The Performance of Rule Identification from Web Pages

Juyoung Kang
Korea Advanced Institute of Science and Technology

Jae-Nam Lee
Korea Advanced Institute of Science and Technology

Follow this and additional works at: http://aisel.aisnet.org/icis2004

Recommended Citation

This material is brought to you by the International Conference on Information Systems (ICIS) at AIS Electronic Library (AISeL). It has been accepted for inclusion in ICIS 2004 Proceedings by an authorized administrator of AIS Electronic Library (AISeL). For more information, please contact elibrary@aisnet.org.
THE PERFORMANCE OF RULE IDENTIFICATION FROM WEB PAGES

Juyoung Kang and Jae Kyu Lee
Graduate School of Management
Korea Advanced Institute of Science and Technology
Seoul, Korea
jykang@kgsm.kaist.ac.kr  jklee@kgsm.kaist.ac.kr

Abstract
In the world of Web pages, there are oceans of documents in natural language texts and tables. To extract rules from Web pages and maintain consistency between them, we have developed the framework of XRML (eXtensible Rule Markup Language). XRML allows the identification of rules on Web pages and generates the identified rules automatically. For this purpose, we have designed the Rule Identification Markup Language (RIML), which is similar to the formal Rule Structure Markup Language (RSML), both parts of XRML. RIML is designed to identify rules not only from texts, but also from tables on Web pages, and to transform to the formal rules in RSML syntax automatically. While designing RIML, we considered the features of sharing variables and values, omitted terms, and synonyms. Using these features, rules can be identified or changed once, automatically generating their corresponding RSML rules.

We have conducted an experiment to evaluate the effect of the RIML approach with real-world Web pages of Amazon.com, BarnesandNoble.com, and Powells.com. We found that 97.7 percent of the rules can be detected on the Web pages, and the completeness of generated rule components is 88.5 percent. This is good proof that XRML can facilitate the extraction and maintenance of rules from Web pages while building expert systems in the Semantic Web environment.

Keywords: Rule identification, rule acquisition, knowledge engineering, knowledge acquisition, XRML, RuleML, XML

Introduction
Web technology was developed to provide a common browsing platform for human comprehension. As the next step for making the Web intelligent, Semantic Web research (Miller et al. 2001) attempts to extract data and rules from Web pages. In order to formally represent the data structure, XML is widely adopted as the basic platform. Moreover, to formally represent the rules, rule markup language research is widely undertaken (RuleML 2003). The primary focus of rule markup language research is to represent rules with appropriate tags, imposing logical inference capability over the markup rules.

A critical issue in adopting rule markup language is the acquisition and maintenance of formal rules from Web pages where the rules are expressed in natural language texts and tables. Concerning rule acquisition, Hulth et al. (2001) pointed out that direct conversion of natural language text to formal representation is extremely complex, so the intervention of a knowledge engineer is unavoidable (Schmidt and Wetter 1998). To handle this issue, we propose the eXtensible Rule Markup Language (XRML) approach as a framework of extracting rules from texts and tables. XRML consists of three components as depicted in Figure 1: Rule Identification Markup Language (RIML), Rule Structure Markup Language (RSML) and Rule Triggering Markup Language (RTML) (Lee and Sohn 2003). RIML identifies the rules implicitly expressed in Web pages; RSML represents the formal rule structure which corresponds to the rule syntax in commercial rule-based systems; and RTML defines the conditions that trigger the inference of certain rules. The process of rule acquisition using XRML is depicted in Figure 1 with the step numbers in parentheses.
In the traditional rule-based system, rules are extracted from their sources without tracing the formal association between them. However, in the XRML environment, we can identify the rules on the Web pages first using RIML, and then transform them to the RSML form. In this way, a fundamental benefit of using RIML is that nearly complete formal rules can be specified on the Web pages so that the identified rules in RIML can be associated with the formal rules in RSML. When either the Web pages or the rule base is changed, the association knowledge can trace its counterpart so that both knowledges can be altered to maintain consistency.

The first stage of XRML 1.0 research has naturally focused on the design of tags that annotate the existence of rules, variables, and values in the texts. In this paper, as the second step of this research, XRML 2.0 investigates the integrated extraction of rules both from natural language texts and from tables. For instance, to compute the delivery cost of items purchased from online shops like Amazon.com, customers need to look at both the relevant texts and tables together to derive a conclusion. In this process, we experience situations where some terms are not explicitly stated (thus omitted because readers can implicitly understand without the terms), some terms used as variables are shared by many other terms whose role is values, and some terms are expressed by other synonyms or pronouns.

So the first objective of this research is the design of XRML 2.0 (which consists of RIML 2.0 and RSML 2.0) considering these features (see the issues section of this paper). We excluded the description on RTML 2.0 here because it is not directly related to the scope of this paper.

The second objective of this research is whether XRML 2.0 can perform well with real-world Web pages. So we have applied the XRML framework to three Web sites of typical online bookstores (Amazon.com, BarnesandNoble.com, and Powells.com) with 36 Web pages in total. The rule identification and generation process is demonstrated with Amazon.com (see the the XRML approach section of this paper).

While we identify the rules from the Web pages, we noticed that we could not identify all of the rules if the pages did not have all the knowledge necessary for the rule base. For instance, the linkages between rules and inferential conclusions are not explicitly expressed in the Web pages. Sometimes, identification on the Web pages is not efficient because the complex numeric expressions and logical relationship are not easily expressed in markup language. In this case, it is desirable to make identification roughly, postponing the precise specification to the rule refinement stage with RSML. Thus, analyzing this issue is the third objective of this research (see the postponement strategy section of this paper).
The fourth objective of this research is the evaluation of the effectiveness and efficiency of the XRML approach. For this purpose, we need to define the measures for effectiveness and efficiency and conduct an experiment. The measures are defined in the performance section, and the experiment is conducted with the 36 Web pages of the 3 online bookstores.

Since rule identification is similar to rule extraction from the natural language, we have reviewed the relevant literature on knowledge acquisition, natural language processing, and machine learning, and discussed their relationship with XRML. We have also contrasted XRML with the other rule markup languages (see the next section).

There are other important issues such as assisting the identification of RIML and refinement of RSML. These issues and limitations will be discussed in the concluding section.

Review on Rule Markup Languages and Knowledge Acquisition

To understand the role of XRML, we review the literature on rule markup languages and knowledge acquisition, and contrast them with the distinctive goals of XRML.

Rule Markup Languages

Recently, there have been many studies on rule markup languages. The RuleML initiative (RuleML 2003) organized the online resources about rule markup languages. The primary purpose of rule markup languages is to express the logical rules with the annotating tags of meta-knowledge, thereby expanding the rules’ expressive power. Typical rule markup standards include DARPA Agent Markup Language (DAML)—rules (Grosof 2002), predictive model markup language (Grossman et al. 2003), attribute grammars’ semantic rules (Psaila and Crespi-Reghizzi 1999), and mathematical markup language (Carlisle et al. 2003). The rule markup languages are also easily compatible with ontology languages like RDF (Lassila and Swick 1999), DAML+OIL (Horrocks 2002), and OWL (Horrocks et al. 2003), because they are also based on XML.

RuleML has attempted to integrate all necessary features into a standard platform. However, it still does not cover the issue of extracting the markup rules from the Web pages, as the XRML approach attempts. Thus the key distinction of XRML with other rule markup languages is that XRML supports the rule identification stage on the Web pages to assist rule generation from the natural Web pages. This framework is also useful to assure the consistency between Web pages and rules during the maintenance stage, because XRML can pinpoint the position of a counterpart when one side of Web pages or rules is changed.

Knowledge Acquisition

The process of extracting rules from Web pages resembles the knowledge acquisition process from natural language. So we need to review the literature on expert’s diagram, natural language processing, and machine learning. Then we can describe their relationship with the XRML approach more clearly.

Expert’s Diagram Approach

Knowledge acquisition has been an everlasting bottleneck in building expert systems. To help domain experts such as lawyers in structuring domain knowledge, the expert’s diagram (Lee et al. 1990), which can be used by experts without the aid of knowledge engineers, is widely utilized. The conceptual graph (Amati and Ounis 2000), decision table (Seagle and Duchessi 1995), and influence diagram (Boose et al. 1993; McGovern et al. 1991) belong to this category. These diagrams aim at assisting experts to explicitly represent knowledge without making logical mistakes by visually confirming the reasoning process. The diagram can be an intermediary representation that assists the communication between knowledge engineers and experts who understand the knowledge in text form.

So the knowledge engineer may mark rules on the Web pages using RIML while constructing diagrams of the eventual rules that should be extracted from the Web pages. The expert’s diagram can be transformed to rules automatically, so it is efficacious to associate the diagram with the rules, but not the original text and the rules. Once a diagram is precisely drawn, the diagram may be complementarily used to assist the rule identification on the Web pages.
Achieving perfect natural language processing is difficult because the natural language texts may imply more than one valid interpretation as Wetter and Nüse (1992) pointed out. Using the natural language processing capability to acquire rules from the Web is also very difficult because Web sites can handle many diverse domains. Recently, ontology has become popular for specifying the knowledge of a particular domain on the Web (Crow and Shadbolt 2001). For certain domains, ontology may help in selecting the right interpretation of vocabulary (Guarino 1997). On the other hand, the natural language processing may be used to automatically extract terms to add to the ontology using grammar analysis (Maedche and Stabb 2000), linguistic patterns (Schmidt and Wetter 1998) such as regulatory sentences (Moulin and Rousseau 1994), and predefined templates (Szpakowicz 1990). However, the quality of automatically extracted knowledge from natural language sources is not accurate enough yet, so the draft should be manually refined by knowledge engineers (Schmidt and Wetter 1998; Wetter and Nüse 1992).

The primary idea of the XRML approach is to assist the knowledge engineer's rule extraction process assuming that natural language processing is not reliable enough. However, some text analysis technologies can be adopted in designing the XRML editor so that the knowledge engineer can search for terms from the text and retrieve their synonyms while identifying rules on the Web pages and editing rules in the rule base.

Machine Learning and Web Mining

Machine learning techniques such as inductive learning, neural networks, and statistical models may be applied under the umbrella term of data mining—specifically, Web mining when the log data are collected from Web pages (Jicheng et al. 1999; Kim et al. 2003). If a structured data set is available, we can induce them to more generalized and abstract knowledge. Several methods and tools (Alani et al. 2003; Apte et al. 1994; Hulth et al. 2001; van Heijst et al. 1997) have been developed using this approach. However, since extracting rules from the natural texts and tables usually aims to acquire the knowledge at the same level of abstraction, inductive learning (Craven et al. 1999) is not the primary issue in extracting rules from Web pages.

The eXtensible Rule Markup Language Approach

This section describes the representation of rule identification and process of rule extraction from Web pages using XRML. The entire process is demonstrated in Figures 2 through 4 with an example of the shipping and return policy of Amazon.com.

**Representation of Rule Identification Markup Language**

To fulfill the goal of RIML described earlier, let us formally define its syntax. Earlier versions of RIML 1.0 covered the simple statements of `RuleGroup`, `Rule`, `variable`, and `value`. These statements are marked up within the paired “< >” symbols as are XML statements. For instance, a variable `items` is identified as `<variable>items </variable>`. In the current version of RIML 2.0, we extend the statements by adding the primitive statements `RuleTable`, `IF`, `THEN`, connectives (such as `AND`, `OR`, and `NOT`), and numeric operators (such as `GT`, `GE`, `LT`, `LE`, `+`, `-`, `*`, `/`). Rules derived from the tables can be integrated with the rules from texts to conduct integrated reasoning.

The statements for RIML 2.0 that we have developed are formally specified in document type definition (DTD) syntax. We intend to use the same tags in RIML and RSML as much as possible to make the symbols mutually comprehensible. However, one unavoidable difference is the addresses of the counterpart resources. RIML has to identify the associated rule base and rules, while RSML has to identify the associated Web pages. Although RSML has a complete tag set for rule specification, RIML may use only the tags that are useful to identify rules in the context of Web pages. Identifying overly sophisticated numeric functions will be neither easy to express with tags, nor effective to specify completely. It will be better to postpone such specifications to the rule refinement stage in RSML as described in the postponement strategy section.

**Process of Rule Extraction from Web Pages by XRML**

As mentioned earlier, the XRML approach aims at identifying rules using RIML on the screen of the Web pages and transforming the identified rules to the draft rules in RSML syntax as depicted in Figure 1. The steps in Figure 1 are noted in parentheses. The process of rule extraction using XRML is composed of the following steps. The process is demonstrated with the example case from Amazon.com in the next subsection.
1. Plan the rule base and select Web pages.
   - Determine the goals and topics of the rule base, and compose the rule groups and rules.
   - Select the Web pages that are relevant to the rule base.

2. Identify rules using RIML.
   - Identify the RuleGroups, Rules, variables, values, IF-THEN relationships, and connectives such as AND and OR on the Web pages.
   - Build the HTML/RIML file which has the RIML statements embedded in HTML statements.

3. Transform the HTML/RIML statements to the draft rules in RSML syntax.
   - The identified rules on the Web pages can be automatically transformed to rules in RSML syntax, but the generated draft rules may be incomplete.

4. Refine the RSML draft rules and add new rules to build a complete rule set.
   - The rule components not specified in the identification step need to be refined at this stage to make the rules complete.
   - Additional rules are necessary to link the generated rules with the conclusion where the inference stops.

The complete RSML rule set generated in this manner may be further transformed to the syntax of target commercial rule-based systems if necessary. The remaining steps (5) through (8) are not described because they are beyond the scope of this paper. The knowledge engineer is involved in the three steps of the rule acquisition process: plan the rule base, identify the rules, and refine the rules. These manual steps should be assisted by the XRML editor, while the transformation step will be automatically executed.

An Example on Shipping and Return Policy

Let us demonstrate the rule extraction process from an example Web page in Amazon.com. This example attempts to build a rule-based expert system on the shipping and return policy, which can be merged to the price comparison site that considers not only the price of items, but also the delivery cost.

Plan the Rule Base and Select Web Pages

Suppose Amazon.com has decided to build an expert system that can consult the complex shipping and return policy. On behalf of the knowledge engineer, we have searched through the Web pages relevant to the subject and selected 21 pages. The selected pages explain the places to which Amazon ships, shipping rates, free shipping conditions, delivery time, the number of days within which a full refund is permitted, and conditions in which a return is not allowed. The rules could be grouped into four categories: Shipping Rates, Modifying Orders, Shipping Guide, and Returns and Refunds. It will be helpful for the knowledge engineer to become familiar with the rules described in the texts and tables, possibly with the aid of knowledge acquisition diagrams.

Identify Rules using RIML

The next step is to identify the existence of rules in the selected pages. We have identified 4 RuleGroups, 120 Rules, 35 RuleTables, 313 variables, 808 values, 13 operators, 119 IF statements, 120 THEN statements, and 107 connectives. In total, we have identified 1,635 rule components.

For instance, suppose we have picked the page as shown in Figure 2, which explains the rules on the shipping rate of books to Seoul, Korea. Note that the page has text, numeric functions, and three tables on Standard International Shipping, Expedited International Shipping, and Priority International Shipping. We can see that the shipping rate depends upon shipping region, items purchased, number of items, and priority type.
On this page, we need to identify the existence of RuleGroup, Rules, variables, values, and IF-THEN relationships, and connectives like AND and OR. Note the screen editing should be supported by the XRML editor although the example is explained with a HTML file. The rules from the tables also need to be identified as RuleTable. The embedded RIML tags are written in italic style in Figure 3. In this example, the RuleGroup is identified with its title = Shipping Rates. The first rule is identified as <Rule rid=1> with its THEN part in which the first variable, <variable vid=1> able to ship</variable>, and its omitted value, True, are identified. The knowledge engineer intentionally added the value True.

Transform the HTML/RIML Statements to the Draft Rules in RSML Syntax

The HTML files with embedded RIML statements can be used for two purposes. One is for display to humans in the HTML file format via the regular browser by eliminating the RIML statements. The other is to transform the HTML/RIML statements to the draft rules in RSML syntax. The RIML statements in Figure 3 can be transformed to the draft rules in Figure 4.

In this manner, we have generated 2,520 rule components as summarized in Table 1. According to the columns IC (number of identified rule components in the RIML stage) and GC (number of all generated rule components in the RSML stage), the numbers of identified Rules, RuleTables, IF, THEN, and operator components in RIML are the same as those in RSML. However, the numbers of variable and value are increased by 614 (66.2 percent) and 119 (12.8 percent), respectively, owing to the shared components. There are also 148 (57.1 percent) AND statements automatically generated as a default connective.
Figure 3. An Illustrative HTML/RIML File with RIML
Refine the RSML Draft Rules to Build a Complete Rule Set

Based on the draft rules demonstrated in Figure 4, we need to refine the rules to make them syntactically and semantically complete. This is an interactive process to the knowledge engineer with the assistance of rule editor. The rule editor may highlight a position containing a syntactic error such as “Missed IF Statement” or “Missed Rule Statement.” In the draft rules in the GC column of Table 1, some generated rules may not be complete, while some rules are not identified at all. The rC column of Table 1 shows the number of rule components added to make the identified rules complete. Three numeric operators and five connectives (AND or OR) are added at the RSML stage to refine the identified rules.
The knowledge engineer also needs to add the statements or rules to link with conclusions or other rules. For instance, in Rule 2 and Rule 3 in Figure 4, we need the statement <Set_Shipping_Rates>Computed</Set_Shipping_Rates> in the THEN part of the rules. Thus, we added 206 rule components to link with the conclusions, which amounts 7.5 percent of all rule components as shown the IC column of Table 1. In this example, we can see that 92.2 percent of the rule components are identified on the Web pages during the identification stage.

**Issues in Rule Identification**

There are four additional issues in rule identification: rule extraction from tables, handling of omitted components, handling of shared components, and handling of synonyms.

**Extracting Rules from Tables**

The earlier study of RIML started with identifying the rules from the natural language texts on the Web pages. However, many rules are effectively represented in tables on Web pages as is the case in Figure 2, which describes the shipping rate. So we need to have a common rule representation that can integrate the rules from both texts and tables. For this purpose, we need to specify the tag <RuleTable>.

The contents in the table correspond to the value of the rules’ THEN parts with their column heading titles as variable names. The record names in the rows correspond to the values of condition statements with their heading titles as their variables. When multiple tables exist in the same format, the table title statement should be added in the conditional statement. If the value statements around the tables have different levels of detail (such as country name Korea and region name Asia & Pacific), we need an additional statement that links them. In the example in Figure 2, customers need to identify the region to which a country belongs because the table is defined by region. So we need to refer to the text below the table that defines the region. This requires adding a rule (the Rule id=4 in Figure 4) from the text “Countries and Territories Included in the Asia & Pacific Islands Shipping Region” at the bottom of Figure 2.

The XRML editor needs to support the highlighting of table, columns, rows, and relevant text to identify the rules from the tables. Even though the identified rules will be internally specified in the HTML statements, the XRML editor should support a user friendly GUI dialogue for identification.

**Omitted Terms**

Some terms used in the rules may not be explicitly expressed in the Web pages. The terms might have been omitted because readers of the natural language texts and tables may be able to understand them without explicit expression. Let us call such terms **Omitted Terms.** This implicit omission is different from the undescribed statements. It is desirable to explicitly identify the omitted terms with their standard terms in RIML to associate them with the relevant terms unambiguously. Particularly when the omitted term is shared by other terms, it is beneficial to specify them in RIML to generate the multiple variable-value pairs of RSML automatically.

**Shared Components**

Some rule components may be defined once and shared multiple times by other components. For instance, a variable may be shared by multiple values, and a statement may be repeated in multiple rules. In the table, the same column title will be repetitively used as a variable with multiple values in the column. So the variable will be shared by multiple values. Variables are frequently shared by multiple values in the context of tables. However, it can also happen in the natural text as well.

The identification of shared variables in the Web pages is very useful because it can reduce the effort of rule coding. We simply do not have to repeat the coding of the same variables and values. Instead, we need to provide the serial identifier so that the corresponding variables and values can be recognized. For this purpose, we have designed the XRML editor to generate the serial number of variables and values to mutually combine them. For an example, see Figure 3: The omitted variable items in line 11 has its ID number vid=2 as expressed <variable vid=2 name = “items”/>.

The books and CDs are the values of items as
expressed <value vid=2>books</value> and <value vid=2>CDs</value>. In this manner, the values books and CDs can share the variable items.

This sharing can improve the degree of automatic rule structuring and reduce the effort of rule maintenance in the RSML stage because the variable may be identified once and revised once when the change is necessary. So the degree of automatic rule generation from the shared variables is an important component in measuring the efficiency of using RIIML. Later in this paper, we will find that 16.8 percent of the rules are generated from the shared variables.

**Synonyms**

Some terms may be phrased in different synonyms. To handle the synonyms, we need to build a thesaurus which pops up whenever the synonyms exist. The synonyms may also have a representative term, namely a *standard term*. The standard term is useful, particularly when we add tags for omitted terms. If the application system already has an ontology, the ontology facility may support the synonyms. Synonyms need to be managed both at an individual site and domain level. According to the experiment presented later, 14 percent of terms in the rules are expressed as synonyms.

**Postponement Strategy in Rule Identification**

The benefit of identifying the rules in the RIML stage is that it is easier to generate the rules from Web pages and to maintain consistency between the Web pages and the rule base. So it is desirable to identify as many rules on the Web as possible. Although most rules on the Web can be effectively identified, some statements are not easy to identify in the context of Web pages. For such cases, the recommendation is to postpone the identification to the refinement stage of RSML stage.

For a given situation, the knowledge engineer should balance effort between rule identification and rule refinement because raising the identification level may cause higher total effort due to identification overload. Thus the best principle in rule management is balancing the effort among rule identification, rule refinement, and rule maintenance so as to minimize the total effort. Typical situations where postponing identification is recommended are described the following subsections.

**Linkages between Rules and Conclusion**

Every rule-based inference needs conclusion rules that stop the inference. The conclusion rules should be intentionally defined by the knowledge engineer who knows the purpose of the expert system. Therefore, the conclusion rules and their linkages with intermediate rules usually do not exist in the Web pages. Thus, it is better to postpone adding such linkages to the RSML stage, because the rule editor for refinement can assist the detection of missing linkages between rules.

**Complex Numeric and Logical Expressions**

When the numeric functions and logical expressions are very complex, identifying only the key arguments postponing the full specifications to the rule refinement stage is recommended. Adding complex operators with the identified arguments will be easier to edit. In Figure 2, there is an equation that calculates the total shipping cost, which is not easy to express with the markup tags.

**Scattered Pages for a Rule**

Frequently, Web pages do not explain a set of rules in appropriate order. When the statements for a rule are scattered in multiple pages, it may not be easy to link the associating identifiers. Multiple windows, possibly with double monitors, may help alleviating this problem to some extent. When the association of all relevant components for a rule is not easy, we may intentionally postpone the full identification. In this case, the knowledge engineer needs to put a warning flag in the rule with a description about its incompleteness so that s/he can be alerted in the RSML stage.
External Sources of Statements

Some rules may require extra statements that do not exist in the Web pages. Some rules may not be allowed to publish on the Web. In this case, the knowledge engineer may also put up a flag on the missing statements along with their source if they are known. This will be an effective helper to make the rule complete during the refinement stage. In the rule refinement stage, the rules in RSML need to be transformed to the AND/OR graphic relationships so that the missed linkages and dead locks can be detected automatically (Lee et al. 1990; Liebowitz 1998; Nguyen et al. 1987).

Performance of Using RIML

Experiment Design for Evaluation

This section evaluates the effect of rule identification using RIML. For this evaluation, we selected three well-known online book-selling Web sites: Amazon.com, BarnesandNoble.com (in short BN), and Powells.com. From these sites, we analyzed the 36 relevant pages on the shipping and return policy, and identified the rules to discover the full potential of the rule identification from the Web pages. The rules generated from these sites were used to build a comparison site which compares book prices including the delivery cost for particular orders as was demonstrated.

To measure the effectiveness of rule identification, we define the metrics at the rule or rule component level as described below. Rule components consist of the statements RuleTable, Rule, Variable, Value, Operator, IF, THEN, and Connectives.

IR: Number of rules identified at the RIML stage
nR: Number of new rules added at the RSML stage
TR: Total number of rules necessary to build a complete rule-based system

The Effectiveness of Rule Identification can be defined in (1) as illustrated in Table 2.

\[
\text{Effectiveness of Rule Identification} \% = \frac{\text{IR}}{\text{TR}}
\]

We can observe the performance more precisely at the rule component level.

\(IC\): Number of rule components identified at the RIML stage
\(sC\): Number of rule components automatically generated from the shared components
\(dC\): Number of rule components automatically generated from the default operators
\(GC\): Number of rule components generated from the RIML statements
\(rC\): Number of rule components interactively added for rule refinement in the RSML stage.
\(RC\): Number of rule components after refinement

\[GC = IC + sC + dC\]

\[RC = GC + rC\]

With these notations, the Effectiveness of Rule Component Identification (in short Effectiveness) is defined as (4) to measure the degrees that the identified rules can cover. The Efficiency Ratio by Sharing is defined as (5) measuring the ratio of automatically generated rules in proportion to the number of identified components.

\[\text{Effectiveness} \% = \frac{\text{GC}}{\text{RC}} \times 100\]

\[\text{Efficiency Ratio by Sharing} = \frac{sC}{IC}\]

Efficiency implies the effort of rule coding that can be saved due to the automatically generated rules from the shared components. A higher ratio means a larger number of rules generated without coding effort during the identification stage. By the same token, we can define Efficiency Ratio by Default as (6). To sum the total effect of sharing and default defined as (7), let us define the term Efficiency Ratio by Automation as (8)
Efficiency Ratio by Default = \( dC/IC \)  
\( aC = sC + dC \)  
Efficiency Ratio by Automation = \( aC/IC \)

To make the complete rule set, the knowledge engineer needs to add the statements or rules to link with conclusions or other rules. These statements are generically impossible to identify on the Web pages because they are necessary by the nature of the rule chain in the rule-based system. To measure the effort level of adding the linkages statements, we define the measure **Linkage Effort (%)** as the percentage of linkage components to total components in (10). Linkage effort can be classified by its type whether it requires linkage statements with conclusions, linkage statements between interim rules, and new rules for conclusions.

\( IC \): Number of rule components added to provide linkages  
\( TC \): Total number of rule components necessary to build a complete rule-based system

\[ TC = RC + IC \]  
\[ \text{Linkage Effort} (%) = \left( \frac{IC}{TC} \right) \times 100 \]

So the potential effectiveness of rule identification is limited by linkage effort requirement. Thus the limit of identification can be defined as (11).

\[ \text{Limit of Identification} (%) = 100 - \text{Linkage Effort} \]

Effectiveness in (2) may be defined as \( GC/TC \) reflecting the potential limitation by linkage requirement. So let us call the metric **Overall Effectiveness** as (12).

\[ \text{Overall Effectiveness} (%) = \frac{GC}{TC} \]

Let us evaluate the three online bookstore sites with these metrics.

**Overall Experiment Results**

The overall experiment results are summarized in Table 2. Amazon has 21 relevant Web pages, BN has 8 pages, and Powells has 7 pages. The number of rule groups is four for each of the three cases. The total number of rules generated from each is 124, 101, and 258, respectively. Note that even though the number of pages in Powells is smaller than Amazon, the number of rules generated is higher because Powells uses large tables to describe the shipping rate.

In Amazon, the four rule groups are scattered in 21 Web pages. A total of 79 rules are generated from 35 tables along with the texts in the titles and annotations, and 41 rules are generated from the texts, which amounts to 34.2 percent of all generated rules. The large number of rules from the tables contributes to the computational role necessary for shipping cost estimation. In all, 120 rules were identified at the RIML stage, and 4 rules should have been added at the RSML stage. This shows that 96.8 percent of the rules could be identified from the context of Web pages. The four new rules were necessary for the logical linkages between rules and conclusion of the inference. Likewise, BN needs four new rules, and Powells three. Since all of the new rules are the ones for conclusions, all rules extractable from the Web pages are identified. Overall, the effectiveness of rule identification is 97.7 percent for the Web pages in three sites, which is very high, showing that RIML is quite dependable.

However, not all identified rules are complete yet. To study the completeness of the generated rules, we need to observe performance at the rule component level as summarized in Table 3. In Amazon, the number of identified rule components at the RIML stage was 1,635. From them, 737 rule components were automatically generated from the shared components, and 148 from default AND connectives. Thus, the total of all rule components generated at the RIML stage is 2,520. To make the incomplete draft rules complete, eight rule components should be added. So the effectiveness of rule component identification is 99.7 percent (= 2520/2528). However, we have to add 206 components to link with the conclusions which amounts to 7.5 percent of all rule components. Thus the generic limit of automatic identification is 92.5 percent, and the overall effectiveness achieved is 92.2 percent. For the three sites, the effectiveness of GC/RC is 99.7 percent, the limit of identification is 88.8 percent, and the achieved overall effectiveness of GC/TC is 88.5 percent. So the potential of rule identification is significant.
### Table 2. Performance of Identifying Rules on the Web Pages

<table>
<thead>
<tr>
<th>Book Store</th>
<th>No. of Web Pages</th>
<th>No. of Rule Groups</th>
<th>No. of Tables</th>
<th>No. of Identified Rules from Tables</th>
<th>No. of Identified Rules from Texts</th>
<th>Total (IR)</th>
<th>No. of New Rules (nR)</th>
<th>No. of Total Rules (TR)</th>
<th>Percentage of IR/TR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon</td>
<td>21</td>
<td>4</td>
<td>35</td>
<td>79</td>
<td>41</td>
<td>120</td>
<td>4</td>
<td>124</td>
<td>96.8%</td>
</tr>
<tr>
<td>BN</td>
<td>8</td>
<td>4</td>
<td>7</td>
<td>54</td>
<td>43</td>
<td>97</td>
<td>4</td>
<td>101</td>
<td>96.0%</td>
</tr>
<tr>
<td>Powells</td>
<td>7</td>
<td>4</td>
<td>3</td>
<td>242</td>
<td>13</td>
<td>255</td>
<td>3</td>
<td>258</td>
<td>98.8%</td>
</tr>
<tr>
<td>Total</td>
<td>36</td>
<td>12</td>
<td>45</td>
<td>375</td>
<td>97</td>
<td>472</td>
<td>11</td>
<td>483</td>
<td>97.7%</td>
</tr>
</tbody>
</table>

Table 3. Performance of Identifying Rule Components on the Web Pages

<table>
<thead>
<tr>
<th>Book Store</th>
<th>Identified at RIML Stage (IC)</th>
<th>Automatically Generated from Shared Components (sC)</th>
<th>Automatically Generated from Default Operators (dC)</th>
<th>All Generated from RIML Statements (GC=IC+sC+dC)</th>
<th>Efficiency Ratio by Sharing (sC/IC)</th>
<th>Efficiency Ratio by Default (dC/IC)</th>
<th>Interactively Added for Rule Refinement (IC)</th>
<th>After Refinement (RC=GC+IC)</th>
<th>Effectiveness (%)= (GC/RC)*100</th>
<th>Added to Provide Linkages (LC)</th>
<th>Total in Complete Rules (TC=RC+LC)</th>
<th>Linkage Effort (%)= (LC/TC)*100</th>
<th>Overall Effectiveness (%)= (GC/TC)*100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon</td>
<td>1635</td>
<td>737</td>
<td>144</td>
<td>2520</td>
<td>0.451</td>
<td>0.091</td>
<td>8</td>
<td>2528</td>
<td>99.7%</td>
<td>206</td>
<td>2734</td>
<td>7.5%</td>
<td>92.2%</td>
</tr>
<tr>
<td>BN</td>
<td>1151</td>
<td>494</td>
<td>109</td>
<td>1754</td>
<td>0.429</td>
<td>0.095</td>
<td>8</td>
<td>1762</td>
<td>99.5%</td>
<td>175</td>
<td>1837</td>
<td>9.0%</td>
<td>90.8%</td>
</tr>
<tr>
<td>Powells</td>
<td>1816</td>
<td>730</td>
<td>484</td>
<td>2830</td>
<td>0.452</td>
<td>0.300</td>
<td>3</td>
<td>2833</td>
<td>99.9%</td>
<td>519</td>
<td>3352</td>
<td>15.5%</td>
<td>84.4%</td>
</tr>
<tr>
<td>Total</td>
<td>4402</td>
<td>1961</td>
<td>741</td>
<td>7104</td>
<td>0.445</td>
<td>0.168</td>
<td>19</td>
<td>7123</td>
<td>99.7%</td>
<td>900</td>
<td>8023</td>
<td>11.2%</td>
<td>88.5%</td>
</tr>
</tbody>
</table>

The efficiency ratio by sharing is 0.445 (= 1961/4402), and the efficiency ratio by default is 0.168 (= 741/4402). Thus 0.613 of the identified components are automatically generated during the transformation process, saving this amount of the rule-coding effort.

**Contribution of Rule Identification from the Text**

We were curious about the relative contribution of text and tables in building a rule base, although it depends upon the characteristics of Web pages rather than the technical potential of the XRML approach. In the domain of the shipping and return policy, the number of rules identified from the natural language text on the Web is 41, 43, and 13 for Amazon, BN, and Powells respectively. They correspond to 34.2 percent, 44.3 percent, and 5.1 percent of all rules excluding the linkage rules to the conclusion. In this example, the effectiveness of rule identification is 100 percent because the example Web pages are quite self-contained and do not need extra rules from other sources.

At the rule component level, 959 rule components in Amazon are identified, and 8 rule components are added during the rule refinement stage. This corresponds to effectiveness of 99.2 percent, while it is 98.9 percent in BN, and 98.0 percent in Powells. The average effectiveness of rule identification from text is 99.0 percent. The result demonstrated that the potential effectiveness of the identifying rules from text is very high.

**Contribution of Tables in Generating Rules**

Rules are generated from not only text but also tables. Amazon has 35 tables, while BN and Powells have 7 and 3 tables respectively. Powells describes the shipping rate for each country without grouping them, so the number of tables is small, and each table is very large. The layout of tables can be quite different in this manner. It is interesting to note that all rules from the tables should be complemented by the text near the tables such as titles and annotations.

The number of rules identified from each is 79 in Amazon, 54 in BN, and 242 in Powells. They correspond to 65.8 percent, 55.7 percent, and 94.9 percent of all rules excluding the conclusion linkage rules. The figure in Powells is particularly high because the site heavily depends upon tables.
We can see that the effectiveness of rule component identification from tables is 100 percent for all of them. We could also observe that the identifications of RuleTable, Rule, IF, THEN, and connectives (such as AND and OR) are complete for all cases, while the identifications of variables and values are not. According to the performance from the tables, we can see that rule extraction from them along with complementary texts is highly reliable. In Amazon, 78 variables and 78 values are generated to link rules together. Powells achieved a relatively low level of variable/value generation (75.3 percent), because the large table required relatively more linkage statements.

**Identification of Implicitly Omitted and Synonymous Components**

We analyzed the impact of omitted components and synonymous components in this experiment. A common thesaurus for the three online bookstores is built with 583 words. For the three sites, the original terms are used at 73.4 percent, 59.9 percent, and 37.0 percent levels from the text, while 78.0 percent, 86.0 percent, and 98.3 percent are from the tables. Original terms are used more often from the tables because titles and headings of tables are more formally specified than ordinary text.

The percentage of omitted terms from text is 13.6 percent, 25.3 percent, and 24.7 percent respectively, and 18.6 percent, 2.0 percent, and 0.4 percent from the tables. It seems that there are fewer omitted terms from tables because the terms associated with tables are usually formally annotated to improve the readability of tables. Adding omitted components in RIML will make the consistency maintenance easier between the Web pages and the rule base.

The percentage of synonyms used is 13.0 percent, 14.8 percent, and 38.3 percent respectively from the texts, and 3.5 percent, 12.0 percent, and 1.3 percent from the tables. Natural texts used more synonyms than tables, because natural texts use many pronouns that are treated as synonyms. To identify synonyms, RIML supports the facility to label the standard term in the tag.

According to the experimental results, effective treatment of omitted terms and synonyms is very important to make rule identification more complete.

**Concluding Summary and Discussion**

Knowledge acquisition and maintaining consistency with original sources have been the fundamental hurdle in knowledge engineering. In the world of Web pages, there are oceans of original documents in natural language text and tables. If we are able to extract rules from Web pages and maintain consistency between them, the Web can be used more intelligently.

To attain this goal, we have developed the framework of XRML (eXtensible Rule Markup Language), which supports the identification of rules on the Web pages and generates the identified rules automatically. For this purpose, we have designed the Rule Identification Markup Language (RIML), which is similar to the formal Rule Structure Markup Language (RSML), both parts of XRML. RIML is designed to identify rules not only from text, but also from tables. So we can generate a rule set that can consider the text and tables together. The beauty of RIML is that the rules identified on the Web pages can be automatically transformed to the formal rules in RSML syntax.

While designing RIML, we considered the feature of sharing variables and values, omitted terms, and synonyms. Handling them in RIML is beneficial because they may be coded or changed once, automatically generating the corresponding RSML rules. The significance of these features is demonstrated by the experiment.

We have conducted an experiment to observe the potential performance of the XRML approach with real-world Web pages in Amazon.com, BarnesandNoble.com, and Powells.com. We found that 97.7 percent of the rules on the shipping and return policy could be detected on the Web pages, and the completeness of the generated rule components is 88.5 percent. This is good proof that XRML can facilitate the extraction of rules from Web pages to build expert systems and ensure the maintenance of consistency between the Web pages and rules.

The application opportunity of the XRML approach seems huge, because every rule-based consulting system on tax, regulations, law, insurance underwriting, loan, funds, budgetary control, and salary systems which have their counterpart Web pages need to adopt XRML to assure consistency. Since a different application may require extending the representation beyond rules, we need domain-specific research for key applications.

There are several topics that need further investigation in the XRML research agenda. The representation of rules should be extended to various target representations such as objects, constraints, mathematical expressions, and programming pseudo codes.
This will open the new horizon of consistency computing among heterogeneous representations. Another direction that requires development research is a user-friendly XRML editor that can assist knowledge engineers during rule identification and refinement. The capability of traditional rule refinement should be adopted to enhance XRML rule refinement. The capability of the XRML editor will be enhanced by adding ontologies that can reuse the rules in the similar applications. XRML can be implemented on Web services platforms, building XRML-S. We expect XRML to become one of the key ingredients in the next generation Semantic Web.

Acknowledgement

Part of the research described here was funded by the National Computing Agency in Korea. We are particularly grateful to Researcher Ho-Kyung Lee for his useful feedback.

References


Grosos, B. “DAML Rules Phase II Announcement,” 2002 (available online at www.daml.org/rules/).


RuleML. “The Rule Markup Initiative,” 2003 (available online at http://www.dfki.uni-kl.de/ruleml/).