knowledge bases, inference, decision units, cluster analysis

Agnieszka NOWAK-BRZEZIŃSKA[1], Roman SIMIŃSKI[1]

KNOWLEDGE MINING APPROACH FOR OPTIMIZATION OF INFERENCE PROCESSES IN MEDICAL RULE KNOWLEDGE BASES

The main aim of the article is to present the modifications of inference algorithms based on information extracted from large rule sets. The article introduces the conception of discovering the knowledge about rules saved in rule bases. It also describes the cluster analysis and decision units conception for this task and presents the optimization of forward and backward inference algorithms as well as selected experimental results.

1. INTRODUCTION

In the domain of decision support systems and data mining area, last decade brought a significant development of new algorithms, tools and applications. Knowledge based systems are created for specialized domains of competence, in which effective problem solving normally requires human expertise. Recently, medical knowledge based systems has proven itself to be a valuable tool for solving hitherto intractable problems. They can help doctors to make the decision faster or can substitute the doctor when they are not available. Complete and full domain knowledge base is the most important element of such systems. Using one of the possible inference methods, the system can work as a human expert: predict the disease, the treatment or just help to collect and organize knowledge about patients. Generally, the knowledge bases (KB) are constantly increasing in volume, thus the knowledge stored as a set of rules or patterns is getting progressively more complex and much harder to interpret or analyze. The most concerning fact is that this phenomenon has got an enormous negative impact on the time of decision making – inference in large rule sets is a time consuming process and the results of inference are in many cases hard to understand. Rules in KB can be both deterministic and nondeterministic, what makes them more difficult to analyze and to use them in inference processes [6].

Methods of inference dedicated to the rule-bases have a long descent and are fairly well known, but no significant changes in inference since RETE algorithm were made[11]. We claim that development of data mining methods may cause changes in inference methods. That is why we propose modifications of inference algorithms based on implicit and directly unreadable information extracted from large rule sets. The purpose of this study is to present a new approach for inference optimization in rule-based systems and evaluation of the effectiveness of the proposed approach. the main goal of this work is to use the information discovered in rule base to transform the rule KB into decision unit net or rules clusters, which in result improves the efficiency of inference algorithms.

The first part of the paper briefly introduces the approach of discovering the knowledge about rules saved in large rule-based KB using the cluster analysis and decision units conceptions. The next section presents the optimization of forward and backward inference algorithms. Chosen experimental results are also included.

[1] University of Silesia, Institute of Computer Science, Będzińska 39, 41-200 Sosnowiec, Poland

2. TOWARDS THE NEW MODULAR STRUCTURE OF KNOWLEDGE BASE

2.1. PRELIMINARIES AND BASIC NOTATION

Proposed approach is dedicated to the rule KB containing the rules stored by using Horn clauses, where literals are coded using attribute-value pairs. If KB contains m rules: $R = \{r_1, r_2, \ldots, r_m\}$, where $r : l_1 \wedge l_2 \wedge \ldots \wedge l_n \rightarrow c$ and n is the number of conditional literals in the rule r, l_i denotes the i-th conditional literal of r rule and c denotes the conclusion literal of r rule. Having $A = \{a_1, a_2, \ldots a_n\}$ (non-empty finite set of conditional and decision attributes) and the set V_a (the value set of $a \in A$: $V_a = \{v_1^a, v_2^a, \ldots, v_k^a\}$), each attribute in $a \in A$ may be a conditional and/or a decision attribute – a conclusion of rule r_i can be a condition in other rule r_j. If this is true, we say that rules r_i and r_j are connected and it is possible that inference chain occurs. For clarity of presentation we will use (a, v) wherever we want to describe the occurrence of any attribute-value pair. There is no assumption about rules in the set R (result of data mining process as well as the result of knowledge engineer work), we expect deep inference chains. Let's take an example rule from breast-cancer knowledge base: Class # no-recurrence-events if age # 70-79 & tumor-size # 10-14, in which the pair Class # no-recurrence-events is the conclusion, and pairs: age # 70-79 & tumor-size # 10-14 are premises of such rule. An example of KB is shown by Figure 1 later in this article we will present clusters and decision units models of this rule set. In order to improve the performance of the proposed system, the authors use the following method of storing the rules:

```
r01: c = 1 if a = 1 and b = 1 | r08: d = 4 if c = 4             | r15: c = 5 if a = 2 and b = 1 | r22: d = 4 if c = 8
r02: c = 2 if a = 1 and b = 2 | r09: f = 1 if d = 1 and e = 1 | r16: c = 6 if a = 2 and b = 2 | r23: f = 3 if d = 1 and e = 3
r03: c = 3 if a = 1 and b = 3 | r10: f = 1 if d = 2 and e = 1 | r17: c = 7 if a = 2 and b = 3 | r24: f = 3 if d = 2 and e = 3
r04: c = 4 if a = 1 and b = 4 | r11: f = 2 if d = 3 and e = 2 | r18: c = 8 if a = 2 and b = 4 | r25: f = 4 if d = 3 and e = 4
r05: d = 1 if c = 1             | r12: f = 2 if d = 4 and e = 2 | r19: d = 1 if c = 5             | r26: f = 4 if d = 4 and e = 4
r06: d = 2 if c = 2             | r13: h = 1 if f = 1 and g = 1 | r20: d = 2 if c = 6             |
r07: d = 3 if c = 3             | r14: h = 2 if f = 2 and g = 2 | r21: d = 3 if c = 7             |
```

Fig. 1. An example knowledge base.

Each line in rule's file denotes one individual rule and the structure of rule itself is stated in Fig. 1.

2.2. DECISION UNITS AS A DECISION MODEL FOR RULE BASED SYSTEMS

The decision units are simple and intuitive models describing relations in rule KB, thus the decision units can be considered as a simple tool for rule KB modeling. The concept of a decision unit is described in [8,9,10]. The idea of decision units allows to divide a set of rules into subsets by elementary decision (the rules forming the subset have exactly the same attribute-value pair in the conclusion), by decision (the rules contained in the subset have exactly the same attribute a in the each attribute-value pair (a, v) in the conclusion) or by the general decision (like above, but we do not take into consideration the information about values appearing in the conclusions of the rules). The decision units model of an example knowledge base from Fig. 1 is shown in Fig. 2.

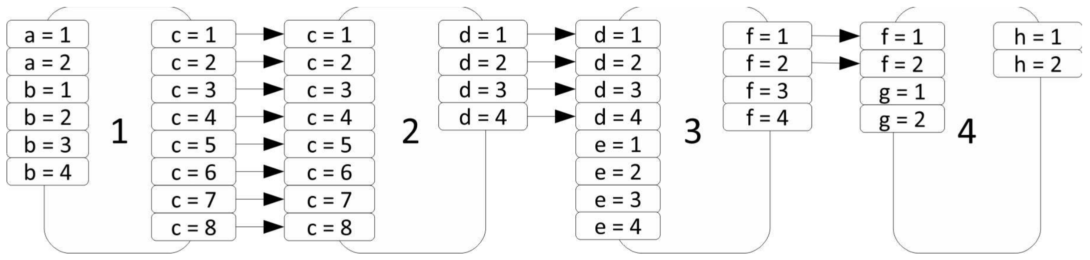

Fig. 2. An example knowledge base as decision units.

In a given KB an example decision unit is i.e. group of all rules with a given class: value of decision attribute. If it was the goal of inference, the role of inference process would be just to find such a decision unit and then search it taking into account the set of input knowledge (facts). The connections between conclusion and conditional literals are usually hidden in rule bases and only during an inference we discover how deep the inference chains sometimes are. Such KB generates a few decision units with connections between input and output entries – in this way we obtain a decision units net. An example usage of the properties of the decision units in the inference optimization task is presented in [3], same issues concerning rule base modeling contains [8]. The complexity of decision units building algorithm is O(n), we need to scan rule list only once and additional memory occupation for data structures isn't very high too.

2.3. RULES CLUSTERS AS A DECISION MODEL FOR RULE BASE

Clustering techniques have to be used whenever the size of input data exceeds the capabilities of the traditional methods of data analysis. In this work we try to find –using cluster analysis – some previously unknown connections in the rules which can help to create more effective systems. At first, the rules are organized into groups of similar rules. Clustering is considered as optimal if each cluster consists of very similar rules and if different clusters are easily distinguished with every another cluster. An example rules' cluster (in the breast-cancer knowledge base) is the group of all rules with pairs: age # 70-79 and tumor-size # 10-14. Assuming that we have such facts as input, the forward inference process would search all rules' clusters and find the one the most similar to such pairs.

The hierarchical clustering algorithms produces the hierarchical structure of the rules, therefore such algorithms will be considered in this work.

By organizing data into clusters, several additional benefits are achieved. Primo, additional information about analyzed data is acquired. The groups of rules can be further analyzed in order to observe the additional patterns which can be used to simplify the structure of KB either by simplifying the rules or by reducing the number of them. Secundo, cluster analysis (used for rules in KB) is a very fast method that allows to look for the relevant rule much faster than it is while we browse through the entire KB. The hierarchical methods are promising due the high possibility of finding the relevant rule by traversing through the tree of rules made by the hierarchical algorithm AHC [3, 4]. Every level of a tree produces more accurate answer and the number of comparisons needed is much lower than in the classical systems. The concept of new hierarchical model of KB is described in [3, 4].

The hierarchical model of KB is represented by a Tree = $\{w_1,...,w_{2n-1}\}$ where each node $w_i = \{d_i, c_i ,f, i, j\}$ (i, j are numbers of clustered groups) is the representative of the conditional parts c_i as well as the decisions d_i of the rules belonging to it. Also the similarity values f: $R \times R \rightarrow R|[0..1]$ of rules forming the cluster w_i to each other is stored.

Additionally, each representative c_i consists of all (or the most frequent) literals ci as conditional parts of the rules clustered in a cluster i (k is the number of conditional literals). Having the set of rules from KB (R), in each step the two most similar rules or clusters of rules are found and joined as a cluster. The process ends when there is only one cluster with every rule from the system in it (AHC) [3]. The created structure is a kind of binary tree so the time efficiency of searching such a tree is $O(\log_2 n)$. It means that inference engine does not have to search the whole KB in the time efficiency of O(n) like it is for the classical KB. In order to have hierarchical structure computed, the clustering algorithm has to be executed only once [3]. The time complexity of AHC algorithm is $O(n^2)$ and is typical for most of the hierarchical clustering algorithms. For large rule sets it could be an issue, fortunately, this phase is executed only once.

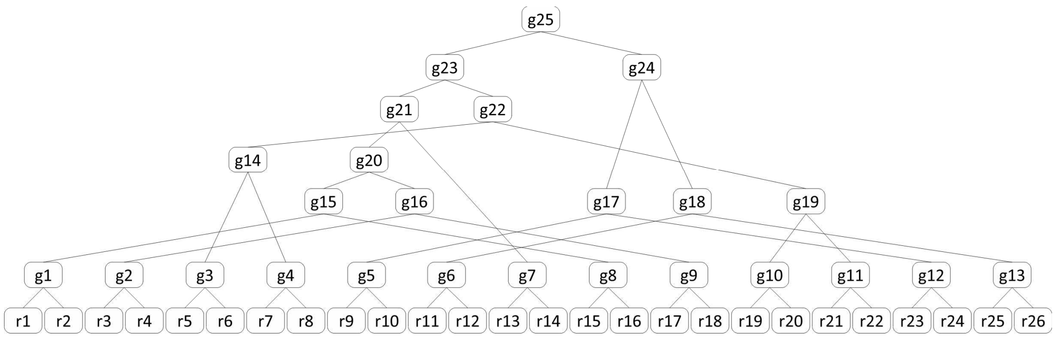
g25
g23
g24
g21
g22
g14
g20
g15
g16
g17
g18
g19
g1 g2 g3 g4 g5 g6 g7 g8 g9 g10 g11 g12 g13
r1 r2 r3 r4 r5 r6 r7 r8 r9 r10 r11 r12 r13 r14 r15 r16 r17 r18 r19 r20 r21 r22 r23 r24 r25 r26

Fig. 3. An example knowledge base as AHC rules clusters.

In Figure 3 KB *bw1* is presented as the result of clustering rules from an example KB (Figure 1) using AHC algorithm. In figure 2 decision unit labeled by 1 consists of rules r01-r04 and r15-r18 (they share the same decision attribute). The same group of rules is here presented as cluster labeled by g20 (rules inside this cluster are similar by conditional part). It lets to the conclusion that those eight rules are very similar in both parts: conditional and decisional.

3. THE CONCEPTION OF KNOWLEDGE MINING

In this paper the concept of data mining will be understood classically –as ,,the nontrivial extraction of implicit, previously unknown, and potentially useful information from large data sets or databases" [1,2]. An interesting paradox arises - data mining, which was to bring out knowledge hidden in large databases, often discovers and provides large sets of rules, which contain unreadable knowledge and are directly useless for domain experts. In this way, in many cases the methods of ,,the nontrivial extraction of implicit (...) and potentially useful information from (...)" ,,large data sets (...)" produce large rule sets with nontrivial, maybe useful but unreadable knowledge for human experts. These experts try to verify discovered knowledge or in some cases simply try to understand this new knowledge, but often without success.

This observation is the result of a series of experiments in the field of data mining which were confronted with domain experts during works realized in the Institute of Computer Science of the University of Silesia in Poland. Difficulties with the interpretation of large sets of rules generated automatically from the data made us to search for methods of extracting more general information from rule-based KBs. We can say that our goal is to discover useful, potentially implicit and directly unreadable information from large rule sets. In such a way we make a paraphrase of the classical definition of data mining and we introduce the meaning of the term knowledge mining. The proposed approach assumes that we reorganize any attributive rule KB from the set of not related rules to groups of similar rules using cluster analysis [4, 5] or decision units [8, 9, 10]. Thus we can decompose rule sets into a hierarchical structure.

This structure will be a source for extracting interesting information about the rules, i.e. which rules are similar and why, which rules create groups working out a selected decision, which attributes represent input data, terminal and indirect decisions, what are the relationships between attributes. This information is useful for understanding the internal structure of the knowledge base, as it describes the insights about knowledge saved in the rule sets. We can use this information to extract the current decision model implicitly stored in the rule sets and for formal verifying and validating the rules against the expert knowledge.

Hierarchically organized KBs tend to be susceptible to readable visualization and graphical management of the knowledge base content. It is an important issue because one of the practical goals of the project is the implementation of visual oriented software tool for knowledge engineering - a new, second version of kbBuilder [8] and HKB_Builder systems [4].

4. MODIFICATIONS OF CLASSICAL VERSIONS OF INFERENCE PROCESSES ALGORITHMS

4.1. MODIFIED GOAL DRIVEN ALGORITHM

The modification of the backward inference algorithm consists on the initiation only promising recursive calls of classical backward algorithm. The decision units net provides information which allow a preliminary assessment of whether the call could potentially confirm the nominated subgoal of inference. According to the model shown in Figure 2, in order to confirm the goal of the inference described by literal (f, 1) it is necessary to confirm the truth of the premise (d, 1) and (e, 1). Second premise isn't connected with any decision units and it will not raise any recursive calls. When (e, 1) is not a fact, the inference process for goal (f, 1) fails. A classic backward inference algorithm will try to determine that (d, 1) is a fact – now it considered as a new subgoal of inference and backward inference algorithm runs itself recursively to solve it (inference will start backwards for (d, 1) set as goal recursively). The classic version of the algorithm doesn't know whether the call is promising – i.e. it is unknown whether there is a rule that fits the new goal of inference. Indeed, in real cases very often there is no such rule and the recursive call is unnecessary.

In the case of ,,deep" inference the conception of verifying the necessity of recursive calls can significantly improve the inference algorithm. In the example shown in Figure 2 there are many possible subgoals of inference, for example (f,1), (d,2), (c,2). Table 1 presents the backward inference algorithm, which uses the information from decision units net described above. Input data for backward inference algorithm: D - the set of decision units, F - the set of facts f_i: $F = \{f_1, f_2, \ldots, f_n\}$, g - the goal of inference. The output data of the algorithm: F - the set of facts with new facts obtained through inference $F = \{f_1, f_2, \ldots, f_n, f_{n+1}, f_{n+2}, \ldots\}$, function result is a boolean value, true if the goal g is in the set of facts: $g \in F$, false otherwise.

The algorithm uses the working variables: d - the working decision unit;

$A \subseteq R(d)$ - the set of rules already activated; $r \in R(d)$ - the rule currently considered; IC(d) - the set of connected input entries IC of decision unit d is the result of the function, $IC(d) \subseteq I(d)$; truePremise - boolean variable, true if currently considered premise of rule r is true, false otherwise; w - currently considered premise's condition of rule r. The introduced algorithm realizes the inference for the single goal of backward inference g but it can be applied also for many goals (performed repeatedly).

The proposed modification of backward inference algorithm is based on elimination of unnecessary recursive calls and the reduction of the number of rules searched in each run of inference. Only promising decision units are selected for further processing (select $d \in D$ where $g \in O(d)$) and only selected subset of the whole rule set (R(d) - A) is processed in each iteration. Finally only promising recursive calls are made ($w \in IC(d)$). In each iteration the set R(d) simply contains proper rule set matching to the goal currently considered, it completely eliminates the process of searching for rules with conclusion matching to the inference goal which is necessary in the classical version of backward inference algorithm.

4.2. MODIFIED DATA DRIVEN ALGORITHM

The modification of the forward inference algorithm consists of two steps: finding the relevant rule first, and then confirming all premises of this particular rule. First step involves the search within created structure (Tree, (T)) by comparing the similarity between the set of facts (F) and the representative of the left and right branch of a given node ((node.c)) at each level of the tree and choosing the higher value. After reaching the rules' level (leafs in the tree) instead of rules clusters (which was achieved at higher levels of the tree) we start to confirm all the premises of a given rule. If all premises are true (match to the set of facts(F)) a conclusion of a given rule is added to the set of facts ($F = F \cup \{node.d\}$).

In our proposed modification, we do not browse through rule one by one as it is in classical version of the forward inference algorithm. We look for the most relevant rule with $O(\log_2(2n-1))$ time complexity (it means that instead of searching the whole KB only a fraction of it has to be analyzed).

Afterwards we check if all premises of selected rule are facts in KB. If so, we finish the inference process successively much faster than it is in classical KB without rules clusters.

Table 1 also presents the pseudocode of forward inference algorithm, which uses the information from rules' clusters tree described above. The input data for forward inference algorithm is: T - the set of rules clusters, F - the set of facts f_i: F ={ f_1, f_2, .., f_n}.

The output data of the algorithm is: F - the set of facts with new facts obtained through inference F ={ f_1, f_2, .., f_n, f_{n+1}, f_{n+2}, ..}, the result of the function given as an boolean value, true if there is at least one rule in a given KB which consists of the set of premises that match to the set of facts, false otherwise. This version of the algorithm does not need to set the goal of inference.

Table 1. Backward and Forward Inference Algorithm.

Backward Inference Algorithm

```
function backwardInferenceDU(D, g, var F) : boolean
 begin
  if g ∈ F do
   return true
 else
  A ← ϕ
  select d ∈ D where g ∈ O(d)
  truePremise ← false
  while ¬ truePremise & { R( d ) - A } ≠ ϕ do
    select r ∈ { R( d ) - A } according to the current strategy
of the selection of rules
    forall w ∈ cond(r) do
      truePremise ← ( w ∈ F )
      if ¬truePremise & w ∈ IC( d )
       then
        truePremise ←backwardInferenceDU( D, w, F )
        if ¬truePremise then
          truePremise ← environmentConfirmsFact( w )
           if ¬truePremise then
             break
             endif
        endif
      endif
    endfor
    if ¬ truePremise then
      A = A ∪ { r }
    Endif
   endwhile
  endif
  if truePremise then
    F = F ∪{ g }
  endif
  return truePremise
end
```

Forward Inference Algorithm

```
Function forwardInferenceCA(T, F) : boolean
  begin
    node ←T[2n-1]
    FactsConfirmation ← false
    while ¬ FactsConfirmation & node.level > 0 do
        leftBranch ← T[node.i]
        rightBranch ← T[node.j]
         if similarity(leftBranch.c,F) > similarity(rightBranch.c,F)
then
           node ← leftBranch
         else
           node ← rightBranch
        endif
     endhile
     check ← 1
     forall cond ∈ node.c do
        if truePremise(cond)  then
           check ← check * 1
        else
           check ← check * 0
        endif
      endfor
      if check ==1 then
         FactsConfirmation ←true
         F = F ∪{ node.d }
      else
         FactsConfirmation ←false
       endif
       return FactsConfirmation
  end
```

We start with the last node of the tree with rules clusters (Tree[2n-1], where Tree={w_1…,w_{2n-1}} and each element (node *w_i={d_i, c_i ,f, i, j}* which has been described earlier in this paper) and go deeper into the tree. At each level we calculate the similarity *similarity(node.c,F))* between the set of facts (F) and the representatives of premises of left (*leftBranch.c*) and right branch (*rightBranch.c*) of the given node (*node.c*). The new node is the child node of node with higher value of similarity. When we select the rule which is the most relevant to the given set of facts, we check all premises of this rule and if they are true we add the conclusion of this rule (*node.d*) to the set of facts (F = F∪{ *node.d*}) and set the boolean value of the variable FactsConfirmation to true. If it is not the case, we do nothing. After that we return the value of variable FactsConfirmation which can be true if the inference process finished successively or false otherwise.

5. EXPERIMENTS

The goal of the experiments is to analyze the level of optimization brought by proposed methods. Compared to the classical version of inference algorithms (both: goal and data driven), those proposed by authors are better, which can be seen in the results of the experiments. For each knowledge bases from medical domain the results of using classical version of algorithms and algorithms based on rules clusters or decision units are presented.

In Figure 4 the results of the experiments performed on four different KBs are presented. Table presented at the left side of the figure presents the number of rules in each KB and a percentage of KB which is necessary to search during the inference process to find a relevant rule. For each KB we can see the efficiency of inference processes for three different cases: data driven (AHC), goal driven (AHC) and goal driven (DU). We did not describe the goal driven algorithm for AHC algorithm (using cluster analysis) in this paper, but we implement it also. In this paper (taking into account the requirements of the page number for an article) we include only the description of forward (data driven) inference algorithm for rules clusters and backward (goal driven) inference algorithm for decision units.

Knowledge Base	rules number	Data Driven (AHC)	Goal Driven (AHC)	Goal Driven (DU)
a	32	44%	31%	87%
b	32	44%	31%	87%
c	32	25%	19%	6%
d	26	38%	46%	30%
e	26	31%	38%	8%
f	26	46%	38%	23%
g	49	24%	20%	12%
h	49	8%	20%	2%
i	23	70%	35%	9%
j	23	26%	35%	9%

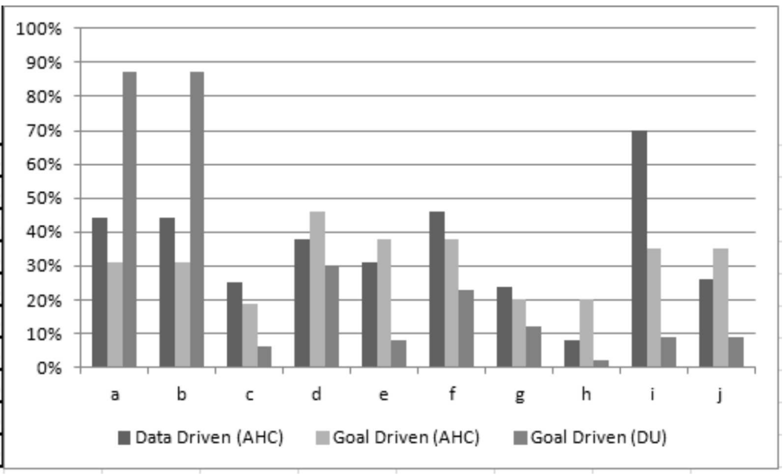

Fig. 4. Results of the experiments with inference optimization.

Cases a, b and c concern KB with 32 rules and different set of facts and goals, cases d, e and f are for KB with 26 rules, cases g and h are for KB with 49 rules and the last two cases (i and j) are for KB with 23 rules. Taking into account the case ‘a’ we see that the KB consists of 32 rules and using for example the data driven inference algorithm with agglomerative hierarchical clustering of rules, finally we will search only 44% of the KB. For case ‘h’ we managed to reduce the number of rules that are searched during the inference process to 8% of KB. The conclusion can be stated in the following way: instead of searching the whole KB, we need to search only a few percentages of all rules (when we want to find proper rule and finish the inference process successfully). There were also singular cases, when during the search process, algorithms based on clustering rules did not find any good rule to activate despite the fact that in a given KB such rule existed. The reason for this case is the fact that the procedure for creating a good descriptions for rules clusters is not always good enough. It is very crucial problem and it will be a main goal of future works. When using decision unit idea - such problem never appears.

Even if we sometimes have problems achieving the optimal results when using rules clusters, it is still worth to reduce the number of analyzed rules in that way. Especially because of providing methods for generating groups of similar rules (similar by conclusions or premises) we get knowledge about knowledge hidden in rules. We get information about relations between rules, we know which decision class is the most dominant, which is rare and if there are any outliers in rules.

6. CONCLUSIONS

In presented work we propose modifications of inference algorithms based on information discovered in rule base. We introduce the approach which uses the ,,knowledge mining approach" - the main goal of this approach is to discover useful, potentially implicit and directly unreadable information from large rule sets. In effect we paraphrase the classical definition of data mining and we introduce the meaning of the term knowledge mining. This is a new challenge, still open for research. The importance of this problem will grow along with the growth of rule based systems.

In this paper we combined two approaches - hierarchical decomposition of large rule bases using cluster analysis and the decision units conception. The thesis of this work is that using information discovered in rule base we transform the rule KB into decision unit net or rules clusters, which in result improves the efficiency of inference algorithms. The decision units are similar to clusters of rules and are thus simple and intuitive models describing relations in rule KB, being direct superstructure of a rule-base model.

The proposed models of KB allows us to browse through only a small fraction of all the rules during the inference process. The modification of backward inference algorithm is based on elimination of unnecessary recursive calls and the reduction of the number of rules searched in each run of inference. Only promising decision units are selected for further processing and only selected subset of the whole rules is processed in each iteration; finally only promising recursive calls are made. The modification of forward inference algorithm is based on the rules' clusters tree, we look for the most relevant rule with O(log 2n-1) time complexity.

An important practical result of the project is the idea of development of an online information system, which is a platform for research on rule-based KBs.

These will allow to analyze large KBs, discover and study the decision models and much more. This software will become a research platform regarding the exploration of rule-based KBs and will be also a place for the exchange of information and cooperation among researchers. A complete skeletal system is also expected to be created, offering the possibility to implement domain KBs, which will be equipped with new, developed in the project, inference algorithms.

BIBLIOGRAPHY

[1] FRAWLEY W., PIATESKY-SHAPIRO G., MATHEUS C., Knowledge Discovery in Databases: An Overview. Fall 1992, AI Magazine, pp. 213-228.

[2] HAND D., MANNILA H., SMYTH P., Principles of Data Mining. MIT Press, 2001, Cambridge, MA.

[3] NOWAK A., SIMINSKI R., WAKULICZ-DEJA A., Towards modular representation of knowledge base, Advances in Soft Computing, Physica-Verlag, Intelligent Information Processing and Web Mining, 2006, Springer Verlag Company, pp. 421-428.

[4] NOWAK A., WAKULICZ-DEJA A, The way of rules representation in composited knowledge bases, Advanced In Intelligent and Soft Computing, Man - Machine Interactions, 2009, Springer-Verlag, pp. 175-182.

[5] NOWAK A., SIMINSKI R., JACH T., XIESKI T., Towards a practical approach to discover internal dependencies in rule-based knowledge bases, 2011, Lecture Notes in Computer Science, LNCS 6954, Springer Verlag, pp. 232-237.

[6] PASZEK P., MARSZAL – PASZEK B., Nondeterministic Decision Rules in Classification Process. In: P. Herrero et al. (Eds.): OTM 2012 Workshops, 2012, LNCS 7567, Springer-Verlag, pp. 485–494.

[7] SKOWRON A., KOMOROWSKI H.J, PAWLAK Z., POLKOWSKI L.T., A rough set perspective on data and knowledge. Handbook of Data Mining and Knowledge Discovery, UK January 2002, Oxford University Press Oxford, pp. 134-149.

[8] SIMIŃSKI R., WAKULICZ-DEJA A., Verification of Rule Knowledge Bases Using Decision Units, in Advances in Soft Computing, Intelligent Information Systems, Physica Verlag, 2000, Springer Verlag Company, pp. 185-192.

[9] SIMIŃSKI R., WAKULICZ-DEJA A., Decision units as a tool for rule base modeling and verification, in Advances in Soft Computing, 2003, Information Processing and Web Mining, Springer Verlag Company, pp. 553-556.

[10] SIMIŃSKI R.: Decision units approach in knowledge base modeling, 2009, Recent Advances in Intelligent Information Systems, Academic Publishing House EXIT, pp. 597-606.

[11] On-line information: Reasoning About Rete, www.haley.com, 2001, The Haley Enterprise, Inc.