Design of Configurable Rule Engine for Information Filtering

Zhang RuiJun, Yin Jun
School of Management, Wuhan University of Science and Technology, Wuhan 430081
E-MAIL: zrjdoctor@126.com

Abstract

It is urgent to filter invalid or even hostile information in web information systems. A configurable rule engine is designed to solve the problem of simpleness and low self-adaptive ability of rule matching method in traditional information filtering algorithm. In the engine, functions are dynamically called by using reflect mechanism to achieve computing of atom rules, logic rules are described in XML file and transferred to AND/OR tree, AND/OR tree is transferred to 0-1 Matrix, efficient reasoning of logic rules is achieved by combining with AND/OR switching principle, and atom rules base and logic rule base are also configured. Finally, the conceptual model and design model for the rule engine are given, the rule engine is running by classifying and definition of lifecycle.

Keywords **:** *Information Filtering; Configurable; Rule Engine; AND/OR Tree;0-1Matrix*

1. Introduction

The theory of information filtering is mainly based on content and cooperation. The former is more common used, in which the method of information filtering based on rule matching is widely adapted and recognized for it's simple, precise and controllable[1].

Mostly, information filtering deal with information which contains a few attributes. Usually, the more complex the information constitutes, the more complex filtering rules become. With the rapid change of business and market, the filtering rules may change constantly, but the information filtering system is not always flexible enough to adapt to new environment due to complexity and changeability of information[2]. In huge information system, because of different style of filtering rules and too simply design prototype, the information filtering components became less reusable and maintainable.

A core algorithm of Configurable rule engine is brought forward in this paper. In the algorithm, logical rules described by XML files are transferred to AND/OR tree and AND/OR tree is transferred to 0-1 matrix. By combining AND/OR switching principle, the algorithm with 0-1 matrix have achieved efficient reasoning for logical rules. As far as the rule engine is concerned, if information and rules for information filtering are input, the score is then worked out. In the algorithm, a whole filtering rule includes several scoring rules, and if one scoring rule is met, the prescriptive score can be gained. Finally, some issues of the rule engine such as the configurability, efficiency, sharing multi-rules and hot deployment are discussed.

2. Reasoning algorithm of rule engine

The essence of filtering rules for rule engine is describing the conditions that filtering information should satisfy. The conditions consist of relatively independent pattern-matching rules which are called atom rules and a series of logic conditions. The logic conditions made up of atom rules are called logic rules[3].

2.1 Atom rules

An atom rule can be abstracted as a method function, in which the parts varied with filtering information are called parameters of the function. In this way infinite atom rules can be abstracted as finite method functions and the set of functions constitutes atom rule base.

Atom rules can be configured by 2 files: Engine Configuration File(ECF) and Rule Configuration File(RCF). ECL describes the name and parameter type of every method function by XML, and provides a simple application indicator for the function[4]. There is only one ECF in the engine. According to the description of method function by ECF, a RCF assembles the atom rules required by special filtering information, by combining the attribute fields of filtering information. Every kind of filtering information has a RCF. All the RCFs share a ECF[5].

When a engine is running, it finds the function body in program by the description of ECF, passes the fields value of filtering information as parameters to the function, and makes the atom rules computing by calling the function. The computing result of atom rules(constantly Boolean type) is saved for future use.

2.2 Transfer logic rules to AND/OR tree

To filter and judge information carefully, the engine will regard scoring value as output result. The logic rule is then the rule that describes scoring logic. Judging each kind of information is divided into several scoring points and every point has a judging logic. The logic rule may be very complex, but it can be described in a AND/OR tree structure[6]. A simple example is given in figure 1, in which ①,②,③,④ are all atom rules.

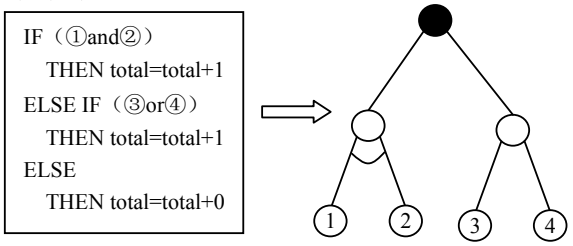

Fig.1 Transfer logic rules to AND/OR tree

Being transferred into AND/OR tree, logic rules become so formal that they can be conveniently described in standard XML format. The logic rules in figure 1 can be described as the following statements by XML:

```
<score value="1">
    <or>
        <and> $atom1</and>
        <and> $atom2</and>
    </or>
```

978-0-7695-3817-4/09 $26.00 © 2009 IEEE
DOI 10.1109/WISM.2009.155

```
        <or>
            <or> $atom3</or>
            <or> $atom4</or>
        </or>
    </score >
```

Where: $atom1,$atom2,$atom3,$atom4 - atom rule.

2.3 Rule engine reasoning algorithm of AND/OR tree based on stacks

Rule engine needs to deduce the AND/OR logic tree described by XML. The steps of reasoning algorithm operated by stacks are described as the following:

(1) The path to visit from the root node to the first atom rule can be stored in a original stack. The bottom element of the stack is root node and the top one is atom rule. Every AND/OR node along the path is a stack element. Judging the result of atom rule at the top is true or false.

（2）Deduce by different branches according to 6 kinds of combining conditions of and/or, true/false, exist/not exist brother nodes. Pass true/false value to root node of AND/OR element, that is, pass the value to the next element in the direction of the bottom of the stack.

（3）Finally, deducing end until true/false is passed to the root node. If the result is true, the score will be assigned to the filtering information, or it can not be assigned. The whole deducing process is presented in figure 2.

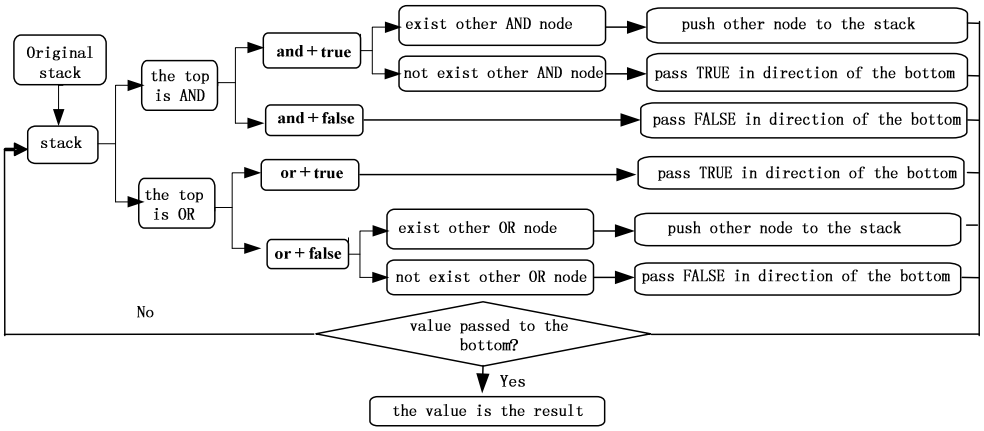

Fig.2 Deducing process of logic rules

Every kind of information has several scoring points and every point is described as scoring logic. After deducing from all the scoring logics by the engine, the total score is the result that the information is computed by the engine.

2.4 Transfer AND/OR tree to 0-1 matrix

When considering advantage of the algorithm, the time complexity and space complexity of deducing based on stack are high. The AND/OR tree can be transferred to 0-1 matrix and stored in the memory, for it can save vast memory space, furthermore, the speed of 0-1 matrix computing is more rapid than stack processing based on XML[7].

The principle of transferring is as following:

Transfer every scoring rule (or AND/OR tree) to a 0-1 matrix.

Represent every AND node with 1, OR node with 0.

Represent separately the path from the root node to every leaf node with a 0&1 row.

The same node can appear in the 0-1 matrix only once. If a node has appeared on a row, the positions of the node under the row should be blank.

The last number of every row in the 0-1 matrix correspond to a atom rule.

A simple example is given in figure 3.

Here define some basic operation concept about 0-1 matrix:

(1) Focus

Focus is the current position which is being concerned in a 0-1 matrix, that is, the crossing position of a certain row and column.

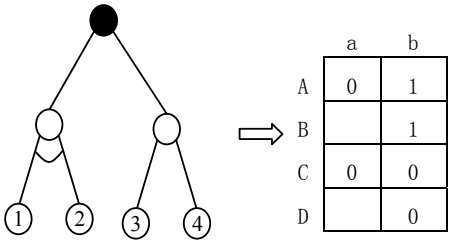

Fig.3 Transfer AND/OR tree to 0-1 matrix

(2) Young brother node

Search downwards from the position of focus. If 2 conditions are both met:

Ⅰ.the first number met is the same as the focus;

Ⅱ.There is no number in the position of the same row and previous column of the number or the number is in the first column.

Then position of the number is the young brother node of the focus, or there is no young brother node.

(3) Father node

Search upwards from the position of the current row and previous column of the focus, the first position with number(includes the start point) is the father node of the focus.

(4) Next atom rule point

The next atom rule point is the point of the last number of the row where the focus stays.

The computing results true and false can also be represented by 1 and 0. In this way, the 6 deducing branches of AND/OR tree in figure 2 can be evolved to the following 6 principles:

(1) 1+1+exist young bother node → transfer the focus to the young bother node

(2) 1 + 0 → transfer both 0 and the focus to the father node

(3) 1+1+not exist young bother node → transfer both 1 and the focus to the father node

(4) 0 + 1 → transfer both 1 and the focus to the father node

(5) 0+0+exist young bother node → transfer the focus to the young bother node

(6) 0+0+not exist young bother node → transfer both 0 and the focus to the father node

Assume that the values of 4 atom rules are: $atom1=true,$atom2=false,$atom3=false,$atom4= true, then the deducing process is given as figure 4.

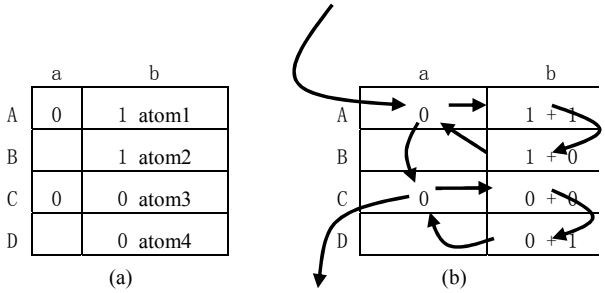

Fig.4 0-1 matrix deducing

There are null values in the 0-1 matrix, which doesn't make it a genuine 0-1 matrix. To build 0-1 matrix by using bit as the basic unit in the memory, AND/OR switching principle is introduced here.

If the subtrees of OR tree are also OR trees, all the branches of the subtrees can be elevated to the upper layer. In this way, the logic represented by the new tree is equal to the old tree, so is the AND tree, as figure 5 shown:

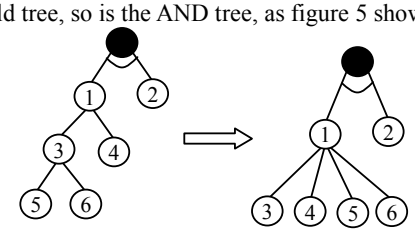

Fig.5 transfer AND/OR tree by AND/OR switching principle

Transferred in this way, all the subtrees of AND/OR tree are formed with AND tree and OR tree by turns, which is so-called AND/OR switching principle. The 0-1 matrix is also changed under the action of the principle, as figure 6 shown:

Following the AND/OR switching principle, when the number at row 1 column 1 is 0, if there is a number at any position in the odd column of the 0-1 matrix, it must be 0; while if there is a number at any position in the even column of the 0-1 matrix, it must be 1. On the contrary, Storing and computing logic rules by 0-1 matrix in memory can not only reduce the occupying space of memory, but also improve the computing speed.

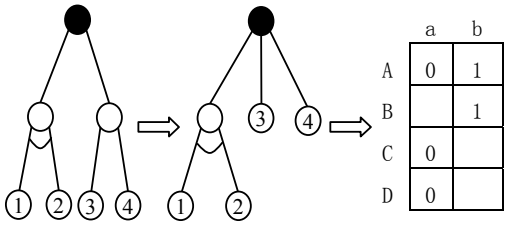

Fig.6 0-1 matrix changed under AND/OR switching principle

when the number at row 1 column 1 is 1, if there is a number at any position in the odd column of the 0-1 matrix, it must be 1; while if there is a number at any position in the even column of the 0-1 matrix, it must be 0.

Define the number deduced above as valid number, the number not met the deducing principle as invalid number. The blank positions of a 0-1 matrix can be filled with invalid numbers, just as figure 7 shown. All the computing will be carrying through by valid numbers, which solved the problem of building 0-1 matrix by the basic unit of bit.

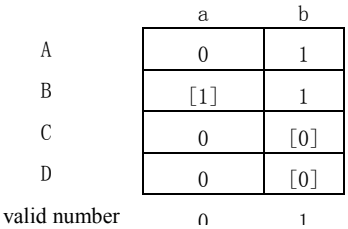

Fig.7 0-1 matrix described by the basic unit of bit

3. Design model of rule engine

3.1 Conceptual model of rule engine with multi-rule space

The rule engine is input with information and output score. The core of rule engine is the user-deployed rule files and engine for executing rule files. To meet the requirement of personal information filtering in several fields, the conception of multi-rule space is introduced. The model of rule engine with 2 rule spaces is given in figure 8. Every rule space has a rule configuration file. Besides rule spaces, a rule engine has an engine space which contains the common rules shared by all rule spaces.

3.2 Request mechanism of rule engine

Request is the whole course from inputting information to outputting the result by engine[8]. Every request contains such information as a storing parameter variable, matching result of atom rules and the reasoning score of logic rules. These information will be saved in the Request Context. To improve the performance and make it convenient, the same parameter information shared by all

requests in a same rule space can be improved to the rule layer, thus the Rule Context is formed. The same information shared by every rule space can also improved to the engine layer, so that the Engine Context is formed. The detailed design model is shown in figure 9.

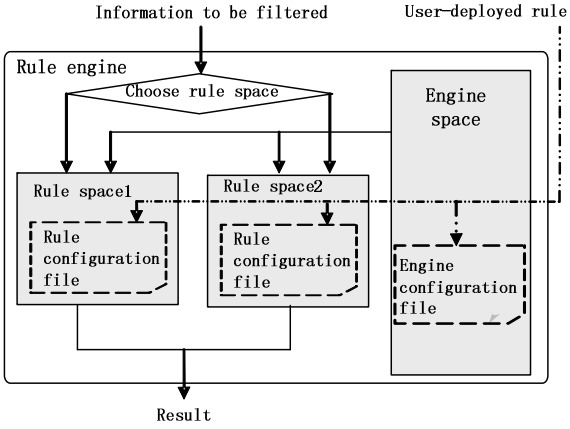

Fig.8 Model of rule engine with 2 rule spaces

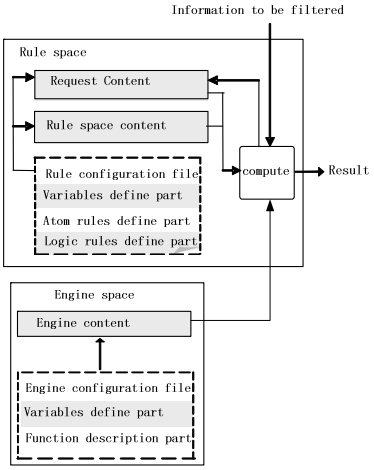

Fig. 9 Request mechanism of rule engine

3.3 Lifecycle of rule engine

The are 3 kinds of engine lifecycle:

1. Lifecycle of engine. The whole course from starting to stopping of the engine.

2. Lifecycle of rule space. The course from being created to being destroyed of a rule space.

3. Lifecycle of request. The course of being required to returning result of information.

The main task and relationship of these 3 kinds of lifecycle are given in figure 10.

In addition, the destroy phase in the lifecycle of rule space is divided into soft destroy and hard destroy. Soft destroy is to detect that all requests in the rule space have entranced the state of destroyed, and change the lifecycle state of the rule space to having been destroyed. Hard destroyed is to change directly the lifecycle state of the rule space to having been destroyed.

4. Conclusion

To solve a series of problem in information filtering, the paper brought forward a design scheme of rule engine to information filtering, in which 3 goals were represented: configurable rule, efficient and multi-rule space coexistence. The scheme achieved the prior 2 goals by designing a 0-1 matrix core algorithm and attained multi-rule space coexistence in design model of the engine. Finally, A configurable rule engine for information filtering was designed.

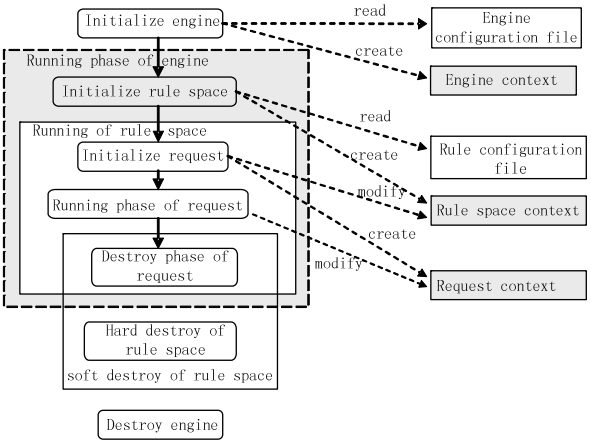

Fig. 10 relationship and main task of 3 kinds of lifecycle

5. Acknowledgements

The research is supported by China Scholarship Council (2008106373) and Doctoral Fund of Wuhan University of Science and Technology.

6. References

[1] Shi Yan, “A new information filtering technology model”, Journal of Library Science, vol.29,no.4,pp.45-47,2007

[2] Liu Yanfang, “Research and implement of Chinese text filtering technology ”, Master Dissertation of Beijing University of Technology,pp.5-6,2007

[3] Qu Shou ning , Zhu Qiang, Lin Brian, etc. “Research and application for rule Engine in airport resource management system”, Journal of JiangXi Normal University(Natural Science), vol. 32, no. 2,pp.142-146,2008

[4] DAI Ming-xing, DU Yan-hui , “Design for Content Search Engine Based on WebLech”, Computer Engineering, vol. 34, no. 9,pp.278-280,2008

[5] Zhang Yuan,Xia Qingguo, “JAVA rule engine based on Rete algorithm”, Journal of Science Technology and Engineering, vol. 6, no. 11,pp.1548-1550,2006

[6]Giarratano J C. Expert system:principles and programming (Fourth Edition). Beijing :China Machine Press ,pp.37-39,2005.

[7] Robert L,Alexander J, “Data Structures and Program Design in C++”,Prentice-Hall,1999.

[8]Ye Zhou ,Wang Dong. Rules engine based data cleansing . Computer Engineering ,vol. 32, no. 23,pp.52-54,2006.