AN ANALYSIS OF RULES-BASED SYSTEMS TO IMPROVE SWRL TOOLS

Adriano Rivolli, João Paulo Orlando and Dilvan A. Moreira
Dept. of Computer Science, ICMC - USP, Av. Trabalhador Sancarlense, 400, São Carlos, Brazil

Keywords: Rule acquisition, Rule visualization, Rule management, OWL, SWRL.

Abstract: The Semantic Web renewed a growing interest in rule based software systems and their development. Semantic Web Rule Language (SWRL) is a rule language that enables Horn-like rules to be combined with Web Ontology Language (OWL) knowledge bases to provide even more expressivity. However, as rule based web system mature, the number of rules they use grows making them difficult to manage. Developers face problems when trying to understand and control the big rule sets they create. In order to address this problem, techniques and tools are necessary to organize, view and create new rules as part of a large rule set. This work presents strategies and techniques developed in order to improve SWRL tools based upon a survey of rule tools, a study of the state of the art and the analysis of representative rule sets.

1 INTRODUCTION

The use of rules in the Semantic Web has grown and contributed to renew and increase interest in rule based software systems (Zacharias, 2008). The Semantic Web Rule Language (SWRL) provides even more expressivity to the Web Ontology Language (OWL), which is a powerful language for building ontologies that specify high-level descriptions of Web content (SWRL Submission, 2004).

With the growing use of rules in the Semantic Web, users and developers have encountered some problems, particularly when the rule set becomes large or the rules are complex (Hassanpour, O'Connor and Das, 2009). Thus, they need tools for the creation, management and visualization of rules allowing: knowledge acquisition without ambiguity, inconsistency and rule duplication; and rule (and rule set) visualization that improves the understanding of knowledge content.

Protégé is the most widely used freely available, platform-independent technology for developing and managing ontologies (Rubin, Noy and Musen, 2007). We take it as a representation of the state-of-the-art in such tools. It uses two tabs for writing SWRL rules: the SWRLTab (O'Connor, Musen and Das, 2009) and Axiomé (Hassanpour, O'Connor and Das, 2009). Both tools emphasize rule visualization. However, their use has shown that support for large SWRL rule bases is deficient and have to be improved.

At the same time, there is a considerable number of Business Rules System (BRS) that were built to solve problems similar to the ones SWRL rule systems face today. Unfortunately, there is little research data about how rule systems are being designed, written and debugged; and what challenges their rule developers face (Zacharias, 2008).

This work is composed by: A survey of rule tools, their main features and user interfaces; The study of the state of the art related to SWRL; and analysis of the rule set characteristics.

2 SWRL RULES

The SWRL format is a simple Horn-like rule structure that combines with an OWL knowledge base to provide more expressivity. Each rule consists of two parts: the antecedent (body) and the consequent (head) that are formed by zero or more atoms. Atoms, in these rules, can be of the form C(x), P(x,y), sameAs(x,y) or differentFrom(x,y), where C is an OWL description, P is an OWL property, and x,y are variables, OWL individuals or OWL data values (SWRL Submission, 2004). The W3C Submission defines six SWRL atom types: class; individual property; data valued property; data

Rivolli A., Paulo Orlando J. and A. Moreira D..
AN ANALYSIS OF RULES-BASED SYSTEMS TO IMPROVE SWRL TOOLS.
DOI: 10.5220/0003439901910194
In *Proceedings of the 13th International Conference on Enterprise Information Systems* (ICEIS-2011), pages 191-194
ISBN: 978-989-8425-56-0

range; same/different; and builtins (Hassanpour, O'Connor and Das, 2009).

Atoms may refer to individuals, data literals, individual variables or data variables. Each SWRL atom type supports a number of arguments and types. Finally, the human readable variables are indicated using the standard convention of prefixing them with a question mark. Using this syntax, a rule would be written like this (SWRL Submission, 2004):

```
parent(?x,?y) ∧ brother(?y,?z) ->
uncle(?x,?z)
```

3 RULE TOOLS

Nowadays, most enterprise systems include a module or use a program to write business rules. A rule, in general, is a declarative instruction that expresses "what" and not "how" things must happen (Braye et al., 2006). Bellow we list and cluster the rule tools surveyed for this paper:

BRS RuleTrack, FICO Blaze Advisor, ILOG JRules BRMS, Oracle Business Rules and **Visual Rules** provide a complete environment for managing business rules that allow the capture and organization of business rule statements.

Drools introduces the "Business Logic integration Platform" which provides a unified and integrated platform for rules. The **LibRT Rule Management System** enables business experts to maintain and test rules in different representation formats. **OpenLexicon** is a Business Rules Engine. The **SAP NetWeaver BRM** component provides support for the various phases of a rule system life cycle: design, execution, modification and optimization of business rules.

RuleXpress is a repository-based tool that can be used to manage vocabulary and rules. This tool is built for business people and business analysts using their vocabulary. It uses the **BRS RuleSpeak,** which is a set of practical guidelines mainly expressing rules in clear, unambiguous, well-structured business English.

The **TRANSlator from LAnguage TO Rules (TRANSLATOR)** allows anyone, even non-experts, to write facts and rules in a formal representation for use in the Semantic Web. This is accomplished by automatically translating natural language sentences written in Attempto Controlled English.

CLIPS is an interpreted language and expert system tool that allows the definition of facts and rules to which functions are applied.

Also included, are the SWRL tabs of Protégé, **Axiomé** and **SWRLTab**. The first has some functionality for visualization, acquisition, browsing and exploring of SWRL rule bases and relationships. The second supports edition and execution of SWRL rules.

Apart from rule tools, there are also notations, like the **Object Rule Modeling (ORM)** a graphic notation, enabling the creation of rules with diagrams. There are different tools to design, maintain and execute rules written in ORM.

4 RULE INTERFACES

During the review of the rule tools, observations were made regarding the interfaces, features and resources used in these tools. In general, the interfaces and interaction approaches are: Text editor; Integrated Development Environment (IDE); Descriptive user-friendly; Graphical editor; UML-based; Spreadsheets; Tree editor; Combination with ontologies; Automatic extraction from data (Zacharias, 2008; SWRL Submission, 2004).

However, the survey also shows more specific features and interfaces.

4.1 Interfaces

Decision tables are composed by rows and columns of rules. They are used to display in tabular form all possible situations that a decision rule might encounter and to specify what actions to take in each of these situations. The key point to keep in mind is that, in a decision table, each row is a rule, and each column in that row is either a condition or an action for that rule.

Decision trees provide the same functionality as decision tables, but are composed of nodes instead of rows and columns. In a decision tree, each rule is represented by the set of nodes going from the tree root to each leaf.

Rule templates allow rule designers to write rule logic (or structure) once and reuse it many times. The main use of this kind of interface is to acquire new rules.

From the tool user viewpoint, **natural language** is the simplest interface for rules. The use of natural language is mainly restricted to rule visualization and not acquisition.

Diagrams can represent rules and rule sets, providing the users ways to create and view rules. The graphic representation can be as expressive as

Entity Relationship (ER) and **Unified Modelling Language (UML)** graphic notations.

A **Rule Flow** uses a Graphic Diagram to show a Decision Tree. It uses geometric forms, colours and symbols to represent parts of a rule.

Text editors are used by technical users that write rules like programmers, in other words, the user has to know the rule language syntax. Some editors are like notepads, others are more sophisticated as an **IDE**.

A **shell** is a piece of software that provides a user interface that accepts and executes commands. It supplies a command line interface that may be used interactively or non-interactively.

4.2 Components and Features

Rule grouping is common in most rule tools. It contributes for the organization and classification of rules. There are two basic types: manual groups (defined by the rule creator) and automatic groups (defined by an algorithm).

Some rule-based systems allow naming and set textual description of rules, so users can manage them more efficiently. Those descriptions can make use of lists, make searches and filter more accurately, especially with a large rule set. Breaking rules in parts (rule segments or atoms) can ease visualization and acquisition.

Finally, icons and symbols are used to represent tool features and assist rule development. Almost all systems use icons and symbols in their interface as a mean to access its features and functions. Icons and symbols are applied in two distinct moments: to provide access to system functionality or specific tasks; and, to be directly used in rule visualization and acquisition.

5 PRELIMINARY RESULTS

5.1 Rule Analysis

We chose the Autism Rules Phenolog ontology (Tu et al., 2008) as a representative of a large and complex SWRL rule system. This ontology of autism extends some ontologies available in the Open Biomedical Ontologies (OBO) Foundry using a combination of description logic and rules. The analysis of this rule system gave us a better understanding of the structure of its parts, atoms and variables. Table 1 presents the general data about the 156 rules analyzed.

Table 1: Overview of Autism Rules Phenolog.

Information	Total	Antecedent	Consequent
Number of atoms	2094	994	1100
Number of distinct predicate atoms	155	147	9
Average of atoms by number of rules	13	6	7
Min and Max number of atoms	10, 20	5, 15	4, 8
Number of arguments	3826	1877	1949
Number of distinct arguments	1304	997	1007
Min and Max number of arguments	18, 46	25, 37	25, 20
Min and Max number of distinct arguments	7, 18	5, 16	4, 10

This analysis also includes the frequency and distribution of predicates, atoms, atom types and argument types in rule parts (antecedents and consequents) as well as in rules as a whole, which allowed us to discover features of this rule set:

- Few predicate atoms occur in many rules while the vast majority occurs in less than 10 rules as observed;
- Only three predicate atoms occur more than once in the same rule;
- Only one predicate appears on both sides of a rule;
- Every rule analyzed does not contain same/different or data range atom types, individual property atoms are not applied to antecedents and built-in types are not applied to consequents;
- Every rule in the rule set contains, at least, two class atoms, one individual property in the consequent, five data valued property and one built-in atom;
- The main argument types, used in the rules, are Individual and data variables.

This data is being used to direct some of the interfaces discussed in the next section.

5.2 Techniques, Strategies and Services

Although this work is still in progress, we have achieved some results that are being carefully evaluated and will be implemented in SWRL rule tools. The tools SWRLTab and Axiomé tabs are our starting point to propose improvements, new techniques and features to support the SWRL language.

A simple and useful feature, not used in both tabs, is SWRL Highlights. Currently SWRL rules

are presented in simple text format, however, the use of distinct colors to represent variables, data values and distinct types of atoms have been shown to be very useful.

We also propose an auto-suggest feature during the rule composition process. For that, we developed an algorithm to determine the number of times each predicate is related to another in a large rule set. To do this, each predicate is mapped to a node in a graph with edges connecting the predicates that appear in the same rules. For each rule, all its predicates are connected and counted. The auto-suggest is based on the frequency that predicates are related in the rules. When a user adds atoms to a rule under construction, related atoms are suggested based on this algorithm.

We are experimentally using Euclidian and Manhattan distances (Salzberg, 1991) among the rules with the aim of measuring rule similarity and then group them based on it. We are using distinct scenarios: using antecedent or consequent rule parts; using both at the same time; and switching among atoms and predicates (without variables).

The atoms/predicates form the columns in a feature array and the rules/rule parts are the rows. This feature array indicates how many times an atom/predicate occurs in a rule/rule part. This technique proved very efficient and useful for the tested rule-based systems. It allows the discovery of very similar or identical rules in a rule set and it also finds rules similar to a given rule.

Finally, the rule similarity values were applied in the development of a K-means (Jain, Murty and Flynn, 1999) clustering method in order to group the rules by similarities. With it, it is possible to determine the number of groups and subdivide them to get more closely related rule groups. Initial tests demonstrate that the formed groups contain rules of different sizes and with different atoms, what is good. However, the K-means method can classify the same rule in different groups and the insertion of a new rule in the rule set requires a new classification and therefore rearrangement. To remedy this problem, we are studying the development of other clustering methods.

6 CONCLUSIONS

The results obtained so far are good and they have shown promising improvements in the creation, visualization and maintenance of SWRL Rules. We have been conducting studies in an attempt to use restricted natural language and the next steps are the development of tools that integrate these new interfaces in a SWRL tab for Protégé.

ACKNOWLEDGEMENTS

This work has been funded by a grant from CNPq-Brazil.

REFERENCES

Braye, L., Ramel, S., Grégoire, B., Leidner, S., Schmitt, M., 2006. State of the Art Business Rules Languages. Public Research Centre Henri Tudor, http://efficient.citi.tudor.lu/cms/efficient/content.nsf/0/4A938852840437F2C12573950056F7A9/$file/BusinessRulesLanguages_D3.1.pdf.

Hassanpour, S., O'Connor, M. J., Das, A. K., 2009. Exploration of SWRL Rule Bases through Visualization, Paraphrasing, and Categorization of Rules. In *RuleML*, pp. 246–261, doi: 10.1007/978-3-642-04985-9_23.

Jain, A. K., Murty, M. N., Flynn, P. J., 1999. Data clustering: A review. In *ACM Computer Survey 31*, 3, pp. 264-323, doi: 10.1145/331499.331.

O'Connor, M. J., Musen, M. A., Das, A. K., 2009. Using the Semantic Web Rule Language in the Development of Ontology-Driven Applications. In *Handbook of Research on Emerging Rule-Based Languages and Technologies: Open Solutions and Approaches, ch. XXII*, pp. 525-539.

Rubin, D. L., Noy, N. F., Musen, M. A., 2007. Protégé: A Tool for Managing and Using Terminology in Radiology Applications. In *Journal of Digital Imaging*, pp 34–46, doi: 10.1007/s10278-007-9065-0.

Salzberg, S., 1991. Distance Metrics for Instance-Based Learning. In *Proceedings of ISMIS'916th International Symposium, Methodolo-gies for Intelligent Systems*, pp. 399-408.

SWRL Submission, 2004. http://www.w3.org/Submission/SWRL.

Tu, S., Tennakoon, L., O'Connor, M. J., Shankar, R., Das, A. K., 2008. Using an integrated ontology and information model for querying and reasoning about phenotypes: the case of autism. In *Proceedings of the American Medical Informatics Association*, pp. 727–731.

Zacharias, V., 2008. Development and verification of rule based systems – a survey of developers. In *Rule Representation, Interchange and Reasoning on the Web: International Symposium*, pp. 6-16, doi: 10.1007/978-3-540-88808-6_4.