A Mathematical Formula Retrieval Method Using Structure Sub-tree

Mingjie Guan[1,2], Xuedong Tian[1, 2*], Fang Yang[1,2], Songqiang Yang[1,2]
(guanmj@sohu.com, zntwcl@126.com, yangfang@hbu.edu.cn, hbuyangsongqiang@126.com)
1. College of Mathematics & Computer Science, Hebei University Baoding, China
2. Hebei key laboratory of Machine Learning and Computational Intelligence, Hebei University

Abstract— **It is quite inadequate in providing formula retrieval function by traditional retrieval techniques used in full-text information retrieval system. The main reason is that there are many difficulties to extract the keywords of the mathematical formulas. In this paper, a detailed analysis of the structural characteristics of mathematical formulas and existing index mechanism of mathematical formula searching engine is fulfilled. Then a full-text index (named SLIndex) of mathematical formulas with B+ tree structure is designed and implemented which extracts the structured logic sub-tree feature as keywords of formulas and employs inverted index. Finally, a formula search engine model based on SLIndex is implemented in Apache 2.0 web server.**

Keywords-mathematical formulas retrieval; full-text retrieval; inverted index; B+ tree

I. INTRODUCTION

With the development of the Internet, text-based information retrieval systems have been widely used in every field. While this type of retrieval systems is limited to support for mathematical formula. Compared with text information, mathematical formulas are more expressive and rigorous with more complex structural features, which results in the search engines based on text information could not used for indexing and retrieval formula information. So, it is necessary to develop a search engine which can support the retrieval of mathematical formulas.

At present, there are already some mathematical formulas retrieval systems such as MathDex, DLMF Search and MathWeb Search.

MathDex [1] is based on Apache and Lucene. It has the following characteristics: supporting the retrial of non-semantic mathematical content; supporting the searching of multiple encoding of mathematical content queries; supporting both mathematical symbols and text retrieval, as far as possible to meet user expectations and not just the text query. MathDex converts all files to the format of XHTML + MathML firstly, and then sorts the content according to the structure and syntax match level with query expressions. In the index build phase, it creates index for each formula and recording the frequency information of sub-formula which is important in complex formula matching.

DLMF Search [2] is an online project for Digital Library of Mathematical Functions at National Institute of Standards and Technology in America. It extended and defined a set of metadata to support fuzzy query expression. Firstly, it maps every mathematical symbol into the alphabet. Then, it standardizes these items to avoid query error due to the inconsistent expressions. The core of DLMF Search is still text-based. There is no difference between it and the conventional text retrieval system in essence.

MathWeb Search [3] is a mathematical search engine which is independent of full-text search engine. It uses non-text query mode in which the expression is parsed to the substitution tree and all sub-formulas belongs to a formula were added to the index library alone.

II. EXTRACTION OF KEY VALUES IN MATHEMATICAL FORMULAS

A. Structural Features of Mathematical Formulas

There are many description modes to express mathematical formulas, such as LaTex, MathML and OpenMath. All formulas can be expressed by an expression tree. For example, the formula $u=\sqrt{5s^3+2s+t^2}$ (Formula 1) can be displayed as the structural tree shown in Figure 1.

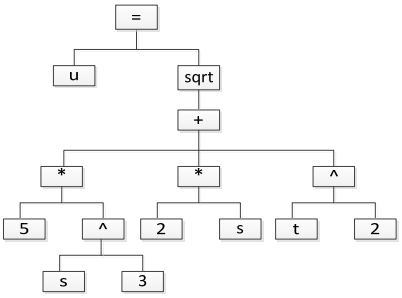

Figure 1. The expression tree of Formula $u=\sqrt{5s^3+2s+t^2}$.

Since all variables and constants are in the leaf nodes of the tree and variables are replaceable, we can define a Structured Logic sub-tree (SL sub-tree) by removing variables from the structural tree of mathematical formulas. The SL sub-tree of the formula in Figure 1 is shown in Figure 2. The removed leaf nodes will be saved which could be used to sorting the search results of formulas in the future.

* Corresponding author.

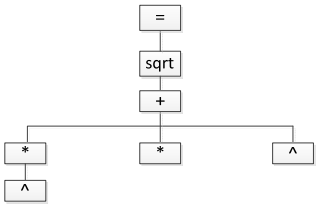

Figure 2. The SL sub-tree of Formula $u=\sqrt{5s^3+2s+t^2}$.

The SL sub-tree expresses the logic relationships of mathematical formula symbols clearly. This relationship, to some extent, reflects the essence of mathematical formulas, and can be used in the extraction of formula keywords.

B. Key Value Base on SL Sub-tree of Mathematical Formulas

1) Definitions of related functions and operators

- Operator "· ": Join two strings to one. For example: "SL" · "Index" ="SLIndex".
- Operator "∪": Combine two index sets. For example: {"SL"} ∪ { "Index"} = { "SL" , "Index"}.
- F(m): The return value of the function is an ASCII mapped. Where m is a mathematical symbol. According to Wikipedia, there are totally 108 mathematical symbols [4], and the standard ASCII has a capacity of 127. So we can use the set of ASCII characters as a mapping table to mathematical symbols. Take chart 1 as example, all the symbols except the square root symbol are already ASCII characters, we use them as targets. So we only need to define the mapping from "sqrt" to a character "s" [5].
- K (t): a function of calculating key string of a SL sub-tree. For example: for a SL sub-tree with preorder traversal sequence d_0, d_1, d_2, … , d_{m-1} , the key string is: $K(sltree)=F(d_0)\cdot F(d_1)\cdot F(d_2)\cdot ...\cdot F(d_{n-1})$
- I(t): a function of calculating key string set of a SL sub-tree. A structure logic tree named "sltree" with height of h, there are m nodes called n_0,n_1,n_2,…,n_{m-1} under its root node r. Then, Its key string set can be calculated as the following:

  While h<2, $I(sltree)=\phi$;

  While h≥2,

  $$I(sltree)=K(r)\bigcup\left(\bigcup_{i=0}^{m-1}K(n_i)\right).$$

- H(t): a string hash function. In order to improve the efficiency of the comparison operation, we map the strings to integer values using a string hash function. Common string hash functions get big integers, so here we use a modified BKDRHash [7] function to limit the return value fall into [0,999].

2) Extracting key value set from a SL sub-tree:

According to the functions defined above, for the formula $f(x,y,z)=3y^2z\left(3+\frac{7x+5}{1+y^2}\right)$ (Formula 2), the steps for extracting the key value set are listed below:

- Generate expression tree.

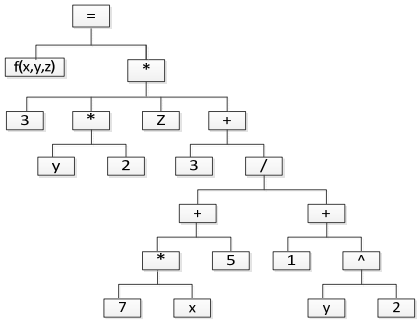

Figure 3. The expression tree of Formula (2).

- Generate SL sub-tree after through removing all leaf nodes.

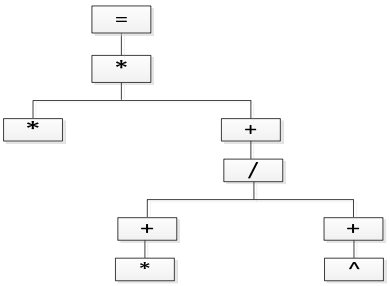

Figure 4. The SL sub-tree of Formula (2).

- By a preorder traversal of the sub-tree, the key string is derived: "=*^+/+*+^".
- Using the child node (named "*") of the root node as new root node, repeat the previous step to obtain a new key string: "* ^ + / + * + ^".
- Recursive the previous step, until the sub-tree height is less than 2, obtained following several strings: "+ / + * + ^" "/ + * + ^", "+ + ^", "+ *"," + ".
- The key string set of Formula (2) is the union set of these strings sequence: { "= * ^ + / + * + ^", "* ^ + / + * +", "+ / + * + ^", "/ + * + ^", "+ * + ^", "+ * "," + "}.
- Map each string in the key string set to an unsigned integer variable by the hash function. The key value set is the union of these variables: **{ 379 , 598 , 538 , 801 , 402 , 675 , 727 }**

C. *Sort the search result by the leaf nodes information set*

The problem to be solved after the completion of retrieval is how to sort the search results set reasonably. On one hand, the search result which is more similar with the retrieval requirement should be closer to the top location in all searching results. On the other hand, the information from the leaf nodes is helpful for us to get more correct information.

- Leaf Nodes Information set(LNI set): LNI set is a character stream by traversing all of the leaf nodes of formula tree to extract the text information of operand delimit with a space symbol. The LNI set of the Formula (2) can be derived through the definition: "$f(x,y,z)\ 3\ y\ 2\ z\ 7\ z\ 5\ 1\ y\ 2$". In the same manner, the LNI set of the formula $y = x^2 + 3$ (Formula 3, key value set is {548, 727}) is "$y\ x\ 2\ 3$".
- Distance function of string LD (*a*, *b*): LD function is used for measuring the difference between two strings. In this paper, the string distance is defined as same as edit distance [6]. For example, the edit distance between "slindex" and "sl" is 5, "slindex" and "saindex" is 1.
- Similarity Function of String S(str1, str2): *S* function is used to measure the similarity between two string. For the two strings str1 and str2, the formula is as follow:

$$S(str1, str2) = 1 - \frac{LD(str1, str2)}{\max(lenth(str1), lenth(str2))}$$

S Function reflects the similar extent of two strings, with a return value in the interval [0, 1]. The greater the value of S function is, the more similar the two strings are. Using S function, the similarity value between string "slindex" and "sl" is 0.286, string "slindex" and string "saindex" is 0.857. It means that string "slindex" and string "saindex" are more similar.

For example, when retrieve the formula $x^2 + 3$:

- Extract its key value: 727. It is in the key value set of Formula (2) and Formula (3). Therefore, the two formulas will be searched at the same time.
- Extract the LNI set of the formula $x^2 + 3$: "$x\ 2\ 3$". Using S function, the similarity value between it and Formula (2) is 0.178, while the similarity value between Formula (2) and (3) is 0.714. Therefore, the position of Formula (3) is closer to the top of the retrieval results than Formula (2). It is consistent with subjective human feelings.

III. INDEX MECHANISM OF MATHEMATICAL FORMULAS

A. *Inverted index*

Fast response is vital in the formula search engine. Therefore, it is more efficient to use the inverted list structure [8] to organize formulas and documents in spite of its efficiency because the indexing is established in background. An inverted index file for mathematical formula is shown in Figure 5.

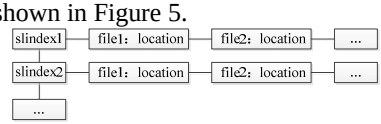

Figure 5. Inverted file structure.

B. *Inverted index based on B+ tree*

With the increase in the inverted table, it should be a bottleneck to search and update the index table quickly for improving the efficiency of search engine. As B+ tree [9] could keep the stability and order of data which makes it has a lower logarithmic time complexity when inserting data or updating, we use it to construct the inverted index of formulas.

Take Formula (1) and Formula (2) as examples, in this paper, a 3 order B+ tree is used to establish index. The procedure is as follows (assuming the initial state of the tree is empty):

- Extract key value set with the algorithm in second chapter. It is {921, 12, 915, and 596} in Formula (1), and {379, 598, 538, 801, 402, 675, 727} in Formula (2).
- Insert every value in these two key value sets into the 3 order B+ tree. In this process, the B+ tree state change as Figure 6:

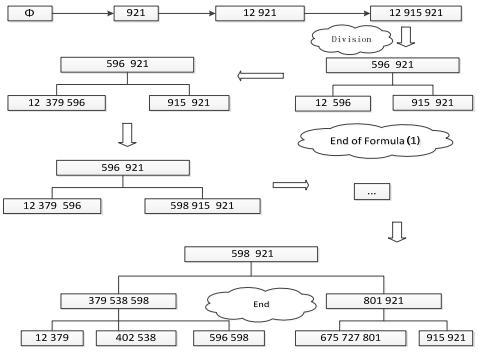

Figure 6. The states of B+ tree.

IV. DESIGN AND IMPLEMENTATION OF SLINDEX

As a full-text retrieval system, there are two modules in SLIndex: query module (in response to the retrieval input) and index maintenance module (maintaining the index structure) to keep better modularity, scalability and portability.

A. *Query module*

As basic data structure, SL sub-tree is the input data type of query module. And the output data type is a string stream corresponding to the key value of the input, with files and positions information. Figure 5 is the flow chart of query module.

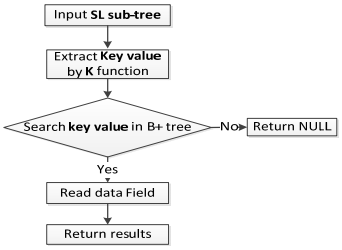

Figure 7. Flow chart of query module.

B. *Index maintenance module*

The information of files and their positions according to the key values is recorded in the item of inverted index file. Since the number of the date files corresponding to each key value changes dynamically, it is very complex to create and maintain the inverted index file. Therefore it is necessary to run index maintenance module in the background to reduce the influence of efficiency and to maximize advantages of inverted index. Figure 6 is the flow chart of index maintenance module.

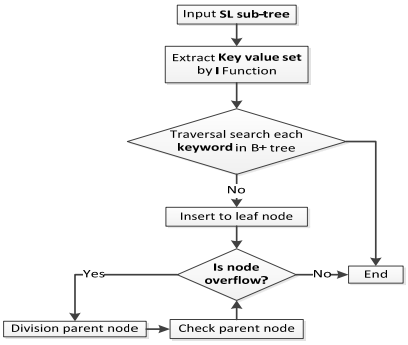

Figure 8. Flow chart of index maintenance module

V. EXPERIMENTS AND ANALYSIS

The query module uses Apache 2.0 as web server, and PHP 5.2.5 as an interactive script. The main page is shown in Figure 9.

Figure 9. The main page of the query module.

The result page shown in Figure 10 is returned after clicking the button "search" includes the URLs of formula, the positions of formula in the page and the contents.

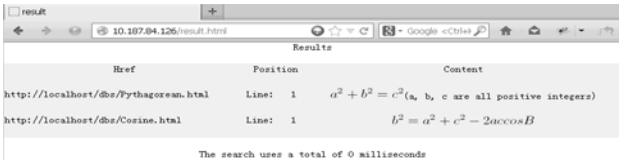

Figure 10. The results page of the query module.

The index maintenance module is compiled with VC++ 6.0, and runs as a background service without output information.

VI. CONCLUSIONS

Based on previous research works related to mathematical formula retrieval, and analyses of the structure characteristics of the mathematical formulas, the concept of SL sub-tree is presented in this paper. The key value set based on SL sub-tree and related algorithms are discussed. B+ tree is used as a dynamic inverted index structure to query and maintenance key values. In theory, the method is more efficient than the traditional text-based retrieval system. Experiment results show the feasibility of the method designed in this paper. But it still can be improved at how to extract SL sub-trees from mathematical formulas more quickly and accurately.

ACKNOWLEDGMENT

This work was supported by Natural Science Foundation of Hebei Province (F2012201020).

REFERENCES

[1] "MathDex search," http://www.mathdex.com:8080/mathdex/search.

[2] "DLMF: Preface," http://dlmf.nist.gov/front/preface.

[3] "MathWeb Search - A Semantic Search Engine – About," http://search. mathweb.org/about.html.

[4] "List of mathematical symbols," http://en.wikipedia.org/ wiki/ List_ of_mathematical_symbols.

[5] Gennady Antoshenkov. "Dictionary-based order-preserving string compression," The VLDB Journal, Volume 1, DOI: 10.1007/s007780050031, 1997.

[6] "BKDRHash/bkdrhash.c at master · hit9/BKDRHash · GitHub," https://github.com/hit9/BKDRHash/blob/master/bkdrhash.c.

[7] "Inverted index," http://en.wikipedia.org/wiki/Inverted_index.

[8] "Levenshtein distance," http://en.wikipedia.org/wiki/Levenshtein_distance.

[9] "B+ Tree," http://en.wikipedia.org/wiki/B%2B_tree.

[10] K. Jing, "Research on math query language and index in web-based math search," unpublished.

[11] T. Luo and J. Yu, "The Design and implementation of web-based formula retrieval system," Micro Processors, No. 2, pp. 102-106, April 2008.

[12] C. Cai and W. Su, "Construction of web based mathematical formula editor software," Computer APPlications, Vol. 27, pp. 235-238, Dec. 2007.