Summary of Mors Project

Abstract

Measure of Report Similarity (Mors) is an education-oriented system for large-scale document comparison and plagiarism detection. Mors determines whether two documents might be similar by calculating the similarity on two aspects. First, comparison between local documents. For each documents, it extracts fingerprint, uses Lucene indexing to calculate similarity, and finally find out documents with high similarity. Second is the use of search API, to find out documents with high similarity on the internet. Thus, not only plagiarism between students, but also plagiarism between students and internet resources will be found. Mors will reduce teachers' workload and improve the teaching quality.

Mors was developed based on the Java technology due to its excellent portability, thus Mors is easy to deploy in different environment. We also provide web service to those who do not have the condition of deployment. Meanwhile, Mors is not requiring powerful computing performance, so it can be deployed in a normal server with low cost.

Key words Plagiarism detection, Fingerprint, Lucene, Web Service

Menu

Abstract 1
1. Brief Introduction to Mors 3
2. System Architecture 3
    2.1 Module Design 3
    2.2 Database Design 5
    2.3 Analysis of Key Technology 5
3. System Detailed Design and Implementation 6
    3.1 Detailed Design and Implementation of Report Manage Module 6
    3.2 Detailed Design and Implementation of Local Plagiarism Detection Module 8
    3.3 Detailed Design and Implementation of Web Plagiarism Detection Module 13
    3.4 Development Environment and Tools 15
4. System Testing and Performance Analysis 16
    4.1 Test Environment 16
    4.2 Performance Analysis 17
5. Funding Details 18
6. Thoughts and Feelings 19
    6.1 Project Leader Juncheng Liu 19
    6.2 Co-Researcher Guohua Tang 19
    6.3 Co-Researcher Guo Jiang 19
7. Conclusion 20
    7.1 Project Completion 20
    7.2 Characteristics and Innovation 20
    7.3 Future Work 20
Thanks 21
References 21

1. Brief Introduction to Mors

With the popularity of the internet, both submit and mark methods of homework have changed a lot. On the one hand, electronic homework can save students' time and is convenient for teacher to mark. On the other hand, the easy-to-get internet resources and the easy-to-copy electronic homework bring greater convenience of plagiarism. How to automatically and effectively detect plagiarism through technical means has become big problem of current teaching.

Plagiarism detection gradually led to attach great importance to the academic community, and The 1st International Competition on Plagiarism Detection [1] held in 2009, its purpose is to measure performance of a variety of plagiarism detection algorithms when processing large-scale data.

Stanford University's Moss (Measure of Software Similarity) [2] is an easy-to-use programming language plagiarism detection system. It is mainly used to calculate the similarity between programs written by C, C + +, Java, Pascal, Ada, ML, Lisp and some other formal languages. Moss has been successfully used in a number of programming courses of School of Computer Science and Technology, Harbin Institute of Technology [3]. However, Moss applies only to formal languages which have certain grammar and does not apply to natural language which has no certain structure. That is, Moss is ineffective on plagiarism detection of homework papers.

The CHECK system [4], established by Si at Hong Kong Polytechnic University, uses keywords' statistical method to measure the similarity between texts. The time complexity is O (n^2), it is not only inefficient for cases with a large number of reports but also more difficult to find plagiarism in Internet situation.

Gang Zhang, who has proposed an algorithm of large-scale elimination of similar web pages [5], used to elimination similar documents in large-scale file systems. The algorithm put forward the concept of text signature, which is based on string matching method to measure the similarity between documents. Our project it to use a similar algorithm, will not only improve the efficiency of comparison between similar reports, but also applies to finding plagiarism text from the Internet. Therefore our project mainly concerns about the algorithm to realize a highly efficient report similarity detecting system.

The project base on the above background, combined with years of teaching practice and research experience to develop a report similarity detecting system. The project is primarily concerned with electronic articles, whose content is a natural language text rather than formal language code, including text, HTML, PDF and DOC format. It can ensure quality of electronic papers to a certain extent, laid the foundation for full play of the student's personality and innovation.

2. System Architecture

2.1 Module Design

The core functionality of Mors is computing similarity of natural language text. Mors determines

whether two documents might be plagiarized on two aspects. First is to find similar documents in specified documents set, second is from the internet aspect, to find local documents and web pages which have similar content. Mors is divided into three main modules; system architecture is shown in figure 1.

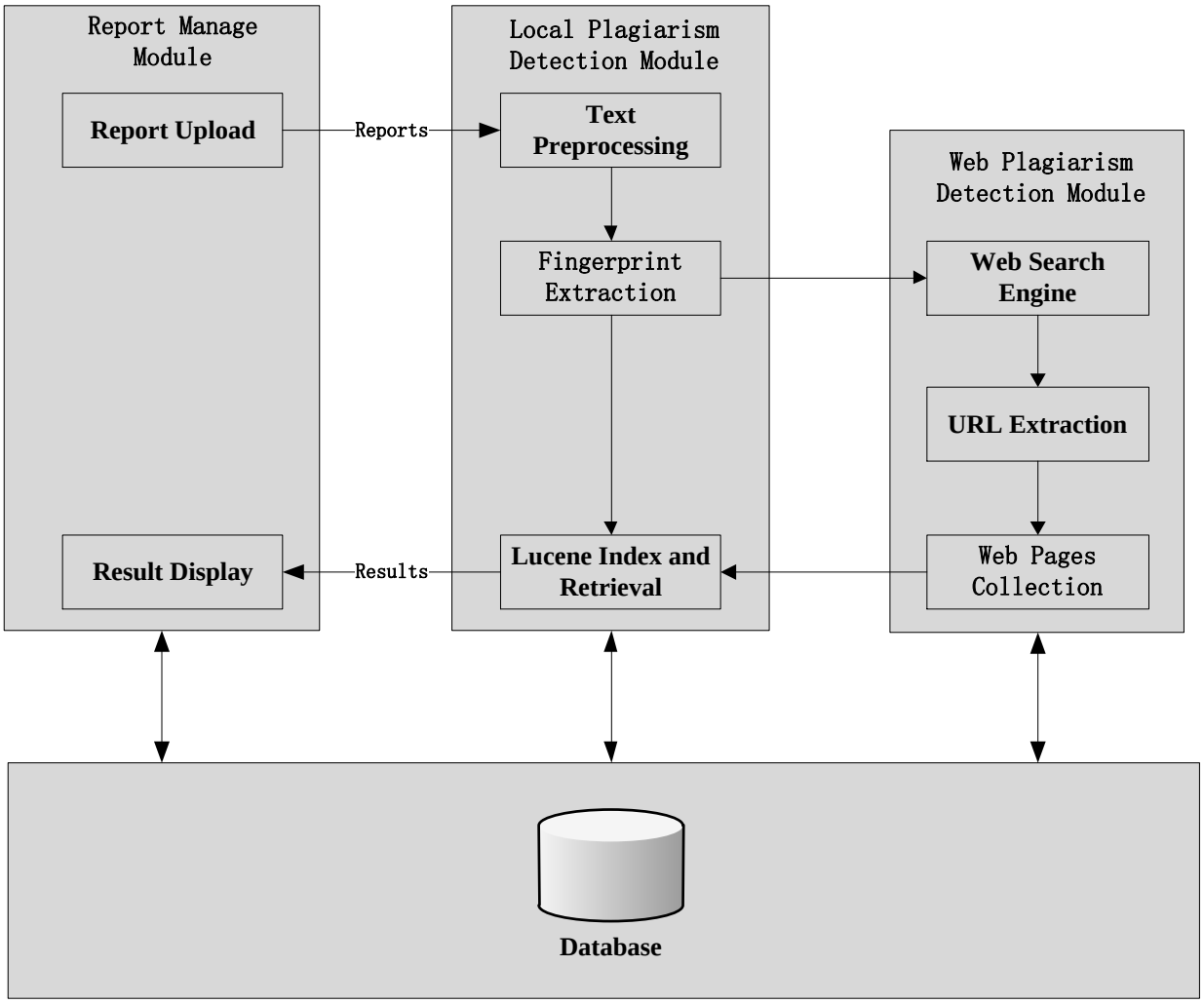

Figure 1: System Architecture of Mors

(1) Report Manage Module

The sub module of report upload provides an interface for users to transmit reports to server; The sub module of results display first get suspected plagiarized documents pair from database, and then calculate the detailed similarities between the two documents by second comparison algorithm, and at last display in a friendly manner.

(2) Local Plagiarism Detection Module

The central process of similarity detecting among documents at local system is as follows: step one, format the uploaded documents into the form of text file while preprocessing; step two, establish inverted index for the documents having been preprocessed; step three, extract the signature randomly and remove the intermediate format after that; step four, with the extracted signature, do search in the established inverted index and calculate the similarity; step five, sort the files by similarity; at last, display the results.

(3) Web Plagiarism Detection Module

Network part of the similarity calculation is based mainly on the Search API provided by search engine (Microsoft Bing). Deploy the search API to the server at the first. Then call the Web Service of a search engine in accordance with relevant protocols. The process is as follows: first, reading from database the feature vector which has been extracted, repartition them, and then recombine; second, put the recombined feature vector into Web search engines, then compare the URL extracted from the results of search engine with URL already existing; then analysis the frequency of the URL to accomplish the similarity detecting.

2.2 Database Design

According to the requirements analysis, and take into account specific functions of Mors, there are four tables in database together. E-R diagram of database is shown in figure 2.

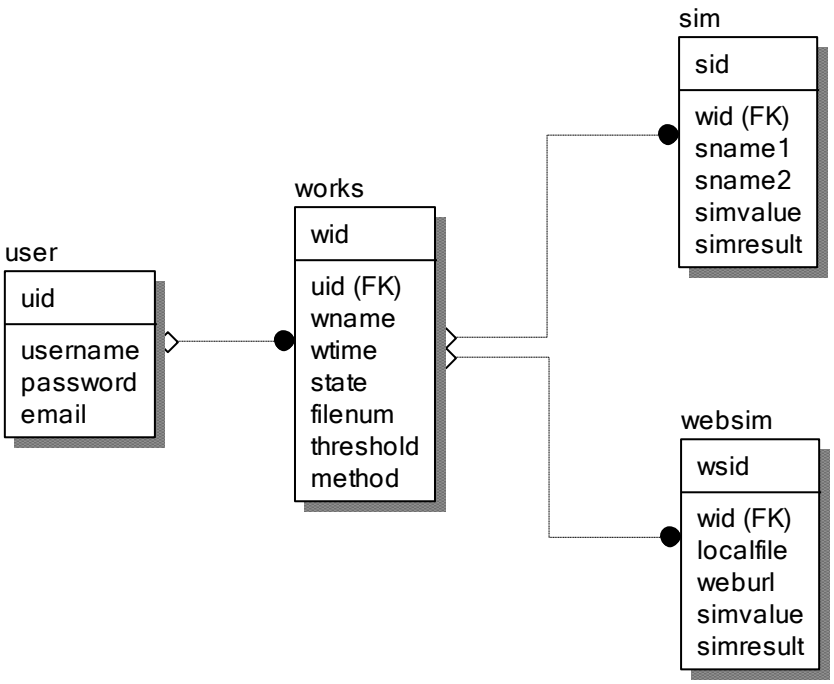

Figure 2: E-R Diagram of Database

2.3 Analysis of Key Technology

(1) The Definition of Plagiarism

It is very difficult to use quantitative indicators to determine plagiarism, but a common approach is to remove the text title, summary and contents notes, etc., if 10% of the content is similar, we definite it is plagiarism. The indicators may be adjusted according feedback from teachers.

At the same time, taking into account the context of the practical application of the different collections for each document, teachers can customize to determine the threshold. The system is only to find literally identical, and not able to recognize copy of ideas. The outcome of the system is a recommendation only; the system will eventually submit similar document to the teacher, the

ultimate determination is done by the teacher.

(2) Text Preprocessing

The first step of text preprocessing is format conversion, that is, convert documents submitted by students in HTML, PDF, MSWord, XML, RTF and other formats into text files, in preparation for processing and analysis of the next step.

For the text files, we need to perform Chinese word segmentation. Word is the basic unit of Chinese, but because Chinese is not written with word segmentation like English, therefore when processing Chinese with computers, word segmentation need to be carried out first, which is the most basic and core problem in natural language processing.

(3) Lucene Full-Text Indexing and Retrieval

Plagiarism determination mainly makes use of the Lucene full-text indexing and retrieval technology, including the establishment of inverted index and inverted index search. Lucene's search principle is to search the index, a typical method of space for time. Therefore index need to be established first. Searching is only operated on index, thus we can quickly get search results. Compared with the retrieval speed of return, paid the price of space is insignificant, because the disk space occupied by the text file is small, and the index file size is only about 30% of the contents.

(4) Web Service Technology

Similarity comparison among the web pages mainly using the web services of search engines, Web services is an XML-based software technology; it provides a standard for communication between the procedures and operation.

Web services defines several protocol specification to achieve the object visit under loosely coupled, the main technical standards are as follows: HTTP (Hyper Text Transfer Protocol Hypertext Transfer Protocol), XML language (eXtensible Markup Language), SOAP (Simple Object Access Protocol), WSDL (Web Service Description Language).

3. System Detailed Design and Implementation

3.1 Detailed Design and Implementation of Report Manage Module

Report manage module is divided into three sub-modules, User Management, Report Management, and Result Display. The primary function of User Management is user registration, login, and view and modify of personal information (name, password and email).

After the user login, it can add a new task, set the similarity threshold, and choose the detection type (local only, web only, or both local and web). These settings can also be viewed and modified later. User can upload files (one or more times) before starting the task. When the user clicks "start" button of a task, a record will be inserted into database, and the settings and files cannot be changed ever since. Daemon will keep query the database and fetch tasks and

executive one by one. For the completed task, user can view the detection results. The program flow chart of Report Management Module is shown in figure 3.

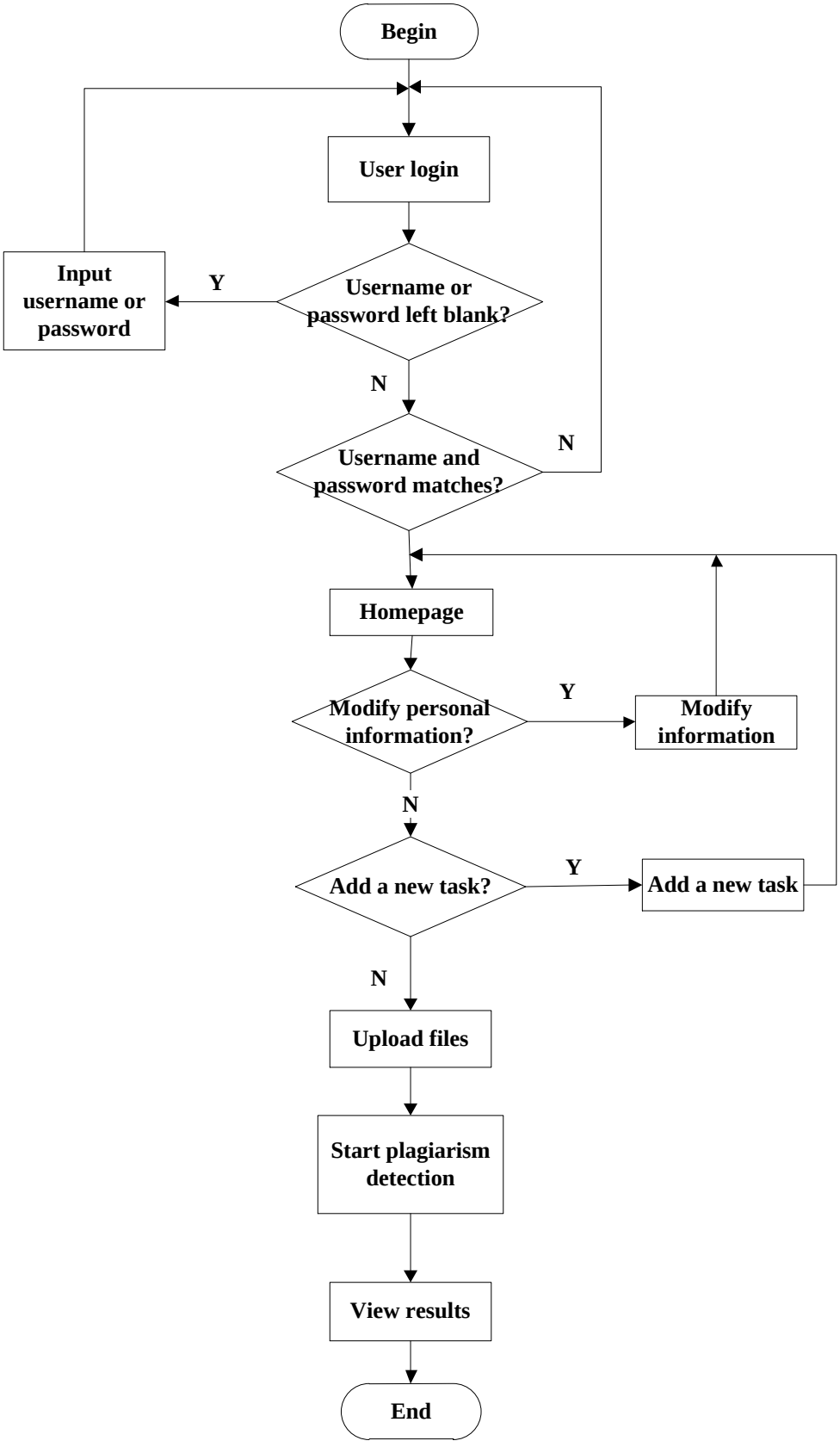

Figure 3: Flow Chart of Report Manage Module

3.2 Detailed Design and Implementation of Local Plagiarism Detection Module

Similar detection in local file system mainly includes text pre-processing, feature extracting, Lucene indexing and retrieval, similarity calculation and sorting. Local module of MORS is mainly based on the Lucene toolkit for similarity calculation to get similar reports. Lucene is a high-performance, full-featured text search engine library written entirely in Java. It is a technology suitable for nearly any application that requires full-text search, especially cross-platform [6]. By implementing its interface, application can easily achieved in the full-text index. The System Architecture and Code Organization of this module as shown in Figure 3:

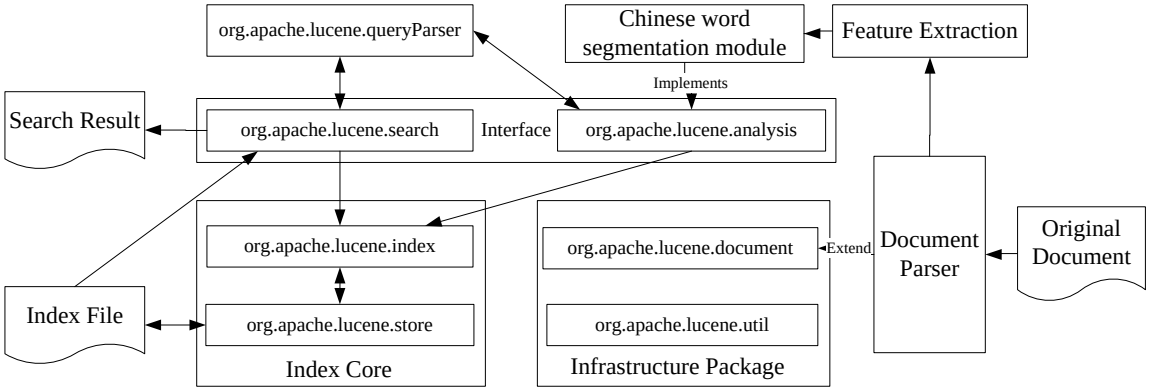

Figure 4: System Architecture and Code Organization Chart

The main process of similarity calculation within local file system is as follows:

First is the text pre-processing. Text pre-processing is mainly for the text extraction of documents submitted by students in HTML, PDF, MSWord, XML, RTF and other formats, to prepare for the following feature extraction and the creation of inverted index.

Next, the establishment of the original text inverted index. We use the “Path” and “Content” field of a Document, where “Path” field to save the path of the original text, “Content” field to save the text body.

Third, the text feature extraction, as well as similar text retrieval. The main process of feature extraction is randomly extracting signatures of certain length and number, and then all of the signatures be combined to form feature vectors. Next, put the feature vectors into the index files to search. The searched result is sorted according to Lucene scoring mechanism. Search result is a closed document collection, within which the similarity can be calculated. The last step is displaying results.

The Flow Chart of the local module as shown in Figure 5:

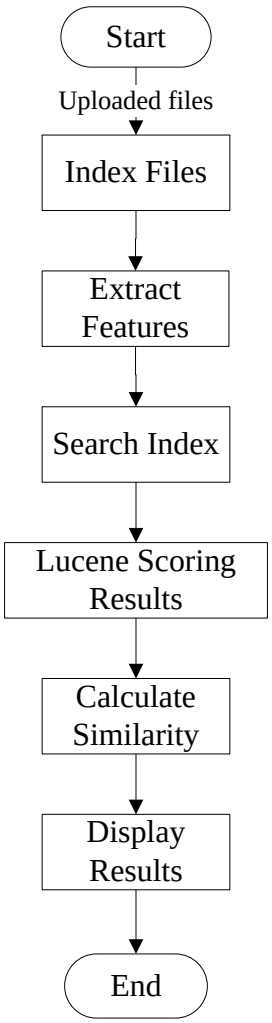

Figure 5: Flow Chart of Local MORS

3.2.1 Text pre-processing

Text pre-processing is mainly for various formats of documents of text extraction, currently with MS Word, PDF, HTML, and TXT supported. The following takes MS Word, PDF, and HTML as examples to introduce the text extracting method of MORS.

(1) MS Word

Text of MS Word format is parsed by the open-source POI toolkit 3.6. Part of the key code is as follows:

```
POITextExtractor extractor = ExtractorFactory.createExtractor(file);
content = extractor.getText();
```

(2) PDF

Text of PDF format is parsed using PDFBox technology. PDFBox is an open source Java PDF library. The library allows users to access the information of PDF files. Through the use of PDFBox API,

text can be extracted from PDF files. Part of the key code is as follows:

```
PDDocument document = PDDocument.load(file);
PDFTextStripper stripper = new PDFTextStripper();
StringWriter writer = new StringWriter();
stripper.writeText(document, writer);
content = writer.toString();
document.close();
```

(3) HTML

With the API provided by HtmlParser, an open-source toolkit, text information of html files can be well extracted. Part of the key code is as follows:

```
Source htmlSource = new Source(new FileInputStream(file));
content = htmlSource.getRenderer().toString();
```

The Flow Chart of text pre-processing as show in Figure 6:

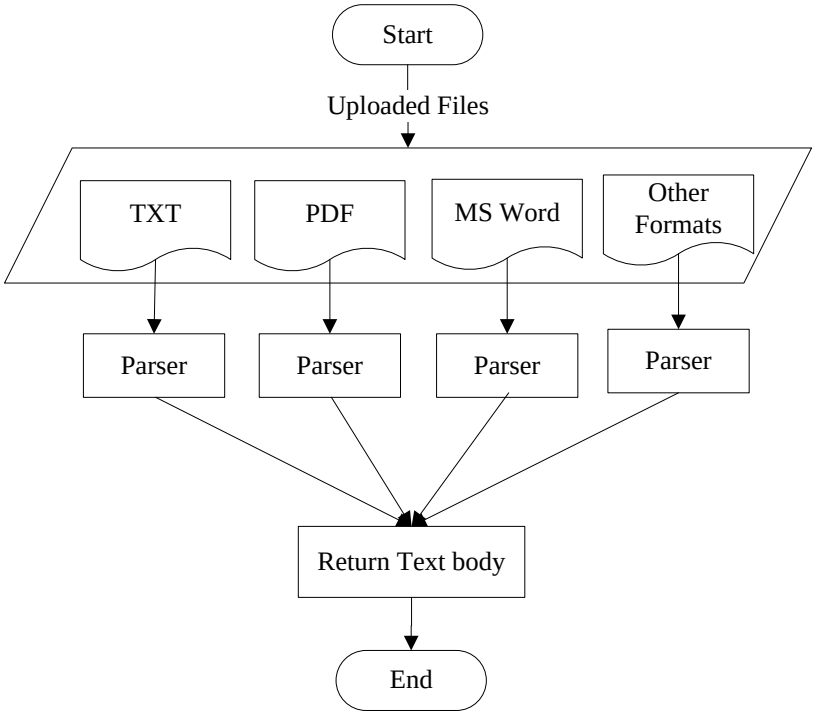

Figure 6: Flow Chart of text pre-processing

3.2.2 The Establishment of Inverted index

The process of indexing is to convert documents of different file formats into searchable index files to facilitate the query. The similarity comparison process is actually a search process with a document keywords searched in a large number of documents. If there is no index of the documents, you need to read the documents into memory orderly, and then with the

string-matching algorithms, to check whether the document contains the keywords being searched, which will cost much more time. The purpose of indexing is to random-access the keywords stored in the index, and then find the documents associated with the keywords. Lucene uses Inverted index mechanism. Inverted index, also known as reversed index, maintains a word / phrase table. For each word / phrase in the table, there is a list contains all the documents with which the word / phrase is associated, as well as the appear-frequency. Search for inverted index can quickly get search results.

The inverted index mechanism as show in Figure 7:

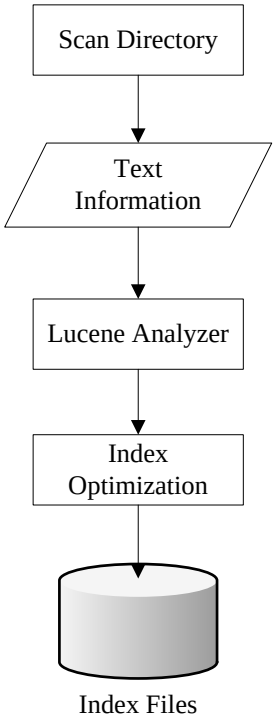

Figure 7: Inverted index mechanism

Based on the index mechanism and toolkits defined by Lucene, as well as Java itself, we managed to establish inverted index for files in directory, and then stored the index files in a separate folder on server.

3.2.3 Signature extraction and similar text detection

Signature extraction is mainly based on randomized algorithms and de-emphasis algorithm for large-scale web pages. Using randomized algorithms can reduce the contingency, and through multiple random signature extraction for similarity calculation, the accuracy of similar-detection will be improved. As we know, the general clustering problem is that elements with similar characteristics form a class, while the similar-detection problem depends on a certain threshold of similarity. Therefore general clustering method is not appropriate. Instead, we determine the similarity according to the signature-based vector similarity calculation.

Take into account that a document should contain punctuations such as full stop, question mark, etc. The feature extracting algorithm first chooses a full stop as an extract position, and then extracts strings of words of L/2 length on both sides of it as a signature. The definition of

signature and the parameter L play important roles. Signature that is too long will bring much more cost in storage and calculation, while signature too short will reduce the ability of recognition. In general L can be taken as 10. In accordance with the definition of N-Gram in information theory, this signature is equivalent to a 10-Gram. Assume that there is totally 6763 words in Chinese, the probability for this signature to be repeated is about to be $1/(6763)^{10}$. Therefore, guaranteed the signature specificity in the document.

After extracting the signature, put it into the index files to search using the Lucene search engine. The searched result is sorted according to Lucene scoring mechanism, and documents are determined to be similar according to the predefined similarity threshold. If the similarity of a pair of documents is greater than the threshold, the pair will be determined similar, and written into database.

Lucene scoring uses a combination of the Vector Space Model (VSM) of Information Retrieval and the Boolean model to determine how relevant a given Document is to a User's query. [7] Formula of Lucene scoring mechanism is as follows:

$$score(q,d) = \sum_{\text{t in q}} \text{tf(t in d)} \bullet idf(t) \bullet \text{boost(t.filed in d)} \bullet \text{lengthNorm(t.field in d)} \quad (2\text{-}1)$$

Factors in the formula shown in table1

Table 1: Factors in Lucene scoring formula

Scoring Factor	Description
tf(t in d)	Frequency of lexical item “t” in document “d”
idf(t)	Frequency of lexical item “t” in inverted documents
boost(t.field in d)	Weighting factor of field “t”, set in the indexing process
lengthNorm(t.field in d)	Standardization of field “t”. usually calculated in the indexing process and stored in the index
coord(q,d)	Coordination factor， number of signatures based on documents
queryNorm(q)	Standardization value of each query. sum-of-squares of the weight of each signature

As the similarity of the document and the document itself must be 1, that is 100%, so the calculation of the search should exclude the document itself and just consider others, which is just a small pruning of the search algorithm. The search algorithm’s time complexity is $O(n^2)$.

Take into account that the threshold set too small will reduce the precision, while threshold too large reduces the recall; the system set the threshold value of 0.5.

The flow chart of similarity calculation as shown in Figure 8:

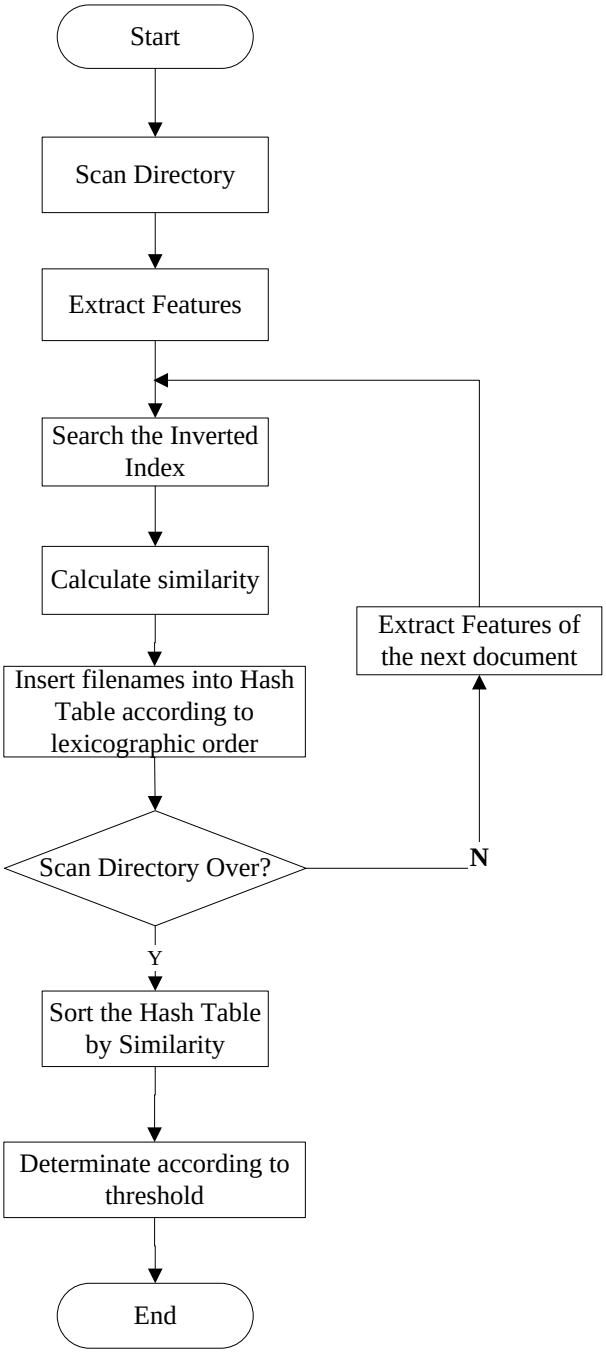

Figure 8: Flow Chart of Similarity Calculation

3.3 Detailed Design and Implementation of Web Plagiarism Detection Module

The web plagiarism detection module mainly depends on the Web Service API provided by Microsoft's main search engine Bing. By calling the Bing search API with the signature extracted from the reports as request parameters, Bing web service returns XML result which contains the title, summary, URL, date and other information of the web pages from Bing searched result. Next, parse the XML file (in MORS, the open-source Dom4J toolkit is used) and extract all of the

URLs. Then, with the URL addresses and Java network toolkit, download the relevant web pages to local file system. The next will be similar to local detection module.

The main steps are as follows:

(1) Extracting the signature of documents. Take into account the randomness of the signature extracting algorithm, so we extract 10 different signatures for each document, thus, to increase the web pages hit rate.

(2) Accessing the web resources. Get the XML result via calling the Bing search API with the extracted signatures as request parameters.

(3) Parsing the XML result of web service. We mainly use the dom4j technology, which is an open-source project.

(4) Collecting web pages. By parsing the XML result of web service, we get a collection of URL addresses. And then, download, if available, all the corresponding web pages and save them at local file system.

(5) Building inverted index. Pre-process the web pages and build inverted index.

(6) Similarity detection. Extracting signatures for every report submitted by students, and search them in index files via Lucene search engine. The process is almost the same with the local similar detection.

The flow chart of Web Plagiarism Detection as shown in Figure 9:

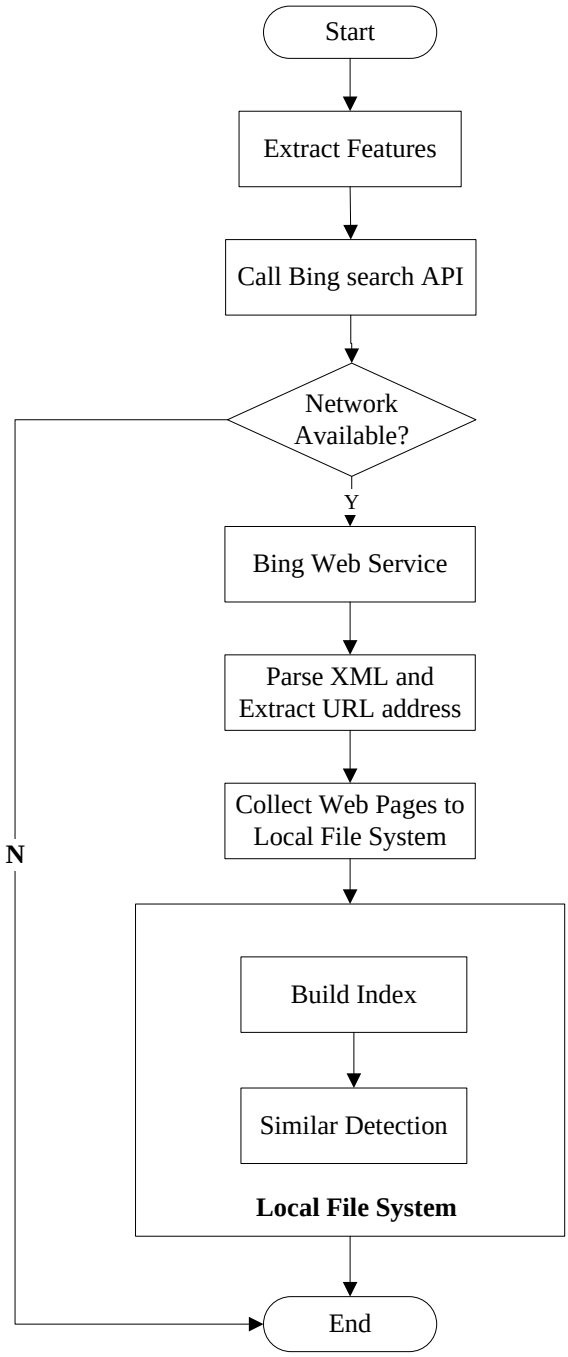

Figure 9: Flow Chart of Web Plagiarism Detection

3.4 Development Environment and Tools

(1) JDK 1.6

Java is an object-oriented language; it is more suitable for the development of Mors than the process-oriented language. There are many high quality Java libraries, Lucene, for example; the use of these libraries can greatly simplify the development. JDK is short for Java SE Development Kit, Mors uses the latest JDK 1.6, and so the project can be easily deployed in different environment due to Java's portable ability.

(2) MySQL 5.1.10

Mors uses MySQL as the data center. MySQL is the most popular open source SQL database management system, with small, fast and low cost advantage. Open-sourced MySQL can not only be used free of charge, but also have a good technical support in open source community. The use of JDBC allows the system and MySQL database easily exchange data.

(3) OpenSolaris 2009.06

Open Solaris is an open-source projects including Solaris operating system. User can set up a free safety, and high-performance operating system with Open Solaris. The system will build on the Open Solaris operating system. However, other users can deploy the system on any operating systems which support Java and MySQL.

(4) NetBeans 6.7.1

NetBeans is full-featured open-source IDE (Integrated Development Environment); can help developers to write, compile, debug and deploy Java applications. NetBeans integrated version control and XML editor among its features; we will use Netbeans to develop Mors.

(5) GlassFish 3.0

GlassFish is a free, open-source Java application server, is a robust business-compatible application server to achieve product-level quality, it can be used free of charge for the development, deployment and re-distributed. In the static file transfer aspects, performance of GlassFish is much stronger than Tomcat, and can support more concurrent access. We use GlassFish as application server.

4. System Testing and Performance Analysis

4.1 Test Environment

Mors kernel was developed using java, and corresponding web site was built on OpenSolaris and Glassfish. The deployment detail can be found in the deployment instructions which submitted together with the report. Test environment is as follows.

(1) Software Environment

Software needed to run Mors:

- OS: OpenSolaris
- Software: Java Runtime Environment (JRE), Java SE Development Kit (JDK) 5.0+
- Web Server: Glassfish 3.0
- Database: MySQL 5.1.10

Be noted that, due to Java language's feature of cross-platform, Mors can be ported to Windows, Linux and other OS, and will not bring cross-platform problems.

(2) Hardware Environment

Minimum hardware required to run Mors:

- CPU: Pentium IV, 550MHZ
- Memory: 1G

Be noted that, as Mors need high memory to run, so more than 1G memory recommended, 2G better.

4.2 Performance Analysis

(1) Document Collection Used

We used report from three computer courses as the test document collection. They are refered as Course A, Course B and Course C as following.

(2) Test Method

We take voting mechanism to evaluate the results manually. With a group of three members, for example, if more than (including) two think that the result is correct, then the result is considered to be correct. For the same reason, we cannot get the recall, so using accuracy as a performance measure.

(3) Test Result

The performance of the three courses is as the following table:

- **Local Detection**

Table 2: Course A

Number of Reports	Average words	Threshold	Plagiarized Reports found by Mors	Plagiarized Reports Checked by voting mechanism	Accuracy
2397	627	0.5	3235	3193	98.70%

Table 3: Course B

Number of Reports	Average words	Threshold	Plagiarized Reports found by Mors	Plagiarized Reports Checked by voting mechanism	Accuracy
994	2696	0.3	3222	3195	99.16%

Table 4: Course C

Number of Reports	Average words	Threshold	Plagiarized Reports found by Mors	Plagiarized Reports Checked by voting mechanism	Accuracy
183	1570	0.3	53	53	100%

- **Web Detection**

Table 5: Course B

Number of Reports	Average words	Threshold	Plagiarized Reports found by Mors	Plagiarized Reports Checked by voting mechanism	Accuracy

994	2696	0.3	26	26	100%

Table 6: Course C

Number of Reports	Average words	Threshold	Plagiarized Reports found by Mors	Plagiarized Reports Checked by voting mechanism	Accuracy
183	1570	0.3	175	170	97.14%

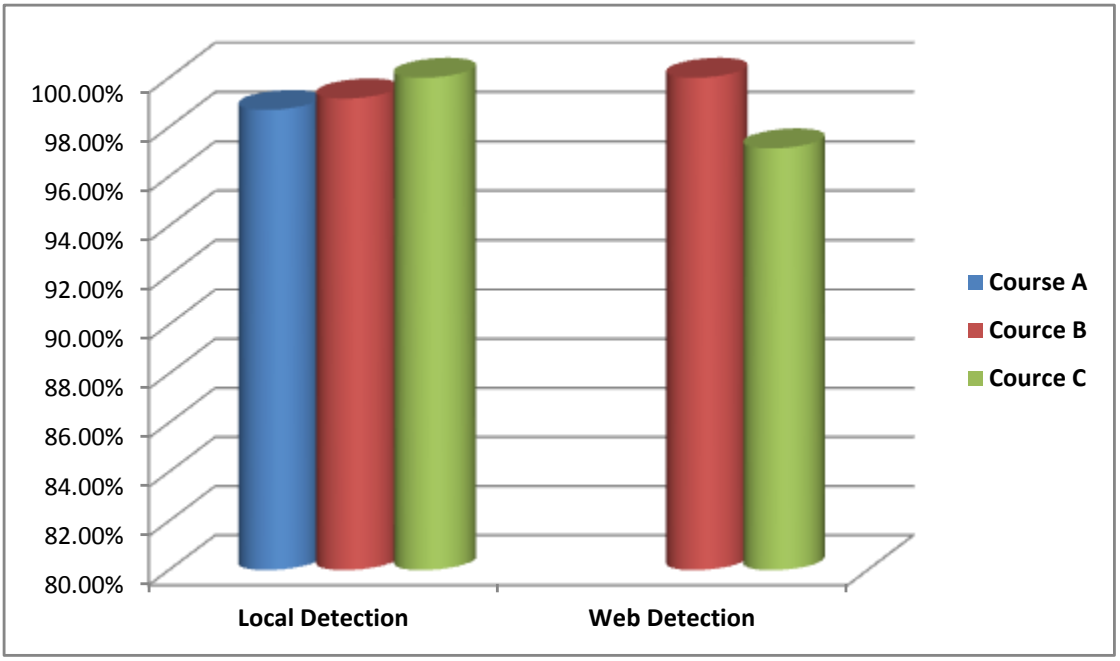

Figure 10: Histogram of Test Results

(4) Performance Analysis

Local results of Course A have very large number of plagiarized documents, it is because that the report of Course A is in Q&A form, and the answers are in high similarity with each other. So for such reports the threshold can be set higher.

Contrast the results found by Mors and checked by voting mechanism, Mors perform well both in local and web detection, the accuracy were above 97% at all of three test data, with very low rate of misjudge.

The three test data we choose are from representative computer science courses. According to the results, Mors has a good compatibility with different courses.

Therefore, the test results show that Mors has good practicability.

5. Funding Details

The Mors project has a total funding of ￥15000, and ￥14975 was used. Details are as following table.

Table 7 Details of Project Funding

Item	**Funding**
Server	￥9000
Books and reference materials	￥5000
Printing, copying and internet access	￥975
Total	**￥14975**

6. Thoughts and Feelings

6.1 Project Leader Juncheng Liu

Once thought, innovation is just racking brains to think of a good idea. But now I finally understand that innovation is not just a good idea. And innovation that been tested in practice is the real innovation. In the process of the project, not everything is smooth. We encountered many difficulties. But good ideas and elegant solutions are often come up at the most difficult time. To begin innovation, the first is to move beyond oneself. People who have not practice will never know what real innovation is.

6.2 Co-Researcher Guohua Tang

In the process of the project, the deepest feeling is curiosity of unknown areas. When a problem is unsolved, we do not know what problems will encounter later, we have to explore gradually. In the exploration, we can often encounter a variety of strange situations. At this time, what we need is curiosity and tenacity. The darkest time of the project is a time for a major breakthrough.
The process of the project is also the process of personal growth, on both academic and practice aspects. Hope that the innovative program will be further promoted, so that more students can participate in this meaningful event too.

6.3 Co-Researcher Guo Jiang

I was responsible for the part of text comparison and similarity calculation of the project. In my work, I tried to reduce the coupling and increasing the cohesion between the various independent modules, and use the strategy pattern to enhance the degree of code reuse. At the same time, efficiency is particularly important in this part. Multi-threaded web search, minimize IO operations, which are the key to improve time efficiency. Because of Java's automatic garbage collection mechanism, memory management becomes a lot simpler. But we still encountered the problem of memory overflow. So optimize the space efficiency is also very important. In the process of the project, Dr. Che gave us a log of valuable guidance. Communication or controversy between teammates also makes some problems be solved. Now Mors has been completed, I'm

sure Mors will withstand the test of practice, and eventually become a practical system.

7. Conclusion

This paper described the design and implementation process of Mors from aspects of introduction, overall design, detailed design, and testing and performance analysis.

7.1 Project Completion

Overall, Mors achieves the desired objectives. Be able to find plagiarized documents in specific documents set, display the paths and file names, and show in detail the specific similarities; be able to achieve web plagiarism detection at a higher accuracy, and display path of local file and hyperlink of web page.

7.2 Characteristics and Innovation

Different from academic papers plagiarism detecting system, Mors is designed for detecting reports plagiarism oriented the students in the field of education. Based on the characteristics of the reports, and through communication with experienced teachers, we set up a unique scheme of realization and criteria for judging for our system which will make it more suitable for teachers to determine the similar report in day-to-day teaching.

Mors is based on the search algorithm of which the time complexity is O (n), much lower than the traditional clustering algorithm with time complexity O (n^2). As the processing time will not significantly increase with the number of documents, the system is suitable for large-scale document processing, and allows users see the results in a relatively short period of time.

A second comparison module was added to Mors. Retrieval-based algorithm is used for the first time of comparison to identify possible plagiarized documents. Iteration of the LCS (Longest common substring) algorithm is used for the second time of comparison, to find out specific positions. The second comparison greatly shortening the running time and improve the efficiency of the system. At the same time can improve the comparison accuracy, display results in a friendly manner, thus led a better user experience.

At present, all the similarity detecting programs or systems are based on given collection of documents. There is not yet one system aimed at the entire Internet. Some systems claim that they can do similarity detecting in the Internet, however, what they do is just to download the relevant Internet resources to local beforehand for following comparison. Comparatively, our system is depending on search engine support to do search in the whole Internet to find identical reports.

7.3 Future Work

At the same time, the system also has some shortcomings to be improved. Firstly, taking into account that the text extracted from html file has lots of redundancy (such as JavaScript and

CSS code), we disable the feature of viewing detailed positions of plagiarism. Secondly, the function of current web site is relatively simple, for example, we only have one type of user, and maybe it's more proper to treat students and teachers separately. Owing to time constraints, these functions will be updated or completed in later versions.

Thanks

First of all, thanks our mentor Dr. Wanxiang Che. Dr. Che's rich knowledge, positive attitude to problems, dedication to science, is our example for life. From Dr. Che, we learned not only knowledge, but also, more importantly, ways of thinking and attitude to life. He pours care and help in every point of our advance. Here, great thanks to Dr. Che.

Thank Zhigang Sun and other teachers to provide test data and help test our systems, and provide recommendations for improvement; Thank other teachers and classmates who provide guidance, care and help during the project.

Special thanks to Sun Microsystems Inc. provided us with such a unique opportunity, allows us to show our talent to maximize extend. Thank Sun Microsystems Inc. to provide us so much powerful open-source tools and technologies, making our project completed smoothly.

References

[1]. http://www.uni-weimar.de/medien/webis/research/workshopseries/pan-09/index.html

[2]. http://theory.stanford.edu/~aiken/moss/

[3]. http://cms.hit.edu.cn/course/view.php?id=44

[4]. SiA, LeongHV, Lau RWH. CHECK: A doeument plagiarism detection system. In: proceedings of the ACM SymPosium for Applied Computing. 1997:70~77

[5]. Zhang Gang, Liu Ting, Zheng Shifu, Che Wanxiang, Li Sheng. Fast Deletion Algorithm for Large Scale Duplicated Web Pages. Singapore ICCC2001

[6]. http://lucene.apache.org/java/docs/index.html

[7]. http://lucene.apache.org/java/2_3_2/scoring.html