Assistance in the management of rule sets for rule-based expert systems

Magali Legast and Axel Legay
UCLouvain
Louvain-la-Neuve, Belgium
{magali.legast, axel.legay}@uclouvain.be

***Abstract*—Rule-based expert systems (RBES) use knowledge about a specific topic, represented as rules, in order to solve particular problems that would otherwise require a human expert. The creation and maintenance of the rule sets used come with several challenges in order to guarantee that they remain free of any error, which could reduce performances or lead to erroneous results.**

In this paper, we present a methodology to provide an automated assistance for domain experts creating and maintaining rules for RBES. This assistance takes the form of automated detection of relationships between rules that can lead to redundancies or conflicts. By reducing the weight borne by the human experts in the verification of the rules, it reduces the chance of errors, which helps increase the relevance of the rule set.

To complete the theoretical methodology, we have implemented a functional prototype allowing for the management of a rule set and the visual highlight of redundancies and potential conflicts.

Our approach, developed in the context of a case study, can be used for rules in any domain as long as they can be described in the same format. The approach can also be extended or modified to account for other types of relationships between rules.

***Index Terms*—rule-based expert system, rule set**

I. INTRODUCTION

Expert systems (ES) are computer applications that contain knowledge about a specific topic with which it solves particular problems that would otherwise require a human expert. Using this knowledge, it can reach a conclusion that can be given to the user [2]–[4], [6], [8], [11], [12].

There exist many kinds of ES, [11] considering eleven categories. Amongst them are fuzzy expert systems, neural networks, case-based reasoning and many others. In rule-based expert systems (RBES), the information is represented as a set of IF-THEN rules with an antecedent and a consequent. The rules are then used to perform operations on data, make inferences and reach a conclusion [4], [8], [11], [12]. Such systems are used in a wide variety of applications, such as disease diagnosis, energy optimization or firewalls. Many other examples of rule-based ES can be found in [14].

Since the result provided by the system depends on the knowledge encoded in its rules, its quality is highly dependent on the rules themselves. In many cases, those rules are written by domain experts. It can be a challenge for them to guarantee that the rule set is and remains free of errors that can at best reduce the performance of the ES or at worst lead to an erroneous result, as is further explained in Section II.

This work tackles the difficulty for humans to write and maintain a good rule set. Our contribution is an approach to assist them in the management of the sets through an automated detection of redundancies or potential conflicts. To do so, we have developed an efficient methodology to identify the different types of relationships between pairs of rules, using matrices and numerical representations. We have built on [5], which shows how to detect anomalies in firewall rule sets, to adapt that work for the identification of other types of relationships. The methodology is detailed in Section III.

We have also implemented a functional prototype, presented in Section IV, to demonstrate how our methodology could be implemented in practice. This prototype allows to create and maintain rule sets, using the assistance provided by the automatic identification of relationships between rules. By alleviating the weight of such verification, it can help domain experts to obtain more accurate rules.

We have applied this problem to a case study, described in Section II. Nonetheless, our approach is not limited to a specific field and can even be extended to identify other types of relationships depending on the needs of the application, as is discussed in Section V.

II. RULE-BASED EXPERT SYSTEMS FOR ENERGY OPTIMIZATION - A CASE STUDY

This research was motivated by a case study aiming at improving an RBES that provides recommendations to optimize energy production and consumption for WeSmart [16]. This company supports energy communities, which are communities in which participants can produce and consume energy together in a more autonomous and efficient way.

The rule-based system considered in this case study has a knowledge base with rules that link conditions on input energy data to an appropriate recommendation. Their construction is similar to many other rules, including those in firewalls [1], [7], [13], [15]. They can be viewed as having the form

$$\text{IF } \langle \textit{condition} \rangle \text{ THEN } \langle \textit{recommendation} \rangle$$

The condition is a set of attributes and their associated values that define a specific set of situations. It can be seen as a boolean expression that is $True$ when the situation considered is in the set described by the condition and $False$ otherwise. When a situation satisfies the condition of a certain rule, those situation and rule are said to match. The recommendations

DOI reference number: 10.18293/SEKE2023-189

TABLE I
EXAMPLE OF ENERGY RULE SET

	Recommendation	YearlyOfftake^a	OfftakeMorning	OfftakeAfternoon	MaxProd^b	InCommunity^c
0	Add 1 photovoltaic panel	$[350, \infty]$	$]-2, \infty]$	$]-2, \infty]$	False	$*$
1	Add 2 photovoltaic panels	$[700, \infty]$	$]-2, \infty]$	$]-2, \infty]$	False	$*$
2	Add 3 or more photovoltaic panels	$[1200, \infty]$	$]-2, \infty]$	$]-2, \infty]$	False	$*$
3	Run highly consuming devices during afternoon	$*$	$[0, \infty]$	$[-\infty, 0]$	$*$	$*$
4	Run highly consuming devices during morning	$*$	$[-\infty, 0]$	$[0, \infty]$	$*$	$*$
5	Add solar panels	$[350, \infty]$	$*$	$*$	False	$*$
6	Join energy community	$*$	$]0, \infty]$	$[-\infty, 0[$	$*$	False
7	Join energy community	$*$	$[-\infty, 0[$	$]0, \infty]$	$*$	False
8	Add maximal number of photovoltaic panels	$*$	$*$	$*$	False	True

^{a} Offtake: Energy consumed but not self-produced (kWh), ^{b}MaxProd: Whether the potential for energy production is fully used (e.g. roof filled with solar panels and no other production possibilities), ^{c} InCommunity: Whether the user is part of an energy community

are pieces of advice that can be given to a user in the specific situations described by the rule condition. An example of such rules is given in Table I, with the recommendation and the attributes of the condition. An unspecified value indicates the absence of constraint for that attribute and can be represented with a wildcard '$*$', as in [7]. For example, the 5th rule in Table I corresponds to IF $\langle$YearlyOfftake $\in [350, \infty]$, MaxProd $= False\rangle$ THEN $\langle Recommendation =$ "Add solar panels"$\rangle$.

The RBES checks the condition of each rule in the set against the situation given as input. It then gives as output all the recommendations associated with the matching rules. Several different recommendations can thus be aggregated in the same output. The ordering of the rules is not significant. The implementation of such an ES is trivial, we have thus not developed it further.

The relevance and accuracy of those rules is crucial since they have a direct impact on how useful the output recommendations will be. Like in many other applications, they are created manually by experts in the field. Those experts can face some challenges in order to guarantee that the rule set allows the ES to have the desired behavior. Besides the correctness of each rule, they also need to avoid unwanted contradictions or redundancies within the rule set, i.e. distinct rules that contain conflicting or identical information. This can be a quite tedious and error-prone task while the number of rules grows, as reported by WeSmart experts. This is thus a difficulty at the creation of the rule set, but also for its maintenance and the addition of new rules as the system and the knowledge evolve.

Since the rule creation and maintenance process is both of high importance and high error risk, the proposed solution is to assist the human experts in this task. This assistance takes the form of an automatic verification of the possibility of conflict or redundancy between each two rules within the set. The verification methodology is described in Section III. Identifying rules that can lead to these problems helps experts easily detect and fix a rule that would not yield the wanted result or would bring undesired redundancy. It thus reduces the chances of mistakes and helps with the obtainment of a more relevant rule set and the overall improvement of the RBES.

It is worth noting that some level of redundancy may be useful to make rules more understandable or reduce their number. Additionally, different recommendations may be complementary to one another, like "Add photovoltaic panels" and "Reduce use of energy at night", while others may be conflicting, like "Run highly consuming devices at night" and "Reduce use of energy at night". For these reasons, WeSmart requested for domain experts to have a strong manual control on the rules. So it was desired to have an automated detection of potential problems, but no automated correction.

III. IDENTIFICATION OF RELATIONSHIPS BETWEEN RULES

The unwanted behaviors mentioned in Section II can happen when the same situation matches two different rules. In such cases, there is redundancy if the two rules have the same recommendation and contradiction if their recommendations are conflicting. In order to detect those cases, we need to look at the relationships between each pair of rules. The relationship between two rules is considered regarding the relationships between the set of situations that can be matched by each of those rules. This is similar to the way firewall rules are considered regarding the set of packets they match [1].

In this section, we will define the different types of relationships considered for the case study in Subsection III-A, describe the matrices and numerical representations used to represent them in Subsection III-B, then describe in Subsection III-C how we can use those representations to detect connections. Finally, we show how to identify each type of connection in Subsection III-D and how the numerical encoding needs to be defined in Subsection III-E. Our contribution uses the work done on the relationships between firewalls rules in [5] and adapts it to other applications.

A. Relationship definitions

Two rules can either be ***disconnected***, if there can be no situation they both match, or ***connected***, if there exists at least one possible situation they can both match. The relationships between two rules, inspired by [1], [5], are defined below :

*a) **Disjunction/Disconnection**:* Two rules r and s are disjoint if the set of situations that are matched by both rules is empty. The values of at least one of their respective attributes are disjoint.

*b) **Equality**:* Two rules r and s are equal if all the situations matched by r are also matched by s and all the situations matched by s are also matched by r. The values of all of their respective attributes are equal.

*c) **Inclusion:*** A rule r is included in a rule s if all the situations matched by r are also matched by s and there exist situations that are matched by s but not by r. For all respective attributes, r's values are either a subset of or equal to s's values, with at least one attribute for which it is a subset.

*d) **Overlap:*** Two rules r and s overlap if there can exist at least one situation that is matched by r and not by s, at least one situation that is matched by s and not by r and at least one situation that is matched by both r and s. For all respective attributes, the values of r and the values of s can't be disjoint and at least one of the two following sufficient conditions must hold:

- There is at least one attribute for which the values of r are a subset of the values of s and at least one attribute for which the values of s are a subset of the values of r.
- There is at least one attribute for which the values of r overlap the values of s, meaning the intersection between the two set of values is not empty, not equal to the set of values of r and not equal to the set of values of s.

Equality, inclusion and overlap are different types of connection, while disjunction is the only type of disconnection. In addition to those relationships, two rules can also have the same or different recommendations.

Reference [5] considers those relationships (although named differently), with a different overlap definition that doesn't consider overlaps between attribute values. It also defines anomalies specific to firewall rules sets and develops the detection methodology for those anomalies. On the other hand, our approach directly considers the relationships, which are more general and can be further specified by indicating if the recommendations are equal or different for both rules. We thus adapt the approach in [5] to identify them.

B. Matrix representation and Inter-Difference Coding

We propose to represent a set of rules in a matrix S where the rows represent the n rules and the columns the $m+1$ fields. Those fields are the m attributes, preceded by the recommendation which is located in the first field. The element v_{ij} thus represents the value of the j^{th} field for the i^{th} rule.

Since a relationship between rules can be defined with regard to the relationships between their respective fields, the latter are represented in Inter-Difference Matrices (IDM). For each attribute, there is a corresponding IDM R that represents the relationships between the values of this attribute for each pair of rules. There is also one IDM for the recommendations. For a rule set of n rules with m attributes, there is thus $m+1$ IDM's of size n x n. Since the relationship between attribute values of rules R_i and R_j are reciprocal, R is a strictly upper triangular matrix. The information for a relationship between R_i and R_j is thus found in the entry (i,j) if $i<j$ and in the entry (j,i) if $j<i$.

Together, the IDM matrices can be considered as several layers of a 3D tensor, creating an IDM layers model. The 0^{th} layer is associated to the recommendation. The other layers, 1^{st} to m^{th}, correspond to each of the attributes. We can define an Inter-Difference Vector (IDV) R_{ij} that represents the relationship between the rules R_i and R_j, where $i<j$. Its elements are the elements (i,j) of each IDM, which includes the recommendation IDM. Its length is thus $(m+1)$. A visual representation of the IDM layer model and an example of vector R_{ij} can be seen in Figure 1.

The concepts of IDM, IDM layers model and IDV have been introduced in [5]. They have been adapted here, primarily to include the recommendations in the IDM layers model and in the IDV, which simplifies the mathematical representation of relationships, allowing the use of a unique vector to do so. Another change is the use of triangular matrices to avoid storing redundant data.

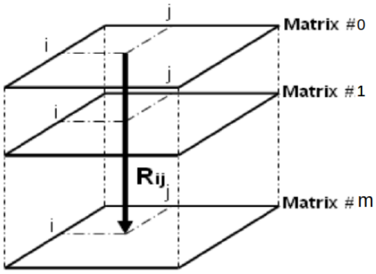


Fig. 1. IDM layers model, illustration taken from [5]

In the IDM's, the relationships between the respective attribute values or recommendations of two rules are encoded using a specific Inter-Difference Coding (IDC). The IDC associates a numerical value to each type of relationship, which will be used in the identification, as exposed in the following subsections. The IDC used for the case study is given in Table II alongside the definitions of the different possible relationships between respective fields. $\Re(v_{ik},v_{jk})$ corresponds to the relationship between R_i and R_j for the k^{th} field, which is represented in the k^{th} layer of the IDM layers model. Two unspecified values are considered to be equal, and a specified value to be a subset of an unspecified value. The use of an IDC was introduced in [5]. We have renamed the inclusions and added an extra value for the overlap.

TABLE II
RELATIONSHIPS BETWEEN ATTRIBUTE VALUES OF TWO RULES: DEFINITIONS AND IDC

$\Re(v_{ik},v_{jk})$ $1 \leq k \leq m$	Definition (relationships between attributes)	IDC code
Disjunction	$val(v_{ik}) \cap val(v_{jk}) = \phi$	0
Equality	$val(v_{ik}) = val(v_{jk})$	1
Inclusion ij	$val(v_{ik}) \subset val(v_{jk})$	2
Inclusion ji	$val(v_{ik}) \supset val(v_{jk})$	3
Overlap	$\begin{cases} val(v_{ik}) \cap val(v_{jk}) \neq \emptyset \\ val(v_{ik}) \not\subset val(v_{jk}) \\ val(v_{ik}) \not\supset val(v_{jk}) \end{cases}$	6

$\Re(v_{ik},v_{jk})$ $k=0$	Definition (relationships between recommendations)	IDC code
Difference	$val(v_{ik}) \neq val(v_{jk})$	-1
Equality	$val(v_{ik}) = val(v_{jk})$	1

C. Relationships between rules with regard to IDC

Thanks to the IDC in Table II, the relationship between rules are defined with regards to R_{ij}, adapting the work in [5] to the relationships defined above and the use of a unique vector to represent them. Regarding the conditions of the rules and their attributes, the relationships are defined as:

- **Disjunction :** $\exists\ x \in \{1, ..., m\},\ R_{ij}(x) = 0$
- **Equality :** $\forall\ x \in \{1, ..., m\},\ R_{ij}(x) = 1$
- **Inclusion of R_i in R_j :** $$\begin{cases} \exists\ x \in \{1, ..., m\},\ R_{ij}(x) = 2 \\ \forall\ e \in (\{1, ..., m\} \backslash \{x\}),\ R_{ij}(e) \in \{1, 2\} \end{cases}$$
- **Inclusion of R_j in R_i :** $$\begin{cases} \exists\ x \in \{1, ..., m\},\ R_{ij}(x) = 3 \\ \forall\ e \in (\{1, ..., m\} \backslash \{x\}),\ R_{ij}(e) \in \{1, 3\} \end{cases}$$
- **Overlap :**
  Either one of these conditions needs to be satisfied:
  $$\begin{cases} \exists\ x \in \{1, ..., m\},\ R_{ij}(x) = 6 \\ \forall\ e \in (\{1, ..., m\} \backslash \{x\}),\ R_{ij}(e) \in \{1, 2, 3, 6\} \end{cases}$$
  or
  $$\begin{cases} \exists\ x \in \{1, ..., m\},\ R_{ij}(x) = 2 \\ \exists\ y \in \{1, ..., m\},\ R_{ij}(y) = 3 \\ \forall\ e \in (\{1, ..., m\} \backslash \{x, y\}),\ R_{ij}(e) \in \{1, 2, 3, 6\} \end{cases}$$

To further specify the relationships between rules with regard to their recommendations, we have:

- **Same recommendation :** $R_{ij}(0) = 1$
- **Different recommendations :** $R_{ij}(0) = -1$

Those definitions allow to prove the existence of a connection between the rules R_i and R_j, with $i < j$, using the product p_{ij} of the elements of the IDV R_{ij} of the rules, defined as:

$$p_{ij} = \prod_{x=0}^{m} R_{ij}(x)$$

Theorem 1 : *A connection exists between rules R_i and R_j, with $i < j$, if and only if $p_{ij} \neq 0$.*

The proof, which follows a similar reasoning as in [5], is trivial and can be found in [10].

D. Identification of relationships using p_{ij}

The product p_{ij} not only allows to detect the existence of a connection, but also to identify the type of the relationship. Indeed, thanks to the definitions of Section III-C, we can express p_{ij} using the corresponding IDC values. The identification is then done using simple mathematical operations, such as `modulo`. Those p_{ij} values are stored in the Product Matrix P that is generated from the $(m+1)$ IDM's, with $P[i, j] = p_{ij}$ for every $i < j$. This takes inspiration from [5], while adapting the representations and conditions to the relationships defined in Section III-A.

The expression of p_{ij} with regard to the IDC values is given below for a selection of relationships taken as examples, as well as the condition on $P[i, j]$ that allows to identify them.

- **Disjunction :**
  - $p_{ij} = 0$
  - Condition : $P[i, j] = 0$
- **Inclusion of R_i in R_j, same recommendation:**
  - $p_{ij} = 1 \times (1^{(m-x)} \times 2^x)$
  - Condition: $\begin{cases} P[i, j] \ \texttt{mod} \ 6 \neq 0 \\ P[i, j] \ \texttt{mod} \ 2 = 0 \\ P[i, j] > 0 \end{cases}$
- **Overlap, different recommendations :**
  - $p_{ij} = -1 \times (1^{(m-x-y-z)} \times 2^x \times 3^y \times 6^z)$
  - Condition : $\begin{cases} P[i, j] \ \texttt{mod} \ 6 = 0 \\ P[i, j] < 0 \end{cases}$

with m the total number of attributes, x the number inclusions ij amongst the attributes, y the number of an inclusions ji and z the number of overlaps. The full list of expressions and conditions is available in [10].

E. Adaptation of IDC to relationships

The values in the IDC that represent the relationships between attribute values need to be chosen in order to facilitate the expression and identification of relationships between rules using simple mathematical operations.

The values for the difference between attribute values has to be 0, the absorbing element of multiplication, in order for the theorem to be true.

When a relationship between rules can be deduced from the presence of one kind of relationship between attributes, like for equality or inclusion, its IDC value has to be a prime number to allow the identification using modulo operations. The choice of 1, the identity element of multiplication, for the equality simplifies the identification conditions.

When a relationship between rules can be the result of different relationships between attributes, like for the overlap, taking it into consideration in the IDC code simplifies the identification condition. Hence, the IDC value for the overlap is the product of the values for inclusion ij and inclusion ji.

The values for the difference and equality between recommendations of course need to be of opposite sign, an absolute value of 1 being the obvious choice to simplify the identification conditions.

Following these principles, other IDC values can be chosen to represent other kinds of relationships.

IV. RELATIONSHIP IDENTIFICATION TOOL

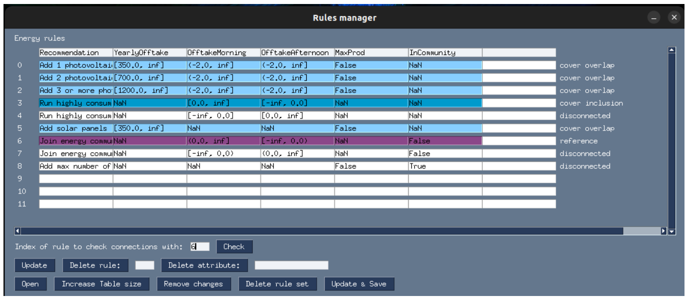
The methodology has been implemented in a functional prototype, which demonstrates how the theoretical methodology exposed in Section III can be implemented and used in practice. Its code is available at [9], alongside rule set examples, including those in Tables I and III. It has been written with Python 3.7 and is distributed under MIT license.

The Relationship Identification Tool (RIT) allows the user, supposedly a domain expert, to manage a rule set trough a graphical user interface (GUI), which can be seen in Figure 2. It highlights and indicates the different types of connections between rules, using the methodology described previously. It also supports the addition, modification and deletion of rules or attributes in order to create and maintain the rule set. The user

TABLE III

EXAMPLE OF ENERGY RULE SET AFTER CORRECTION USING THE RIT

	Recommendation	YearlyOfftake	OfftakeMorning	OfftakeAfternoon	MaxProd	InCommunity
0	Add 1 photovoltaic panel	$[350, 700[$	$]-2, \infty]$	$]-2, \infty]$	False	False
1	Add 2 photovoltaic panels	$[700, 1200[$	$]-2, \infty]$	$]-2, \infty]$	False	False
2	Add 3 or more photovoltaic panels	$[1200, \infty]$	$]-2, \infty]$	$]-2, \infty]$	False	False
3	Run highly consuming devices during afternoon	$*$	$]0, \infty]$	$[-\infty, 0[$	$*$	$*$
4	Run highly consuming devices during morning	$*$	$[-\infty, 0[$	$]0, \infty]$	$*$	$*$
5	Join energy community	$*$	$]0, \infty]$	$[-\infty, 0[$	$*$	False
6	Join energy community	$*$	$[-\infty, 0[$	$]0, \infty]$	$*$	False
7	Add maximal number of photovoltaic panels	$*$	$*$	$*$	False	True



	Recommendation	YearlyOfftake	OfftakeMorning	OfftakeAfternoon	MaxProd	InCommunity		
0	Add 1 photovoltai	[350.0, inf]	(-2.0, inf]	(-2.0, inf]	False	NaN		cover overlap
1	Add 2 photovoltai	[700.0, inf]	(-2.0, inf]	(-2.0, inf]	False	NaN		cover overlap
2	Add 3 or more pho	[1200.0, inf]	(-2.0, inf]	(-2.0, inf]	False	NaN		cover overlap
3	Run highly consum	NaN	[0.0, inf]	[-inf, 0.0]	NaN	NaN		cover inclusion
4	Run highly consum	NaN	[-inf, 0.0]	[0.0, inf]	NaN	NaN		disconnected
5	Add solar panels	[350.0, inf]	NaN	NaN	False	NaN		cover overlap
6	Join energy commu	NaN	(0.0, inf]	[-inf, 0.0)	NaN	False		reference
7	Join energy commu	NaN	[-inf, 0.0)	(0.0, inf]	NaN	False		disconnected
8	Add max number of	NaN	NaN	NaN	False	True		disconnected
9								
10								
11								

Fig. 2. RIT highlighting relationships between rule 6 and each other rule

can thus successively analyze them and their relationships, then modify the rule set until the intended behavior is obtained.

We tested the tool with a simplified and modified version of WeSmart rule set [1] which contain interesting test examples. We started with the rule set presented in Table I, which is composed of 9 rules, thus involving 36 relationships. With the tool, we can select each rule to view the relationships it has with each of the other rules, as is done for rule 6 in Figure 2. For the complete rule set, there is 23 overlaps and 8 cover inclusions. We detected that 5 of these overlaps and 6 of these inclusions involved conflicting recommendations, thus bringing unwanted behavior. To fix this rule set, we deleted rule 5 which was conflicting with 4 others (rules 0, 1, 2 with inclusions and rule 8 with overlap). We corrected attribute values for rules 3 and 4 to remove their overlap. This was also done for the 3 first rules so they wouldn't be included within each other any more and to avoid the overlap with rule 8. As a result, there remain the 8 rules (24 relationships) listed in Table III, out of which the 5 firsts had some attribute values updated. Between those rules, there are 18 overlaps and 2 inclusions, all desired in order to provide complementary recommendations when applicable. More detail about the correction process in relation with the methodology can be found in [10].

While we tested the tool with the energy rules of the case study, it can be used for rules of any domain. Its implementation can also be easily extended to account for other relationship types between rules. Further, the `RuleSet` class, which allows for the management of the ruleset objects, is independent from the GUI and can thus be reused as is in other implementations.

V. DISCUSSION AND FUTURE WORK

The methodology presented provides an efficient way to assist in the management of a rule set and the relationships between its rules. For a new set, the time complexity to identify all the relationships between pairs of rules is quadratic in the number of rules and linear in the number of attributes. The matrices used also allow to store those relationships. They can thus be updated in linear time when a rule is added or modified, which is particularly interesting for the maintenance of the rule set. New attributes can be added with the update of the matrices in quadratic time in the number of rules. All those complexities simply follow from the way the different matrices are built.

Our methodology, while developed for energy rules in a specific context, is generic and can be used as is to identify the relationships defined in Section III-A for rules of any domain. Furthermore, our solution and the relationships considered, which already adapt [5] that was designed for firewall rules, can also be modified and extended to identify other types of relationships depending on the need of a particular application. Indeed, other relationships between attributes can be represented with the IDC and lead to the identification of new types of relationships between rules, following the same process.

Beyond that extension, it would also be interesting to study the possibility to consider related attributes. Indeed, the methodology considers all attributes independently from each other. They can however be linked, like the average daily energy consumption and the average consumption over a certain period of time during the day. Such situation can lead to more connections between rules being detected than if those relationships between distinct attributes had been taken into account. This is thus a current limitation of the tool. Considering how the attributes may interact would give another insight into the relationships between the rules and give more precise results.

Another interesting enhancement would be to provide users with recommendation on potential correction, in addition to the display of relationships type. For example, the system could give more information on the potential conflicts between rules. With the current solution, the user has to determine whether the recommendations of two connected rules are

---

[1] For confidentiality reasons, the actual rules and recommendations used in WeSmart system cannot be presented here. Therefore, all examples presented in this article have been created by the authors for illustration purpose. They remain plausible and useful to the discussion.

complementary or in conflict. It would thus be useful to identify recommendations that are always in conflict with one another and recommendations that are always complementary. This information could then be included in the feedback given to the user and would provide an extra assistance compared to the current solution.

It would also be useful to build on this model to add an automated correction of some or all conflicts and redundancies, depending on the specificity of the domain of interest and the RBES in which the rule set is used.

Lastly, more experiments on larger and more diverse rule sets would be interesting to better quantify the impact of our approach. Key points to study include the measured improvement of accuracy of the rule set and its RBES, the number of rules in the sets and the gain in time and quality of experience for the domain experts during the creation and maintenance process.

VI. CONCLUSION

In this work, we have considered a case study to improve an existing RBES used by a company in the field of energy communities. One of the key elements for the accuracy and efficiency of such systems is the quality of the set of rules in which its knowledge is contained. The creation and maintenance of those rules, typically handled by domain experts, can be a tedious process with high chances of errors.

In response to this challenge, we developed a methodology to assist domain experts in the management of such rule sets. This assistance takes the form of a tool that automatically verifies if there is a possibility of conflict or redundancy between each two rules within the set. By highlighting rules that can lead to these problems, it helps experts easily detect and fix a rule that would not yield the wanted result or that brings undesired redundancy. By reducing the weight of the verification borne by the domain experts, it reduces the chances of mistakes and helps with the obtainment of a more relevant rule set and with the improvement of the RBES.

The proposed methodology relies on matrices and basic mathematical operations, which makes it efficient and easy to implement. While it has been developed for a specific case study, it can be applied to any rule set with the same redundancy and conflict concerns. It can also be easily adapted and extended to detect other types of relationships between rules.

Beyond the theoretical presentation of the methodology, we have implemented and tested it with a functional prototype which code is available at [9].

REFERENCES

[1] M. Abedin, S. Nessa, L. Khan, and B. Thuraisingham, “Detection and resolution of anomalies in firewall policy rules,” in *Data and Applications Security XX*, ser. Lecture Notes in Computer Science, Berlin, Heidelberg: Springer, 2006, pp. 15–29.

[2] S. Abu Naser, “SL5 object: Simpler level 5 object expert system language,” *The International Journal of Soft Computing, Mathematics and Control*, vol. 4, no. 4, pp. 25–37, 2015.

[3] S. Abu-Naser, R. Aldahdooh, A. Mushtaha, and M. El-Naffar, “Knowledge management in ESMDA: Expert system for medical diagnostic assistance,” *ICGST International Journal on Artificial Intelligence and Machine Learning, AIML*, vol. 10, pp. 31–40, 2010.

[4] M. Adams, F. Gaillard, and M. A, *Rule-based expert systems*. Radiopaedia.org, 2017.

[5] A. Bouhoula, Z. Trabelsi, E. Barka, and M. A. Benelbahri, “Firewall filtering rules analysis for anomalies detection,” *International Journal of Security and Networks*, vol. 3, no. 3, p. 161, 2008.

[6] J. Durkin, *Expert systems: design and development*. Macmillan, 1994.

[7] E. W. Fulp, “Optimization of network firewall policies using directed acyclical graphs,” in *Proceedings of the IEEE Internet Management Conference*, Nice, France: IEEE, 2005, p. 4.

[8] C. Grosan and A. Abraham, “Rule-based expert systems,” in *Intelligent Systems: A Modern Approach*, ser. Intelligent Systems Reference Library, Berlin, Heidelberg: Springer, 2011, pp. 149–185.

[9] M. Legast. “Relationship identification tool.” (2021), [Online]. Available: https://gitlab.com/legastm/Relationship-Identification-Tool.

[10] M. Legast, “Rule-based expert system for energy optimization : Detection and identification of relationships between rules in knowledge base,” Master’s thesis, UCLouvain, 2021.

[11] S.-H. Liao, “Expert system methodologies and applications—a decade review from 1995 to 2004,” *Expert Systems with Applications*, vol. 28, no. 1, 2005.

[12] A. Ligêza, *Logical Foundations for Rule-Based Systems* (Studies in Computational Intelligence), red. by J. Kacprzyk. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, vol. 11.

[13] A. X. Liu and M. G. Gouda, “Complete redundancy detection in firewalls,” in *Data and Applications Security XIX*, ser. Lecture Notes in Computer Science, Berlin, Heidelberg: Springer, 2005, pp. 193–206.

[14] N. Masri, Y. A. Sultan, A. N. Akkila, *et al.*, “Survey of rule-based systems,” *International Journal of Academic Information Systems Research (IJAISR)*, vol. 3, no. 7, pp. 1–23, 2019.

[15] A. Papagrigoriou, P. Petrakis, and M. Grammatikakis, “A firewall module resolving rules consistency,” in *13th Workshop on Intelligent Solutions in Embedded Systems (WISES)*, Hamburg: IEEE, 2017, pp. 47–50.

[16] WeSmart. “WeSmart,” WeSmart. (2023), [Online]. Available: https://wesmart.com/en/ (visited on 03/10/2023).