules have been implemented in several systems for machine learning and data mining [15,25,29,30,34,41-44]. A representative simplified algorithm is presented below. It is aimed at generating close to minimum covering of the lower (or upper) approximation of the target class with the support sets of selected value reducts.

**Algorithm MinRul: Computation of Minimal Rules**

1. Compute lower (or upper) approximation of the target class. Initialize the list \( L \) of value reducts to empty.
In the decision table, consider one by one every row corresponding to the selected lower (or upper) approximation of the target set. Compute a single value reduct from the current row. If the support set of the computed value reduct is contained in the union of support sets of the value reducts in the list \( L \), go to the next row; otherwise add the computed value reduct to the list \( L \).

Output the list \( L \). At the end of the process, the list \( L \) will contain non-redundant set of value reducts whose support sets will cover the selected approximation of the target set. These reducts will directly correspond to the set of rules discriminating either the lower, or the upper approximation of the target class.

For illustration, let’s compute minimal rules for \( T = 1 \) from the decision table shown in Table 2. The lower approximation of \( supp(T = 1) \) is the union of elementary classes \( E_2, E_4, E_5, E_7, E_8 \). Consequently, we will compute value reducts for rows 2,4,5,7,8 of the decision table using VRED algorithm (the assumed order of value elimination is from left to right). It can be easily verified that the value reduct of rows 2,4,7 and 8 is \( \{(C = 1)\} \). The value reduct of row 5 is \( \{(H = 2)\} \). Since the support set of \( \{(H = 2)\} \) is not covered by the support set of \( \{(C = 1)\} \) the result consist of the value reducts \( \{(C = 1)\} \) and \( \{(H = 2)\} \), and corresponding to them rules:

\[
\text{(CURRENT } = 1\text{) } \rightarrow \text{(TEMPERATURE } = '1'\text{)}
\]
\[
\text{(HEIGHT } = 2\text{) } \rightarrow \text{(TEMPERATURE } = '1'\text{)}
\]

For knowledge discovery applications, finding all possible minimal rules rather than minimal covering rules is more desirable. However, the problem of computing all minimal rules is NP-hard. All minimal rules can be computed, if the problem size permits, using the techniques of discernibility matrix [10], or decision matrix [9] which is described below.

Before we define the concept of a decision matrix, we will assume some notational conventions. That is, we will assume that all classes containing objects belonging to \( supp(d = V) \) and all classes with objects belonging to the complement of \( supp(d = V) \) are separately numbered with subscripts \( i \) \( (i = 1, 2, ... \gamma) \) and \( j \) \( (j = 1, 2, ... \rho) \) respectively. The modified example decision table with additional indexes is shown in Table 4.

A decision matrix \( M(S) = (M_{ij}) \) of a decision table with respect to value \( V \) of the decision attribute \( d \) is defined as a matrix whose entries are sets of attribute value pairs

\[
M_{ij} = \{(a, a(E_i)) : a(E_i) \neq a(E_j)\}
\]

where \( a(E_i) \) denotes the common value of the attribute \( a \) on all objects belonging to the class \( E_i \). The set \( M_{ij} \) contains all attribute-value pairs \( (attribute, value) \) whose values are not identical between \( E_i \) and \( E_j \). In other words, \( M_{ij} \) represents the complete information distinguishing \( E_i \) from \( E_j \). For example, the entry \( M_{11} \) of the decision matrix given in Table 5 reflects the fact that class \( E_1 \) differs from the class \( E_2 \) in values of attributes \( SIZE, ENERGY, \) and \( CURRENT \). These values are \( SIZE = 1, ENERGY = 2 \) and \( CURRENT = 1 \) for the class \( E_2 \).

The set of all minimal value reducts of the collection of attribute value pairs corresponding to row \( i \) of the decision table, and consequently the set of all minimal decision rules for that row can be obtained by forming the Boolean expression

\[
(CURRENT = 1) \rightarrow (TEMPERATURE = '1')
\]
\[
(HEIGHT = 2) \rightarrow (TEMPERATURE = '1')
\]
\[ B_i = \bigwedge_j \bigvee M_{ij} \]

where \( \wedge \) and \( \vee \) are respectively generalized conjunction and disjunction operators\[9,10\].

The Boolean expression, called a decision function \( B_i \), is constructed out of row \( i \) of the decision matrix, that is \((M_{i1}, M_{i2}, ..., M_{iρ})\), by formally treating each attribute-value pair occurring in the matrix entry \( M_{ij} \) as a Boolean variable and then forming a Boolean conjunction of disjunctions of attribute-value pairs belonging to each set \( M_{ij} \).

The decision rules are obtained by turning each decision function into disjunctive normal form and using the absorption law of Boolean algebra to simplify it. The conjuncts, or prime implicants of the simplified decision function correspond to the minimal decision rules [9,10].

The major steps in computing all minimal rules for the selected approximation (lower or upper) of the target class are summarized in the following algorithm.

**Algorithm AllRul: Computation of All Minimal Rules**

1. Split the rows of the input decision table into two categories: lower (or upper) approximation rows of the target set versus all other rows.
2. Construct the decision matrix by comparing decision table rows belonging to the categories obtained in Step 1.
3. Form decision functions out of rows of the decision matrix and turn them into disjunctive normal form.
4. Identify prime implicants of the simplified decision functions obtained in Step 3.
5. Perform the set union operation on the sets of prime implicants associated with the rows of the decision matrix.
6. Output minimal rules for lower (or upper) approximation of the target class by translating all prime implicants obtained in Step 5 into rule format. Each prime implicant is translated into rule conditions with the rule conclusion corresponding to the selected value \( V \) of the decision attribute.

For example, for the target class corresponding to \( TEMPERATURE = 1 \) of the decision table shown in Table 4, we can compute and simplify the following decision functions, as derived from the decision matrix presented in Table 5.

\[
B_1 = ((S, 1) \vee (E, 2) \vee (C, 1)) \wedge ((E, 2) \vee (C, 1)) = (E, 2) \vee (C, 1)
\]
\[
B_2 = ((H, 2) \vee (C, 1)) \wedge ((S, 0) \vee (H, 2) \vee (C, 1)) = (H, 2) \vee (C, 1)
\]
\[
B_3 = ((S, 1) \vee (H, 2)) \wedge ((H, 2)) = (H, 2)
\]
\[
B_4 = ((S, 1) \vee (H, 2) \vee (E, 2) \vee (C, 1))
\]

Table 5: Decision matrix for \( TEMPERATURE = 1 \) of Table 3.

<table>
<thead>
<tr>
<th>i</th>
<th>CLASS</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>E₂</td>
<td>(S,1)(E,2)</td>
<td>(H,0)(E,2)</td>
<td>(E,2)(C,1)</td>
</tr>
<tr>
<td>2</td>
<td>E₄</td>
<td>(H,2)(C,1)</td>
<td>(S,0)(H,2)</td>
<td>(S,0)(H,2)</td>
</tr>
<tr>
<td>3</td>
<td>E₅</td>
<td>(S,1)(H,2)</td>
<td>(H,2)</td>
<td>(H,2)</td>
</tr>
<tr>
<td>4</td>
<td>E₇</td>
<td>(S,1)(H,2)</td>
<td>(E,2)(C,1)</td>
<td>(E,2)(C,1)</td>
</tr>
<tr>
<td>5</td>
<td>E₈</td>
<td>(E,2)(C,1)</td>
<td>(S,0)(H,0)</td>
<td>(S,0)(E,2)</td>
</tr>
</tbody>
</table>
The prime implicants are $(E, 2)$, $(C, 1)$ and $(H, 2)$. After translation, the following rules are obtained:

- $(\text{ENERGY} = 2) \rightarrow (\text{TEMPERATURE} = '1')$
- $(\text{CURRENT} = 1) \rightarrow (\text{TEMPERATURE} = '1')$
- $(\text{HEIGHT} = 2) \rightarrow (\text{TEMPERATURE} = '1')$

All the minimal rules for the decision class “0” can be computed in a similar way.

The level of difficulty with applying the decision matrix method in practical situations depends on the complexity of the decision table obtained through classification of the original data based on values of generalized secondary attributes. Typically, the attributes used to construct a decision table are the functions of the original data values, their number is small and each attribute has a small number of domain values. This leads to relatively small and constrained number of classes of the decision table. The size of the decision table is constrained by the product of cardinalities of attribute domains and is independent of the size of the original data set. Consequently, careful definition of the decision table attributes is an essential prerequisite before applying the decision matrix method and, in fact, any other rough set-based rule discovery method. On the other hand, the computation of prime implicants associated with decision functions, the most time consuming aspect of the algorithm AllRul, can be done independently, for each row of the decision matrix. This means that the decision matrix method can take full advantage of a parallel implementation in a multiprocessor system to speed up the rule computation process.

4 Final Remarks

The theory of rough sets is a fundamental mathematical model which has been extensively studied by mathematicians, logicians and computer scientists. The simplicity and mathematical clarity of the model makes it attractive for both theoreticians and application-oriented researchers. Although its introduction was not motivated by any particular application, it touched the essence of many old problems in AI, pattern recognition and control. More recently, it became clear that rough set theory provides a framework for studying data mining and for developing data analysis algorithms for this research area. However, it should be emphasized that, while providing sound theoretical basis for designing new algorithms, the rough set theory is not associated with any particular method or algorithm for data mining or other applications. The choice of rough set-based techniques for data mining presented in this article reflects only author’s preferences and experiences and by no means pretends to cover the whole subject area.

5 Acknowledgments

The research reported in this article was partially supported by a research grant awarded to the author by Natural Sciences and Engineering Research Council of Canada. Many thanks to Jan Zytkow and the anonymous referees for precise and inspiring comments.

References


