Fundamenta Informaticae 100 (2010) 1–18
DOI 10.3233/FI-2010-272
IOS Press

A Local Version of the MLEM2 Algorithm for Rule Induction

Jerzy W. Grzymala-Busse*
Department of Electrical Engineering and Computer Science, University of Kansas
Lawrence, KS 66045, USA
and
Institute of Computer Science, Polish Academy of Sciences, 01–237 Warsaw, Poland
jerzy@eecs.ukans.edu

Wojciech Rzasa
Department of Computer Science, University of Rzeszow, 35–310 Rzeszow, Poland
wrzasa@univ.rzeszow.pl

Abstract. In this paper, we present the newest version of the MLEM2 algorithm for rule induction, a basic component of the LERS data mining system. This version of the MLEM2 algorithm is based on local lower and upper approximations, and in its current form is presented in this paper for the first time. Additionally, we present results of experiments comparing the local version of the MLEM2 algorithm for rule induction with an older version of MLEM2, which was based on global lower and upper approximations. Our experiments show that the local version of MLEM2 is significantly better than the global version of MLEM2 (2% significance level, two-tailed Wilcoxon test).

1. Introduction

In this paper we present the newest version of the MLEM2 (Modified Learning from Example Module, version 2) algorithm for rule induction, a basic component of the LERS (Learning from Examples based on Rough Sets) data mining system. The LERS data mining system has been developed at the University of Kansas. Its first component, called LEM1 (Learning from Example Module, version 1) was implemented for the first time in Franz Lisp in 1988 [6]. In 1990, the basic algorithm of LERS, called LEM2,

*Address for correspondence: Department of Electrical Engineering and Computer Science, University of Kansas, Lawrence, KS 66045, USA

was added to LERS [3, 9]. The main difference between LEM1 and LEM2 is the type of coverings: LEM1 uses global coverings while LEM2 uses local coverings.

Rule sets, induced by LERS or other data mining systems, are usually used for classification of new, unseen cases that were not used for rule induction. The first classification system was added to LERS in 1994 [10]. LERS deals with inconsistent data (i.e., data with conflicting cases; for two such cases all attribute values are the same yet the decision values are different) using lower and upper approximations. Thus, before running LEM1 or LEM2, such approximations must be computed first.

The LERS system is equipped with a number of discretization algorithms to handle numerical attributes. Both LEM1 and LEM2 algorithms need discretization as a preliminary process for numerical attributes. These discretization techniques were described, e.g., in [4, 5, 24].

In the nineties LERS used typical, traditional approaches to missing attribute values, before rule induction, i.e., as preprocessing. Such methods include deleting cases with missing attribute values from the data set, replacing a missing attribute value by all possible values from the attribute domain, replacing a missing attribute value by the most frequent value from the attribute domain for symbolic attributes and by the mean of all values from the attribute domain for numerical attributes [16].

In 1997 a new approach to missing attribute values, based on inducing rules only from known data, was introduced [20]. This interpretation of missing attribute values is known as *lost value* [13].

The algorithm called MLEM2 was introduced in 2003 [12]. MLEM2 induced rule sets directly from raw data, i.e., data not only with numerical attributes but also with missing attribute values. However, this algorithm needed a preprocessing: computing lower and upper approximations. Note that even though MLEM2, like LEM2, uses local coverings, yet is based on global approximations.

A new acquisition to LERS was the program IRIM (Interesting Rule Induction Module), able to induce all rules with a defined number of conditions and with a defined strength (number of training, correctly classified cases by the rule) [15].

Note that the LEM2 algorithm was successfully implemented and used in many places, see, e.g., [1, 2, 19], under many names such as ELEM2, MODLEM, etc.

In 2006 local approximations were combined with MLEM2 [8]. Local approximations are defined using blocks of attribute-value pairs. The same idea of an attribute-value block is used in both LEM2 and MLEM2, so it was possible to combine both ideas, local approximations and MLEM2 and modify MLEM2 again. This time MLEM2 does not need any preprocessing, since computing of local lower and upper approximations as well as handling numerical attributes and missing attribute values are done within the same algorithm.

In this paper, we present a slightly modified algorithms for determining local lower and upper coverings from those presented in [18]. Additionally, we present results of experiments on the local version and global versions of MLEM2. In the global version of MLEM2, global lower and upper approximations are computed during a preliminary step, as preprocessing. For each data set, we selected the best results for local MLEM2 and global MLEM2 and then compared the overall performance using a nonparametric test, the Wilcoxon matched-pairs signed-rank test.

2. Blocks of Attribute-Value Pairs

An example of a data set is presented in Table 1. Rows of the table represent *cases*, while columns are labeled by two types of *variables* called *attributes* and a *decision*. The set of all cases will be denoted

Table 1. An incomplete data set

	Attributes			Decision
Case	Temperature	Headache	Cough	Flu
1	39.8	yes	yes	yes
2	?	yes	yes	yes
3	40.8	yes	?	yes
4	?	no	no	yes
5	39.8	no	no	no
6	36.8	yes	no	no
7	38.4	no	yes	no

by U. In Table 1, U = {1, 2,..., 7}. The set of all attributes will be denoted by A. In Table 1, A = {*Temperature*, *Headache*, *Cough*}. The decision, denoted by d, is *Flu*. The fact that for a case x an attribute a has the value v will be denoted by $a(x) = v$. Similarly, if for a case x the value of d is w, we will denote it by $d(x) = w$. A table with some missing attribute values will be called *incomplete* or a table with missing attribute values.

In general, missing attribute values are *lost values* (the values that were recorded but currently are unavailable, denoted by "?"), *do not care conditions* (the original values were irrelevant, denoted by "*"), and *attribute-concept values* (these missing attribute values may be replaced by any attribute value limited to the same concept, denoted by "$-$"), see, e.g., [14].

For the rest of the paper we will assume that all decision values are specified, i.e., they are not missing. Additionally, we will assume that for each case at least one attribute value is specified.

An important tool to analyze decision tables is a *block of an attribute-value pair*. Let (a, v) be an attribute-value pair. For *complete* decision tables, i.e., decision tables in which every attribute value is specified, a block of (a, v), denoted by $[(a, v)]$, is the set of all cases x for which $a(x) = v$.

3. Numerical Attributes

The attribute *Temperature* from Table 1 is numerical. For data mining, numerical attributes must be converted into symbolic ones, or in different words, numerical values should be converted into intervals. For a numerical attribute, the first step is to sort numerical attribute values. For *Temperature* the list of sorted values is: 36.8, 38.4, 39.8 and 40.8. The next step is to select cutpoints. In MLEM2, the potential cutpoints are averages of consecutive values of the sorted list of all attribute values. In our example, such potential cutpoints are 37.6, 39.1, and 40.3. Thus, the potential intervals are, e.g., 36.8..37.6 and 37.6..40.3. In the current, local MLEM2 algorithm, there are two options of selecting potential cutpoints: *all cutpoints* and *selected cutpoints*.

3.1. All cutpoints option of MLEM2

If we use the option *all cutpoints*, for every potential cutpoint the MLEM2 algorithm creates two *primary intervals*, the first containing all numerical values smaller than the cutpoint and the second containing all numerical values greater than the cutpoint. Thus, for Table 1, the list of all primary intervals is 36.8..37.6, 36.8..39.1, 36.8..40.3, 37.6..40.8, 39.1..40.8, 40.3..40.8. The first interval, 36.8..37.6 contains just one value: 36.8, the second interval, 36.8..40.8 contains values 38.4, 39.8, and 40.8. In different words, the first interval is *represented* by the set $\{36.8\}$, and the second interval is represented by the set $\{38.4, 39.8, 40.8\}$. The all cutpoints option is the only option of the MLEM2 global version and one of two options of the MLEM2 local version. The other option of the MLEM2 local version is *selected cutpoints*.

3.2. Selected cutpoints option of MLEM2

In the all cutpoints option of MLEM2, the decision is not taken into account, while in the selected cutpoints option of MLEM2 the algorithm chooses only some selected cutpoints on the basis of the corresponding decision values. In general, if for all occurrences of the two consecutive values of the sorted list of values of a numerical attribute the decision value is the same, the corresponding cutpoint is ignored in creating primary intervals. In Table 1, there are unique values of 36.8 and 38.4, for both the decision value is the same (*no*), so the potential cutpoint 37.6 is ignored. The value 39.8 occurs twice, for cases 1 and 5, with decision values *yes* and *no*, so we cannot ignore the cutpoint 39.1, the mean for 38.4 and 39.8. By the same token, we cannot ignore the cutpoint 40.3 . Thus, the selected cutpoints are 39.1 and 40.3, and the primary intervals are 36.8..39.6, 36.8..40.3, 39.6..40.8 and 40.3..40.8. We are following here the principle *the cutpoint will always occur on the boundary between two classes* [7], though this principle is valid only for cutpoints selected using entropy minimization. Moreover, as we will see later, this principle - in general - will not always produce better results.

Rules induced by different discretization options of MLEM2 differ from each other. For example, using the all cutpoints option of MLEM2, the following rule set is induced from the *bankruptcy* data set:

1, 30, 30
(a2, $-308.9..-3.55$) –> (Prediction, bankruptcy),
2, 23, 23
(a3, $-280.0..-1.4$) & (a1, $-185.1..24.2$) –> (Prediction, bankruptcy),
3, 33, 33
(a2, $-3.55..68.6$) & (a4, 44.8..771.7) & (a3, $-15.1..34.1$) –> (Prediction, survival),

while the selected cutpoints option induces, for the same data set, the following rule set:

2, 28, 28
(a2, $-308.9..7.85$) & (a4, 0.7..91.05) –> (Prediction, bankruptcy),
2, 30, 30
(a3, $-280.0..2.8$) & (a2, $-308.9..21.15$) –> (Prediction, bankruptcy),
3, 33, 33
(a2, $-3.55..68.6$) & (a4, 44.8..771.7) & (a3, $-15.1..34.1$) –> (Prediction, survival),

where a1 = Working_capital/Total_assets,
a2 = Retained_earnings/Total_assets,
a3 = Earnings_before_interest_and_taxes/Total_assets, and
a4 = Market_value_equity/Book_value_of_total_debt.

Every rule is presented in the LERS format. Such rules are preceded by three numbers: the total number of attribute-value pairs on the left-hand side of the rule, the total number of cases correctly classified by the rule during training, and the total number of training cases matching the left-hand side of the rule.

3.3. Operations on Intervals

In the MLEM2 algorithms, both versions, global and local, some operations are performed on intervals. The MLEM2 algorithm may select intervals associated with the same attribute as conditions of a rule. Such intervals are eventually *merged*. For example, let us say that MLEM2 selected two intervals of the same attribute *Temperature*, the first one is 36.8..40.3 and the second is 39.1..40.8. These two intervals will be merged into 39.1..40.3. The first interval is represented by the set {36.8, 38.4, 39.8}, the second interval is represented by {39.8, 40.8}, the merged interval is represented by the intersection of sets represented by both intervals, i.e., by the set {39.8}. The interval 39.1..40.3 will be called a *common part* of 36.8..40.3 and 39.1..40.8.

Two intervals with the common part equal to the empty set are called *disjoint*. For example, intervals 36.8..39.1 and 39.1..40.3 are disjoint since the first interval is represented by {36.8, 38.4} and the second interval is represented by {39.8, 40.8}.

We say that an interval *includes* another interval if it is represented by a set that is a superset of the other interval. For example, 39.1..40.8 includes 40.3..40.8 since the first interval is represented by the set {39.8, 40.8} and the second interval is represented by the set {40.8}. We will denote it by $39.1..40.8 \supseteq 40.3..40.8$.

4. Characteristic Relation

For incomplete decision tables the definition of a block of an attribute-value pair must be modified.

- If for an attribute a there exists a case x such that $a(x) = ?$, i.e., the corresponding value is lost, then the case x should not be included in any blocks $[(a, v)]$ for all values v of attribute a,
- If for an attribute a there exists a case x such that the corresponding value is a "do not care" condition, i.e., $a(x) = *$, then the case x should be included in blocks $[(a, v)]$ for all specified values v of attribute a,
- If for an attribute a there exists a case x such that the corresponding value is an attribute-concept value, i.e., $a(x) = -$, then the corresponding case x should be included in blocks $[(a, v)]$ for all specified values $v \in V(x, a)$ of attribute a, where

$$V(x, a) = \{a(y) \mid a(y) \text{ is specified}, \quad y \in U, \ d(y) = d(x)\}.$$

In this section we are concerned with cases that can be distinguished using attributes. Therefore, in computing blocks for numerical attributes, we will use specific numerical values instead of intervals.

For Table 1

[(Temperature, 36.8)] = {6},
[(Temperature, 38.4)] = {7},

[(Temperature, 39.8)] = $\{1, 5\}$,
[(Temperature, 40.8)] = $\{3\}$,
[(Headache, yes)] = $\{1, 2, 3, 6\}$,
[(Headache, no)] = $\{4, 5, 7\}$,
[(Cough, yes)] = $\{1, 2, 7\}$.
[(Cough, no)] = $\{4, 5, 6\}$.

For a case $x \in U$ the *characteristic set* $K_B(x)$ is defined as the intersection of the sets $K(x, a)$, for all $a \in B$, where the set $K(x, a)$ is defined in the following way:

- If $a(x)$ is specified, then $K(x, a)$ is the block $[(a, a(x)]$ of attribute a and its value $a(x)$,
- If $a(x) = ?$ or $a(x) = *$ then the set $K(x, a) = U$,
- If $a(x) = -$, then the corresponding set $K(x, a)$ is equal to the union of all blocks of attribute-value pairs (a, v), where $v \in V(x, a)$ if $V(x, a)$ is nonempty. If $V(x, a)$ is empty, $K(x, a) = U$.

For Table 1 and $B = A$,

$K_A(1) = \{1, 5\} \cap \{1, 2, 3, 6\} \cap \{1, 2, 7\} = \{1\}$,
$K_A(2) = U \cap \{1, 2, 3, 6\} \cap \{1, 2, 7\} = \{1, 2\}$,
$K_A(3) = \{3\} \cap \{1, 2, 3, 6\} \cap U = \{3\}$,
$K_A(4) = U \cap \{4, 5, 7\} \cap \{4, 5, 6\} = \{4, 5\}$,
$K_A(5) = \{1, 5\} \cap \{4, 5, 7\} \cap \{4, 5, 6\} = \{5\}$,
$K_A(6) = \{6\} \cap \{1, 2, 3, 6\} \cap \{4, 5, 6\} = \{6\}$, and
$K_A(7) = \{7\} \cap \{4, 5, 7\} \cap \{1, 2, 7\} = \{7\}$.

Characteristic set $K_B(x)$ may be interpreted as the smallest set of cases that are indistinguishable from x using all attributes from B, using a given interpretation of missing attribute values.

The characteristic relation $R(B)$ is a relation on U defined for $x, y \in U$ as follows

$$(x, y) \in R(B) \; if \; and \; only \; if \; y \in K_B(x).$$

The characteristic relation $R(B)$ is reflexive but—in general—does not need to be symmetric or transitive. Also, the characteristic relation $R(B)$ is known if we know characteristic sets $K(x)$ for all $x \in U$. In our example, $R(A)$ = {(1, 1), (2, 1), (2, 2), (3, 3), (4, 4), (4, 5), (5, 5), (6, 6), (7, 7)}.

For decision tables, in which all missing attribute values are lost, a special case of a characteristic relation was defined by J. Stefanowski and A. Tsoukias in [26, 27]. For any decision table in which all missing attribute values are lost, the characteristic relation is reflexive and transitive, but—in general—does not need to be symmetric . For decision tables where all missing attribute values are ”do not care” conditions a special case of characteristic relation was defined by M. Kryszkiewicz in [21], see also, e.g., [22]. Such a relation is reflexive and symmetric but—in general—not transitive.

5. Global Approximations

For completely specified decision tables lower and upper approximations are defined on the basis of the indiscernibility relation introduced by Z. Pawlak [23]. Any finite union of characteristic sets, that are called here *elementary* sets, associated with B, will be called a *B-definable set*. The empty set is

definable. Let X be any subset of the set U of all cases. The set X is called a *concept* and is usually defined as the set of all cases defined by a specific value of the decision. In general, X is not a B-definable set. However, set X may be approximated by two B-definable sets, the first one is called a *B-lower approximation* of X, denoted by $\underline{B}X$ and defined as follows

$$\{x \in U | [x]_B \subseteq X\}.$$

The second set is called a *B-upper approximation* of X, denoted by $\overline{B}X$ and defined as follows

$$\{x \in U | [x]_B \cap X \neq \emptyset\},$$

where $[x]_B = K_B(x)$. The above shown way of computing lower and upper approximations, by constructing these approximations from singletons x, will be called the *first method*. The B-lower approximation of X is the greatest B-definable set, contained in X. The B-upper approximation of X is the smallest B-definable set containing X.

As it was observed in [23], for complete decision tables we may use a *second method* to define the B-lower approximation of X, by the following formula

$$\cup\{[x]_B | x \in U, [x]_B \subseteq X\},$$

and the B-upper approximation of x may de defined, using the second method, by

$$\cup\{[x]_B | x \in U, [x]_B \cap X \neq \emptyset\}.$$

In this paper we quote three different definitions of lower and upper approximations. Again, let X be a concept, let B be a subset of the set A of all attributes, and let $R(B)$ be the characteristic relation of the incomplete decision table with characteristic sets $K(x)$, where $x \in U$. Our first definition uses a similar idea as in the previous articles on incompletely specified decision tables [21, 22, 25, 26, 27], i.e., lower and upper approximations are sets of singletons from the universe U satisfying some properties. Thus, lower and upper approximations are defined by analogy with the above first method, by constructing both sets from singletons. We will call these approximations *singleton*. A singleton B-lower approximation of X is defined as follows:

$$\underline{B}X = \{x \in U | K_B(x) \subseteq X\}.$$

A singleton B-upper approximation of X is

$$\overline{B}X = \{x \in U | K_B(x) \cap X \neq \emptyset\}.$$

In our example of the decision table presented in Table 1 let us say that $B = A$. Then the singleton A-lower and A-upper approximations of the two concepts: {1, 2, 3, 4} and {5, 6, 7} are:

$$\underline{A}\{1, 2, 3, 4\} = \{1, 2, 3\},$$
$$\underline{A}\{5, 6, 7\} = \{5, 6, 7\},$$
$$\overline{A}\{1, 2, 3, 4\} = \{1, 2, 3, 4\},$$
$$\overline{A}\{5, 6, 7\} = \{4, 5, 6, 7\}.$$

The second method of defining lower and upper approximations for complete decision tables uses another idea: lower and upper approximations are unions of elementary sets, subsets of U. Therefore we may define lower and upper approximations for incomplete decision tables by analogy with the second method, using characteristic sets instead of elementary sets. There are two ways to do this. Using the first way, a *subset* B-lower approximation of X is defined as follows:

$$\underline{B}X = \cup\{K_B(x)|x \in U, K_B(x) \subseteq X\}.$$

A *subset* B-upper approximation of X is

$$\overline{B}X = \cup\{K_B(x)|x \in U, K_B(x) \cap X \neq \emptyset\}.$$

For the same decision table, presented in Table 1, the subset A-lower and A-upper approximations are

$$\underline{A}\{1,2,3,4\} = \{1,2,3\},$$
$$\underline{A}\{5,6,7\} = \{5,6,7\},$$
$$\overline{A}\{1,2,3,4\} = \{1,2,3,4,5\},$$
$$\overline{A}\{5,6,7\} = \{4,5,6,7\}.$$

The second possibility is to modify the subset definition of lower and upper approximation by replacing the universe U from the subset definition by a concept X. A *concept* B-lower approximation of the concept X is defined as follows:

$$\underline{B}X = \cup\{K_B(x)|x \in X, K_B(x) \subseteq X\}.$$

Obviously, the subset B-lower approximation of X is the same set as the concept B-lower approximation of X. A concept B-upper approximation of the concept X is defined as follows:

$$\overline{B}X = \cup\{K_B(x)|x \in X, K_B(x) \cap X \neq \emptyset\} = \cup\{K_B(x)|x \in X\}.$$

For the decision table presented in Table 1, the concept A-lower and A-upper approximations are

$$\underline{A}\{1,2,3,4\} = \{1,2,3\},$$
$$\underline{A}\{5,6,7\} = \{5,6,7\},$$
$$\overline{A}\{1,2,3,4\} = \{1,2,3,4,5\},$$
$$\overline{A}\{5,6,7\} = \{5,6,7\}.$$

Note that for complete decision tables, all three definitions of lower approximations, singleton, subset and concept, are reduced to the same definition. Also, for complete decision tables, all three definitions of upper approximations are reduced to the same definition. This is not true for incomplete decision tables, as our example shows.

For incomplete data sets, a set X will be called *B-globally definable* if it is K_B-definable, i.e., if X is a union of members of the family K_B. A set that is A-globally definable will be called *globally definable*.

A set T of attribute-value pairs, where all attributes belong to set B and are distinct, will be called a *B-complex*. Any A-complex will be called—for simplicity—a *complex*. Obviously, any set containing a single attribute-value pair is a complex. For the rest of the paper we will discuss only *nontrivial complexes*, i.e., such complexes that the intersection of all attribute-value blocks from a given complex is not the empty set.

Set X *depends* on a complex T if and only if

$$\emptyset \neq [T] = \bigcap \{[t] \mid t \in T\} \subseteq X.$$

Set T is a *minimal complex* of X if and only if X depends on T and no proper subset T' of T exists such that X depends on T'.

Let $\mathcal{T}$ be a nonempty collection of nonempty sets of attribute-value pairs. Then $\mathcal{T}$ is a *local covering* of X if and only if the following conditions are satisfied:

(a) each member T of $\mathcal{T}$ is a minimal complex of X,

(b) $\cup\{[T] \mid T \in \mathcal{T}\} = X$, and

(c) $\mathcal{T}$ is minimal, i.e., $\mathcal{T}$ has the smallest possible number of members.

Both LEM2 and global version of MLEM2 are based on global approximations, however, they are using local coverings. The input set X for such algorithms is either a lower or upper global approximation of some concept. The original LEM2 rule induction algorithm was described in many papers, see, e.g., [11]. For description of MLEM2 see, e.g., [12].

For an incomplete decision table and a subset B of the set A of all attributes, a union of intersections of attribute-value pair blocks of attribute-value pairs from some B-complexes, will be called a *B-locally definable* set. A-*locally definable* sets will be called *locally definable*. Any set X that is B-globally definable is B-locally definable.

The singleton upper approximation of the concept {1, 2, 3, 4} is not A-locally definable since all blocks of attribute-value pairs containing case 4 contain case 5 as well.

The importance of the idea of local definability is a consequence of the following fact: A set is locally definable if and only if it can be expressed by rule sets. This is why it is so important to distinguish between locally definable sets and those that are not locally definable.

6. Local Approximations

Let X be any subset of the set U of all cases. In general, X is not a B-definable set, locally or globally. Let $B \subseteq A$. The B-*local lower* approximation of the concept X is defined as follows

$$\bigcup \{[T] \mid T \ is \ a \ complex \ of \ X, \ [T] \subseteq X\}.$$

The B-*local upper* approximation of the concept X is defined as the minimal set containing X and defined in the following way

$$\bigcup \{[T] \mid \exists \ a \ family \ \mathcal{T} \ of \ complexes \ T \ of \ X \ with \ \forall \ T \in \mathcal{T}, \ [T] \cap X \neq \emptyset\}.$$

Table 2. Data sets

Data set	Number of cases	attributes	concepts
Bankruptcy	66	5	2
Breast Slovenia	286	9	2
Breast Wisconsin	625	9	9
Bupa	345	6	2
Echocardiogram	74	7	2
Glass	214	9	6
Hepatitis	155	19	2
Horse	299	21	2
House	434	16	2
Iris	150	4	3
Lymphography	148	18	4
Primary Tumor	339	17	21
Segmentation	210	19	7
Wine	178	13	3

Obviously, the B-local lower approximation of X is unique and it is the largest B-locally definable set contained in X. Any B-local upper approximation of X is B-locally definable, it contains X, and is, by definition, the smallest. Note that a concept may have more than one local upper approximation [17].

For a set T of attribute-value pairs, the intersection of blocks for all t from T will be denoted by $[T]$. Let X be a nonempty subset of the universe U. For the rest of the paper we will assume that any set T consists of attribute-value pairs with all different attributes (thus, any set T may consists of at most $|A|$ attribute-value pairs). Let $\mathcal{T}$ be a family of sets T of attribute-value pairs.

A set $\mathcal{T}$ will be called a *local lower covering* of X if and only if the following three conditions are satisfied:

(1) $\bigcup_{T \in \mathcal{T}} [T] \subseteq X$,

(2) every $T \in \mathcal{T}$ is minimal, i.e., no proper subset T' of T exists with $[T'] \subseteq X$,

(3) $\mathcal{T}$ is minimal, i.e., for every $T \in \mathcal{T}$, $\bigcup_{S \in \mathcal{T} - \{T\}} [S] \neq \bigcup_{S \in \mathcal{T}} [S]$.

Note that the *local lower covering* should not be confused with the *local covering*. The former is computable for any subset X of U, e.g., a concept, the latter is computable only for a lower or upper global approximation of the concept. The procedure for determining a single local lower covering, based on the MLEM2 algorithm, is presented below.

Table 3. Numerical data sets

Data set	Local MLEM2 algorithm				Global MLEM2 algorithm	
	Certain rules		Possible rules		Certain rules	Possible rules
	All cutpoints	Selected cutpoints	All cutpoints	Selected cutpoints		
Bankruptcy	4.55%	6.06%	4.55%	6.06%	4.55%	4.55%
Echocardiogram	29.73%	27.03%	27.03%	27.03%	40.54%	40.54%
Glass	28.50%	32.71%	33.18%	29.44%	29.44%	29.44%
Hepatitis	18.71%	17.42%	20.65%	18.71%	17.42%	20.65%
Horse	33.78%	35.12%	39.80%	40.13%	35.45%	40.80%
Iris	4.67%	4.67%	4.67%	4.67%	4.67%	4.67%
Wine	10.67%	11.80%	10.67%	11.80%	11.24%	11.24%

Table 4. Symbolic data sets

Data set	Local MLEM2 algorithm		Global MLEM2 algorithm	
	Certain rules	Possible rules	Certain rules	Possible rules
Breast-Slov.	27.97%	29.02%	29.72%	29.72%
Breast-Wisc.	20.64%	21.12%	21.12%	20.96%
Bupa	35.65%	35.65%	34.78%	34.78%
Glass	32.24%	30.37%	30.84%	30.84%
Hepatitis	17.42%	15.48%	17.42%	17.42%
House	4.84%	7.14%	4.84%	6.45%
Lymphography	19.59%	15.54%	18.92%	18.92%
Primary Tumor	69.62%	61.36%	69.05%	61.36%
Segmentation	19.05%	16.67%	19.05%	19.05%
Wine	6.74%	7.87%	6.18%	6.18%

Procedure for determining a single local lower covering
input: a set X (a subset of U),
output: a single local lower covering $\mathcal{T}$ of the set X,
begin
$G := X$;
$\mathcal{T} := \emptyset$;
$\mathcal{J} := \emptyset$;
while $G \neq \emptyset$
begin
$T := \emptyset$;
$T_s := \emptyset$;
$T_n := \emptyset$;
$T(G) := \{t \mid [t] \cap G \neq \emptyset\}$;
while $(T = \emptyset$ **or** $[T] \not\subseteq X)$ **and** $T(G) \neq \emptyset$
begin
select a pair $t = (a_t, v_t) \in T(G)$ such that $|[t] \cap G|$ is maximum;
if a tie occurs, select a pair $t \in T(G)$ with the smallest cardinality of $[t]$;
if another tie occurs, select first pair;
$T := T \cup \{t\}$;
$G := [t] \cap G$;
$T(G) := \{t \mid [t] \cap G \neq \emptyset\}$;
if a_t is symbolic {let V_{a_t} be the domain of a_t}
then
$T_s := T_s \cup \{(a_t, v) \mid v \in V_{a_t}\}$
else {a_t is numerical, let $t = (a_t, u..v)$}
$T_n := T_n \cup \{(a_t, x..y) \mid$ disjoint $x..y$ and $u..v\} \cup$
$\{(a_t, x..y) \mid x..y \supseteq u..v\}$;
$T(G) := T(G) - (T_s \cup T_n)$;
end {while};
if $[T] \subseteq X$
then
begin
for each numerical attribute a_t with $(a_t, u..v) \in T$ **do**
while (T contains at least two different
pairs $(a_t, u..v)$ and $(a_t, x..y)$ with
the same numerical attribute a_t)
replace these two pairs with a new pair
$(a_t,$ common part of $u..v$ and $x..y)$;
for each t in T **do**
if $[T - \{t\}] \subseteq$ X **then** $T := T - \{t\}$;
$\mathcal{T} := \mathcal{T} \cup \{T\}$;
end {then}
else $\mathcal{J} := \mathcal{J} \cup \{T\}$;

$G := X - \cup_{S \in \mathcal{T} \cup \mathcal{J}}[S]$;
end {while};
for each $T \in \mathcal{T}$ **do**
if $\bigcup_{S \in \mathcal{T} - \{T\}}[S] = \bigcup_{S \in \mathcal{T}}[S]$ **then** $\mathcal{T} := \mathcal{T} - \{T\}$;
end {procedure}.

Note that for a local lower covering $\mathcal{T}$ of X, the set $\bigcup_{S \in \mathcal{T}}[S]$ is a lower approximation of X, however it does not need to be the best lower approximation, i.e., the local lower approximation (excluding complete decision tables).

Let us illustrate this procedure. We will induce rules for the concept $[(Flu, yes)]$ = {1, 2, 3, 4} from Table 1. Initially, we need to compute blocks for all attribute-value pairs. This time, values for numerical attributes are intervals, since our goal is to induce rules. Let us use selected cutpoints option of MLEM2. Thus, the set of all attribute-value pair blocks is:

[(Temperature, 36.8..37.6)] = {6},
[(Temperature, 37.6..40.8)] = {1, 3, 5, 7},
[(Temperature, 36.8..39.1)] = {6, 7},
[(Temperature, 39.1..40.8)] = {1, 3, 5},
[(Temperature, 36.8..40.3)] = {1, 5, 6, 7},
[(Temperature, 40.3..40.8)] = {3},
[(Headache, yes)] = {1, 2, 3, 6},
[(Headache, no)] = {4, 5, 7},
[(Cough, yes)] = {1, 2, 7}.
[(Cough, no)] = {4, 5, 6}.

The set $T(G)$ of all relevant attribute-value pairs with G = {1, 2, 3, 4} is {(*Temperature*, 39.1..40.8), (*Temperature*, 36.8..40.3), (*Temperature*, 40.3..40.8), (*Headache*, *yes*), (*Headache*, *no*), (*Cough*, *yes*), (*Cough*, *no*)}.

The set $[t] \cap G$ is the largest for t = (*Headache*, *yes*) (and is equal to {1, 2, 3}), so we select (*Headache*, *yes*). The attribute *Headache* is symbolic, T_s = {(*Headache*, *yes*), (*Headache*, *no*)}. Since [(*Headache*, *yes*)] $\not\subseteq \{1, 2, 3, 4\}$, we need to go through a second iteration of the inner **while** loop. This time $G = \{1, 2, 3\}$, and the set $T(G)$, after subtracting $T_s \cup T_n$, is {(*Temperature*, 39.1..40.8), (*Temperature*, 36.8..40.3), (*Temperature*, 40.3..40.8), (*Cough*, *yes*)}.

For two attribute-value pairs t, (*Temperature*, 39.1..40.8) and (*Cough*, *yes*), the intersection of $[t]$ and G is the largest. Also, results of the second criterion, size of $[t]$ are the same for both attribute-value pairs, so we select the first, i.e., (*Temperature*, 39.1..40.8). Thus T = {(*Headache*, *yes*), (*Temperature*, 39.1..40.8)}, $[T] = \{1, 3\} \subseteq \{1, 2, 3, 4\}$, so the first candidate for an element of the local lower covering is identified, it is T. There is a **for** loop to check whether T is minimal, T is minimal, so it is the first minimal complex and $\mathcal{T} = \{T\}$.

Our new goal is $G = X - [T] = \{1, 2, 3, 4\} - \{1, 3\} = \{2, 4\}$. The set $T(G)$ of all relevant attribute-value pairs associated with our new G is {(*Headache*, *yes*), (*Headache*, *no*), (*Cough*, *yes*), (*Cough*, *no*)}. The first criterion, related to $|[t] \cap G|$, does not select any attribute-value pair. The second criterion, the size of $[t]$, end up with three candidates: (*Headache*, *no*), (*Cough*, *yes*), (*Cough*, *no*), so we select the first attribute-value pair among these three: t = (*Headache*, *no*). Note that $[t] = \{4, 5, 7\}$

and $G = \{4\}$. Since $[T]$ = [{(*Headache*, *no*)}] $\not\subseteq \{1, 2, 3, 4\}$, we need a second iteration of the inner **while** loop. The only relevant attribute-value pair is (*Cough*, *no*), so T = {(*Headache*, *no*), (*Cough*, *no*)}, $[T] = \{4, 5\} \not\subseteq \{1, 2, 3, 4\}$, and $G = \{4\}$, however, $T(G) = \emptyset$. This set T becomes an element of $\mathcal{J}$.

The next goal is $\{1, 2, 3, 4\} - (\{1, 3\} \cup \{4, 5\}) = \{2\}$. The only relevant attribute-value pairs, members of $T(G)$, are (*Headache*, *yes*) and (*Cough*, *yes*). It is not difficult to see that in the two consecutive iterations of the inner **while** loop both attribute-value pairs will be selected and that [{(*Headache*, *yes*), (*Cough*, *yes*)}] $= \{1, 2\} \subseteq \{1, 2, 3, 4\}$. This set will pass unchanged through the first **for** loop, so it is a minimal complex. Thus, the local lower covering $\mathcal{T}$ is

{{(*Headache*, *yes*), (*Temperature*, 39.1..40.8)}, {(*Cough*, *yes*), (*Headache*, *yes*)}}.

Again, it is not difficult to see that the second **for** loop, designed to eliminate redundant minimal complexes, will not change $\mathcal{T}$. Therefore, the set of certain rules describing the concept [(*Flu*, *yes*)] = {1, 2, 3, 4} is

2, 2, 2
(Headache, yes) & (Temperature, 39.1..40.8) –> (Flu, yes),
2, 2, 2
(Cough, yes) & (Headache, yes) –> (Flu, yes).

Additionally, the lower approximation of {1, 2, 3, 4} is {1, 3} ∪ {1, 2} = {1, 2, 3}. For the concept [(*Flu*, *no*)] = {5, 6, 7} this procedure will return

$\mathcal{T}$ ={{(*Temperature*, 36.8..39.1)}, {(*Temperature*, 39.1..40.8), (*Headache*, *no*)}},

so the corresponding *certain* rules are

1, 2, 2
(Temperature, 36.8..39.1) –> (Flu, no),
2, 1, 1
(Temperature, 39.1..40.8) & (Headache, no) –> (Flu, no).

The lower approximation of {5, 6, 7} is the same set.

A set $\mathcal{T}$ will be called a *local upper covering* of X if and only if the following three conditions are satisfied:
(1) $X \subseteq \bigcup_{T \in \mathcal{T}} [T]$,
(2) every T is minimal, i.e., no proper subset T' of T exists with $[T'] \subseteq \bigcup_{T \in \mathcal{T}} [T]$,
(3) $\mathcal{T}$ is minimal, i.e., for every $T \in \mathcal{T}$, $X \not\subseteq \bigcup_{S \in \mathcal{T} - \{T\}} [S]$.

Again, the *local upper covering* should not be confused with *local covering*. The former is defined for any subset X of U, e.g., a concept, the latter is defined only for a lower or upper global approximation of the concept. The modified MLEM2 procedure for determining a single local upper covering is presented below.

Procedure for determining a single local upper covering
input: a set X (a subset of U),
output: a single local upper covering $\mathcal{T}$ of the set X,

begin
$G := X$;
$D := X$;
$\mathcal{T} := \emptyset$;
while $G \neq \emptyset$
begin
$T := \emptyset$;
$T_s := \emptyset$;
$T_n := \emptyset$;
$T(G) := \{t \mid [t] \cap G \neq \emptyset\}$;
while $(T = \emptyset$ **or** $[T] \not\subseteq D)$ **and** $T(G) \neq \emptyset$
begin
select a pair $t = (a_t, v_t) \in T(G)$ such that $|[t] \cap G|$ is maximum;
if a tie occurs, select a pair $t \in T(G)$ with the smallest cardinality of $[t]$;
if another tie occurs, select first pair;
$T := T \cup \{t\}$;
$G := [t] \cap G$;
$T(G) := \{t \mid [t] \cap G \neq \emptyset\}$;
if a_t is symbolic {let V_{a_t} be the domain of a_t}
then
$T_s := T_s \cup \{(a_t, v) \mid v \in V_{a_t}\}$
else {a_t is numerical, let $t = (a_t, u..v)$}
$T_n := T_n \cup \{(a_t, x..y) \mid$ disjoint $x..y$ and $u..v\} \cup$
$\{(a_t, x..y) \mid x..y \supseteq u..v\}$;
$T(G) := T(G) - (T_s \cup T_n)$;
end {while};
$D := D \cup [T]$;
$\mathcal{T} := \mathcal{T} \cup \{T\}$;
$G := D - \cup_{S \in \mathcal{T}}[S]$;
end {while};
for each $T \in \mathcal{T}$ **do**
for each numerical attribute a_t with $(a_t, u..v) \in T$ **do**
while (T contains at least two different
pairs $(a_t, u..v)$ and $(a_t, x..y)$ with
the same numerical attribute a_t)
replace these two pairs with a new pair
$(a_t,$ common part of $(u..v)$ and $(x..y))$;
for each $t \in T$ **do**
if $[T - \{t\}] \subseteq D$ **then** $T := T - \{t\}$;
for each $T \in \mathcal{T}$ **do**
if $\bigcup_{S \in \mathcal{T} - \{T\}}[S] \supseteq X$ **then** $\mathcal{T} := \mathcal{T} - \{T\}$;
end {procedure}.

For a local upper covering $\mathcal{T}$ of X, the set $\bigcup_{S \in \mathcal{T}} [S]$ is an upper approximation of X, however it does not need to be the best upper approximation, i.e., the local upper approximation (excluding complete decision tables).

In our example from Table 1, for the concept [(*Flu*, *yes*)] = {1, 2, 3, 4}, our algorithm returns the following local upper covering of {1, 2, 3, 4}:

{{(*Temperature*, 39.1..40.8)}, {(*Headache*, *no*), (*Cough*, *no*)}, {(*Cough*, *yes*), (*Headache*, *yes*)}}.

Thus, the corresponding upper approximation of {1, 2, 3, 4} is $\{1, 3, 5\} \cup (\{4, 5, 7\} \cap \{4, 5, 6\}) \cup (\{1, 2, 7\} \cap \{1, 2, 3, 6\}) = \{1, 2, 3, 4, 5\}$.

For the concept [(*Flu*, *no*)] = {5, 6, 7}, the algorithm returns the following local upper covering of {5, 6, 7}:

{{(*Temperature*, 36.8..39.1)}, {(*Temperature*, 39.1..40.8), (*Headache*, *no*)}},

(the same as the local lower covering for the same concept). The corresponding upper approximation of {5, 6, 7} is $\{6, 7\} \cup (\{1, 3, 5\} \cap \{4, 5, 7\}) = \{5, 6, 7\}$.

Furthermore, *possible* rules are:

1, 2, 3
(Temperature, 39.1..40.8) –> (Flu, yes),
2, 1, 2
(Headache, no) & (Cough, no) –> (Flu, yes),
2, 2, 2
(Cough, yes) & (Headache, yes) –> (Flu, yes),
1, 2, 2
(Temperature, 36.8..39.1) –> (Flu, no),
2, 1, 1
(Temperature, 39.1..40.8) & (Headache, no) –> (Flu, no).

7. Experiments

For our experiments we used 14 data sets, summarized in Table 2. All of these data sets, except *bankruptcy*, are available on the Machine Learning Repository at the University of California at Irvine. The bankruptcy data set was used by E. Altman to predict bankruptcy of companies.

Note that some of these data sets (*bupa*, *glass*, *segmentation* and *wine*) were discretized using the agglomerative cluster analysis method [4, 5] and the *hepatitis* data set was discretized using the divisive cluster analysis method [24], both methods are implemented in the LERS data mining system. Thus, some of these data sets may appear in both tables, 3 and 4.

Additionally, some of these data sets are incomplete (*breast Slovenia*, *hepatitis*, *horse*, *house* and *primary tumor*. For incomplete data sets, the same interpretation of missing attribute values (*lost*) was used in all experiments. For data sets with all missing attribute values interpreted as *lost*, local approximations are reduced to global approximations [8]. Thus, in our experiments the fact that our new algorithms are based on local approximations was not crucial. However, there is a number of other differences between local and global versions of MLEM2, the main is both the local versions compute local lower and upper coverings from scratch while the lower and upper approximations are also computed as a side effect. Also, there are other differences, e.g., in handling numerical attributes.

For the older, global version of the MLEM2 algorithm all approximations were of the type *concept*. In the Tables 3 and 4 error rates, computed as a result of *ten-fold cross validation*, are presented.

8. Conclusions

We conducted experiments comparing the new, local version of MLEM2 with the older, global version of MLEM2. The local version of MLEM2 starts from a concept while in the older version of MLEM2 starts from the previously computed global lower or upper approximations of the concept. There are two new MLEM2 algorithms, for computing certain and possible rules. There is only one older MLEM2 algorithm, if it starts from a lower approximation of the concept, it produces certain rules; if it starts from the upper approximation of the concept, it computes possible rules. Thus, the new, local version of MLEM2 computes rules from a raw data, which may be inconsistent, may have numerical attributes, and may be incomplete.

Results of our experiments show that the new approach is better: we combined results presented in both tables, 3 and 4, for every data set we selected the smallest error rate among local MLEM2 algorithm options and the smallest error rate among the global MLEM2 algorithm options, and then we used the Wilcoxon matched-pairs signed-ranks test. This test shows that the new, local version of MLEM2 is significantly better (2% significance level, two-tailed test) than the older, global version of MLEM2.

References

[1] An, A., Cercone, N.: ELEM2: A learning system for more accurate classifications, *Proceedings of the 12-th Biennial Conference of the Canadian Society for Computational Studies of Intelligence*, 1998.

[2] Bazan, J. G., Szczuka, M. S., Wojna, A., Wojnarski, M.: On the evolution of rough set exploration system, *Proceedings of the Rough Sets and Current Trends in Computing Conference*, 2004.

[3] Chan, C. C., Grzymala-Busse, J. W.: *On the attribute redundancy and the learning programs ID3, PRISM, and LEM2*, Technical report, Department of Computer Science, University of Kansas, 1991.

[4] Chmielewski, M., Grzymala-Busse, J. W.: *Global discretization of continuous attributes as preprocessing for inductive learning*, Technical report, Department of Computer Science, University of Kansas, 1992.

[5] Chmielewski, M. R., Grzymala-Busse, J. W.: Global discretization of continuous attributes as preprocessing for machine learning, *International Journal of Approximate Reasoning*, **15**(4), 1996, 319–331.

[6] Dean, J. S., Grzymala-Busse, J. W.: *An overview of the learning from examples module LEM1*, Technical report, Department of Computer Science, University of Kansas, 1988.

[7] Fayyad, U. M., Irani, K. B.: On the handling of continuous-valued attributes in decision tree generation, *Machine Learning*, **8**, 1992, 87–102.

[8] Grzymala-Busse, J. W., , Rzasa, W.: Local and global approximations for incomplete data, *Transactions on Rough Sets*, **8**, 2008, 21–34.

[9] Grzymala-Busse, J. W.: LERS—A system for learning from examples based on rough sets, in: *Intelligent Decision Support. Handbook of Applications and Advances of the Rough Set Theory* (R. Slowinski, Ed.), Kluwer Academic Publishers, Dordrecht, Boston, London, 1992, 3–18.

[10] Grzymala-Busse, J. W.: Managing uncertainty in machine learning from examples, *Proceedings of the Third Intelligent Information Systems Workshop*, 1994.

[11] Grzymala-Busse, J. W.: A new version of the rule induction system LERS, *Fundamenta Informaticae*, **31**, 1997, 27–39.

[12] Grzymala-Busse, J. W.: MLEM2: A new algorithm for rule induction from imperfect data, *Proceedings of the 9th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, 2002.

[13] Grzymala-Busse, J. W.: Rough set strategies to data with missing attribute values, *Workshop Notes, Foundations and New Directions of Data Mining, in conjunction with the 3-rd International Conference on Data Mining*, 2003.

[14] Grzymala-Busse, J. W.: Three approaches to missing attribute values—A rough set perspective, *Proceedings of the Workshop on Foundation of Data Mining, in conunction with the Fourth IEEE International Conference on Data Mining*, 2004.

[15] Grzymala-Busse, J. W., Hamilton, J., Hippe, Z. S.: Diagnosis of melanoma using IRIM, a data mining system, *Proceedings of the 7-th International Conference on Artificial Intelligence and Soft Computing*, 2004.

[16] Grzymala-Busse, J. W., Hu, M.: A comparison of several approaches to missing attribute values in data mining, *Proceedings of the Second International Conference on Rough Sets and Current Trends in Computing*, 2000.

[17] Grzymala-Busse, J. W., Rzasa, W.: Local and global approximations for incomplete data, *Proceedings of the RSCTC 2006, the Fifth International Conference on Rough Sets and Current Trends in Computing*, 2006.

[18] Grzymala-Busse, J. W., Rzasa, W.: Approximation space and LEM2-like algorithms for computing local coverings, *Fundamenta Informaticae*, **85**, 2008, 1–13.

[19] Grzymala-Busse, J. W., Stefanowski, J., Wilk, S.: A comparison of two approaches to data mining from imbalanced data, *Proceedings of the 8-th International Conference on Knowledge-Based Intelligent Information & Engineering Systems (KES 2004)*, 2004.

[20] Grzymala-Busse, J. W., Wang, A. Y.: Modified algorithms LEM1 and LEM2 for rule induction from data with missing attribute values, *Proceedings of the Fifth International Workshop on Rough Sets and Soft Computing (RSSC'97) at the Third Joint Conference on Information Sciences (JCIS'97)*, 1997.

[21] Kryszkiewicz, M.: Rough set approach to incomplete information systems, *Proceedings of the Second Annual Joint Conference on Information Sciences*, 1995.

[22] Kryszkiewicz, M.: Rules in incomplete information systems, *Information Sciences*, **113**(3-4), 1999, 271–292.

[23] Pawlak, Z.: *Rough Sets. Theoretical Aspects of Reasoning about Data*, Kluwer Academic Publishers, Dordrecht, Boston, London, 1991.

[24] Peterson, N.: Discretization using divisive cluster analysis and selected post-processing techniques, 1993, Internal Report, Department of Computer Science, University of Kansas.

[25] Stefanowski, J.: *Algorithms of Decision Rule Induction in Data Mining*, Poznan University of Technology Press, Poznan, Poland, 2001.

[26] Stefanowski, J., Tsoukias, A.: On the extension of rough sets under incomplete information, *Proceedings of the RSFDGrC'1999, 7th International Workshop on New Directions in Rough Sets, Data Mining, and Granular-Soft Computing*, 1999.

[27] Stefanowski, J., Tsoukias, A.: Incomplete information tables and rough classification, *Computational Intelligence*, **17**(3), 2001, 545–566.