Rough Sets Inspired Extension of Forward Inference Algorithm

Roman Siminski and Alicja Wakulicz-Deja

Institute of Computer Science, University of Silesia, Poland
{roman.siminski,alicja.wakulicz-deja}@us.edu.pl

Abstract. The main goal of this work is to introduce theoretical background of the extended forward inference algorithm. Proposed algorithm allow to continue inference after its failure. Inference failure means that the inference engine is unable to obtain the solutions — the new facts or goals confirmation. Two-phase extension of classical inference algorithm is considered. In the first phase, classical forward inference is executed. If inference fails, second phase is activated and targeted search for additional facts is executed in the interactive mode. Inference extension proposed in this work is inspired be the rough sets theory which provides the conception of lower and upper approximations of particular sets.

Key words: knowledge base, forward inference, rules groups

1 Introduction

Rule-based systems are a well known solvers for specialized domains of competence, in which effective problem solving normally requires human expertise. This approach to solving ill-structured and non-algorithmic problems has been known for many years and it seemed that in this field everything had been said. But the rules are still the most popular, important and useful tool for constructing knowledge bases. During the last decade we could observe the growth of interest on rule knowledge bases applications.

In the presented work, the extension of the forward inference algorithm will be considered. The modifications and extensions of the inference algorithms for rule-based systems were described in the previous works [5, 3]. The main goal of this work is to introduce another extension of forward inference algorithm which allow inference to continue after its failure. Inference failure means that the inference engine is unable to obtain the solutions: the new facts or goals confirmation. Two-phases extension of classical inference algorithm is considered. In the first phase, classical forward inference is executed. If inference fails, second phase is activated and targeted search for additional facts is executed in the interactive mode. Proposed approach is limited to systems that have the ability to acquire new facts from the environment.

2 The Problem, Related Works, Proposed Approach

The real-word rules bases often contain only rules in the Horn clauses form. Inference with such rules can be done through the forward and backward chaining algorithms.

Forward chaining inference represents a data-oriented approach that searches for the solution space from an initial state to a final goal state. In a forward chaining system, the initial facts are processed using the rules to draw new facts as the conclusions of the applied rules. Contrary to the backward inference, forward chaining algorithm operates in the batch mode and it does not require an interaction with the system's environment. The environment nature may vary depending on the system application. Typically the user interacts with the system, but in the context of embedded systems, information for inference can be provided by the technical equipment (for example: sensors, detectors). In the literature we can find forward inference optimization algorithms [4]. Regardless of the applied particular methods of forward inference, unsuccessful inference is an important problem. This leads to a negative assessment of knowledge base system, inference failure is interpreted as inability to obtain any solution by the system. The literature studies allow us to identify three main approaches to the unsuccessful inference.

First group of approaches finishes any consideration after inference failure. This is typical for currently used main expert system shells and frameworks for expert systems. Second group attempts to continue inference using question-asking strategies. If the initial information is insufficient, the system will ask the user for additional data. There are few articles which discuss the question-asking problem analytically. Some of the early applications of experts systems consider the issue of question asking strategies — the system EXPERT [2], PROSPECT [1]. In [10] the authors define an unconfirmed observable set of assertions P_i, as a set of unconfirmed assertions that, if all were confirmed true, P_i would be proved true, but if even one was false, then P_i could not be concluded from the others. A smallest unconfirmed observable set over all the unconfirmed observable sets of top level assertions is called a global minimum inquiry set of the Horn clause system. The authors proved [10, 8] that the problem of finding a global minimum inquiry set of a Horn clause system is NP-hard, therefore they introduced and discussed the Minimum Usage Strategy — an efficient strategy approximating sub-effectiveness, in which question selection is a natural extension of the deduction process. Labelling Algorithm selects a next question for this strategy by using dynamic programming. In the [8] different strategies of question-asking strategy are considered. Other work [9] describes question-asking strategies for a Horn system in which the response costs of the questions and tile probabilistic estimates of the answers are given, the authors introduce a question sequencing rule and enhance an efficient question-asking strategy. Methods mentioned above try to deal with inference failure using variations of logic and probabilistic methods.

Third group of approaches treats inference failure as a symptom of incompleteness. The most popular approach to process incomplete data is based on the uncertain reasoning. Reasoning under uncertainty has been studied in the fields of statistic, probability theory and decision theory. The oldies methods applied in uncertain reasoning are probability theory, including Bayesian networks, certainty factors, relatively newer are Dempster-Shafer theory, fuzzy logic and fuzzy set, rough sets, several non-monotonic logic have been also proposed. Methods mentioned above differ significantly, they try to quantitative estimate the degree of truth information, which can not be true from the logical point of view.

In this work we want to introduce different proposals of deal with inference failure. We propose an extension of classical inference algorithm, which includes two steps. In the first stage, classical forward inference is executed. If inference is successful, its result are presented in the typical way and the second stage is unnecessary and not activated. If inference fails, we propose simple idea: resumption of the inference and the acquisition of the missing facts. It could be done if a system is able to work in the interactive mode and the source of information about missing facts exist. Typically, the end-user may provide such information, in general, such information can be provided by any system environment, which is able to operate interactively (for example technical equipment with sensors, detectors). The acquisition of the missing facts seems to be a naive idea, however, we propose targeted search for additional facts, based on the results of failed inference from first stage. Detailed description of proposed approach is the topic of the next section. Introduced approach is similar to the described above question-asking method, the main difference is the utilisation of rough set inspired method of evaluating the acceptable fact extension set. In contrast to described question-asking method, proposed approach appends to resume classical forward inference and acceptable fact extensions a trigger for new inference.

3 Methods

3.1 Preliminary Issues

In presented work the knowledge base is defined as a pair $\mathcal{KB} = (\mathcal{R}, \mathcal{F})$ where $\mathcal{R}$ is a non-empty finite set of rules and $\mathcal{F}$ is a finite set of facts. $\mathcal{R} = \{r_1, r_2, \ldots, r_n\}$, each rule $r \in \mathcal{R}$ will have a form of Horn's clause $r : p_1 \wedge p_2 \wedge \cdots \wedge p_m \rightarrow c$, where m — the number of literals in the conditional part of rule r, and $m \geq 0$, p_i — i-th literal in the conditional part of rule r, $i = 1 \ldots m$, c — literal of the decisional part of rule r. For each rule $r \in \mathcal{R}$ the following functions are defined:

- $concl(r)$ — the value of this function is the conclusion literal of rule r: $concl(r) = c$;
- $cond(r)$ — the value of this function is the set of conditional literals of rule r: $cond(r) = \{p_1, p_2, \ldots p_m\}$,
- $literals(r)$ — the value of this function is the set of all literals of rule r: $literals(r) = cond(r) \cup \{concl(r)\}$,
- $csizeof(r)$ — conditional size of rule r, equal to the number of conditional literals of rule r: $csizeof(r) = |cond(r)| = m$,
- $sizeof(r)$ — whole size of rule r, equal to the number of conditional literals of rule r increased by the 1 for single conclusion literal, for rules in the form of Horn's clause: $sizeof(r) = csizeof(r) + 1$.

We will also consider the *facts* as clauses without any conditional literals. The set of all such clauses f will be called *set of facts* and will be denoted by $\mathcal{F}$: $\mathcal{F} = \{f : \forall_{f \in \mathcal{F}}\ cond(f) = \{\} \wedge f = concl(f)\}$.

In this work, rule's literals will be denoted as the pairs of attributes and their values. Attributes are defined in a manner that is quite similar to that of a rough set theory. Let

A be a nonempty finite set of conditional and decision attributes[1]. For every attribute $a \in A$ the set V_a will be denoted as the set of values of attribute a. Attribute $a \in A$ may be simultaneously conditional and decision attribute. Also a conclusion of a particular rule r_i can be a condition in other rule r_j. It means that rule r_i and r_j are connected and it is possible that inference chains to occur. The literals of the rules from $\mathcal{R}$ are considered as attribute-value pair (a, v), where $a \in A$ and $v \in V_a$. Furthermore the notation (a, v) and $a = v$ is equivalent.

3.2 Unsuccessful Inference

Typically, inference failure is the result of incompleteness — it is possible to consider incompleteness of facts actually describing problem to solve, and/or the incompleteness or rules base. The condition $p \in cond(r)$ of rule r will be true, if it is a fact: $p \in \mathcal{F}$. For each rule r and non-empty set of facts $\mathcal{F} \neq \emptyset$ it is possible to show following cases of matching the conditional part of each rule $r \in \mathcal{R}$: $cond(r)$ and set of facts $\mathcal{F}$:

1. $cond(r) \subseteq \mathcal{F}$ — full matching, all rule's conditions are facts: $\forall_{p_i \in cond(r)} p_i \in \mathcal{F}$, rule r is *fireable*, and r is able to draw new fact $concl(r)$.
2. $(cond(r) \not\subseteq \mathcal{F}) \wedge (cond(r) \cap \mathcal{F} = \emptyset)$ — the lack of matching, all rule's conditions are not facts: $\forall_{p_i \in cond(r)} p_i \notin \mathcal{F}$, rule r is definitively not *fireable*, and is not able to draw new fact.
3. $(cond(r) \not\subseteq \mathcal{F}) \wedge (cond(r) \cap \mathcal{F} \neq \emptyset)$ — partial matching, not all rule's conditions are facts: $\exists_{p_i \in cond(r)} p_i \notin \mathcal{F}$, rule r is not *fireable*, and is not able to draw new fact.

The degree of matching rule r to the facts $\mathcal{F}$ we will describe using *rule to facts matching factor* $MF(r)$ defined in the following way:

$$MF(r) = \frac{|cond(r) \cap \mathcal{F}|}{|cond(r)|}$$

For cases described above: $MF(r) = 1$ for case 1, $MF(r) = 0$ for case 2, $0 < MF(r) < 1$ for case 3. From the inferential point of view, both cases 2 and 3 cause the impossibility of obtaining new facts. However, in the third case the rule r is promising — it will be fireable when we will succeed to asset as true conditions, that were considered to be false until now. The higher the $MF(r)$ is, the greater is rule r usefulness. For the clarity of the further presentation, let $\mathcal{R}^F$ be the set of rules fully matched to the facts, $\mathcal{R}^P$ be the set of rules partially matched to the facts and $\mathcal{R}^N$ denotes the set of rules that do not match to the facts at all:

$$\mathcal{R}^F = \{r \in \mathcal{R} : cond(r) \subseteq \mathcal{F}\}$$
$$\mathcal{R}^P = \{r \in \mathcal{R} : (cond(r) \not\subseteq \mathcal{F}) \wedge (cond(r) \cap \mathcal{F} \neq \emptyset)\}$$
$$\mathcal{R}^N = \{r \in \mathcal{R} : cond(r) \cap \mathcal{F} = \emptyset\}$$

[1] Decision attributes are attributes that are at least once included in conclusion of any rule from $\mathcal{R}$.

3.3 Rough Set Inspiration

Inference extension proposed in this work is inspired be the rough sets theory [7]. Rough sets theory provides the conception of lower and upper approximations of particular sets [6]. In general, the proposed approach is inspired by the set approximation, but relation with the rough set theory is not strict. The description presented bellow informally refers to the lower and upper approximations.

Based on the knowledge provided by the rule $r \in \mathcal{R}$ and nonempty set of facts $\mathcal{F}$, it is possible to identify the set of rule r conditions, which can be with certainty classified as the facts: $\underline{\mathcal{F}}(r) = \{c \in cond(r) : cond(r) \subseteq \mathcal{F}\}$. Informally, the set $\underline{\mathcal{F}}(r)$ is conceptually similar to the lower approximation of the fact set $\mathcal{F}$ under the knowledge described by the rule r.

It is also possible to identify set of rule r conditions, which can be only classified as possible facts: $\overline{\mathcal{F}}(r) = \{c \in cond(r) : cond(r) \cap \mathcal{F} \neq \emptyset\}$. The items of $\overline{\mathcal{F}}(r)$ are facts or they appears in the rule r premises partially matched to the facts set. Informally, the set $\overline{\mathcal{F}}(r)$ is conceptually similar to the upper approximation of the fact set $\mathcal{F}$ under the knowledge described by the rule r. The set $\mathcal{F}_{BN}(r) = \overline{\mathcal{F}}(r) - \underline{\mathcal{F}}(r)$ contains conditions representing missing facts. If these conditions will be the facts, the rule r will be fireable. It is informally similar to the boundary region of $\mathcal{F}$, and thus consists of those conditions that cannot be classified as the facts on the basis of knowledge described in the rule r, but are interesting in the context of inference after failure. The set $\mathcal{F}_{BN}(r)$ will be called *basic fact extension* set for rule r.

It is possible to consider approximation-like total sets described above, for any nonempty set of rules $R \subseteq \mathcal{R}$:

$$\underline{\mathcal{F}}_T(R) = \{\underline{\mathcal{F}}(r) : r \in R\}$$
$$\overline{\mathcal{F}}_T(R) = \{\overline{\mathcal{F}}(r) : r \in R\}$$
$$\mathcal{F}_{TBN}(R) = \overline{\mathcal{F}}(R) - \underline{\mathcal{F}}(R)$$

The first naive extension of forward inference proposed in this work, will use the set $\mathcal{F}_{BN}(R)$ it will be called *fact extension* set for rules R. The pseudo-code 3 presents the implementation of classical forward inference algorithm (CFI). The input data are — rule base: $\mathcal{R} = \{r_1, r_2, ..., r_m\}$, facts set: $\mathcal{F} = \{f_1, f_2, ..., f_k\}$. The output data are — new facts in $\mathcal{F} = \{f_1, f_2, ..., f_k, f_{k+1}, ..., f_{nk}\}$, function result: **true** if new facts inferred, **false** otherwise.

The first proposed *extended forward inference algorithm* (*EFI*) is presented by the pseudo-code 4. At the beginning the *EFI* algorithm calls classical forward inference *CFI*. If inference is successful, function *CFI* returns true and the first stage of the *EFI* algorithm is the last one — the *EFI* works like *CFI*. If inference is unsuccessful, function *CFI* returns false and the second stage of algorithm is activated. On this stage, the promising rules subset R is selected and the $\mathcal{F}_{BN}(R)$ is determined. The $\mathcal{F}_{BN}(R)$ set contains sets of conditions representing possible facts. This set can be ordered by the selected strategy — for simplicity the minimal subset will be considered first.

Next, the algorithm looks for first acceptable extension of the facts set $e \in \mathcal{F}_{BN}(R)$. Decision about the truth of the fact is taken by the function *accepted*, which return true if proposed extension e is acceptable, false otherwise. In object-oriented implementation of *EFI* algorithm, function *accepted* is defined as the abstract function or method. It

Algorithm 3: *CFI* — Classical Forward Inference algorithm

function *CFI*($\mathcal{R}$, $\mathcal{F}$) : **boolean**
var $A \leftarrow \emptyset$
var $NF \leftarrow \emptyset$
begin
select rules subset $\mathcal{R}^F$ **from** $\mathcal{R}$ **according to** $\mathcal{F}$
while $\mathcal{R}^F \neq \emptyset$ **do**
  $r \leftarrow$ **select rule from** $\mathcal{R}^F$ **according to current selection strategy**
  $NF \leftarrow NF \cup \{concl(r)\}$
  $\mathcal{F} \leftarrow \mathcal{F} \cup NF$
  $A \leftarrow A \cup \{r\}$
  select rules subset $\mathcal{R}^F$ **from** $\mathcal{R} - A$ **according to** $\mathcal{F}$
end while
return $NF \neq \emptyset$
end function

may be implemented in any way, e.g. through interaction with the system user or, in general, with system environment. First acceptable extension e is added to the fact set and *EFI* function calls itself recursively. Next call of *CFI* function should be successful — accepted facts extension should fire proper new rule. Thus, it is possible to skip recursive call and to call *CFI* directly. Recursive version is more general and allows the consideration of different methods of evaluating possible facts extensions.

Algorithm 4: *EFI* — Extended Forward Inference algorithm

function *EFI*($\mathcal{R}$, $\mathcal{F}$) : **boolean**
begin
if *CFI*($\mathcal{R}$, $\mathcal{F}$) **then**
  return true
else
  select rules subset $R \subseteq \mathcal{R}$ **where** $\forall_{r \in R}\ cond(r) \cap \mathcal{F} \neq \emptyset$ // (1)
  determine $\mathcal{F}_{TBN}(R)$ **ordered by facts extension strategy**
  for all $e \in \mathcal{F}_{TBN}(R)$ **do**
    if *accepted*(e) **then**
      $\mathcal{F} \leftarrow \mathcal{F} \cup e$
      if *EFI*($\mathcal{R}$, $\mathcal{F}$) **then**
        return true
      end if
    end if
  end for
end if
end function

3.4 Criticism of the Proposed Solution

Proposed method of inference after failure looks for first acceptable facts sets extension $\mathcal{F}_{TBN}(r)$. Method of calculation the possible facts extension set described above, ensures successful classical inference after acceptance of any set's element. It can trigger only one step of classical inference, or it may cause many iteration of the algorithm. Unfortunately, the results of inference may be unpredictable and distant from expectation. Additionally, iterative acquisition of acceptable fact extension may be uncomfortable for the user — the number of possible query for large rule bases can be high, thus unacceptable in the real-world applications.

Proposed algorithm is also sensitive to the order of selection of the $\mathcal{F}_{TBN}(R)$ set's elements. Different order of the elements selected from $\mathcal{F}_{TBN}(R)$ can lead to different inference results. It is not clear whether the obtained inference results are useful and satisfactory from the point of view of the problem being solved. Although the proposed solution may be capable of providing potentially useful results, it is possible to formulate more appropriate methods of determination of $\mathcal{F}_{TBN}(R)$ set and selection of promising facts extension. As presented below, extraction of the internal rules dependencies will be a source of information for improving selection of possible facts.

3.5 Rules Groups as Simple Decision Model

For each rule base $\mathcal{R}$ with n rules, it is possible to consider a partition PR of a set $\mathcal{R}$. PR is grouping of set's $\mathcal{R}$ rules into non-empty subsets, in such way that every rule is included in one and only one of the subsets: $\emptyset \notin PR$, $\bigcup_{R \in PR} = \mathcal{R}$ and if $R_i, R_j \in PR$ and $R_i \neq R_j$ then $R_i \bigcap R_j = \emptyset$. The sets in PR will be called the rules groups of the partition. In this work only *simple partitioning strategies* will be considered. The *membership criterion function* decides about the membership of rule r in a particular group $R \subseteq PR$ according to the membership function mc. Simple strategy divides the rules by using the algorithm with time complexity not higher than $O(n \cdot k)$, where $n = |\mathcal{R}|$ and $k = |PR|$. Simple strategy creates final partition PR by a single search of the rules set $\mathcal{R}$ and allocation of each rule r to the proper cell R, according to the value of the function $mc(r, R)$ described bellow.

Proposed approach assumes that the membership criteria is defined by the mc function, which is defined individually for every simple partition strategy. The function: $mc : \mathcal{R} \times PR \rightarrow [0..1]$, return the value 1 if the rule $r \in \mathcal{R}$ with no doubt belongs to the group $R \subseteq PR$, 0 in the opposite case. The value of the function from the range $0 < mc < 1$ means the partial membership of the rule r to the group R. The method of *determining* its value and its *interpretation* depends on the specification of a given partition method. It is possible to achieve many different partitions of rule base using single mc function.

The *simple strategy* partitioning algorithm is presented [3], it is simple and for this reason will be omitted. The input parameters are: the knowledge base $\mathcal{R}$, the function mc that defines the membership criteria, and the value of the threshold T. Output data is the partition PR. In connection with the main problem, two partitioning strategies will be discussed. The first is *basic decision oriented partition* PS_B which creates groups of

the rules from $\mathcal{R}$ by grouping rules with the same conclusions. The membership criteria for rule r and group R is given by the function mc defined as follows:

$$mc(r,R) = \begin{cases} 1 \text{ if } \forall_{r_i \in R}\ concl(r_i) = concl(r) \\ 0 \text{ otherwise} \end{cases} \tag{1}$$

By using the *simple partition algorithm* [3] with the mc function defined in this way, we obtain the following groups: $R = \{r \in \mathcal{R} : \forall_{r_i \in R}\ concl(r_i) = concl(r)\}$. The number of groups in the partition depends on the number of *different decisions* included in conclusions of such rules. When we distinguish different decisions by the different conclusions appearing in the rules, we get one group for each conclusion. All rules grouped within a rule set take part in an inference process confirming the goal described by the particular attribute-value — for each $R \in PR_B$ the conclusion set $|Concl(R)| = 1$. Similarly, in decision tables $DT = (U, A \cup \{d\})$ — considered in the rough set theory [7, 6] — values of decision attributes d defines partition of objects from U into the *decision class* $CLASS_A(d) = \{X_A^1, X_A^2, \ldots X_A^{r(d)}\}$, where $r(d)$ is the cardinality of value set for attribute a and X_A^i is the i-th decision class of A. Each group of rules from an elementary partition is similar to i-th decision class X_A^i.

The second partitioning strategy considered in this work is *decision oriented partition* denoted PR_D. It also uses the *simple partition algorithm* with the mc function defined in the following way:

$$mc(r,R) = \begin{cases} 1 \text{ if } \forall_{r_i \in R}\ attrib(concl(r_i)) = attrib(concl(r)), \\ 0 \text{ otherwise.} \end{cases} \tag{2}$$

Each generated group of the rules have the following form: $R = \{r \in \mathcal{R} : \forall_{r_i \in R}\ attrib(concl(r_i)) = attrib(concl(r))\}$. When we distinguish different decisions by the different attribute from conclusions appearing in the rules — we obtain one group for each decision attribute. Thus any rule set $R \in PR_D$ can be described as $R = \{\bigcup R' \in PR_B : \forall r_i, r_j \in R'\ attrib(concl(r_i)) = attrib(concl(r_j))\}$. It means that the decision produced by the ordinal decision partition can be constructed as the composition of the basic decision partitions. By analogy, decision oriented partition is similar to the decision tables $DT = (U, A \cup \{d\})$, where d is single decision attribute.

The partitioning strategy PR_B and PR_D describes global dependencies appearing in the rule base. To express and utilize dependencies between rules in any partition PR, it is possible to define *partitions connection graph* $G_{PR} = (PR, C)$. PR represents nodes of G_{PR}, $C \subseteq PR \times PR$ represents relation which defines edges of G_{PR}. In the context of connection graph, we assume that for any rules group R we consider two sets — $In(R)$ and $Out(R)$. $In(R)$ and $Out(R)$ denote respectively the set of *input* and *output* data of the rules set. Items of those sets are literals appearing in the rules from R. The structure of $In(R)$ and $Out(R)$ depends on the currently considered partition strategy. The C relation can be defined in the following way: $C = \{(x, y) \in PR \times PR : Out(x) \cap In(y) \neq \emptyset\}$. In the introduced modification of inference, the decision partition strategy is considered and following mappings are proposed: $Out(R) = Concl(R)$, $In(R) = Cond(R)$. Thus, the C relation connects the group of rules with the common attributes in conclusion and conditions. The subsets can be defined: connected inputs sub-set $In_C(R) \subseteq In(R)$ can be considered:

$In_C(R) = \{(a, v) \in In(R) : \exists_{r \in \mathcal{R}} \ (a, v) = concl(r)\}$, and connected output sub-set $Out_C(R) \subseteq Out(R) : Out_C(R) = \{(a, v) \in Out(R) : \exists_{r \in \mathcal{R}} \ (a, v) \in cond(r)\}$.

3.6 Rules Groups in After Failure Inference Algorithm

Simple decision models provided by the two previously described partitioning strategies allow to direct the searching of the fact extension set. For relatively small rule sets, or „flat" rules sets ($\forall_{R \subseteq PR} In_C(R) = \emptyset \wedge Out_C(R) = \emptyset$), basic decision partition can be used. The partitions connection graph G_{PR} provides enough information to identify such situations. For rules bases containing big number of rules, basic decision partition may produce large number of rules group and ordinal decision partition strategy may be used. Apart of used partitioning method, decision model allows to consider different strategies of selection the promising rules set for facts extension searching.

First, it is possible to reject rules sets from PR with empty intersection with facts set. Let's assume that $PR_F \subseteq PR$ is subset of partition PR with non empty intersection with the facts set: $PR_F = \{R : \exists_{r \in R} \ cond(r) \bigcap \mathcal{F} \neq \emptyset\}$. For each group $R \in PR$ it is possible to determine fact extension set $\mathcal{F}_{TBN}(R)$. Thus, the set of fact extension sets can be considered: $FES = \{\mathcal{F}_{TBN}(R) : R \in PR_F\}$.

Selection of the first acceptable fact set extension will consist of two steps. In first step, the most promising rules group from PR_F will be selected. In the second step, the most adequate facts extension set for selected rules set will be promoted for acceptance. The combination of two above steps provides the different strategies of selecting facts extension set. On the level of the most promising rules group R selection, it is possible to indicate the following possibilities:

- The connected output sub-set: $Out_C(R) = \emptyset$ — inference will produce the one or multiple new facts from $Out(R)$ and the inference will stop. The new facts are unable to trigger any other rule. This rule selection strategy offers shortest inference path and predictable effects — only facts from $Out(R)$ are expected.
- The connected output sub-set: $Out_C(R) \neq \emptyset$ — inference will produce the one or multiple new facts from $Out(R)$ and inference will likely continue. The new facts are able to trigger other rules from rules group connected with R. This rule selection strategy offers the ability to obtain new facts not only from the $Out(R)$.
- It is possible to select the rule group R as a start point of longest path in the G_{PR}. This strategy offers the possibility of obtaining a variety new facts.
- The rules group with maximal facts coverage can be selected — $MF(R) = \frac{|Cond(R) \cap \mathcal{F}|}{|Cond(R)|}$.

The modified after failure inference algorithm differs in two lines and therefore will not be presented as separate pseudo-code. The line number (1) in the algorithm 4 should be replaced by the following two lines:

1. $PR =$ **createDecisionPartition**($\mathcal{R}$)
2. **select most promising rules subset** $R \subseteq PR$

4 Implementation Issues and First Experiments

The first version of *EFI* algorithm (4) has been implemented as a part of *kbExplorator* desktop system. *kbExplorator* is the system which integrates two components — desktop and web application. Web application allows to create, edit and manage rules bases which are stored on the *kbExplorator* server. The knowledge bases are assigned to the users registered in the web application. Desktop application allows to perform different operations on the knowledge base stored on the server, operations jointly referred to as *exploration*. Web application is mainly dedicated to the knowledge engineers as a tool for knowledge base building and improving. Desktop application is the tool for more sophisticated knowledge exploration. The kernel of the desktop application consist of object-oriented packages implemented in Java. Packages offer, among other things, different types of reasoning, both classic and modified versions and also new algorithms, including *EFI* proposed in this work. Applications now have the status of a prototype, they will be available as the free software online in the coming months. Till now, *kbExplorator* is one of the few expert system building tools which allows to continue inference after its failure.

The first version of *EFI* algorithm selects a minimal acceptable facts extensions. The main advantage of this algorithm is its simplicity. Unfortunately, experiments confirmed earlier expectations, the results of inference may be unpredictable and iterative acquisition of acceptable fact extension have proven to be uncomfortable for the user. For large rule bases, iterative acquisition of missing fact was not acceptable for the real-world applications.

Proposed algorithm is also sensitive to the order of selection of the fact extension set elements. Different order of the elements selections can lead to different inference results. It is not clear whether the obtained inference results are useful and satisfactory from the point of view of the problem being solved. Although the proposed solution may be capable of providing potentially useful results, modification proposed in the previous section will be considered in the future work. Modified *EFI* algorithm utilizes the internal dependencies between rules divided into the decision oriented group of rules. Detailed experiments will be made in the next stage of research and the experimental results will be presented in the next publications. One of the reasons is the need of implementation of similar methods for a comparative study. Unfortunately, there is a lack of detailed source information which presents enough information about the implementation details on the proper level of detail. Additional literature studies are needed and future research could be focused on a comparative study of algorithms proposed in this work with the question-asking oriented methods.

Main tool of proposed optimisation are decision oriented partitions of rule base. The complexity of decision partition is $O(n \cdot k)$, where $n = |\mathcal{R}|$, $k = |PR|$, where the number of groups in the partition $k : 1 \leq k \leq n$ typically is significantly smaller than the number of rules n. We have to store additional information for created rules partitions, however additional memory or disk space occupation for data structures seems acceptable. For n rules and k rules group we need approximately $is \cdot n + ps \cdot m$ bytes of additional memory for data structures (is âĂŤ size of integer, ps âĂŤ size of a pointer or reference).

5 Conclusions

In the presented work, the conception of the extended forward inference algorithm was presented. This algorithm allows to continue the inference after its failure. Inference failure means that the inference engine is unable to obtain the solutions: the new facts or goals confirmation. It often happens that the initial facts are not sufficient for a knowledge-based system to reach any conclusion, and more information is needed. In this work two-phase extension of classical inference algorithm was considered. In the first phase classical forward inference is executed. If inference fails, second phase is activated and targeted search for additional facts is executed in the interactive mode. Proposed solution is similar to a question-asking method described in [10, 8]. Question-asking is to figure out what additional information should be known if no useful results can be proved with the available data. The QA problem is computationally hard in a propositional knowledge-based system, even if the system is composed only of Horn clauses. The existing approaches are based on logic. Described solutions try to examine an unconfirmed observable assertion set of a top level conclusion. Inference extension proposed in this work is inspired be the rough sets theory, which provides the conception of lower and upper approximations of particular sets.

In general, the proposed approach is inspired by the set approximation, but relation with the rough set theory is not strict. The description presented in this article informally refers to the lower and upper approximations. Based on the knowledge provided by the rules and set of facts, it is possible to identify the set of rules conditions which can be with certainty classified as the facts. It is conceptually similar to the lower approximation of the fact set. It is also possible to identify set of rules conditions which can be only classified as possible facts, because they appear in the rules premises partially matched to the facts set. It is conceptually similar to the upper approximation of the fact set. The boundary set contains conditions which represent missing facts. If these conditions are facts, some rules will be fireable. It is informally similar to the boundary region of facts, and thus consists of those conditions that cannot be classify as the facts on the basis of knowledge described in the rules base, but are interesting in the context of inference after failure. This work introduced theoretical background of the algorithm, detailed experiments will be made in the next stage of research and the experimental results will be presented in the next publications.

Acknowledgements

This work is a part of the project „Exploration of rule knowledge bases" founded by the Polish National Science Centre (NCN: 2011/03/D/ST6/03027).

References

1. Duda, R., Gaschnig, J., Hart, P.: Model design in the prospector consultant system for mineral exploration. Expert systems in the microelectronic age 1234, 153–167 (1979)
2. Hayes-Roth, F., Waterman, D., Lenat, D.: Building expert systems (1984)

3. Nowak-Brzezinska, A., Siminski, R.: New inference algorithms based on rules partition. In: Proceedings of the 23th International Workshop on Concurrency, Specification and Programming, Chemnitz, Germany, September 29 - October 1, 2014. pp. 164–175 (2014)
4. On-line information: Reasoning About Rete. www.haley.com (2001)
5. Simiński, R.: Extraction of rules dependencies for optimization of backward inference algorithm. In: Kozielski, S., Mrozek, D., Kasprowski, P., Mąsysiak-Mrozek, B., Kostrzewa, D. (eds.) Beyond Databases, Architectures, and Structures, Communications in Computer and Information Science, vol. 424, pp. 191–200. Springer International Publishing (2014)
6. Skowron, A., Peters, J., Suraj, Z.: An application of rough set methods to control design. In: Proc. of the Workshop on Concurrency, Warsaw. pp. 214–235 (1999)
7. Skowron, A., Komorowski, J., Pawlak, Z., Polkowski, L.: Handbook of data mining and knowledge discovery. chap. Rough Sets Perspective on Data and Knowledge, pp. 134–149. Oxford University Press, Inc., New York, NY, USA (2002)
8. Triantaphyllou, E., Wang, J.C.: The problem of asking the minimum number of questions in horn clause systems. Mathematical and computer modelling 20(9), 75–87 (1994)
9. Wang, J., Triantaphyllou, E.: A cost effective question-asking strategy for horn clause systems. Annals of Mathematics and Artificial Intelligence 17(2), 359–379 (1996)
10. Wang, J., Vate, J.V.: Question-asking strategies for horn clause systems. Annals of Mathematics and Artificial Intelligence 1(1-4), 359–370 (1990)