American Journal of Computer Science and Technology
2023; 6(1): 25-32
http://www.sciencepublishinggroup.com/j/ajcst
doi: 10.11648/j.ajcst.20230601.14
ISSN: 2640-0111 (Print); ISSN: 2640-012X (Online)

Research and Implementation of Word Detection System Based on Improved DFA in China

Feng Kai, Tuyatsetseg Badarch*

Department of Information Technology, School of Information Technology and Design, Mongolian National University, Ulaanbaatar, Mongolia

Email address:

ba.tuyatsetseg@mnun.edu.mn (Tuyatsetseg Badarch)

*Corresponding author

To cite this article:

Feng Kai, Tuyatsetseg Badarch. Research and Implementation of Word Detection System Based on Improved DFA in China. *American Journal of Computer Science and Technology*. Special Issue: *Advances in Computer Science and Future Technology*. Vol. 6, No. 1, 2023, pp. 25-32. doi: 10.11648/j.ajcst.20230601.14

Received: January 12, 2023; **Accepted**: February 20, 2023; **Published**: March 4, 2023

Abstract: Since the second half of the last century, the intensive usage of digital texts and textual databases produced the need for efficient search methods and data structures. Even though there are many traditional pattern matching algorithms such as regular matching, AC algorithm, and WM algorithm, in this paper, we use on a word detection method based on an improved DFA algorithm. We focus on the implementation of content matching technology using an improved DFA algorithm. We used the approach that can retrieve the emoticon icon, half corner character, repeated word based on ConcurrentSkipListMap to construct the tree of the word filtering system. We introduce the architecture of the system that mainly depends on the middleware, database, and data processing parts. The algorithm performs functions including filtering the word to match multiple pattern strings, to share a common prefix of a string that can reduce repeated lookups and save memory space. We use the pre-trained word vector model to achieve good results for the expansion and improvement of the sensitive lexicon. The system realizes the functions of word matching, including initializing, changing, matching, and highlighting of the word database, various processes that are tested and analyzed. We did a simulation to capture relevant word data and import it into MySQL database for storage. The method for message sensitive word recognition effectively improves the speed and accuracy of the algorithm recognition, the efficiency of word matching. We emphasize the DFA algorithm is the best approach compared to AC algorithm and other algorithms. Through function test, system test, and performance test, some valuable results are obtained. As a result of the tests, valuable results are founded from functional tests, system tests, performance tests. The system realizes the characteristics of large thesaurus and high matching efficiency of long text. It can meet the requirement of network real-time transmission, so it can be applied in the network. This paper proposes an improved multi-mode matching algorithm for word detection based on DFA. The algorithm maximizes the speed of problem detection and response efficiency and purifies the network space by optimizing the algorithm for the characters of the text content, the number of basic words and the detection efficiency. As a result of our research, we have shown the data from different sources of the system can be reused to reduce repeated construction costs.

Keywords: DFA, MySQL, Word Detection System, Word Changing, Word Matching

1. Introduction

Due to the continuous development of Internet technology, the number of netizens increases year by year, and the Internet has entered a stage of rapid development. People gradually realize that the network has brought us an important role, and through the network of information transmission and resource sharing to achieve the purpose of communication. However, while disseminating favorable information on the Internet at home and abroad, it also provides an opportunity for criminals to take advantage of it. The quality of the network communication platform and the content are not even multiplied, and the network dissemination platform takes advantage of its open source and public characteristics to spread bad information [1], such

as: abuse, advertising, pornography and other content, make some people with weak ability to distinguish between right and wrong are seriously affected, and even harm the security of the country and the stable development of society. Therefore, while the Internet enhances people's communication, it also needs the support of more stable filtering technology. Network information filtering is mainly aimed at the problem of mixed information on the Internet, but it can also provide some new ideas to solve the information society people want to independently set the way to obtain information and content and other personalized customization needs.

The advanced DFA algorithm based word detection system is used to construct the Trie tree for sensitive word recognition, which effectively improves the algorithm speed. It can be recognized as a typical word detection system based on the improved DFA algorithm. In order to filter the words in the text, many traditional pattern matching algorithms such as regular matching, AC algorithm and WM algorithm [2-5]. In the Chinese content research: some researchers proposed an improved fast multi-pattern matching algorithm based on the idea of converting Trie tree into even group form [6]. Other scholars have proposed matching word decision tree information filtering algorithm [7]. This paper proposes an improved multi-mode matching algorithm for word detection based on DFA [8, 9]. Although the DFA algorithm can solve part of the word detection problem, it can maximize the speed of problem detection and response efficiency and purify the network space by optimizing the algorithm for the characters of the text content, the number of basic words and the detection efficiency. On this basis, the overall process of the system is further analyzed and explained. On the other hand, data from different sources of the system can be reused to reduce repeated construction costs.

2. Research Background

In the early stage, the main functions and performance indicators of the monitoring system are mainly servers or hardware, etc., mainly monitoring the overload or crash of servers caused by external network system intrusion, and monitoring the operation of the network and other devices, etc. [10]. In the 21st century, the Internet has become the main channel for the government, universities and enterprises to carry out business and publicity services. A variety of network information is presented in a blowout manner.

With the development of Internet technology, more and more users are participating. In the process of enjoying and participating in entertainment content, some ill-intentioned users will release undesirable sensitive and illegal information, such as violent pornography, online gambling fraud and other information. These information through a variety of means in a variety of carriers spread in the network, so that users inevitably suffer from information infringement. This is where the importance of content policing grows. Due to the convenience of the Internet, wide audience, fast transmission and other characteristics, it is difficult to use manual inspection to effectively find problems in the first time.

Therefore, the monitoring system has gradually expanded from monitoring the function and performance of hardware equipment to monitoring the network information content, which has generated a new demand for the monitoring of network content. This paper draws on the advantages of DFA algorithm to quickly construct search tree in retrieval [11].

The main functions of the word detection system are as follows:

(1) Detection of political-related content: accurately identify the illegal texts of political-related figures, political events, religion, reactionary division and terrorism in various scenes.
(2) Pornographic content detection: accurately detect pornographic content such as obscenity, sexual attraction, erotic love, pornographic innuendo and soft pornography, and support pornography classification.
(3) Prohibited content detection: accurate identification of gambling, knives and guns, drugs, counterfeiting, counterfeit selling and other illegal articles and violations.
(4) Abusive content detection: accurately identify abusive content such as insults, abuse and slander in various scenes.
(5) Garbage content detection: accurate identification of water stickers, brush screen, meaningless and other garbage content, to achieve intelligent anti-garbage.
(6) Advertising content detection: accurately identify the contents of illegal junk advertisements released by wechat, mobile phone number, email, wechat, etc.
(7) Text symbol detection: accurately identify the text semantics of various emojis and character expressions.
(8) Custom detection: supports the user-defined thesaurus, and carries out directional filtering of illegal content through literal comprehensive judgment.

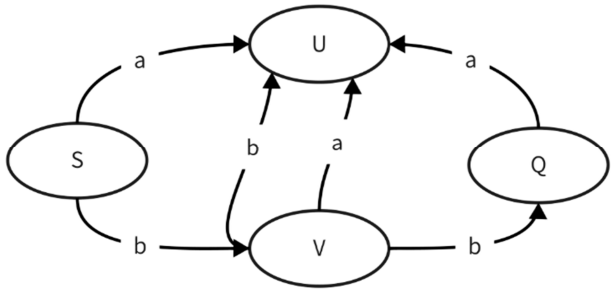

Figure 1. *DFA Diagram of state.*

3. DFA Algorithm

The full name of DFA is Deterministic Finite Automaton, that is, the deterministic Finite automaton [12]. Its characteristics are: there is a finite set of states and some edges from one state to another state, each edge marked with a symbol, one of the states is the initial state, some states are the final state. But different from the uncertain finite automata, there will be no two side signs starting from the

same state with the same symbol in DFA [13, 14].

In a nutshell, it's going to get the next state from event and the current state, that is, event+state=next state. It means that there are multiple nodes in the system, and the route to another node is determined by passing the incoming event, while the node is limited. this picture shows Diagram of state (Figure 1).

4. Proposed Design and Implementation of Word Detection System

Combining with the solution of word detection system based on DFA algorithm, the proposed system is designed. This paper introduces the architecture of the whole system and describes the composition and function of the system.

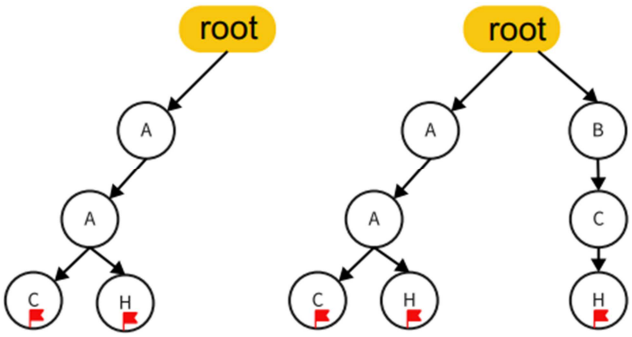

Figure 2. *System Architecture.*

4.1. System Architecture

The whole system depends on the middleware and only Mysql, RocketMQ support. this picture shows system architecture (Figure 2).

(1) Word detection service: Provides word detection system filtering service and text highlighting service through http interface.

(2) Word detection management service: It is responsible for maintaining the life cycle of thesaurus, notifies the filtering service asynchronously through message queue, and provides the functions of adding, modifying, deleting, and querying words. At the same time, the automatic invalidation of the word can be set through the timing scheduling.

The trie thee based data structures based in improved DFA algorithm were necessary in order to support the storage of textual data in a manner in which a preprocessing stage would present the possibility of having subsequent search, insertion and deletion operations.

In general, a trie allows the prefix of terms to be used as a search mechanism. The trie requires pre-processing the data in to a data structure that has very efficient lookups.

4.2. Initialize the Matching Structure

Trie tree (dictionary tree, also known as word lookup tree or key tree) is a tree-like data structure, mainly used for character matching, character statistics, character storage. Compared with the traditional string matching algorithm, the Trie tree has the following advantages:

(1) It can be used to match multiple pattern strings [15];

(2) Sharing a common prefix of a string can reduce repeated lookups and save memory space [16].

Regarding the Trie tree algorithm performance, we describethe main process of the root, its build steps as follows:

(1) The Trie tree composed of AAC and AAH, which share the prefix AA.

(2) Add BCH, which has no common prefix with the previous two words.

(3) Add AAB, which can share the AA prefix with AAC and AAH.

(4) If AA is added, A closing mark needs to be added to the second word A. The picture shows the Trie tree build process (Figure 3, Figure 4).

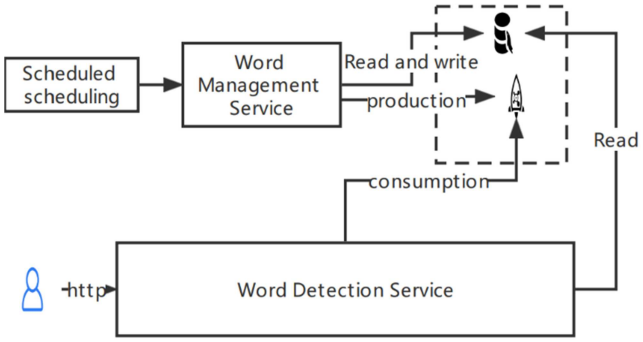

Figure 3. *Trie tree building process.*

After the words in the text to be detected are filtered out according to the Trie tree, other information such as the rank and description of the words need to be obtained. You can use a hash table to store a list of words stored in the database. The result is the following data structure which has two technical details including word cache, the Trie tree construction.

4.2.1. Word Cache

The fundamental difference between the cache and the underlying dependent hash table is that the cache can save memory by reclaiming stored items through a cull policy. The elimination strategy is to predict which data is most likely to be used again in the short term, so as to improve the cache hit ratio, which is an important feature of the cache library.

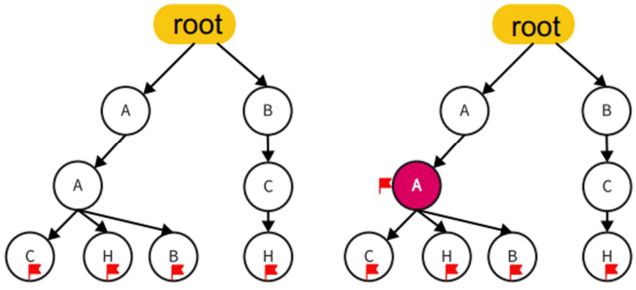

Figure 4. *Filtration Process.*

Caffeine is a high-performance Java Cache library, near optimal caching library. Caffeine provides an in-memory cache using a Google Guava inspired API. Caffeine uses ConcurrentHashMap for its underlying data storage. ConcurrentHashMap has added red-black trees to enable good read performance even when hash conflicts are severe.

4.2.2. Trie Tree Construction

The word detection system in this paper uses ConcurrentSkipListMap to construct the Trie tree of the word filtering system. ConcurrentSkipListMap is a thread-safe ordered hash table suitable for high-concurrency scenarios with the following characteristics:

ConcurrentSkipListMap and TreeMap: are both ordered hash tables. But they have different thread-safe mechanisms. TreeMap is non-thread-safe, whereas ConcurrentSkipListMap is thread-safe. ConcurrentSkipListMap is implemented through a jump list, while TreeMap is implemented through a red-black tree.

ConcurrentSkipListMap and ConcurrentHashMap: The keys of ConcurrentSkipListMap are ordered and ConcurrentSkipListMap supports higher concurrency. The ConcurrentSkipListMap access time is log(N), almost independent of the number of threads. With a given amount of data, ConcurrentSkipListMap can benefit from having more concurrent threads.

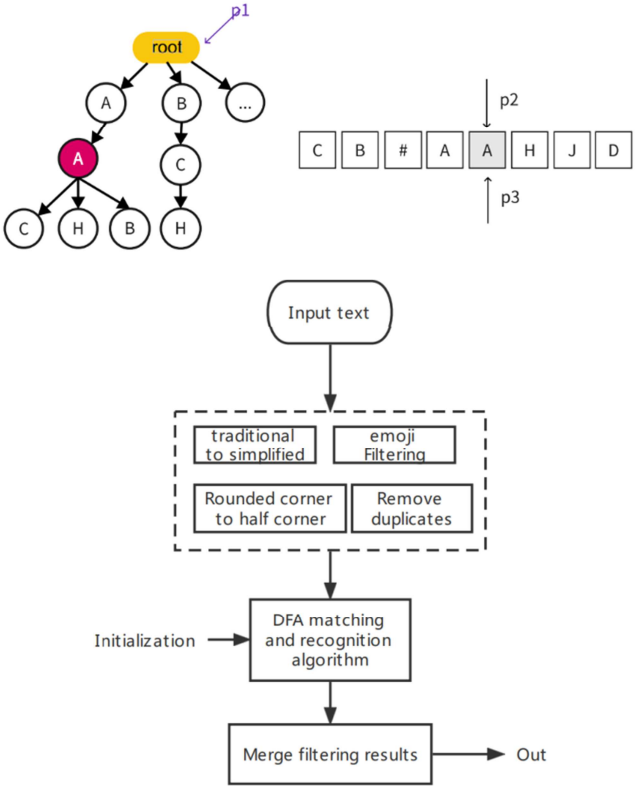

Figure 5. *Filtering process.*

4.3. Text Filtering Process

For the purpose of text filtering process, the system indexes a number of names. Names like "A", “B”. The main goal will be to filter names in a text box, and to do that really fast.

As the user types in a name, we need to filter them. As the user types, we can use the Trie as follows. The process includes preprocessing, matching, and integrating the output that are shown in Filtering process (Figure 4).

(1) Pre-processing: such as text word writing, pre-processing word AAO and then filtering,

(2) Trie tree matching: Find words in the text,

(3) Merged output: Obtain more information of the word from the hash table according to the word ID of the hit word, it is integrated into further processing such as counting the number of the hit words, and then output is the result (Figure 5).

4.4. Change Process

(1) Establish three references p1, p2 and p3. p1 points to the root of the Trie tree, and p2 and p3 point to the beginning of the text (Figure 6).

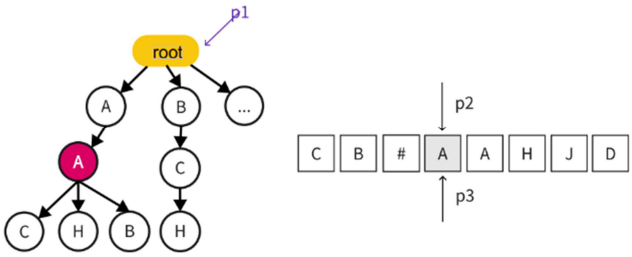

Figure 6. *The root of Trie tree.*

(2) The first character C does not exist in the children of the root node, and moves p2 and p3 to the next character (Figure 5).

(3) The character B of p2 exists at the root node of the Trie tree. Therefore, move p1 to node B below the root node of the Trie tree. Move p3 to the next character # of B, where # is A nonsense character, and move p3 to the next character a of # (Figure 7, Figure 8).

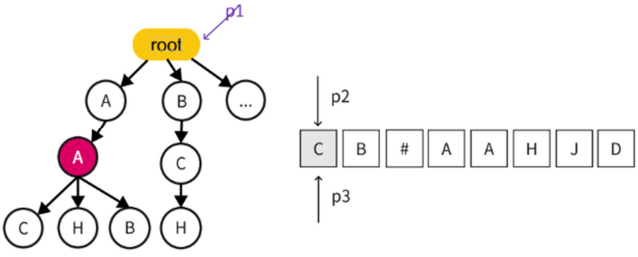

Figure 7. *The root expression.*

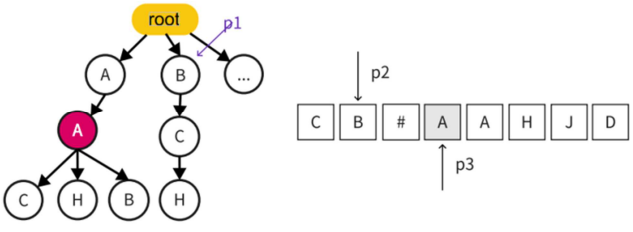

Figure 8. *Change in the second character.*

(4) The character A at p3 does not exist in the subtree of p1, that is, there is no matching character. In this case, p3 points to the next character A of p2 (skip the # character without meaning), and p1 points to the root node (Figure 9).

(5) If two characters A exist in the child node at p1 (Trie root node), move p3 to the next character and p1 to the child node A (Figure 9).

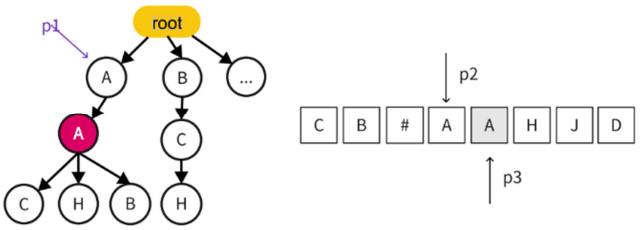

Figure 9. *The root of Trie tree, changes.*

(6) The character A at p3 exists in the subtree of p1. Move p1 to the child node A of p1 where the ending mark exists and the word AA is hit. Record the position of the hit word in the text (start position 3, end position 4), and obtain the ID of the hit word from the content of the ending mark and record it. Move p3 to the next character H (Figure 10).

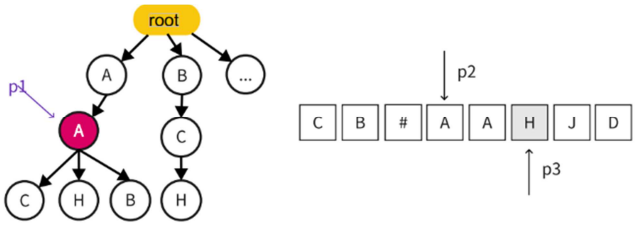

Figure 10. *Change in subtree.*

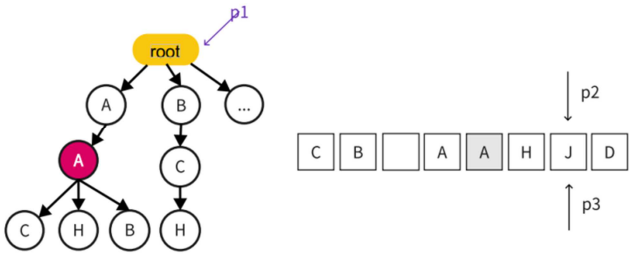

Figure 11. *Change in subtree, continuous.*

(7) The character H at p3 exists in the subtree at p1. If p1 is moved to the child node H, the node has the ending mark, the hit word AAH, and the relevant information is recorded. p2 moves to the next character A, p3 goes back to p2, and p1 points to the root of the Trie, from which the next round of matching begins.

(8) Similarly, character A at p2 exists at the root node, and p3 moves the next character H, and p1 points to the child node A. The character H at p3 does not exist in the subtree of p1, so p2 moves to the next character H, p3 points to p2, and p1 points to the Trie root node. Character H at p2 does not exist in the p1 subtree. Move p2 and p3 to the next character J, as shown in the figure above (Figure 11).

4.5. Word Change Process

When a word changes, the change message is broadcast over RocketMQ, each word service instance consumes the changed message, and the corresponding changes are made to the word cache and Trie tree. It should be noted that the consumption of the same word message should be ensured in an orderly manner and the consistency between the data on the word filtering service instance and the data in the word database should be maintained.

There are three kinds of change messages, which are added, modified, and deleted. New words are added to the word cache and then added to the Trie tree in the same process as initialization. Modification can be understood as removing old words and adding new ones. Deleting a word is similar to adding, except that the operation changes to removing the corresponding node from the Trie tree.

The deletion of words is mainly introduced here. In order to minimize the modification of the data structure caused by the deletion of words, only the ending tag of the word is removed in the actual deletion. This picture shows that Trie tree structure is shown after removing keywords AA, AAB and BCH (Figure 12 and Figure 13).

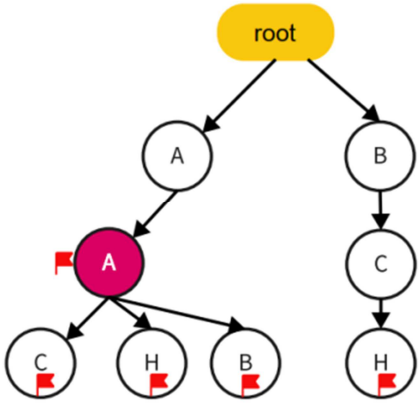

Figure 12. *Removing Keywords AA, AAB and BCH.*

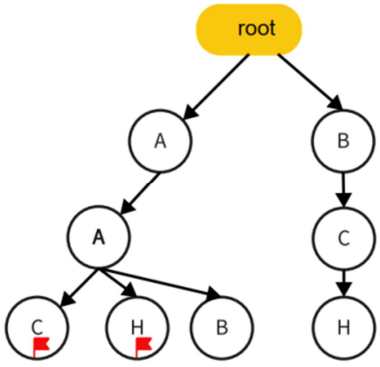

Figure 13. *Removing Keywords AA, AAB and BCH.*

4.6. Matching Steps

Thesaurus construction: To describe the two words good

morning and go to school, the lexicon is first constructed. This picture shows the binary tree construction of these two words is as follows (Figure 14).

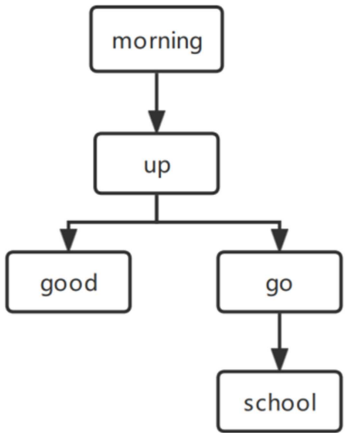

Figure 14. *The binary tree construction.*

This picture shows the hash table construct whose name is MyTreeMap: (Figure 15).

Figure 15. *Hash Table Construction.*

This picture shows Word filtering process. Assuming the keyword entered here is: Morning, the flow chart is as follows: (Figure 16). A trie is a tree-based data structure allowing the organization of prefixes on a digital basis using the bits of prefixes to direct the branching. Each node has at most two children in a binary trie [19]. Each prefix maps to a node in the binary trie of which the path and the level are determined by the prefix value and the length, respectively.

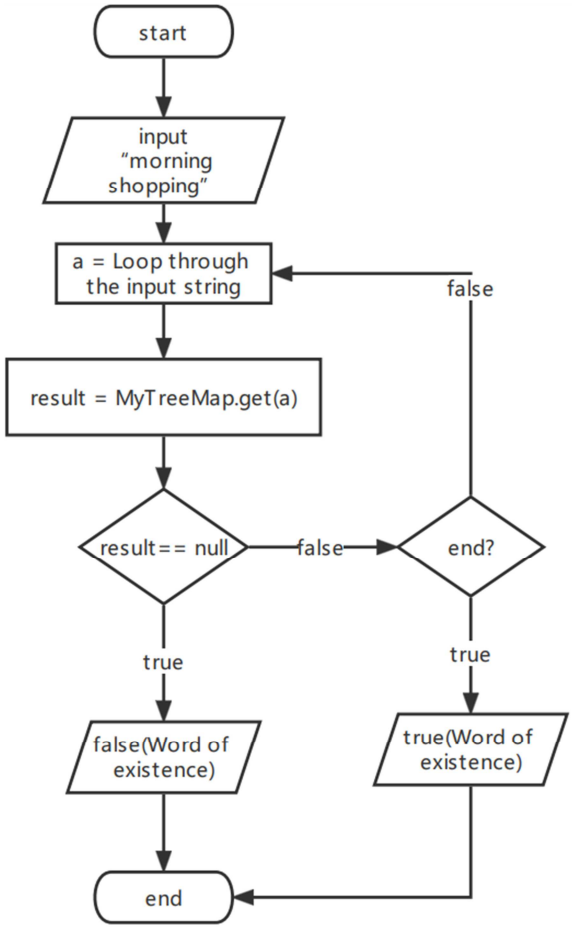

Figure 16. *Word Filtering Process.*

5. System Test and Algorithm Validation

5.1. Experimental Environment

In the test, we propose to capture relevant word data and imported it into MySQL database for storage. The experiment is based on the following environment (Table 1). The testing environment can help to do functional tests that is subdivided into three part including initialization related function testing, application deployment function testing, and application invocation function testing. Our experiment can be made before the expiration.

In general, the process of building requires special environment. We use a test environment that system functional test in this paper belongs to the black box test of software test [17]. This type of experiment mainly uses the infrastructure to test and verify the word detection system (Table 1).

Table 1. *Testing environment.*

Hardware	CPU	8 core E5-2650 v4 @ 2.20GHz
	Internal storage	16 GB
Software	Language	JAVA
	Operating system	CentOS6
	Docker	Version 17.05
	Database	MySQL Version 5.6
	RocketMQ	4.1.0

Files area used to store knowledge during a memory device for good. File handling provides a mechanism to store the output of a program during a file and to perform numerous operations on that [18]. We simulated an experiment-related scene, then captured relevant word data and imported it into MySQL database for storage. After building all the parts of the system, we write the Java program code and release it to the CentOS6 system with 16GB memory and 500GB hard disk. We prepare the Java environment in advance. This test mainly verifies the functions of adding, updating, deleting and matching words.

5.2. Function Tests

Functional tests verify that the system performs as expected when a particular operation is performed. Functional testing is further subdivided into three parts: initialization related function testing, application deployment function testing, and application invocation function testing. Among them. Initialization related functional tests include thesaurus, tree building, MySQL connection, RocketMQ and other tests [20]. The test results of related functions in the initialization phase are as follows (Table 2).

Table 2. *Relevant functional tests during the initialization phase.*

	Case description	Excepted results	Results
1	The program links to the MySQL database	The database is successfully connected	Yes
2	The program connects to RocketMQ middleware	RocketMQ was successfully connected. Procedure	Yes
3	The program initializes the tree tree	The matching tree is successfully created. Procedure	Yes
4	The program initializes the highlighting format	The highlighting mode was successfully loaded	Yes
5	Word expiration data	Not added to word tree	Yes

The results of functional tests about application calling are shown below (Table 3).

Table 3. *Functional tests of application calling.*

	Case description	Excepted results	Results
1	An unsubscribed user invokes the interface to initiate a request.	Call failed	Yes
2	The call interface passed parameter values of the wrong type	Call failed	Yes
3	The calling interface sends no ID	Call failed	Yes
6	Add a database, a table name, and a data table field to the database access control list	Access to these fields was denied	Yes
7	Add a brand new word	The request interface can match	Yes
8	Delete a word	The request interface did not match. Procedure	Yes
9	Set the expiring word	A match can be made before the expiration, but cannot be made after the expiration	Yes

5.3. Performance Test

In the performance test, we test by changing the amount of calling character data. DFA algorithm, AC algorithm based on prefix search pattern [21, 22] and MV algorithm based on suffix search pattern [23, 24] were used to detect the time consumed by the containing word content. Here it is. (*Table* 4).

According to the statistics on the detection time of the three algorithms in Table 4, the detection time increases with the length of the text content [25, 26]. The overall comparison of word detection time is as follows: DFA algorithm is the best approach compared to AC algorithm and other algorithms. The word filtering algorithm proposed in this paper has fast response speed and can be well applied to the network environment [27, 28].

Table 4. *Performance testing results.*

Word Count	Categories	Time Consuming (ms)
500	DFA	131
	AC	450
	MV	487
5000	DFA	243
	AC	810
	MV	870
50000	DFA	512
	AC	1202
	MV	1302
100000	DFA	1020
	AC	2180
	MV	2660
1000000	DFA	1821
	AC	3989
	MV	4121

6. Conclusion

In this paper, we implement a content matching technology based on improved DFA algorithm. The algorithm can retrieve or highlight the emoticon icon, half corner character, repeated word and so on. The word detection system in this paper uses ConcurrentSkipListMap to construct the Trie tree of the word filtering system. We found some valuable results in the results of functional tests, system tests, performance tests, etc. And verify the superiority of DFA algorithm. The system realizes the characteristics of large thesaurus and high matching efficiency of long text. It can meet the requirement of network real-time transmission, so it can be applied in the network.

References

[1] Zhao Junjie. A calculate way of rapid string precision used for keyword index matches. Computer Systems & Applications, 2010, 19 (2): 189-191.

[2] Kurniawan D H, Munir R. A new string matching algorithm based on logical indexing//Proc of International Conference on Electrical Engineering and Informatics. Piscataway, NJ: IEEE Press, 2015: 394-399.

[3] AHO A V, CORASICK M J. Efficient string matching: An aid to bibliographic search. Communications of the ACM, 1975, 18 (6): 333-340.

[4] WU S, MANBER U. A fast algorithm for multi-pattern searching. Tucson, AZ: University of Arizona,

[5] Liu Chuan, Wang Wenyong, Wang Meng, et al. An efficient instance selection algorithm to reconstruct training set for support vector machine. Knowledge-Based Systems, 2017, 116 (1): 58-73.

[6] Deng Yigui, Wu Yuying. Information filtering algorithm of text content-based sensitive words decision tree. Computer Engineering, 2014, 40 (9): 300-304.

[7] Chen Yongjie, Wushour·Silamu, Yu Qing. An improved multi-pattern matching algorithm based on Aho-Corasick algorithm. Modern Electronics Technique, 2019, 42 (4): 89-93.

[8] Guan Donghai, Yuan Weiwei, Lee Y K, et al. Improving supervised learning performance by using fuzzy clustering method to select training data. Journal of Intelligent & Fuzzy Systems, 2008, 19 (4): 321-334.

[9] Liu Lijun. Design and Optimization of DFA Word Segmentation Algorithm based on Keyword filtering system. Computer Application and Software, 2012 (1): 284-287.

[10] Majed AbuSafiya. Automata-based Algorithm for Multiple Word Matching. International Journal of Advanced Computer Science and Applications (IJACSA), 2021, 12, (3), 54-65.

[11] Cheng Yuanbin. Translating a kind of NFA into DFA straightly. Computer Systems & Applications, 2012, 21 (10): 109-113.

[12] Xu Qiang. Design and implementation of regular expression engines based on deterministic finite automata. Xi'an: Xidian University, 2012.

[13] Cavalcanti G D C, Soares R J O. Ranking-based instance selection for pattern classification. Expert Systems with Applications, 2020, 150: 113269.

[14] Pinkerton A, Boerhout J I, Bottalico T. Using an embedded web server to allow a standard multi-tasking operating system to manage, control and display live or recorded condition monitoring data from real time hardware: US, US14816238. 2016-03-31.

[15] Liu J, Bian G, Qin C, et al. A fast multi-pattern matching algorithm for mining big network data. China Communications, 2019, 16 (5), 121-136.

[16] Proux D, Cheminot E, Guerin N. Method and system for phishing detection: US, 11/443240. 2010-02-23.

[17] Zhao Wei. Research and Practice of Software Testing Strategy based on Black Box Testing. Management and Technology of Small and Medium-sized Enterprises (Upper issue), 2017, (01), 144-145.

[18] Ranjan R. College Database Management System, 2021.

[19] Lim H, Lee N. Survey and Proposal on Binary Search Algorithms for Longest Prefix Match. IEEE Communications Surveys & Tutorials, 2012, 14, (3), 681-697.

[20] Dong Mei, Chang Zhijun, Zhang Runjie. A multi-pattern matching algorithm for incremental data specification of scientific literature metadata. Data Analysis and Knowledge Discovery, 2021, 5 (6), 10.

[21] Pande A, V Pant, Gupta M, et al. Design Patterns Discovery in Source Code: Novel Technique Using Substring Match. TEM Journal, 2021, 10, (3), 1166-1174.

[22] Wu S, Manber U. A fast algorithm for multi-pattern searching. US: Department of Computer Science, 1994: 1-11.

[23] LeCun Y, Bengio Y, Hinton G. Deep learning. Nature, 2015, 521 (7553): 436-444.

[24] Xu Jianhua. Designing nonlinear classifiers through minimizing VC dimension bound/ /Proc of International Symposium on Neural Networks. Berlin: Springer, 2005: 900-905.

[25] Yao R, Cao Y, Ding Z, et al. A Sensitive Words Filtering Model Based on Web Text Features//Proceedings of the 2018 2nd International Conference on Computer Science and Artificial Intelligence. 2018: 516-520.

[26] Becchi M, Crowley P. A-DFA: A time-and space-efficient DFA compression algorithm for fast regular expression evaluation [J]. ACM Transactions on Architecture and Code Optimization (TACO), 2013, 10 (1): 1-26.

[27] Xue Pengqiang, Wushouer, Lamu. Sensitive Information Filtering Algorithm based on Network Text Information. Computer Engineering and Design, 2016, 37 (9): 2447-2452.

[28] Zhang Zhi-Yue. Research and implementation of website word monitoring system based on improved DFA algorithm.