A Web Crawler System Design Based on Distributed Technology

Shaojun Zhong
Jiangxi University of Science and Technology/ Faculty of Science, Ganzhou, China
infor2000@qq.com

Zhijuan Deng
Jiangxi University of Science and Technology/ Faculty of Science, Ganzhou, China
66162815@qq.com

Abstract—A practical distributed web crawler architecture is designed. The distributed cooperative grasping algorithm is put forward to solve the problem of distributed Web Crawler grasping. Log structure and Hash structure are combined and a large-scale web store structure is devised, which can meet not only the need of a large amount of random accesses, but also the need of newly added pages. Experiment results have shown that the distributed Web Crawler's performance, scalability, and load balance are better.

Index Terms—Search Engine, Web Crawler, Grasping Strategy, Distributed System

I. INTRODUCTION

The production, transmission, collection and query of information are one of the most basic human activities. Considering information with writing as a carrier, traditionally libraries, corresponding cataloguing systems and professionals help us quickly find the information we need with “book” or “article” as the grain size. With the development of computer and information technology, there comes the field of Information Retrieval (IR) as well as the retrieval system of the whole text about books or literatures, making it convenient for us to obtain the relevant information with the grain size of “key words”.

The openness of World Wide Web and the widespread accessibility of the information on it greatly encourage people to create while bringing new opportunities for development and technological challenges for the information retrieval of World Wide Web.

The scale of traditional IR is relatively limited and the retrieved objects usually undergo serious screening and pretreatment. The number of queries it responds to is generally not very big. However, the information inquiry system (meaning search engine here), which provides services on web, is different with traditional IR both in scale and response time. Search engine has to deal with large-scale information (information swarms in and some are even fake) and a great number of accesses, which still requires fast response.

Search engine is an application system, which develops based on IR, suits the features of web (or www) and provides information query service. Search engine is generally defined as a kind of software system used on web, which collects and discovers information with certain strategies, deals with and organizes the information and finally offers web information query service for users. How does a software system like search engine work? If software system works on a data set, the data it operates includes not only unpredictable user queries but also huge web pages with dynamic change in number and these web pages will not come to the system automatically but need the system to grasp them. But in face of a large amount of user queries, it is impossible for the system to “search” online whenever there is an inquiry. So, the basis for large-scale search engine should be a batch of web pages gathered beforehand [1].

Therefore, web page catcher is also called Web Crawler. As a foremost part of search engine, it is an all-important studying object. Like the dynamic system carrying rocket system in aerospace, Web Crawler is the basis of search engine and all of the data it collects come from the work of Web Crawler in a smart, reasonable and powerful way.

Search engine is one of the most high-end and complex Internet technologies and all companies keep the core technology to themselves. Some big companies have already had a mature solution to large web crawlers and have already put them into use. However, these large search engines can only provide ordinary users with common and non-customized search services. They could not take into consideration of various requirements of different users and single web crawlers fall down on their jobs in many cases. The flexible customization and the incomparable information acquisition speed and scale of the distributed web crawlers have satisfied people’s growing demand on user-oriented web information. Therefore, this paper presents a distributed design method of web crawler, and strives to achieve a robust, scalable and efficient hybrid strategy of a distributed search engine.

II. CORE TECHNOLOGY OF DISTRIBUTED WEB CRAWLER
A. Priority Strategy of Webpage Grasping

Priority strategy of Webpage grasping determines the grasping efficiency. Grasping strategies can be roughly divided into three kinds, i.e. depth-first strategy, breadth-first strategy and best-first strategy. Depth-first strategy could be employed when the amount of information is not huge. However, the rapid development of the Internet and the massive existence of web data will inevitably run into huge data by adopting depth-first algorithm strategy. Therefore, grasping strategies of the search engine will generally be breadth-first strategy and best-first strategy, as well as some of their improved algorithms [2].

B. Diameter of the World Wide Web

Diameter of the World Wide Web or ‘Web Diameter’ is defined as ‘If d is used to represent a path from Web u to Web v, then the average length of the shortest path formed by all the different pairs of connected pages on the World Wide Web is called Web Diameter. According to this definition and the calculation of large-scale web pages, it can be known that Web Diameter is about 17[3].

The calculation formula of Web Diameter is:

\[ d = 0.35 + 2.06 \log (N) \]

(1)

Study shows that the diameter of China’s World Wide Web is 16.26[4], namely if there is a path between any two web pages, click less than 17 times on average, you can reach one web page from another, which is shown in Figure1.

After analyzing the Diameter of the World Wide Web, the following two conclusions are obtained:

1) Traversing Algorithm has affected the crawler’s efficiency to a large extent. The World Wide Web page structure is not that deep as we have imagined, but unexpectedly wider. Therefore, the traversal mode of the crawler generally adopts the breadth-first one. Certainly, there is the reason of the importance of web pages, and this kind of means can help to grasp more important web pages.

2) The World Wide Web is so complex that a chosen grasping circuit cannot necessarily and invariably guarantee the best. In order to prevent this problem, the diameter of the web needs to be fully considered, and "depth-first strategy" should be adopted to control the grasping depth. In this way, the problem can be perfectly solved [5].

Let’s look at the following example:

Suppose starting from seed site A, seed site B and seed site C, there are three paths to web page P, the lengths respectively being 3, 19 and 127 (CostAP=3; CostBP=19; CostCP=127). As to grasp web page P from seed site A is very quick while seed site B and C reach P after a long path, it is apparently not economic enough.

To prevent the Crawler from unlimited breadth-first grasping, a certain depth must be limited. Once reaching this depth, grasping should be stopped. The value of this depth is the length of diameter of the World Wide Web. When stopping at the maximum depth, those excessively deep un-grasped web pages always expect to reach from other seed sites in a more economic way. For example, seed site B and C stop grasping once reaching the depth of 17, leaving the opportunity for grasping web page P to the Crawler starting from seed site A to grasp. It is not hard to see that limiting the grasping depth destroys conditions of infinite loops and loops, if there are, will stop after limited times. Moreover, the combination of depth strategy and breadth-first strategy can effectively guarantee the closeness in the course of grasping, namely always grasping web pages under the same domain name in the process of grasping while web pages under other domain names are rare[6].

C. Judgement of the Web Importance

While maintaining the priority strategy of web page grasping, please grasp important web pages first to ensure those more important web pages can be arranged with limited resources. Which web pages are more important? How to measure the importance?

The measure of importance is decided by the following three aspects, i.e. IB (P), IL (P) and ID (P).

1) IB(P)

It is mainly decided by the number and quality of back links. Firstly, the more links (a great many back links) a web page has, the more it is recognized by other pages. Furthermore, there will be more opportunities for it to be visited by net-citizen and its importance is more obvious. Secondly, the more it is pointed to by more important web pages, the more important it is. The most classic is cheating web pages, which artificially set lots of Backlinks pointing to their own web pages to increase the importance of web pages. If the quality is not considered, local optimal will appear, rather than problem of global optimal.

2) IL (P)

It is a function of URL string which only investigates the string itself. IL (P) is realized mainly through some models, for example, it attaches more importance to URL containing ‘com’ or ‘home’. It also regards that the URL with fewer slashes is more important.

3) ID(P)

ID (P) represents that in a seed site set; there is a link (breadth-first traverse rules) in every seed site that can arrive at the web page. ID (P) is another important index
of the web pages. The closer it is to the seed site, the more opportunities it has to be visited. Therefore, it is more important and the seed site is where the most important web pages are. The farther it is to the seed site, the less important it is.

D. Non-Repeated Grasping Strategy

Massive web page images are other important characteristics of web. According to the 24 million page statistics by Google system, 22% of the web pages are images. The existences of a lot of duplicated web pages are unfavorable to the users’ query. It not only wastes the storage space of search engines, but also decreases the system efficiency.

The reasons, on the one hand, are that the collecting program does not clearly record the visited URLs. On the other hand, the domain names and IP addresses have a multiply corresponding relation. The first problem can be solved by making a record of the visited URLs, and making a contrast between the new URLs and the visited ones every time. The second problem is relatively complex, because different URLs may refer to the same IP.

There are four kinds of corresponding relationships between the domain names and IP addresses, namely: one-to-one, one-to-many, many-to-one and many-to-many. One-to-one relationship won’t cause repeated collection, but the others are likely to do so.

1) Algorithm Based on B-tree

Due to the huge amount of web pages, web page grasping requires network bandwidth, machines, time and so on. The repeated grasping of the same web page greatly reduces the efficiency of the system, so the Crawler system should design a strategy to avoid repeated web page grasping to ensure that a web page is grasped only one time in a certain period of time [7].

B-tree is a kind of balanced multiway search tree. What the file system of operating system uses is the search algorithm of B-tree, which can also be used to design the algorithm matching URL to avoid repeated grasping in the Crawler. B-tree can be empty or multiway tree. A B-tree of m order must meet the following requirements:

(1) A tree can have m subtrees at most;
(2) If the root node is not the leaf node, at least two subtrees are necessary;
(3) All non-terminal nodes except root have at least two subtrees;
(4) All non-terminal nodes contain the following information data: (n, A0, K1, A1, K2, A2, ..., Kn, An, )

Each node includes n pointers pointing to each keyword record. Ki(i=1, ..., n) is keyword and Ki<Ki+1(i=1, ..., n-1); Ai(i=0, ..., n) is the pointer pointing to the root node of subtree and keywords of all nodes of pointer Ai-2 are less than Ki(i=1, ..., n) while keywords of all nodes of the subtree pointed by An are greater than Kn, n(1≤n≤m) is the number of keywords (or n+1 is the number of subtrees ). All leaf nodes are in the same layer and carry no information (they can be seen as external nodes or nodes failing to search. Actually, these nodes do not exist and pointers pointing to these nodes are empty).

What Web pages use now is marking language like HTML, XML, the feature of which is that they can be represented in the form of trees. If the tree structure of a Web page can be established, it is easy to manage and search with B-tree and compare the similarity of web pages by comparing nodes to judge whether the web pages are same, eliminating repeat before grasping.

2) Algorithm Based on Hash

The key of avoiding repeat is to remember history for only by remembering past can repeat be eliminated. The programming without technological history is usually very simple, namely model of finite state machine as far as computer is concerned. Each state is only the accumulation of history rather than the memory of history and such work, which only involves limited storage, is generally very simple. Programs needing to remember history are relatively complex, such as labyrinth solution program, and need a stack structure to remember where they have passed to recall when they fail and continue to find the way out. But to look for optimal solution program through dynamic planning also needs the function of remembering history to guarantee no-repeat calculation.

It is through Hash table that Crawler records history, to see whether each record is stored into a slot of the Hash table by the grasped information. If a web page has been grasped at some point in the past, the value of its corresponding slot is set as 1, otherwise as 0, but it is the Hash function that decides to which slot it maps.

In August, 1992, Ronald L. Rivest described a kind of MD5 algorithm signature principle in an important file submitted to IETF. Thanks to its openness and safety, this algorithm was widely used in the 90s. MD5 signature, a Hash function, can change any data flow into a fixed number (generally four integers, namely 128 digits). This number is called “Signature of Data Flow” or “Digital Finger Print” and any tiny change in the data flow can cause signature value to change.

The integer space of standard MD5 signature is very large as a 128-digit integer can represent a number as enormous as the 128th power of 2, but the space of actually allocated Hash table is very limited. Theoretically, the maximum addressing range of an ordinary 32-bit processor is the 32nd power of 2, namely a 4G memory. Apart from the memory of kernel of operating system and other applications, the limit memory Crawler can use is actually very limited.

Therefore, the signature value is mapped to the actual Hash table (this can be considered as that the Hash table is stored in the memory) after modular arithmetic in actual processing. So the actual Hash function is MD5(URL)%N, which makes URL to be mapped to a slot (its value is N) of the Hash table.

Hash table is a simple sequence list, namely array. To see from the perspective of practical application, this array should be large enough and be totally stored in memory to ensure that the signature of each URL can
make sure whether it has been grasped by looking into the Hash table.

E. Webpage Revisiting Strategy

The popularity of web results from the information web brings. Information is constantly changing, and the webpage information updating is unavoidable. However, the earlier grasped information may be out-of-date or of no use at all. A strategy is thus needed to solve the problem of timeliness of information, and it is called webpage revisiting strategy. Through revisiting, these webpages can keep pace with the changes of the World Wide Web.

In 2000, Cho and Garcia-Monlina of Stanford University randomly chose 500,000 web page samples and found that 23% of the web pages were updated on a daily basis while 40% of the web pages with .com as the suffix of their domain names was updated every day. The half-life of web pages is 10 days. In addition, study shows that the process web pages change boils down to model of Poisson process [8].

To describe the model of Poisson process, X(t) is used to represent the number of changes of web pages in the period of (0, t) and the Poisson distribution with \( \lambda \) as its parameter meets the following nature.

As for \( s=0, t=0 \), random variable \( X(s+t) - X(s) \) conforms to Poisson distribution, namely

\[
Pr(X(s+t) - X(s) = k) = \frac{(\lambda t)^k}{k!} e^{-\lambda t}
\]

(1)

In which \( k=1, 2, 3\ldots \)
The expected value of random variable \( X(s+t) - X(s) \) is \( \lambda t \).

\[
E[X(s+t) - X(s)] = \lambda t
\]

(2)

It can be proved through a simple method. Suppose that time cycle (time interval) is 1, then

\[
E[X(t+1) - X(t)] = \sum_{k=1}^\infty k \cdot Pr[X(t+1) - X(t)] = \sum_{k=1}^\infty \frac{\lambda t^k}{k!} = \lambda
\]

(3)

In which, \( \sum_{k=1}^\infty \frac{\lambda t^k}{k!} = e^{\lambda t} \), with (2), it can be obtained.

\[
E[X(t+1) - X(t)] = \lambda t = \lambda
\]

(4)

Through the trace analysis of 500,000 random web pages, Cho and Garcia-Molina came to the important conclusion that the update of most web pages belonged to Poisson distribution [9].

F. Robots Protocol

Robots Protocol is a standard Web Crawler should conscientiously observe with Robots.txt document as its main content. In general conditions, Crawler writers will observe this protocol. A Crawler can still acquire web information without observing Robots.txt standard; but if a webmaster finds that a Crawler has problems, he will connect with its owner through its logo, or even prevent this Web Crawler from extracting some web pages in other ways. So Crawler developers shall conscientiously observe this protocol [10].

After entering a web page, web spider will first visit the text file equipped with Robots Protocol, which is usually in the root directory of web server, such as www.163.com/Robots.txt. With the protocol file, Robots.txt, webmasters can define the directories Web Crawler can not visit or the specific directories certain Web Crawlers can not visit [11]. For instance, if the executable directory and temporary file directory of some web pages do not want to be searched by search engine, webmasters can define these two directories as directories which deny access.

The file format of Robots is as follows.

User-agent:

It is the name of Crawler. In the file “Robots.txt”, if more than one User-agent records show that many Crawlers are limited by this protocol, this file shall have at least one User-agent record. If the value of this record is set as *, this protocol is effective for any Crawler. In the file “Robots.txt”, record like “User-agent:*” can only have one.

Disallow:

It is used to describe a URL which does not want to be visited. This URL can be a complete path or part of it. Any URL started with Disallow can not be visited. This file format cannot be visited by Robot [12].

For example:


B: “Disallow:/help/*” means that Crawler can grasp /help.html but can not grasp /help/index.html.

C: If the record of Disallow is empty, all pages of this website can be grasped by Crawler and in file “/robots.txt”, there are two or more Disallow records. If “/robots.txt” is an empty file, this website is open to any Crawler and can be grasped. Apart from observing Robots Protocol, Crawler should do its best to reasonably plan grasping strength by weakening the grasping strength during daytime while moderately increasing grasping strength at night when visit of Web host is low. Because of time difference, when it is daytime in Eastern Hemisphere, Western Hemisphere is at night. So the Crawler can enhance the strength of grasping American and European websites during the day while increasing the strength of grasping websites of its own country at night [13].

Even so, Crawler always inevitably brings trouble to Web host of other World Wide Web. So monitoring program of website grasping is indispensable. This program records the grasping traffic of every website to avoid problems caused when grasping strength is occasionally excessive.
III. A DISTRIBUTED DESIGN OF WEB CRAWLER SYSTEM

Thousands of WWW servers on the web form mass information through the web link between them, with each connection between the hosts being relatively independent. Single processor system is restricted by the CPU handling capacity, disk storage capacity and network bandwidth resources, etc. It is impossible to boast the ability of dealing with such huge amounts of information, not to mention to catch up with the rapid growth of web information. The distributed technology becomes a choice. As a design of distributed system, it pursues the following goals: (1) The grasping ability of a single machine should not decrease a lot when the number of grasping machines increases, i.e. the communication and management expenses of the system should be reduced to the minimum while pursuing load balance. (2) Starting from the actual operation, dynamic configuration of the system should be considered, i.e. to allow the addition or removing of one or more machines during the operation.

A. A Distributed Structure Design of Web Crawler System

To design a robust and efficient web crawler, it is needed to make the task distribution across multiple machines in concurrent processing. Huge webpages should be independently distributed on the network and they should provide adequate possibility and rationality for concurrent accesses. Meanwhile, concurrent distribution will save network bandwidth resources. Besides, in order to improve the recall ratio, precision and search speed of the whole system, the internal algorithm of the search should boast certain intellectualization. Therefore, the distributed web crawler adopts the following structure design as in Figure 3.

In system design, it is needed to make the task distribution across multiple machines in concurrent processing. Huge web pages should be independently distributed on the network and they should provide adequate possibility and rationality for concurrent accesses. Meanwhile, concurrent distribution will save network bandwidth resources. Besides, in order to improve the recall ratio, precision and search speed of the whole system, the internal algorithm of the search should boast certain intellectualization.

The core of system distribution is data distribution. The chief dispatcher is responsible for distributing URL to every distributed crawler. The distributed crawlers grasp webpages according to the HTTP protocol. In order to improve the speed, hundreds of distributed crawlers can usually be launched simultaneously. Distributed crawlers simultaneously analyze and dispose of the collected web pages, extract URL links and other relevant information, submit to their respective dispatchers, and their respective dispatchers submit them to the chief dispatcher.

B. Basic Process Design for a Distributed Web Crawler Grasping

Figure 4 is a brief flow chart which only shows page processes with no errors. In this process, the web crawler will start working when one URL is added to the waiting queue. So long as there is one webpage or web crawler disposing of one webpage in the waiting queue, the web crawler program will continue its working. When the waiting queue is null and there is no disposing of any webpages, the web crawler will stop working.

C. The Design of a Cooperative Grasping Algorithm of the Distributed Web Crawler

In the circumstance of multiple crawlers grasping, how the workload will be decomposed becomes the major problem. If the division is not clear, it is probable that multiple crawlers have grasped the same web, thus causing additional expenses. There are two options to solve it.

Scheme 1: To decompose through the web host’s IP address and make a certain crawler grasp only the webpages of a certain section of addresses.

Scheme 2: To decompose through the domain names of a web and make a certain crawler grasp only the webpages of a certain section of the domain names.

World Wide Web determines the location of host according to the IP address in the network infrastructure, but as the IP address is dotted decimal, it is hard to remember. So domain name is adopted to map the IP address. Due to the kindness of domain name towards people, such a problem arises: many domain names correspond to the same IP. Medium-sized and small websites usually use this method to provide different Web services. It only takes economic factor into consideration, for only one server is needed; but large websites, like Sina, Sohu and other portals, generally adopt load balance IP multicast technology, which means

![Figure 3. A distributed structure design of web crawler system](image)

![Figure 4. Basic process design for a distributed web crawler grasping](image)
that the same domain name corresponds to many IP addresses. In this way, robustness of the system is enhanced and load balance is achieved.

Given the condition that many domain names correspond to the same IP address or that the same domain name corresponds to many IP addresses, a fairly good way is to decompose tasks according to domain names, which means that as long as the web pages of large websites are not repeatedly grasped, small websites can accept strategy allocation tasks even if they repeatedly grasp. This method of allocation allocates domain names to different Crawlers to grasp and a Crawler can only grasp web pages of “appointed” domain name set. For example, sina.com.cn is “appointed” to be grasped by spider1, jxust.cn to be grasped by spider2 and sim.jx.cn is “appointed” to be grasped by spider3.

The main differences between these two kinds of solutions can be further understood by the following two examples.

Suppose that we have 3 spiders to analyze 2 websites, www.jxust.cn and www.sim.jx.cn. They have different domain name and have the same IP address (218.87.136.5). The homepages are: http://www.jxust.cn/index.html and http://www.sim.jx.cn/index.html. After DNS, they are actually both http://218.87.136.5/index.html. The domain decomposition scheme will make spider2 and spider3 repeatedly grasp this page. However, since the information of this site is not so much, the loss resulted from repeated grasps can be tolerated.

The IP distribution scheme to grasp tasks is different. For example, sina.com.cn(71.5.7.138) is “appointed” to be grasped by spider1, sina.com.cn(71.5.6.136) to be grasped by spider2, jxust.cn(218.87.136.5) to be grasped by spider3 and sim.jx.cn(218.87.136.5) is “appointed” to be grasped by spider3. In this allocation scheme, there is no repetition in the problem of different domains pointing to the same IP, and the grasping tasks of jxust.cn and sim.jx.cn are both completed by spider3. However, sina.com.cn corresponds to several IPs, and the allocation is by spider1 and spider2 respectively. In this way, the grasping task of spider1 and spider2 repeat with each other. Obviously, sina is a large-scale web and the loss resulting from this repeated grasping will be huge.

Through the comparison, the domain decomposition strategy is more reasonable which takes into consideration of the large website. Therefore, in Crawler system, the work of decomposing grasping tasks according to the domain name should be dealt with by a general scheduling to schedule web pages to different Crawlers to grasp through domain name decomposition.

A formal scheduling distribution is as follows:

Firstly, we suppose that n crawlers can work concurrently, and can define a function domain which can extract an URL domain name, such as:


Domain (URL) =news.163.com

(1) For any URL, it will use the function domain to extract the domain name of URL.

(2) Use MD5 signature function for the signatures domain, MD5(domain (URL)).

(3) Use MD5 signature value to do mould operations on n, int spider no=MD5(domain (URL)) %n.

(4) Allocate this URL to crawler numbered spider no to grasp.

A mold operation can help a universal set be divided into several equivalence classes. Therefore, the union of equivalence classes is equal to the universal set, and the elements in an equivalence class certainly do not belong to another equivalence class. A formal equivalence relation can be expressed as follows.

Set U as an universal set, and it is mapped respectively to S1, S2, ..., Sn through a certain equivalence relation. It satisfies the following two conditions:

(1) S1 ∪ S2 ∪ ... ∪ Sn=U

(2) if(a ∊ Si)&(b ∊ Sj)&(Si≠Sj) then a≠b

Generally, n is the integral power of 2. For example, the mod of 4, 8, 16, 32...can be rapidly obtained by the means of digit and (&), i.e. int spider no=MD5 (domain (URL)) & (n-1). Generally, to mod the integral power of 2, the means of & (n-1) could be employed (In it, n must be the integral power of 2) for rapid calculation.

D. Large-Scale Web Storatge Structure Design

The World Wide Web keeps changing all the time, so a web page database must be able to delete the old version after deletion of web pages. In this way, storage voids may be left. Updating can be understood as addition after deletion and the addition of application order to the web database. Therefore, some disk space compact technologies have to be adopted to recover the storage voids. Besides, updating and visiting should be mutually excluded to avoid synchronization of the errors. Therefore, a good page storage structure can bring excellent access performance.

To combine log structure and Hash structure based on its advantage is quite a good choice. For new web pages, the page's signature could be calculated through the URL. Then through modeling computation, a web page will be mapped to a unit on the Hash table, with each Hash table unit corresponding to the location of a log file. These newly added pages are mapped to Hash [1] through the calculation of Hash function, and then to the document Log1. You may want to randomly read an already accessed web page of URL, or still map to specific log files through similar Hash function calculation. Then you can search the B-tree index on the log file for corresponding page documents. You can acquire equivalent or even slightly better random access effect with log files (random access object files greatly decreased). What is worth mentioning most is that this kind of means can adopt processing batch writing-in, which will greatly improve the pure Hash structure. In each log file, writing-in queue will be added. Only when it has accumulated a certain amount of files, the processing batch can be realized, as shown in Figure. 5.
A Hash table can help to change the uncertainty of the insertion of those newly added web pages into certainty. Therefore, the addition of an inserted queue can be inserted into the target log files in batch mode. Through the Hash function decomposition, the size of each log on the basis of the Hash structure is far less than that in the log structure, and at the same time outweighs Hash barrels in the Hash structure a lot.

Besides, it must be ensured that each log can be stored in memory. So to determine the size of the Hash table in Hash-Log, it is necessary to consider the size of actual physical memory and the scale of web pages which need to be stored.

Table I gives a qualitative evaluation of three storage ways of web pages.

To sum up, without lots of opportunities for random access, log structure can be the best way to store web pages. As for the possible great deal of random access and the need of many new web pages, Hash-Log is a more ideal way to store web pages, for it can effectively support distributed web page storage and effectively distribute web page storage to every storage node to increase the reliability and stability of web page storage in the condition that multi-machine is used to store web pages in a larger environment. The overall search effect will not be affected a lot even if a storage node goes wrong.

IV. SYSTEM EVALUATIONS

A. Operating System Environment

B. Performance Evaluation

The data statistics results show that different sites have quite different grasping rates and the grasping amount of webpages which depends on the access speed of each site, and some sites have restrictions on crawlers’ grasping. These restrictions include speed restrictions, as well as some web accessibility restrictions. Under the condition of three-node-distributed cooperative grasping, the average rate can reach 7 pages per second, which renders very satisfactory results.

C. Scalability Evaluation

A system with good scalability can bring linear growth to its performance with the addition of cost. It is also easy to be streamlined or expanded.

Below is the influence on the grasping result of different numbers of cooperative grasping nodes. Figure 4 shows the operating result of the four kinds of different systems of scale respectively (the number of inspection cooperative grasping nodes = 1, 2, 4, 10 etc.) during the earlier 10 hours. Among them, the abscissa denotes the running time of the crawler system, with the unit being
hour. The y-coordinate represents the accumulated quantity of the grasped webpages.

Figure 6 shows that along with the increase of the number of cooperative grasping nodes, the basic system performance linearly increases. Therefore, this distributed system boasts good scalability and stability.

D. Task Load Balance Evaluation

The load balance of the system is based on the distributed web crawler cooperative grasping algorithm, which utilizes Hash function to allocate URL dynamically among the nodes. Since only one process is considered, one can not evaluate whether it has attained the load balancing only depending on the number of URLs allocated to each node in the process. Instead, all phases of the whole cooperative grasping process should be analyzed to evaluate the effect of load balance (The whole grasping process is divided into several phases in time). The experiment is carried out with 3 nodes cooperatively grasping 163.com. TABLE III shows the URL distribution of each node in the whole process of 5 hours’ running of the system.

It is shown in TABLE IV that each node has grasped a basically equal number of webpages. The system load balance of distributed web crawler has reached the expected elementary objective.

REFERENCES


