Impact of Conflict Set Resolution Strategies on Inference Efficiency in Rule-Based Systems

Agnieszka Nowak-Brzezińska
University of Silesia in Katowice
Faculty of Science and Technology
Institute of Computer Science
Katowice, Poland

Daniel Nowak
University of Silesia in Katowice
Faculty of Science and Technology
Katowice, Poland

Abstract

This study explores the impact of 4 conflict set resolution strategies-random, recency, textual order, and specificity-on the efficiency of forward reasoning in rule-based expert systems. Experiments were conducted on 7 datasets, with knowledge bases ranging from over 100 to 150,000 rules. We evaluated inference time, success rate, and the number of new facts generated. The recency strategy proved most efficient, yielding the shortest inference time and fewest new facts, while the specificity strategy was the slowest. Inference failures occurred only with minimal input data (1% of facts), affecting less than 5% of cases.

krywords: inference algorithm, rule-based knowledge bases, conflict set, forward inference, expert system

1. Introduction

Expert systems (ES) are a branch of artificial intelligence that has remained relevant for decades. Regardless of their intended purpose, expert systems-like all IT solutions-require optimization [10].

A traditional expert system consists of two core components: a knowledge base and an inference engine. The inference engine operates according to principles of logic and typically uses one of two reasoning methods: forward chaining or backward chaining. Inference is considered successful when the system is able to derive new knowledge. In forward chaining, the goal is to activate rules whose premises match the currently known (or confirmed) facts-i.e., the input information. If no specific inference goal is defined, the process continues until all applicable rules have been activated. If a specific goal has been defined, the process stops as soon as a rule is activated whose conclusion matches the goal. In backward chaining, the reasoning goal must be defined from the outset. The system attempts to verify this goal by recursively examining rule conclusions and the available facts, searching for those that can confirm the primary or intermediate goals.

Regardless of which reasoning method is employed, a large knowledge base presents challenges: the time required to analyze the rules and, more importantly for our research purposes, the issue of rule selection determining which rule to activate when multiple options are available at a given inference step.

In our research, we evaluated four different inference conflict set resolution strategies and their impact on various aspects of forward reasoning efficiency, including execution time, the amount of newly inferred facts, and the frequency of successful inference. This paper presents the results of experiments conducted on seven distinct knowledge bases. Both the inference algorithm and the conflict set resolution strategies were implemented independently.

1.1. Article structure

The structure of the paper is as follows. After introducing the main idea in Section 1, Section 2 discusses the inference process in expert systems, briefly comparing forward and backward inference methods, with a greater emphasis on forward reasoning-the focus of our study and conflict set resolution strategies applied in the experiments. Section 3 outlines the research methodology, including the description of datasets used (with information about the algorithm to generate rules using the RSES environment), the number of experiments conducted, and the key factors analyzed. Section 4 presents the most significant results along with their interpretation. The paper concludes with a summary in Section 5.

2. Inference processes in rule clusters knowledge base - description of the idea

The knowledge in expert systems may be either expertise, or knowledge that is generally available from books, magazines, and knowledgeable persons. In this sense, knowledge is considered to be at a lower level than the more rare expertise. One common method of representing knowledge is in the form of IF THEN type rules, such as: *IF the light is red THEN stop*. If a fact exists that the light is red, this matches the pattern "*the light is red*." The rule is satisfied and performs its action of "*stop* [4]. "

Essentially, inference algorithms allow the system to simulate human reasoning by applying formal rules to known facts and deriving new information. These algorithms are a crucial part of artificial intelligence (AI), enabling machines to make decisions, solve problems, and support users in various domains, such as medicine, engineering, and finance. The process of inference is often guided by a set of rules and facts. Facts are the pieces of knowledge that the system already knows or has observed. The role of an inference algorithm is to identify relevant facts and rules in the knowledge base, to apply logical reasoning to derive new facts or conclusions from those facts and to repeat the process until a specific goal is achieved or no more conclusions can be drawn.

2.1. Forward Inference vs. Backward Inference

Classical forward inference starts with known facts and applies rules to move towards conclusions. It is typically used when the system needs to explore all potential outcomes based on the available data [12]. In this approach, the system repeatedly scans its knowledge base to find rules whose premises (conditions) match the current contents of the fact base. When such a rule is found, it is considered eligible for activation. If multiple rules are eligible at the same time, the system uses a conflict set resolution strategy to decide which rule to apply first (Algorithm 1). Once a rule is activated, its conclusion is added to the fact base, and a rule is blocked. The algorithm proceeds iteratively until: a specified goal has been derived (if one was defined), or no new rules can be activated. If a rule is blocked once activated, it will primarily save time on analyzing this rule in subsequent iterations - if such are possible. The reasoning process is iterative, which means that if we have more than one rule to activate in a given iteration in the knowledge base, all of them feed the conflict set and we choose only one rule to activate, using one of the 4 strategies: recency, textual order, random or specificity. The remaining rules are not activated in this particular iteration. The rule that was activated is blocked. All this allows us to search for rules to activate in the next iteration again, but without the one that was activated previously. If the previous activation contributed to the generation of a new fact, then it can now affect the activation of other, new rules that were not possible to activate before.

Forward inference is especially effective in environments where new data continuously en-

Algorithm 1 Forward Chaining Inference Algorithm with Conflict Set Resolution

```
Require: Fact base $FB$, Knowledge base $KB$ (rules with premises and conclusion), Optional
    goal $G$, Conflict resolution strategy $S$
Ensure: Updated fact base; success/failure if goal is specified
 1: $inferred \leftarrow$ true
 2: while $inferred$ is true do
 3:     $inferred \leftarrow$ false
 4:     $ConflictSet \leftarrow \emptyset$
 5:     for all rule $r$ in $KB$ do
 6:         if $r.premises \subseteq FB$ and $r$ is not blocked then
 7:             Add $r$ to $ConflictSet$
 8:         end if
 9:     end for
10:     if $ConflictSet \neq \emptyset$ then
11:         Select $r_{fire}$ from $ConflictSet$ using strategy $S$
12:         if $r.conclusion \notin FB$ then
13:             Add $r_{fire}.conclusion$ to $FB$
14:             $r.blocked \leftarrow$ true
15:             $inferred \leftarrow$ true
16:         end if
17:         if goal $G$ is specified and $G \in FB$ then
18:             return SUCCESS
19:         end if
20:     end if
21: end while
22: if goal $G$ is specified then
23:     return FAILURE
24: else
25:     return updated $FB$
26: end if
```

ters the system, and the goal is to automatically evaluate the implications of that data in real time.

While forward inference starts from known facts and applies rules to derive new conclusions, backward inference works in the opposite direction: it begins with a goal (a hypothesis to be proven) and attempts to determine whether that goal can be logically deduced from the known facts and rules in the knowledge base.

In backward inference, also known as goal-driven reasoning, the system starts by checking whether the goal is already present in the fact base. If not, it looks for rules whose conclusions match the goal. The system then attempts to verify whether the premises of such rules can be satisfied using the known facts-or whether they too must be proven using other rules. This process is recursive, as each unsatisfied premise becomes a new sub-goal. This algorithm is often more efficient than forward chaining when the number of goals is small and specific, since it avoids unnecessary rule activations. Backward inference is widely used in expert systems where the aim is to confirm or reject a specific hypothesis-such as in medical diagnosis, troubleshooting systems, or recommendation engines-where only a small subset of possible conclusions is of interest at a given time.

2.2. Conflict set strategies

In the literature, the problem of selecting a single rule to activate from a set of eligible rules is known as *conflict set* resolution. Over time, several strategies have been proposed to manage this aspect of inference control. One of the most intuitive methods is the *textual order* strategy, which simply activates the first rule on the list - essentially relying on the sequence in which rules appear. Another notable approach is the *recency* strategy, which prioritizes either the most recently added rules in the knowledge base or those whose premises match the most recently added facts in the fact base. The *random* strategy, as the name implies, selects a rule at random from the conflict set. Meanwhile, the *specificity* strategy gives preference to rules with the largest number of premises. The reasoning behind this approach is that a rule validating more conditions may be considered more reliable than one that checks fewer [11]. In the first phase of our research, we wanted to investigate the efficiency of the inference process for various strategies. Although the strategies themselves have been described in the literature, there are no known research results (for real knowledge bases) to which we could compare ourselves when implementing the next stage of research, i.e. clustering similar rules and inferring on rule clusters.

3. Methodology of the experiments

In our experiments, we used seven diverse datasets, varying in domain, size, and data type, including both small and large sets. Crucially, we applied the same experimental setup to each knowledge base: five distinct scenarios were executed using five different sets of facts and four inference control strategies. As a result, we conducted a total of 100 experiments for each of the seven knowledge bases - details are given in Table 1.

Table 1. Knowledge bases description

knowledge base	number of attribues	number of items	number of rules
winequality-red [8]	12	1599	427
winequality-white [9]	12	4899	154
diabetes [2]	9	769	567
diabetes_d [3]	20	10 000	2113
monkRses [5]	7	432	236
dataGeneric [1]	7	690	2635
satData [7]	7	4436	147784

Each dataset underwent a standard preprocessing procedure, including discretization, cleaning, and duplicate removal. Rules were then generated using the RSES (Rough Set Exploration System) tool with exhaustive algorithm, which generates all possible minimal sets of attributes (called reducts) that preserve the classification ability of the original attribute set. This is computationally expensive but guarantees completeness.

To evaluate the behavior of the inference control strategies-the primary focus of this study-five scenarios were randomly generated for five fact sets, representing 1%, 5%, 10%, 25%, and 50% of the possible attribute-value pairs. Subsequently, for each knowledge base and each of the four tested inference control strategies (random, recency, textual, and specificity), the following parameters were analyzed:

- inference success (True/False): Indicates whether the inference process successfully derived a fact related to the decision attribute. Unfortunately, we encountered some failures with one of the datasets, which required manual correction of missing data.

- inference time in seconds: The duration of the inference process, measured from start to finish for a single execution. This serves as an indicator of the strategy's efficiency.

- new facts: The number of new facts added to the fact base during inference. This excludes the initial facts provided by the scenario and includes only those inferred through rule application.
- used rules (Total): The total number of rules that were fired during the inference process for a given scenario and strategy.
- activated rules: The total number of rules included in conflict sets across all inference steps. For example, if 15 rules were in the conflict set at step 1 and 5 in step 2, the total would be 20.
- analyzed rules: The total number of condition evaluations performed on rules throughout the reasoning process. Since our implementation checks all unused rules at each step without optimization, this value reflects the overall computational effort required to process rule conditions.

3.1. Conflict set resolution strategies

We implemented and tested the following inference control strategies in the system:

- *random* strategy, which selects one of the applicable rules at random, with no preference or ordering.
- *textual order* strategy, which follows the static order of rule definitions in the rule base. The rule that appears first in the file is selected.
- *specificity* strategy, which prefers more detailed rules-those with a greater number of conditions (i.e., more restrictive) or those that involve a larger number of unique attributes in their premises. Rules with more conditions are considered more specific due to their narrower applicability, while those using more distinct attributes capture a broader range of information. This strategy prioritizes rules that are both restrictive and information-rich, selecting one at random from the group with the highest specificity.
- *recency* strategy, which favors rules whose conditions are satisfied by the most recently added facts. Each fact is assigned a "recency" tag corresponding to the inference step in which it was introduced (initial facts are assigned step 0, with subsequent steps numbered incrementally). For each rule, the highest recency value among its conditions is computed, and the rule with the highest such score is selected. If multiple rules share the highest recency score, the selection among them is made randomly [11].

3.2. Environment

The environment in which all experiments were implemented (the inference algorithm, all tested conflict set resolution strategies, experiment scenarios, as well as the entire pre-inference data preparation process for analysis were implemented) was Python 3.12.5. Dedicated libraries (pandas, NumPy) were also used [6].

4. Experiments

In the experiments we wanted to examine the influence of the applied inference conflict set resolution strategy on the inference time but also on the amount of new knowledge or the frequency of successful inference.

The presented results are the first stage of a larger-scale study. Ultimately, the rules in the knowledge base will be subject to a machine learning algorithm, specifically algorithms for clustering similar rules. A representative will be created for a group of rules and in the inference process, not individual rules will be analyzed as it is now, but groups of rules, or rather their representatives. In order to be able to check to what extent it will be possible to improve the efficiency of inference compared to the classical approach, both in terms of inference time and the frequency of successful inference, it was necessary to implement inference algorithms, taking into account various inference control strategies, for complex knowledge bases with a rule structure created automatically using the RSES tool. The results obtained in this way are presented in this paper. This allows for verification of which inference control strategies are associated with a shorter inference time, which more often end inference with success, which involve fewer rules as a result, which generate fewer new facts. All these factors contribute to the efficiency of inference. In the next stage of research, the rules will be grouped and then we will develop an inference algorithm that will analyze groups (representatives) of rules instead of individual rules. We will then compare the effectiveness of the classical approach with that based on rule clusters and answer the question of whether clustering the rules allows maintaining high inference efficiency in terms of successful inference while reducing the inference time by reviewing only group representatives instead of each rule separately.

4.1. Conflict set strategy vs. inference efficiency

The Table 2 presents the frequency of successful inference for each of the four tested strategies. We can see that for 3 strategies (recency, textual order and specificity) only 2 cases out of 175 in total ended in failure. We were also interested in how each strategy influenced the number of new

Table 2. Conflict Set resolution strategy vs inference efficiency

strategy	success	failure	new facts number (avg)	inference time [s]
random	175 (100%)	0 (0%)	2.257143	553.351
recency	173 (98.86%)	2 (1.14%)	2.085714	31.057
specificity	173(98.86%)	2 (1.14%)	3.045714	5667.516
textual	173 (98.86%)	2 (1.14%)	3.022857	1683.313

facts-that is, the amount of new knowledge-generated during the inference process. It turned out (Table 2) that the *recency* strategy produced the fewest new facts, likely because it completed the inference process in the shortest time. One of the most interesting-though perhaps expected-findings was the variation in reasoning time, which proved to be statistically significant across all four conflict resolution strategies analyzed. The *recency* strategy consistently resulted in the shortest inference time, followed by *random*, *textual*, and finally *specificity*. The differences in execution time were substantial, in some cases differing by several orders of magnitude.

4.2. Fact set size vs. inference success

The Table 3 presents the frequency of successful inferences across five scenarios, each with varying sizes of input fact sets $(1\%, 5\%, 10\%, 25\%, 50\%)$. It is evident that inference failures occurred only in the scenario where the input facts comprised just 1% of all possible facts. In this case, there were six failures, representing 4.29% of all experiments.

4.3. Conflict set strategy vs. number of analyzed. activated and used rules

In section 3, we introduced three distinct factors for rules that, while seemingly similar, each play a crucial role in accurately evaluating the effectiveness of a given inference approach: *used*

Table 3. Percent of facts vs inference efficiency

% of facts	success	failure
1	134 (95.71%)	6 (4.29%)
5	140 (100%)	0 (0%)
10	140 (100%)	0 (0%)
25	140 (100%)	0 (0%)
50	140 (100%)	0 (0%)

rules-the total number of rules that were fired during the inference process for a specific scenario and strategy, *activated rules*-the cumulative number of rules included in conflict sets across all inference steps. This metric reflects the overall activity level of the system during reasoning, and finally, *analyzed rules* - all unused rules are re-evaluated at each step without optimization, making this metric a direct measure of the computational workload involved in rule checking.

As shown in Table 4, when considering the number of rules actually used, the *random* strategy results in the fewest, followed by *recency*, *textual*, and *specificity*. However, when examining the number of analyzed or activated rules, the *recency* strategy clearly performs better, requiring significantly fewer rule evaluations and activations. The table presents the results for individual parameters, including mean values with standard deviations (denoted as avg ± std), and minimum and maximum values (denoted as (min; max)).

The chi2 test was used to demonstrate differences between the analyzed conflict resolution strategies. At the level of ($p< 0.05$) the presented strategies differ statistically significantly for all analyzed parameters (i.e. reasoning time, number of new facts, number of rules used, number of analyzed and activated rules). The meaning of the parameters: used rules, activated rules, analyzed rules and new facts is presented in the Section 3.

Table 4. Conflict set strategies' comparison

strategy	used rules	activated rules	analyzed rules	new facts
random (min;max)	5.14 ±5.95 (1;41)	60585.7 ± 192831.9 (60;1746640)	66593.4 ±205935.5 (153;1773342)	2.257143 ± 0.98 (1; 5)
recency (min;max)	19.68 ±163.67 (2;1548)	41202.8 ± 138781 (83;1198926)	51095.6 ±200840.4 (207;1799938)	2.09 ± 0.28 (2; 3)
specificity (min;max)	127.21 ±347.74 (1;1637)	315539± 748575 (83;4703208)	337966.7±777750.1 (207;4756541)	3.05± 0.90 (1 ;6)
textual order (min;max)	49.97 ± 191.19 (2;1548)	153964.8 ± 305268.4 (83;1537991)	175707.2±358136.7 (207;1799938)	3.02± 0.73 (2; 4)

5. Summary

Forward reasoning involves iteratively scanning previously unused rules and activating one that aligns with the selected inference conflict set resolution strategy. If the strategy chooses, for example, the first rule, the last rule, a random rule, or the one with mire conditions, it is evident that such a choice will influence inference time. However, our aim is to determine whether it also impacts other aspects of expert system efficiency that rely on inference algorithms. In this study, we focused exclusively on forward reasoning.

In our research, we examined four different inference conflict set resolution strategies (*ran-*

dom, *recency*, *textual order* and *specificity*) across seven diverse datasets. These datasets varied significantly in both size and domain. The smallest knowledge base contained just over 100 rules, while the largest comprised approximately 150,000 rules. The number of attributes used in the rules ranged from 7 to 20.

We evaluated the proposed inference conflict set resolution strategies based on their impact on the frequency of successful inference, inference time, and additional factors such as the number of new facts generated during reasoning. Our findings indicate that the strategy yielding the shortest inference time is the *recency* strategy, which prioritizes rules whose conditions are satisfied by the most recently added facts. This strategy also tends to generate fewer new facts and activates fewer rules compared to the others. Conversely, the strategy associated with the longest inference time is the *specificity* strategy-a result that aligns with existing literature on the subject, as discussed in [11].

We also confirmed that the number of input facts influences the frequency of successful inference. Inference failures occurred only when using the smallest set of input facts (1%), and even then, they accounted for fewer than 5% of all experiments.

In future work, we plan to explore how rule clustering-shifting from reasoning based on individual rules to reasoning based on clusters of similar rules-affects inference efficiency. To this end, we will experiment with various data clustering algorithms and different methods for representing rule clusters.

References

[1] datageneric.csv. `https://www.mimuw.edu.pl/ szczuka/rses/start.html` (2025), accessed March 10, 2025

[2] diabetes.csv. `https://www.mimuw.edu.pl/ szczuka/rses/start.html` (2025), accessed March 10, 2025

[3] diabetes_d.csv. `https://www.kaggle.com/datasets/marshalpatel3558/` (2025), accessed March 10, 2025

[4] Expert systems. `http://www.sci.brooklyn.cuny.edu/ dzhu/cis718/` (2025), accessed March 10, 2025

[5] monkrses.csv. `https://www.mimuw.edu.pl/ szczuka/rses/start.html` (2025), accessed March 10, 2025

[6] Python documentation. `https://docs.python.org/3.12/library/index.html` (2025), accessed March 15, 2025

[7] satdata.csv. `https://www.mimuw.edu.pl/ szczuka/rses/start.html` (2025), accessed March 10, 2025

[8] winequality-red.csv. `https://archive.ics.uci.edu/dataset/186/wine+quality` (2025), accessed March 10, 2025

[9] winequality-white.csv. `https://archive.ics.uci.edu/dataset/186/wine+quality` (2025), accessed March 10, 2025

[10] Buchanan, B.G., Shortliffe, E.H.: Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project. Addison-Wesley, Reading, MA. (1984)

[11] Jackson, P.: Introduction to Expert Systems. Addison-Wesley, Harlow, England (1998)

[12] Luger, G.F.: Artificial Intelligence: Structures and Strategies for Solving Complex Problems. Addison-Wesley, Boston (2005)