The Search Engine Weighted on Resource

Quality for Education and Learning

Youwei Huang  
School of Science, Software Engineering, Monmouth University, West Long Branch, NJ US 07764

S1296394

6/1/2020

# HIGHLIGHTS

1. Research search engine technologies used on learning resources.
2. Define the value of learning resources via quantitative methods.
3. Propose resources storage and rank in a search engine system.
4. Build a web application that supports searching of education and learning resources.
5. Design a verification method to locate high quality resources.
6. Apply user behavior and big data analysis to discover resource value.

Keywords: search engine, big data, education, learning resources, DLRV, RD

# ABSTRACT

Nowadays the most common and convenient way to look for information is searching on Internet. The top Internet search engines are handful, such Google, Bing, Yahoo, Baidu, and so on. They use web crawler technologies to sniff the whole visible network and then provide users simple indexes and links as the results to help users find the source of resources. Most of those Internet search engines have good performance in terms of speed and precise keyword search ability, but wide-range search engines have their cons, such as massive potential results with uncertain quality. The goal of this project is to propose a new search engine, targeting on education, to provide “high quality” learning resources to users. High quality could be an subjective judgement and limited to the resources available for searching. In this project, it refers to means “user-trusted” or “user-liked”, based on common practices in education and learning. A few new search engine technologies are proposed to support efficient storage and enhanced searching for high-quality learning resources, in particular, to address two main issues: (1) how to build such a high-performance search engine; (2) how to define the quality of resources. The details include the strategies designed to optimize general information querying, storage, ranking, and most importantly, finding the relevant learning resources that are “high-quality” for users. Prototyping and experimental study are conducted to conceptually prove this research.

[HIGHLIGHTS 0](#_Toc1667126164)

[ABSTRACT 0](#_Toc885799423)

[GRAPHICAL ABSTRACT 0](#_Toc704605653)

[1 INTRODUCTION 1](#_Toc704110189)

[1.1 Related Work 1](#_Toc1714198217)

[1.2 Web Crawler 1](#_Toc695741281)

[1.2.1 Googlebot 1](#_Toc1612442107)

[1.2.2 Bingbot 1](#_Toc228637218)

[1.2.3 Conclusion 1](#_Toc419471504)

[1.3 Distributed Storage System 2](#_Toc1874285451)

[1.4 Query and Rank 2](#_Toc2025839819)

[1.5 Resource Crawler 2](#_Toc1104356449)

[2 PROBLEMS 3](#_Toc1790727889)

[2.1 Problems in specific areas 3](#_Toc26875705)

[2.2 Problems to specific groups 3](#_Toc1427766779)

[3 SOLUTIONS 4](#_Toc1627091707)

[3.1 Resource characteristics 4](#_Toc1685749051)

[3.2 Query and Search Improvement 5](#_Toc804876229)

[4 DEFINITIONS 6](#_Toc1479825617)

[5 REQUIREMENT 6](#_Toc1944161708)

[5.1 Hardware requirements 6](#_Toc1314793512)

[5.2 Software requirements 7](#_Toc599527944)

[5.2.1 MVC Design pattern 7](#_Toc274309517)

[5.2.2 Related application software 7](#_Toc1853433421)

[5.3 Non-functional requirements 7](#_Toc2070078554)

[5.3.1 Performance 7](#_Toc1585352468)

[5.3.2 Reliability 8](#_Toc69679870)

[5.3.3 Security 8](#_Toc1979340131)

[6 DESIGN 8](#_Toc1236337363)

[6.1 Software Engineering Process 8](#_Toc1648483093)

[6.2 System deployment structure design 9](#_Toc1893749271)

[6.3 UML 11](#_Toc755979879)

[6.3.1 Database and E-R diagrams. 11](#_Toc386798868)

[6.3.2 Use Cases 12](#_Toc450871276)

[6.3.3 Package Diagram & Class Diagram 12](#_Toc1460090068)

[6.3.4 Sequence Diagram 16](#_Toc2100997086)

[7 Algorithms and Methods 17](#_Toc1146612557)

[7.2 RD Way 17](#_Toc925048528)

[7.2.1 Storage and Query 17](#_Toc182150656)

[7.2.2 Rank of Results 18](#_Toc1566084061)

[7.3 DLRV Way 20](#_Toc651850331)

[7.3.1 Disadvantages of traditional RD way 20](#_Toc60506827)

[7.3.2 Storage and Query 21](#_Toc522956863)

[7.3.3 Relational Database 22](#_Toc295094572)

[7.3.4 Non-Relational Database On Disk 22](#_Toc87382532)

[7.3.5 Non-Relational Database Cache 25](#_Toc1950723642)

[7.3.6 Three Layered Search 26](#_Toc1922186279)

[7.3.7 Cache Switch Method 28](#_Toc1773131583)

[7.3.8 Rank based on DLRV 30](#_Toc608116223)

[7.3.9 Static and Dynamic 30](#_Toc1254528248)

[7.3.10 Suitability 31](#_Toc1569809644)

[7.3.11 Reliability 32](#_Toc1922909735)

[7.3.12 Practicability 32](#_Toc1854056192)

[7.3.13 Feedback 32](#_Toc1844119161)

[7.3.14 Popularity 33](#_Toc1628859508)

[7.3.15 Cost 33](#_Toc1776651098)

[7.3.16 Static Value 33](#_Toc1281987981)

[7.3.17 Dynamic Value 33](#_Toc1698539379)

[7.3.18 General formula 33](#_Toc1608507582)

[8 TEST 34](#_Toc370841697)

[8.1 Purposes 34](#_Toc1199538824)

[8.2 Test Methods 34](#_Toc1354773205)

[8.2.1 Black-box Testing 34](#_Toc1126821576)

[8.2.2 White-box and Control Group 35](#_Toc1586337693)

[8.3 Test Conditions 35](#_Toc1805644482)

[8.3.1 Storage and Query Condition 35](#_Toc439427997)

[CITATIONS 37](#_Toc1539851131)

# 1 INTRODUCTION

There are many search engines available for online users. Some support wide range Internet search, such as Google, Bing, and Yahoo. Some are built within systems, such as YouTube, which has its own search engine; There are also search engines tailored to meet certain needs, like better privacy, copyright protection, and so on. For many years, and currently, Google has been dominating over 80% of market share. For this reason, Google is the main search engine considered for comparison in our research. Imagine there is someone interested in learning Java. One might Google “learn java”, and then a bunch of results are displayed: some labeled as ‘Ad’; some are videos; and many more other links, while you can keep clicking ‘see more’ to get more results. There are huge collection of results of different kinds, different sources, and certainly different qualities. It could be quite intimidating if learning java is something new for this user. Making a choice alone could become the big time-consuming step before actually starting to learn. We propose a new and unique search engine that targets educational resources, to promote and support self-learning. Providing “high quality” learning resources to users is the essential for this search engine. Here “High quality” is an subjective judgement from users, but based on common practices in education and learning. The more users are for it, the higher quality it is proved to be. Therefore, the score of quality is indeed limited to the resources that can be verified or tested by users. It will be an improving process over the time. High-quality can be understood as “user-trusted” or “user-liked”.

In this project, a few new search engine technologies are proposed to support efficient storage and enhanced searching for high-quality learning resources, in particular, to address two main issues:

1. how to build such a high-performance search engine.
2. how to define the quality of resources.

Figure 1 illustrates the overview of the major components prototype in the system. The processes are divided into four parts: collecting data (collecting), resources search (recommendation or rank), data analysis (improvement), verification of valuable resources (verification or test). Normally, search engines are based on Web 2.0 technology today. Here DLRV (degree of learning resource value) is the method to define and improve the definition of resources value (quality), which will be explained and discussed in the later chapters.



Figure 1 The general framework of the whole system and key technologies adopted by the search engine.

Finally, experimental and test study are conducted to conceptually prove this research. The details will be presented in the later chapters.

# 2 RELATED WORK

Search engines have been studied for many years. All vast majority of search engine providers try to design superior algorithms to rank the quality of links and strategies to improve storage-query speed. The following is some work related to most popular search engines.

## 2.1 Web Crawler

Web crawler is a core component of most search engines. Web crawler provides the function of data collection, which can update the database behind the search engine to ensure that the data for user query is up-to-date.

The result of crawling is a collection of websites at a central or distributed location.[[1]](#endnote-0) There are a few web crawlers behind current mainstream search engines. Here list two of them.

1. **Googlebot**

The most famous search engine as we know, Google, uses a kind of web crawler named Googlebot. Googlebot collects web pages and build a searchable index for Google Search engine. This name is used to refer to two different types of web crawlers: a desktop crawler (to simulate desktop users) and a mobile crawler (to simulate a mobile user)[[2]](#endnote-1).

1. **Bingbot**

Bingbot is a web-crawling robot (type of [internet bot](https://en.wikipedia.org/wiki/Internet_bot" \o "Internet bot)), deployed by [Microsoft](https://en.wikipedia.org/wiki/Microsoft" \o "Microsoft) October 2010 to supply [Bing](https://en.wikipedia.org/wiki/Bing_(search_engine)" \o "Bing (search engine)).[[3]](#endnote-2) Bingbot has the same principle and tasks with Googlebot. Bingbot collects web page information from Internet nodes and stores it in distributed system.

The work of web crawlers is very similar. They crawl the information of web pages from Internet nodes as the resource library content of search engines. The data obtained by these crawlers is provided by the meta information of HTML pages (always are the title, description, keywords). There are some characteristics for these search engine bots: (1) It is impossible to crawl all the data from the Internet; (2) Crawlers do not consider data correctness or quality; (3) Crawler is a kind of automated script. (4) Both of the bots have one thing in common, that is allowing users to block crawlers.

## 2.2 Distributed Storage System

Almost all search engines that need to use distributed storage system to store a large quantity of resources, Google is used as an example in this thesis.

Google gets billions of resources from the Internet crawler every day. Data storage in a search engine system needs special storage system. Google published some technologies of Google FS in 2003[[4]](#endnote-3), which is Google’s own file system format. Massive data must be stored in such special file system, and the original data on the server node can be found by one or some index servers.

At the same time, in order to ensure the data security, the same data have multiple backups stored on different distributed storage nodes. There are many algorithms and methods developed for GFS (Google File System) , about RD Way, about storage and query, which will be discussed in chapter 7.

## 2.3 Query and Rank

The main purpose of efficient storage is for efficient query. Fast search speed is one of important user experiences expected for all search engines. Google, Bing, Baidu and other major search engines commonly use the following cycles to process a user’s search request:

1. Accept a user query
2. Parse query strings
3. Figure out the keyword order
4. Look up the information in databases
5. Rank the results
6. Send back the results

In order to search for related resources among thousands or more data, traditional search engine systems use cache, pre-fetching results, memory indexes and other methods to shorten or speed up the search life cycle.

One basic ranking method that search engine systems use is Vector-Space[[5]](#endnote-4). Modern search engines generally collect users' behaviors, which nowadays have a lot of compliance to follow for privacy and security reasons, so search results returned by search engines are highly related to users' interests and habits, unless such feature is chosen to be disabled.

# 3 CHALLENGES

There are many challenges to address to collect valuable resources. First of all, how to determine whether a recourse is valuable? The Internet Crawler “spiders” are not something smart like human beings, the only thing they do is collecting and bringing the copies of the information back. Much meaningless and even fake or unhealthy information is obtained too. The search engine should not only filter bad impropriety or dangerous information, but also recommend the content according to the user's interest; however, that is still far from the definition of being valuable resources. Some issues can be magnified to different areas and user groups.

## 3.1 Issues related to resources

In the field of education, there are many issues exposed by traditional search engines. It could be directly regarding resource quality. Below are some of the practical questions related to resources:

* Is the resource reliable? (Does it provide correct information? Is the provider reliable?)
* How relevant is the resource to the user's search? (e.g. matching keywords.)
* Is the resource up-to-date? (When was the resource last updated? )
* Is information appropriate? (Is the resource safe?)
* Is the information redundant? (Are they repeated copies of the content?)
* How much time is needed to go through the resource? (e.g. length of a video.)
* What is the monetary cost? (Is it free? How is the price?)
* How many positive feedbacks? (Do other users find the resource useful?)

## 3.2 Issues to user types

Even if the resources are reliable and safe, for different user groups, they can be still be good or bad, because users of different ages, level of educations, skill experiences, and even learning preferences have different needs and expectation of what they look for. Ideally, search results should be close to the ability of different users. Below are some questions more from user perspective.

* What age group is the recourse suitable for?
* What grade level is the resource for?
* What type of resource is it? (e.g. recording, cartoon, for user preference)
* Is it for professionals? What field?
* What experience level is the resource for? (e.g. entry-level, intermediate, senior)
* Is it made for users of special ability? (e.g. language, disability)

# 4 PROPOSAL

The new search engine is designed to meet educational purpose, to support different types of users, returning valuable search result and maintain query efficiency in term of speed. Several techniques are created. We summarize them into two problems to address:

* How to collect and manages education resources?
* How to define a valuable learning resource?
* How to quickly process queries from users ?

Before starting the detailed discussion, Table 1 lists the terminologies that are used this thesis.

Table 1. The terminologies

|  |  |
| --- | --- |
| **Term** | **Definition** |
| **DLRV** | Degree of Learning Resource Value. This system measures the value of resources, and value is an index for search engines to use for providing ranking and recommending resources. |
| **Suitability** | Resource title, content and tags match search keywords. |
| **Cost** | All the cost of finding and using a resource, both time and money. |
| **Popularity** | Search, click, read and comment trend. |
| **Feedback** | User feedback, positive or negative comments, mark on a resource. |
| **Practicability(Usage)** | Amount of a resource usage, reference in courses and other resources. |
| **Reliability** | Resource reliability refers to whether the source of resources is reliable and whether it has been certified or audited by authority, this value is decided by the publisher. |

## 4.1 Resource Crawler

This search engine proposed in this thesis is to be used for education specifically. It helps users to find useful learning resources. Different from the traditional search engine, this learning resource search engine does not use crawlers to obtain web page contents. The system only obtains and stores the information about the learning resources. Though it may not be a major advantage, without storing resources objects, this search engine can be a significantly light system to achieve high performance. The copyright issues and system over-storage problems can be avoided. To collect the information of resource, “resource crawler” is needed. It collects the meta-data of resources, such as titles, locations, tags, publishers, descriptions, comments, and etc., many of which obviously require input from users. Meta-data are organized in system database, supporting inquires. Through these information, the value (quality) of resources can be defined or estimated, and users can search and locate the original resource content. It helps users find useful resources and study.

**[My editing mainly stops here today.]**

## 4.2 Resource characteristics

In particular it is difficult to identify resources within a firm if there is no agreed definition of what ‘valuable’ means.[[6]](#endnote-5)

Valuable resources can generate three types of competitive advantage: cost advantage, the ability to premium price, and volume-based advantage.[[7]](#endnote-6) The above three competitive advantages are used to demonstrate the valuable resources on business and management.

From the analysis of the above problems combined with the advantages that a valuable resource needs, a high-quality learning resource should keep the positive side of all the following characteristics.

1. Cost
2. Reliability
3. Practicability
4. Suitability
5. Popularity
6. Feedback

The 6 characteristics can be used to define a valuable and high-quality learning resource.



Figure 2

Notes: These six characteristics are used to evaluate the value of resources, of which light blue is the dynamic characteristic and gray ones are static characteristics.

## 4.3 Query and Search Improvement

To the question2, the user experience of the system is also very important.

Users need quick search, which depends on a high performance design of the search engine system. In particular, the system needs an efficient storage structure. A database contains billions of resources information, and the amount of data is growing rapidly every day. Every user query is filtered from such a huge amount of data. The system uses a 3 layered storage structure and encoded keyword mapping method to improve the search efficiency. It is helpful to improve the user's massive search experience. For the application in the actual production environment, the storage needs to adopt the distributed system design.

# 5 REQUIREMENTS

## 5.1 Hardware requirements

The whole search engine platform requires several high-performance servers which can undertake billions of requests from users. Users will search useful learning results listed in the browser. We call these web applications or B/S architecture. Generally speaking, the performance of the server and the configuration of hardware devices in the cluster are determined by the number of users and the number of resources. Therefore, in the early stage, there is no detailed requirement report for the whole set of server-side hardware, but we have given some basic requirements.

* The dual core processor E3 based on X86 system
* More than 16GB RAM
* 80GB disk storage
* Independent database server
* The distributed deployment server based on Linux is prepared with container and virtualization technology, but will not be used in the experimental time
* Data analysis server and other micro services

User-end or test-end, we require PC and mobile devices to test all the web pages and functions work well on various browsers.

## 5.2 Software requirements

The complexity of software requirements is much higher than that of hardware. All algorithms, technical details and functional requirements are implemented and verified by software programming. We can use the normal web development environment, tools, languages and related SDK.

### 5.2.1 MVC Design pattern

MVC pattern is a very classic design pattern in software engineering. MVC pattern was first proposed by Trygve Reenskaug in 1978[[8]](#endnote-7), which is widely used in web development. The framework based on this design pattern can be called MVC framework. All the development and implementation described in this paper are based on this design pattern. This set of search engine system adopts MVC framework. MVC is a design pattern that separates model, view and controller. In actual development, model is data level, view is front-end, and controller is part of business logic. MVC design pattern can achieve high cohesion and low coupling, and separate data, view and business. MVC improves the development efficiency, code cleanliness, and has higher scalability. The purpose of using this mode is to make the search engine easy to optimize and expand the function in the experiment.

### 5.2.2 Related application software

* MVC framework based on Node and express
* The view layer is based on vue, also called front end, which is used by users
* Reverse proxy server and http server are based on Nginx
* MySQL database, a kind of relational database, stores a large number of data generated by the search engine system

## 5.3 Non-functional requirements

### 5.3.1 Performance

Google answers 100 billion searches per month[[9]](#endnote-8), that means the average of a day is at least 3 billion and this is the statistics in 2012. Our learning resource search engine doesn’t need such huge search performance because we are targeting at a special area instead of all the users and resources on the Internet.

Around 2017, there are more than 30 million children use Google education apps[[10]](#endnote-9), and adults and college students are not included in 30 million, so our system needs at least double of this amount(children) for users’ requests so that it needs to accommodate 60 million users per day. In computer area, it is called DAU (Daily Active User) and DAU is at least 60 million. To ensure the smooth requests from the increasing DAU, we raise the performance bottleneck to 100 million DAU.

### 5.3.2 Reliability

The operation of all the services are 24 hours, Users get the results within 1 second after starting the search that is to say, the response speed is less than 1 second for each research

### 5.3.3 Security

**System layer security:**

1. Firewall between server nodes, access control on blacklist, white-list and iptables technologies.
2. Data backup to prevent the data loss disaster.
3. When main servers crash, use the reserve servers instead.
4. Quick recovery plan for crashed servers.

**Business layer security:**

1. User verification
2. API requests security
3. User behaviour logs
4. Cookie or cache security
5. User privacy
6. Encrypt and decrypt data

# 6 SYSTEM DESIGN

This part is the system design description, including engineering design, system architecture deployment design, unified modeling language.The final implementation, online system and theoretical verification of the system will follow all of the following design principle.

## 6.1 Software Engineering Process

The design of software process follows the life cycle of software engineering and adopts agile model.



Figure 3

I divide the whole implementation process into 6 parts. The concept of the system is described in the background and problems above parts of the paper. Software requirements will not be described in the paper, there are other requirements documents. Framework design, UML will be described in the next parts of this chapter, development and implementation will not be described, only display through algorithms and methods research, for detail please refer to development documents and code. Testing and verification will be presented in the last part of the paper.

## 6.2 System deployment structure design

The following is the simulation deployment picture of the search engine system, which follows the normal web deployment mode.



Figure4

Server deployment is distributed, running multiple servers in the same intranet or multiple networks, and each server has its own work task and provides API or open port to other connections. This is a very popular deployment method of Web services, which can handle large concurrent requests, reduce the coupling between services and improve security. Multiple servers can be managed by different teams or individuals, making it easier and more efficient to work. From top to bottom, from left to right, there are user clients(PC or mobile with browsers), CDN, reverse proxy server, firewall, business logic server group, business server and database connection. There is firewall in the middle to control access. Part of database server data is stored in high-speed non relational database, such as Redis or MongoDB, to deal with some hight-frequecny search engine requests.

The purpose of CDN is to speed up the existence of static files. The full name of CDN is: a content delivery network, or content distribution network[[11]](#endnote-10), static files can be distributed on multiple nodes of the Internet. When users access, they request the nearest fastest server, which improves the user experience.

The reverse proxy server distributes user requests to upstream servers, which can effectively reduce the possibility of congestion.

There must be a firewall between the reverse proxy server and the cluster server to control the access list, which can be a white list and prohibit illegal users from directly accessing the cluster.

There are many servers in the server cluster, most of them are controllers dealing with business logic, and there are also some servers specialized in processing big data. DLRV’s calculation is included. These servers, which are responsible for data processing, work without rest, sort and classify the resources and tags from the database, score the resources based on DLRV algorithms and provide the core business for users to search the accurate resources.

Non-relational database, such as Redis and MongoDB, these are very important parts of the whole system. Search engines have high requirements for the speed of search and data acquisition, and the structure of relational database is too complex. For some simple tag search, non relational database and even cache database based on memory can provide search engine with more efficient results. Redis can save high-frequency search keywords in memory based on some page switching algorithms, which can effectively improve the search speed. This will be described and tested in the methodology of the second half of the paper.

## 6.3 UML

### 6.3.1 Database and E-R diagrams.

1. R diagram is a kind of diagrams to display the entity and relations between different data structures, in database, we call them tables. Attention: non-relational database is not included in the diagram.

Any modern web development is based on database driven, and the design of relational database follows E-R diagram. The following is the complete database design diagram of search engine system.



Figure 5

Note: each table(entity) has its own primary key which is named id. ‘is\_effect’ is used to mark a row of data as available or non-available status. It is not safe to completely delete information or to display uncontrollable data, by using ‘is\_effect’ attribute, the system can control the validity of data, hide or display data flexibly.

User table is used to store user’s information, resources table stores resources from the internet, users are the owners the resources, one user to many resources, users use resources to organize a course, one course to many resources.

‘Users are the owners of the resources’, this become a extremely important point which makes this search engine different from others. Users have the right to monitor the quality of the resources. This is the key point of reliability evaluation in DLRV system.

### 6.3.2 Use Cases



Figure 6

This is the use case diagram of the search engine system which covers 7 main use cases. These are just the most basic just needed functions, and the core use case is the search function. A large number of functional details are not covered here. Users are learners. They can search resources, upload resources manually, set up courses by adding resources as a set, and evaluate other courses or resources.

Note: these use cases: review resources, reference resources, review course and upload will affect the DLRV system define the value of resources. They are called as user behaviour or callback effects.

### 6.3.3 Package Diagram & Class Diagram

Package is a [namespace](https://www.uml-diagrams.org/namespace.html) used to group together elements that are semantically related and might change together. It is a general purpose mechanism to organize elements into groups to provide better structure for system model.[[12]](#endnote-11) For the server side of the search engine system, the following figure shows the packages, the system is developed based on object-oriented language. Packages are mapped to different folders or collections. Different packages have dependencies on other packages.

1. Controller, this is a package includes all the controllers, we have talked that the system is designed on MVC mode. Controllers deal with all the business logic, take responsibilities for connecting data and views, accept users’ requests and responses. Controller depends on private libs and public modules.
2. Server, the entrance of the whole system, this package manage the files to work as a web server. The data from user-side enter this package first. It depends on controller because the request and data from user-side need controller to serve them. It depends on public modules too.
3. Config, a package manages the connection configuration of various servers, like mail server, database server, OSS server and all other servers we needed in this search engine system. At the same time, this package contains some configuration of the system itself. Config package doesn’t depend on any other packages.
4. Private libs, the package contains all the private modules, plugins which used only in this system. It needs to depend on public modules and Model package.
5. Model, the central component of the pattern. It is the application's dynamic data structure, independent of the user interface.[[13]](#endnote-12) It can manage the data, logic and rules in the system.
6. Modules, public modules are used by the system



Figure 7

Note: in this diagram, Config and Modules two package have the most dependence by other resources, Config contains all the configuration of the system. ‘Modules’ is a class library referenced from public libs and used by the system in many places.

**Controller Package**



Figure 8

The controller package is the core package of the whole system. It contains classes which are used to deal with the user's business logic, and the search class solves all the user's search requests. The Config class returns the configuration of the site to the front end. Resource will be responsible for data collection, resource classification, deletion and other functions. Through the Resource class, users can also edit courses and get course categories and lists.

**Private Package**



Figure 9

The core class of private package is the class Public, which means private public library. It is a public library specially developed for only this system, it includes functions such as get random numbers, format date, check user info and etc which are used with high frequency. It can be imported and used by controllers to reduce code redundancy and coupling. It is a very common design idea in software engineering. High cohesion, loose coupling[[14]](#endnote-13). Class DB provides a set of methods to operate database like inset, query and delete.

**Server Package**



Figure 10

The server package is the web server start-up entry. It listens and distributes the user's request to the controllers. The mapping mode used between the server package and the controllers is called Convention Routing. The figure below explains what is the convention routing mode.



Figure 11

The solution is to automatically map the user's router to the same class name and action name under the controller. The characteristic and advantage of the convention routing is that it does not need to configure the route files, which reduces the development time and the writing of method documents. It can reduce the possible misunderstanding between the front-end and back-end communication.

In this system, the static file has its own unique mapping way, which is different from the controller, so it can isolate the access of code and media file, which is a guarantee for security.

### 6.3.4 Sequence Diagram

The main function of search engine system is to search valuable resources, so in many sequences series, this part mainly studies the logic of search sequence. The following is the UML design of search sequence.



Figure 12

The whole search process of the system is that after getting the user's request from the server, the keywords are sent to the Search controller to screen the data in resource database. When the matching and approximate data results are found, the Search controller calculates and verifies the value of the resources through the DLRV module service, and finally returns the organized results to the controller. Finally, the controller returns a sorted list of results to the user browser through the HTTP server. Users get valuable resources that match their request.

# 7 ALGORITHMS

This chapter describes the algorithm and some methods in learning resource search engine in order to let users get a good experience. ‘Good User Experience’ means 1. Fast access to search results and 2. Valuable learning resources, which is in line with the theme of the paper. We can simplify these two points to ‘search efficiency’ and ‘resource sorting and filtering’.

Let's first do some research on definitions and methods in traditional search engines(RD). Then it will explain the improved or innovative algorithms used in the learning resource search engine system(DLRV). The algorithm and methodology discussed are also based on the above two ways. The first section reviews the traditional search engine methods to speed up storage/query and rank valuable resources. The second section will put out the weakness of traditional general search engine. Finally, based on the traditional algorithms and methods, aiming at the field of Internet learning resources search engine, DLRV, we improve and enhance the ranking algorithm, storage-query algorithm for this search engine.

## 7.2 RD Way

RD (Resource Discovery), this is a process of searching valuable information on the Internet.[[15]](#endnote-14) The IETF-RD argues that resource discovery should provide the user consistent, organized view of information.[[16]](#endnote-15) RDS(Resource Discovery Server) can return a set of resources, these resources are links or indexes of web pages from the internet. Various of search engines support RD such as: Google, Bing, Baidu, etc. In particular one keyword or topic can typically contain from thousands to millions resources.

When you type ‘English Learning’ keywords on Google, it returns 10,010,000,000 results within only 0.48 seconds. How does search engine do such a fast query?

Users only concern with a few useful results of all the resources results. How does search engine rank them and give a matched list of valuable results?

### 7.2.1 Storage and Query

Google uses its own file system named GFS(Google file system), GFS is a scalable and classical distributed file system for large distributed data-intensive applications[[17]](#endnote-16). GFS has been used in Google since 2003, and GFS is not open source. However, from Google's GFS related paper, we get some basic technical details, which are used in Google's search engine resources storage.

The following is the basic structure and work principle of GFS.



Figure 13

When the search engine application query keywords, the processed keywords first go to a master server. The master server doesn’t store any chunks or files, it only stores the file system namespace and mappings to the chunk locations. A large number of resources are stored on the chunk servers, files are divided into multiple chunks. The main server can return the corresponding address of the chunk, then the search engine can directly find chunks through chunk servers.

The reason why Google does this is because Google gets billions of resources from the Internet crawler every day, which can not be stored by a single server, or in a database or a data table. These resources, web pages and web addresses can only be divided into several small chunks and stored in the distributed file system. Moreover, one same chunk may exist on different servers for many times. The chunk server itself can copy these chunks to prevent loss. Chunks may be stored in any corner of the GFS servers, but since the master server records all the chunks mapping relationships, chunks are very easy to find and search engines get the data sets from the file system in a very short time.

In the distributed system, it is not very difficult to store and query millions of data. The specific search algorithm in the distributed system is no longer redundant in this paper. You can refer to the related papers of distributed system or GFS. What we need to discuss here is, in a traditional search engine system or a resource management system, distributed storage is the only or a general solution.

### 7.2.2 Rank of Results

A general RDS(Resource Discovery Server) can help user quickly rank useful resources through effective filtering and sorting algorithms. The ranking principle follows a score of significance. First we give the main function, Rank().

Function Rank() includes a set of algorithms(will be discussed in detail below), it will return a list of sorted valuable resources after measuring the different aspects of resources. In this method, the search keywords of users’ query(Queries) are essential in this function which are used to match the resources.

Traditional search engine uses Vector-Space model to compute the results of rank function. In Rank() function, there are two main parameters, ‘Queries’ and ‘Resources’. Let’s simplify the two parameters to Q and R in the following context. Q is a vector, in computer language, it can be an array like [ Q(1), Q(2) ... Q(i) ] and R is another array [ R(1), R(2) ... R(j) ]. It comes to ‘many Q to many R’, the Rank() function needs to calculate the count of Q(1) appears in R(1), the count of Q(2) appears in R(1), to summarize as the following formula.

SR(j) is the score of R(j), count() is another function which is used to count the number of times that Q(i) appears in R(j).

V is the final value of a resource, SR has been explained, Freq is the frequency of the resource shows in the whole internet(in the range of search engine statistics). In the example, there are j resources that the search engine need to sort. When the SR(score of a resource) is higher, the more valuable of the resource(here is web page) for the user is. The more frequency of the resource appearing on the network, the more multiples, leading to higher value, he more ahead the resources are ranked in the final results. In order to compare and sort all the resources, we have to calculate from V(1) to V(j) which means to calculate all the values of resources.

The final V is used to rank the resource list and display to the users. The order is as follows.

The first parameter is all the final values of all the resources, second parameter means it uses descending sort.

The above two sections discuss about the traditional storage-query model and competitive searching sorting which are used by RDS. This is also the basic mode adopted by main current search engines.

## 7.3 DLRV Way

DLRV(Degree of Learning Resource Value), discusses a method helps users to acquire valuable resources. The same with RD is that it definitely contains two parts, storage-query and results-rank. In the above paper, in traditional RD methods, we have discussed distributed file system to store and query the resources and Vector-Space model to compute and rank the value of results. But in the special search engine of learning resources, these methods have some disadvantages.

### 7.3.1 Disadvantages of traditional RD way

Note: that the following disadvantages are only proposed in the case of a special learning resource search engine in this paper when comparing with common search engines based on RD.

The distributed system takes a large place of storage and needs hundreds of distributed servers. In this system, we do not save the original files or resources, we only save the links and main information of resources. This logic is like the master server used in GFS. Therefore, if we distribute resources to multiple servers, it will waste a lot of physical resources, and in terms of software design, it is also extremely complicated to create such a system.[[18]](#footnote-0)

The biggest difference between learning resource search engine and traditional RDS is the second part, results ranking methods. Go back to chapter **Problems**, we have listed a number of search engine problems in this particular area, learning resources. If we use the basic ideas given in the last section: to count the number of times keywords appear in a document, then multiply the count number and frequency of appearance on the network, finally we get a score to reflect the value, or significance of a resource. This score is too simple for learning resources. Besides the degree of matching keywords and appearing frequency, learning resources have more important attributes, which determine the quality of the resources. The recommendation ranking of the search results of learning resources must combine the quality of the resources themselves, and also consider the matching degree and frequency in traditional RD. In chapter **Solutions**, the figure 2, displays the 6 vital attributes of a learning resource: Suitability, Cost, Popularity, Reliability, Practicability, Feedback. Among above attributes, RD can measure two, suitability and popularity. High matching can be involved in suitability, frequency of appearance on the internet can be included in popularity, that means high suitability and popularity in RDS are used commonly.

Therefore, in the learning resource system, the remaining missing attributes become the defects of traditional search engines, and sorting the values of resources based on the above two points does not work well in learning resource search engine system.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Engine  Attributes | **Suitability** | **Popularity** | **Reliability** | **Practicability** | **Feedback** | **Cost** |
| **RD** | Yes | Yes | No | No | No | Not sure |
| **DLRV** | Yes | Yes | Yes | Yes | Yes | Yes |

In addition to the methods in resource storage and ranking aspects, there are some other features of the shortcomings compared to common search engines, here to do a simple list.

The common search engines can not allow users to comment, reply, like on the resources.

The common search engines can not allow users to reference resources.

These search engines don’t bring user feedback into DR, this approach leads to the lack of human action values when search engine do the query and sort. Starting from these shortcomings of DR, in the next chapter we will show how DLRV improves all these attributes of learning resources in storage-query and results-rank two main aspects.

### 7.3.2 Storage and Query

There is no need to use distributed storage in the learning resource system, especially the distributed file system. We use a combination of relational database and non-relational database to save data. The database saves the information and attributes of the original resources, including the title, introduction, link, file type, price, and thumbnail of the resources. The following is a diagram of the storage system in DLRV way.



Figure 14

In the DLRV resources system, resources are stored in the traditional web storage mode, ‘database center driver’. It means all the basic information and relationships of resources are stored in the databases. Databases are divided to three parts in this system, one relational database, one non-relational database in disk, one non-relational database in memory cache.

### 7.3.3 Relational Database

A relational database is a digital database based on the relational model of data.[[19]](#endnote-17) This means that the things stored in such database become more complex. In addition to entities, it also stores the relationships between entities. The basic unit of a data structure in a database is a table. Each instance of the data is called a row, which is stored in a table. In DLRV, relational database is used as our main storage container because the information of the learning resource itself is a specific entity and it has a one-to-one, one to many relationship with publishers or reference objects. The figure 15 shows some relationships and entities in the database, the resources stored in data are absolutely not independent individuals. These complex relationships are reflected in the business logic function points of this search engine website, such as reference resources, publish resources, comment resources and other more functions.

However, relational database is not very effective in searching, sorting and analyzing, especially in the case of large amount of data, so most of the role of relational database here is to store resource information. The following will introduce how relational database cooperates with non-relational database to carry out data quickly, how to improve the speed of user search and reduce the time cost. Time cost is one of the evaluation attributes of resource value.



Figure 15

### 7.3.4 Non-Relational Database On Disk

The characteristic of non-relational database is that the storage structure in it is very simple, without dependent relationship between data and data. So when retrieving a piece of data, its reading speed is relatively faster, especially in the case of large amount of data. From figure 14, it is a set of storage solutions. The figure shows three databases, two of which are permanent storage database which store data on hard disks and one is cache database which stores data in memory. Between the three, the reading speed of data is:

Cache Database(Non-relational database) > Hard Disk Non-relational database > Hard Disk Relational database

Please refer to query performance of different modern databases in the following data table.[[20]](#footnote-1)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Operation | Oracle | MySql | MsSql | Mongo | Redis | GraphQL |
| Insert | 0.091 | 0.038 | 0.093 | 0.005 | 0.010 | 0.008 |
| Update | 0.092 | 0.068 | 0.075 | 0.009 | 0.013 | 0.012 |
| Delete | 0.119 | 0.047 | 0.171 | 0.015 | 0.021 | 0.018 |
| Select | 0.062 | 0.067 | 0.060 | 0.009 | 0.015 | 0.011 |

Among these databases, Mongo, Redis and GraphQL are all non-relational databases, our search engine system uses Mongo to store tags and short words as the keys. Therefore, after getting the user's keywords, the speed of searching these short words in the non-relational database is very fast, and these short words will be further indexed, which will also be described in this paper later. And a unique index can further speed up queries.



Figure 16

As figure 16, this is the ‘key-value’ storage data structure In Mongo. The ‘key-value’ data structure is very normal in non-relational database which means one key to one value, we can find value by key. The ‘key-value’ data structure actually is stored as hash map on the computer so its time complexity is O(1).

Because the same keyword may map to multiple resources, which means a keyword can query multiple resources. In this system, the key is a keyword(tag, title and short words etc) and the value is a set of resource ids, ids are stored like [id1, id2, id3, id4].

Note, although there are multiple resource ids here, they are stored in one space as a string split by a special table like ‘,’ or ‘|’ which is easily divided to array by any computer language.

This is very important here, which is a key to accelerate keywords query. If in a relational database, these resource ids are related to keywords, and they will be foreign keys of resource data. This requires the database to search multiple rows, read the entire library or use search optimization algorithms, such as binary tree search. But if we only use one ‘key-value’, the database will only query once, the time complexity is O(1). There is only a hash map searching in the database.

In addition, MD5 is used to encode the key in ‘key-value’. MD5 is an algorithm for inputting variable length information and outputting 128 bits of fixed length. The purpose of storing keywords in this way is to ensure that the key stored is always 128 bit space taken. This makes the storage for key is space complexity O(1) too. But MD5 is not decode-able, does this affect the search? Let’s see the following table first, it is the storage example for ‘key-value’ in our search engine system. The left side are the keys they are MD5 encoded. They can be tags, keywords, titles and any other short words. The right side are the corresponding ids for the resources.

|  |  |
| --- | --- |
| **Key(keywords, tags, title)** | **Value(resource id)** |
| ba0a6ddd94c73698a3658f92ac222f8a | 1,2,3 |
| c31b32364ce19ca8fcd150a417ecce58 | 4,5,6,7,8,9,11,12 |
| 4dbe9ff7f2742c912b53b9feab9f343e | 6 |

The searching process is as the following diagram, figure 17. In this flowchart, we can see the steps as follows: first, enter the search keyword. Then we encode the keyword by using MD5 to get the key, use the key map to value of the IDs for resources, and then get the resources’ related information according to the IDs list. Here, ID is a primary key in the MySQL resource table, so the search speed is fast.



Figure 17

### 7.3.5 Non-Relational Database Cache

The cache database of the system adopts Redis, which is a database based on memory, which can also store the data as key-value in database persistently. Cache database can be used to speed up the search.

1. Redis can store a large amount of data. It supports the 2^32 keys in hash map, and the maximum size of each key or value is 512M.
2. Redis can be scalable and distributed.

The reason for using the cache non-relational database is that although MongoDB mentioned in the previous section can speed up the reading query, when the amount of data reaches one-day 100 million rows storage, the storage space will be insufficient first, and then the query speed will be significantly reduced.

The effect of using cache database like Redis is that:

1. Using distributed storage can divide data to several servers, to balance the storage pressure.
2. The speed of reading and writing memory is far higher than that of hard disk, the frequently read and write data should be stored in memory.

Now here we get a problem. Although the memory speed is much faster than the hard disk, on the other hand the storage space is far less than the disk space. Take a personal computer with 16GB memory and 1TB hard disk as an example, the memory is only 1% of the hard disk.[[21]](#footnote-2)

So we only store part of the content in memory, so what kind of data needs to be stored in the cache non-relational database and how to store these data effectively(storage methods).

### 7.3.6 Three Layered Search

Question: What kind of data needs to be stored in memory? It is explained in figure 18.

To sum up, the data stored in memory are the keywords with high search frequency, we put these keywords in cache non-relational database as keys, and the corresponding values are the IDs of the resources. We have already discussed ‘key-value’ mode in the previous section. As shown in Figure 19, when a user searches for resources, the input keywords first enter the cache non-relational database, query the IDs of the resources, and then enter the relational database to get relevant resources information. The whole process is direct, no longer passing through the disk non-relational database. When the keywords can’t be found in the cache non-relational database, then it searches in disk non-relational database. Note that the disk non-relational database saves all possible keywords and tags, but that doesn’t mean cover all the resources. If the controller still can’t find the keywords in disk non-relational database, it will search in the relational database directly finally. Normally, the process will not go to the last strategy, searching directly in relational database.

For the whole storage structure of the layered search, logically, this search engine system storage structure can be divided into three layers by query speed.

The third layer is the super high speed layer, the layer where the query occurs first, the second layer is the high speed layer, and the first (basic) layer is the normal layer. The speed of query in layers is from low to high. Please refer to figure 20 to understand the 3-layers’ search storage structure.



Figure 18



Figure 19



Figure 20

### 7.3.7 Cache Switch Method

Memory is limited, even if we use distributed storage. We need to make sure that the memory is used within a certain range and that the original operating system runs smoothly. In this way, we need a set of algorithms or methods to ensure that the storage size of the non relational database memory area is kept within a reasonable range. Therefore, it is impossible to save all the keywords as indexes in the cache non-relational database. Here we refer to the idea of page switching in operating system area.

About paging, it is a technology of memory management in an operating system that enables the main memory of computer to use data stored in auxiliary memory. The OS will divide the data in auxiliary storage (usually disk) into fixed size blocks called “pages”. When it is not needed, the page is moved from main memory, when necessary, the data is retrieved and loaded into main memory.[[22]](#endnote-18)

Page is a storage unit and exchange unit in OS paging technology. In our search engine system, a ‘page’ is a key-value unit in redis(a cache non-relational database). As the keyword is encrypted by MD5, as described above, an MD5 encrypted hash value is 128bit, so it can be understood that a page needs at least 128bit. However, please note that the resources’ IDs are also stored in a key-value unit(a page), so the resources’ IDs need extra storage space. In redis a value of string can be 512M. Google allows about 25 billion results for each search. We can take this number as the max search results. Therefore, the value in a ‘key-value’ unit can store 25 billion IDs at most. Let's calculate the storage space of a key-value in the maximum case.

Conditions:

1. 25 billion IDs in a value
2. The max length of id is 11
3. 1 English character needs 1 byte with UTF-8 encoding.

Result:

From the calculation results, the max value 262260MB is more than 512MB which means a value needs at most 512 times (262260/512=512).

Absolutely, one hash map index only maps to a list of resources id, so we also need to calculate the search frequency of resources, and only store the high frequency resource ID into the value so that this value will not be more than 512MB.

So far, we need to pick up high-frequency keywords, high-frequency resource IDs. The former is for the lack of memory for keywords storage, and the latter is to solve the problem of limited storage in redis.

In order to achieve effective page switch(key-value unit switch) work, the commonly used algorithms are as follows.

LFU(Least Frequently Used), LRU(Least Recently Used)[[23]](#endnote-19), FIFO(First In First Out) and Clock. Among the above, LFU is suitable for this search engine system.

By using LFU rule, in cache non-relational database we remove the lowest search frequency key-value unit. For example, we store the frequency with IDs in value for each key. A table shows keywords, IDs and frequency in cache non-relational database.

|  |  |  |
| --- | --- | --- |
| **Keywords** | **ID** | **Frequency** |
| Key1 | 1,2,3,4,5 | 10000 |
| Key2 | 6,7,8,9,10 | 20000 |
| Key3 | 1,3,9,11,12 | 30000 |
| Key4 | 12,23,45,222,657,12321 | 50000 |

The switching process is shown in figure 21.



Figure 21

If, these are the units already in the cache database and the database is full. At this time, a new keyword key5 that is not in the cache is searched more than the lowest keyword key1, then key1 will be replaced by key5

### 7.3.8 Rank based on DLRV

The biggest difference with normal search engines is that a learning resources search engine needs to be evaluated more strictly and on more features.

In the section ‘Value resource attributes’. We give a basic model to evaluate learning resources, which contains six characteristics (Figure2). We use these six directions to evaluate the value of a learning resource. This basic model is explained below. We call it the DLRV model. We get the value of resources by quantifying the six characteristics, and then return them to users according to the value from high to low.

### 7.3.9 Static and Dynamic

Characteristics are divided into static and dynamic. Dynamic means that the value of this characteristic is generated dynamically, and different values are generated each time because of different conditions. It has no direct relationship with the resource itself, and different conditions can produce different results even on the same resource.

The most obvious example is that when users search for two words ‘English’ and ‘Chinese’, to the DLRV system, the value of different resources are different, to the users, the search results are different.

Among the six characteristics, only one is dynamic, which is the ‘usability’. It is related to the user's search keywords. The other characteristics, cost, reliability, practically, popularity, review are all static. Which means they are determined by the status of a resource itself. And the values of static characteristics are stored in the database statically. Static is not permanent. It is just that static characteristics will not be changed in real time and will not change due to user search conditions. Static value will also be changed due to the change of resource quality and nature, such as views, evaluation, publisher, etc.

The methods to calculate the final value of learning resources in a DLRV system are as follows:

The total value of a learning resource, V(j), is the dynamic value of characteristics, Dynamic(j), multiply by static value of characteristics, Static(j).

### 7.3.10 Suitability

This is a dynamic characteristic of DLRV, which means, the value of this characteristic is not determined by the resource itself, it is not a static value, and will change as well each search behaviour.

Suitability means: the results can match the user's search keywords and purposes. If it is different from what users want, or even the opposite result, we call it an unsuitable resource. How to fix what users want? We can set all the keywords into an array, as [ k(1), k(2), k(3), k(4) ... k(i) ], and set the resources as [ r(1), r(2), r(3), r(4) ... r(j) ] this step is the same with traditional RD search engine in section ‘Introduction of Traditional RD Way, Rank of Results’. We still use keywords array to match the information and content of resources. Because these keywords represent the general purposes of the users.

The concept and method are shown above. Add the frequency of each keyword in the resource title, tag and description. The higher the frequency of occurrence, the higher value of resource suitability. Finally, accumulate the number of times keywords appear in the title, description and tags to get a value of suitability. The value of the suitability should more than 0.

S(j) is the value of suitability for r(j), r(j) is a resource.

### 7.3.11 Reliability

Reliability is determined by the source of resources, which we call resource publishers. Resource publishers can be divided into personal accounts or unit accounts, and personal account can be divided into certified or uncertified account. The unit must be certified, unit is generally an organization or an enterprise.

The reliability value for a resource is calculated as follows, a user can be generated as user(i):

In this function, R(j) is the reliability value of a resource r(j).

This is a piece wise function:

1. For the first piece, if a user is not certified (variable cer(i) = 0) then the R(j) is 0.
2. The second piece is under the condition that a user is certified but not a unit account (not an organization or a company), then the R(j) equals k1 multiply the total usage of this user’s resources so far. So far means this value only calculated before the resource r(j) being published. If a user u(i) publishes resources [ r(1), r(2), r(3) ... r(k) ], the reference or usage is [ u(1), u(2), u(3) ... u(k) ]

k1 is a weight number. It can be adjusted according to the experiment to fix the value of reliability.

1. The same way to understand the third piece. It is under the condition that a user is a unit account as an organization or a company. k2 is another weight number set for unit account. The same, we need to sum all the usage of the resources the user has published.
2. The count of the usage in this system is how many times a resource itself has been referenced in courses or other resources.

### 7.3.12 Practicability

Practicability is also called usage. The more times a resource is used, the more practical it is. The value of practicability of r(j) is U(j). U(j) is the total number of times a resource is referenced.

### 7.3.13 Feedback

User feedback is a direct reflection of a resource quality. Users' evaluation of resources can be divided into good or bad. Good feedback can add resource value and the bad feedback reduce the value. We use the proportion of positive comments to reflect the value of resources. If a resource has no feedback, we set this value 1.

In the function, F(j) is the value of feedback. p(j) is the positive feedback which is called good feedback. n(j) is the negative feedback which we call bad feedback. t(j) is the total feedback, it equals n(j) plus p(j). So the value F(j) actually is the rate of good feedback for a resource.

### 7.3.14 Popularity

Popularity is a trend that describes the resources’ increasing views. The value of popularity is P(j).

V is a number of accumulative views of a resource. Vd is the accumulative views of today, Vd-1 is the accumulative views of last day. The number of views added today is the value of popularity.

### 7.3.15 Cost

Cost includes time and money. C(j) is the value of cost, T(j) is the value of time cost, M(j) is the value of money cost.

### 7.3.16 Static Value

The characteristics: reliability, practicability, feedback, popularity and cost are the static characteristics. They don't change because of the query keywords, so not dynamically changed when searching the resources. They are the static values stored with resources. We store the following value of all the static characteristics.

Static value of a learning resource adds reliability, usage, positive feedback multiply popularity and then the value is divided by cost of the resource.

### 7.3.17 Dynamic Value

There is only one dynamic characteristic among the six characteristics in the DLRV model. Suitability is changed dynamically when users search the resources. Different query keywords can course different suitability value.

### 7.3.18 General formula

Refer to section ‘Static and Dynamic’:

In the section, we have described the general formula is multiple the static value and dynamic value. So V(j) means the final value of a resource, it equals to suitability multiply the total static value.

# 8 TEST

In the field of software engineering, software testing and quality assurance is an important part of software development. The method of testing is to use virtual data to test the functions of software. All the data obtained in the test can be used to evaluate the software performance, functional logic, algorithms and security.

## 8.1 Purposes

The main purpose of testing is to meet all the requirements and qualify the quality of the software.

We report the test in this section for the following purposes:

* To test the performance of 3-layered storage structure.
* To test the accuracy of DLRV ranking results.

The first purpose is to verify the query-storage part of the methods. The second part is to verify the DLRV ranking algorithm. These two parts are the core of this learning resource search engine, so we test them in this paper. Other requirements and functions can be verified or tested through similar methods, we will not repeat here.

The expected results of the test are

* Less search response time.
* More high-value resources are ranked topper.

## 8.2 Test Methods

Software testing is based on requirements and specifications of design. There are some common and mature testing methods in software engineering and we briefly introduce and apply some methods to learning resource search engine.

### 8.2.1 Black-box Testing

Black-box treats the software as a "black box", examining functionality without any knowledge of internal implementation, without seeing the source code. The testers are only aware of what the software is supposed to do, not how it does it.[[24]](#endnote-20)

Black-box testing will be used in this search engine system. It is used for testing the ranking part of the search engine. Black-box testing is very suitable for the whole user searching process, from keywords to the list of all the most valuable resources. The search function is a black box, testers don’t need to understand the principle of the searching algorithms and ranking methods. The testers only verify the value of the resources and rank results. Resources with high value are ranked at the top in the list, while those with low value are arranged at the back. Value itself is a more subjective concept. Later in this section, we will discuss how to measure the user's recognition of resource value.

### 8.2.2 White-box and Control Group

White-box testing (also known as clear box testing, glass box testing, transparent box testing, and structural testing) verifies the internal structures or workings of a program, as opposed to the functionality exposed to the end-user. In white-box testing, an internal perspective of the system (the source code), as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs.[[25]](#endnote-21) [[26]](#endnote-22)

We use white-box method to test the storage-query part, especially the 3-layered storage structure. This is not strictly a white box test. According to the previous section, white-box testing asks users to choose different test paths according to the logic of the code. However, here we are not changing the input paths to get the expected results for the code logic of a function. In this part, we query the same amount of data for different storage structures, and design different experiments through the same keyword and the same network. Anyway, the logic is very similar with the white-box, we need to change the structure by changing the code which means the code is transparent to testers. Testers need to test searching business by following different logic codes for different storage structure.

The purpose is to test the query performance and storage performance of the search engine system through different experimental control groups and verify a best storage solution.

## 8.3 Test Conditions

In the test, there are variables and constants in the same experimental group. We define the invariant conditions in the experimental environment.

### 8.3.1 Storage and Query Condition

In testing the storage and search process, we should ensure that some environments are fixed:

* Same local network
* A server with the same configuration
* A PC with the same configuration, the same browser

Test controls the network environment to prevent network problems from affecting the query speed. The server configuration is the same because different server configurations will course different test results. All test data must be based on the same set of server configuration, including hardware and software. The client used in the test must also be the same, less strict than the previous two items.

Local Network: a home router, the server and PC are connected to the same router through wired LAN port, 100M fiber.

Server Software

* Linux operating system Ubuntu distributed, version 20.04.
* NGINX Tengine 2.2.3 proxy HTTP server.
* Relational database MySQL5.7
* Non-relational database MongoDB
* Non-relational database in memory Redis
* Node.js latest version

Server Hardware

* 16G DDR4 PC Memory Cards
* Intel i7 7700
* NVME SSD 512GB Samsung pm961
* ASUS motherboard
* Intel 1000M network card

Client Side

* Chrome Browser
* JS script, automated testing page

# CITATIONS

1. Patil, Yugandhara; Patil, Sonal (2016). ["Review of Web Crawlers with Specification and Working"](http://www.ijarcce.com/upload/2016/january-16/IJARCCE%2052.pdf) (PDF). International Journal of Advanced Research in Computer and Communication Engineering. [↑](#endnote-ref-0)
2. ["Googlebot"](https://support.google.com/webmasters/answer/182072?hl=en). Google. 2019-03-11. Retrieved 2019-03-11. [↑](#endnote-ref-1)
3. ["BingBot Crawl Activity Surging?"](https://www.seroundtable.com/bingbot-crawling-much-16273.html). Retrieved 2016-07-16. [↑](#endnote-ref-2)
4. Sanjay Ghemawat, Howard Gobioff, Shun-Tak Leung. The Google file system[C] Proc of SOSP 2003.New York:ACM,2003:29-43 [↑](#endnote-ref-3)
5. Dell Zhang \*, Yisheng Dong, An efficient algorithm to rank Web resources, 2000 Published by Elsevier Science B.V

    2000 Published by Elsevier Science B.V [↑](#endnote-ref-4)
6. Cliff Bowman, Veronique Ambrosini,Identifying Valuable Resources,European Management Journal,Volume 25, Issue 4,2007,Pages 320-329,ISSN 0263-2373 [↑](#endnote-ref-5)
7. Jordan Friedman, U.S. News Data: The Average Online Bachelor's Student, April 4, 2017 [↑](#endnote-ref-6)
8. Reenskaug, Trygve. [THING-MODEL-VIEW-EDITOR: an Example from a planningsystem](http://heim.ifi.uio.no/~trygver/2007/MVC_Originals.pdf) [↑](#endnote-ref-7)
9. Sullivan, Danny. ["Google: 100 Billion Searches Per Month, Search To Integrate Gmail, Launching Enhanced Search App For iOS."](http://searchengineland.com/google-search-press-129925) Search Engine Land. August 8, 2012 [↑](#endnote-ref-8)
10. Natasha Singer. “How Google Took Over the Classroom”. The New York Times, May 13, 2017 [↑](#endnote-ref-9)
11. ["Globally Distributed Content Delivery, by J. Dilley, B. Maggs, J. Parikh, H. Prokop, R. Sitaraman and B. Weihl, IEEE Internet Computing, Volume 6, Issue 5, November 2002"](https://people.cs.umass.edu/~ramesh/Site/PUBLICATIONS_files/DMPPSW02.pdf)(PDF). [Archived](https://web.archive.org/web/20170809231307/http://people.cs.umass.edu/~ramesh/Site/PUBLICATIONS_files/DMPPSW02.pdf) (PDF) from the original on 2017-08-09. Retrieved 2019-10-25. [↑](#endnote-ref-10)
12. OMG™ Unified Modeling Language™ (OMG UML®) specifications, Kirill Fakhroutdinov 2007-2016 [↑](#endnote-ref-11)
13. Burbeck, Steve (1992) [Applications Programming in Smalltalk-80:How to use Model–View–Controller (MVC)](https://web.archive.org/web/20120729161926/http://st-www.cs.illinois.edu/users/smarch/st-docs/mvc.html) [↑](#endnote-ref-12)
14. Loosely Coupled: The Missing Pieces of Web Services by [Doug Kaye](https://en.wikipedia.org/w/index.php?title=Doug_Kaye&action=edit&redlink=1" \o "Doug Kaye (page does not exist)) [↑](#endnote-ref-13)
15. Dell Zhang \*, Yisheng Dong, An efficient algorithm to rank Web resources, 2000 Published by Elsevier Science B.V

     2000 Published by Elsevier Science B.V [↑](#endnote-ref-14)
16. C.M. Rowman, Scalable Internet resource discovery: research problems and approaches, Communications of the ACM 37 (8) (1994) 98–107. [↑](#endnote-ref-15)
17. Sanjay Ghemawat, Howard Gobioff, Shun-Tak Leung. The Google file system[C] Proc of SOSP 2003.New York:ACM,2003:29-43 [↑](#endnote-ref-16)
18. In the aspect of HTTP, this system uses distributed structure, which divides users' HTTP requests to the upstream server, which is different from the distributed file system here [↑](#footnote-ref-0)
19. [Codd, E. F.](https://en.wikipedia.org/wiki/Edgar_F._Codd" \o "Edgar F. Codd) (1970). "A Relational Model of Data for Large Shared Data Banks". [Communications of the ACM](https://en.wikipedia.org/wiki/Communications_of_the_ACM" \o "Communications of the ACM). 13 (6): 377–387. [doi](https://en.wikipedia.org/wiki/Doi_(identifier)" \o "Doi (identifier)):[10.1145/362384.362685](https://doi.org/10.1145/362384.362685). [↑](#endnote-ref-17)
20. The database performance table is from Roman Čerešňák, Michal Kvet, Comparison of query performance in relational a non-relation databases, Transportation Research Procedia, Volume 40, 2019, Pages 170-177, ISSN 2352-1465, [↑](#footnote-ref-1)
21. 1TB=1024GB=1024\*1024MB=1024\*1024\*1024KB [↑](#footnote-ref-2)
22. [RAM, virtual memory, pagefile, and memory management in Windows](http://support.microsoft.com/kb/2160852/en-us). Microsoft. [2012-11-26]. [↑](#endnote-ref-18)
23. Donghee Lee; Jongmoo Choi; Jong-Hun Kim; Noh, S.H.; Sang Lyul Min; Yookun Cho; Chong Sang Kim. [LRFU: a spectrum of policies that subsumes the least recently used and least frequently used policies](http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=970573&isnumber=20937). IEEE Transactions on Computers [↑](#endnote-ref-19)
24. Patton, Ron (2005). [Software Testing](https://archive.org/details/softwaretesting0000patt) (2nd ed.). Indianapolis: Sams Publishing. [ISBN](https://en.wikipedia.org/wiki/ISBN_(identifier)" \o "ISBN (identifier)) [978-0672327988](https://en.wikipedia.org/wiki/Special:BookSources/978-0672327988" \o "Special:BookSources/978-0672327988). [↑](#endnote-ref-20)
25. Limaye, M.G. (2009). [Software Testing](https://books.google.com/books?id=zUm8My7SiakC&pg=PA108). Tata McGraw-Hill Education. pp. 108–11. [ISBN](https://en.wikipedia.org/wiki/ISBN_(identifier)" \o "ISBN (identifier)) [9780070139909](https://en.wikipedia.org/wiki/Special:BookSources/9780070139909" \o "Special:BookSources/9780070139909). [↑](#endnote-ref-21)
26. Saleh, K.A. (2009). [Software Engineering](https://books.google.com/books?id=N69KPjBEWygC&pg=PA224). J. Ross Publishing. pp. 224–41. [ISBN](https://en.wikipedia.org/wiki/ISBN_(identifier)" \o "ISBN (identifier)) [9781932159943](https://en.wikipedia.org/wiki/Special:BookSources/9781932159943" \o "Special:BookSources/9781932159943). [↑](#endnote-ref-22)