International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 01, January-2019

Improved Resource Sharing Through Optimized Search Engine

[1] Trust Mutero
[1] Lecturer,
Department of Accounting & Information Systems,
Great Zimbabwe University,
Zimbabwe

***Abstract*:- Search engine development has been explored extensively in the past two decades. In general up to date there are two main types of search engines, directories and Crawler based search engines. Some have recently come up with a hybrid of the two. There also exist some new developments in the search area with the use of the Bit torrent Protocol. This paper brings a hybrid search engine which strongly relies on concepts from crawler based search engines and bit torrent. It should be noted that not everything from the two concepts have been absorbed. The solution presented in this research introduces the concepts of initial seeder, seeder and lecher to combine with the traditional crawler search engine to extend search engine functionality. A prototype has been developed that tries to incorporate some of the ideas discussed in this paper. The prototype functions as a search engine that allows one to download resources be it books or videos from any other server which has the resource which in a way increases system reliability. The production system is meant to be used in institutions such as universities and colleges mainly for researches and sharing of scholarly materials as physical copy books are generally scarce.**

Key words: Resource, search engine, crawler based search engine, human powered directories, bit torrent, leecher, seeder, GoMut algorithm (GMT).

INTRODUCTION

As each day passes by, more and more people are being connected to the internet one way or the other, some for leisure, some for connecting with their friends worldwide, some for games, some for learning, some for electronic commerce (e-commerce) etc. It is worth noting that great potential during the learning process can be harnessed through several emerging technologies and can ease the burden for the learner as they can now determine the pace of learning they would manage considering their other daily chores. Several technologies have been developed to date which allow sharing of resources be they search engines like google, yahoo etc. Bit torrent protocol is worth mentioning as it is a newer technology that facilitates sharing of resources.

In general more is to be done in the development of systems that facilitate sharing of resources for academic purposes be they at intranet, extranet and Internet level in universities worldwide, in particular Africa. Physical books are generally scarce in libraries as the student- resource ratio is too high hence the possibility of a semester coming to an end with some students never having the opportunity to access a book as it would have been borrowed by some classmates and they maintain their cartel such that the rest of the class will not have access to the resources but they only circulate amongst themselves. This makes measuring individual performance of students difficult as some have access to resources whilst others do not. In light of this main challenge, this paper attempts to introduce a concept that can be used in a production system that can be used in institutions such as universities and colleges mainly for researches and sharing of scholarly materials (softcopy). This will allow equal access to resources for all hence a level playing field in terms of academic performance.

To date basically there exist two main types of search engines, crawler based search engines and human powered directories [1]. Of course there exist hybrids of the two. This paper aims to come up with a prototype search engine which will be designed from the concepts which will be discussed shortly. The purpose of any search engine is to be able to retrieve content from the World Wide Web (www). A major challenge being faced today is the fact that most search engines retrieve irrelevant content.

During the recent years, another technology termed BitTorrent protocol has been developed and is currently being used for file and video sharing. The technology was good but the major problem is it is now used more for illegal video sharing and piracy and some are facing charges from some record companies because of illegal music downloading. Not much emphasis has been placed on developing search engines with a scholarly focus. Most students are not able to retrieve relevant information pertaining to their subject area at times because of poor querying but at times because of poor search engine algorithms.

PROBLEM DEFINITION

- .BitTorrent has been used mainly for illegal purposes. The question one can pause is why not implementing it for legal purposes. Paper proposes implementing the protocol in a legal environment and for this research it will be implemented in an educational institution and in particular at University of Zimbabwe.
- The BitTorrent protocol provides no way to index torrent files. There usually exist torrent hosting sites.
- BitTorrent relies on a protocol named the “Choking” algorithm. This protocol sabotages one’s download ability by reducing the speed if one is more of a gainer (leecher) and not a giver (seeder).

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 01, January-2019

RESEARCH AIM AND OBJECTIVE

The main focus of this research study is to come up with a hybrid search engine which combines concepts from the BitTorrent protocol, crawler and directory search engines, so as to improve the quality of search results whilst maintaining simplicity.

Justification

BitTorrent has had more side effects rather than gains. If well used it can have a positive impact. The study is to show that the protocol can be implemented for researches and scholarly purposes. Currently there is no search engine in universities around Zimbabwe to help students in their researches. Usually it is the case that there are not enough physical books on any subject matter. Currently all the students now have access to computers and the internet. Developing an intranet system that allows students to share resources would be a brilliant idea. In developed countries lectures can be conducted via video conferencing. Such videos can be disseminated through the system and be beneficial even to students who did not afford the chance to attend the lecture.

GENERAL SEARCH ENGINE FUNCTIONALITY

In general the main aim of all the types of search engine is to be able to retrieve the required information at the time required. According to Paul de Vrieze in his masters thesis "improving search technology" on page 9 he says it comprises of two main parts a robot and a retrieval system. The robot/spider is responsible for retrieving web pages whilst the retrieval system is mainly responsible for querying the database and presenting results to the user. The results one retrieves after querying the search engine are the links that are stored in the search engine server which the robot had already gone through and compiled. That is why searching over the internet is fast otherwise it would take ages to retrieve information had it been the case that the robot looks for what you want at the time you want it [4].

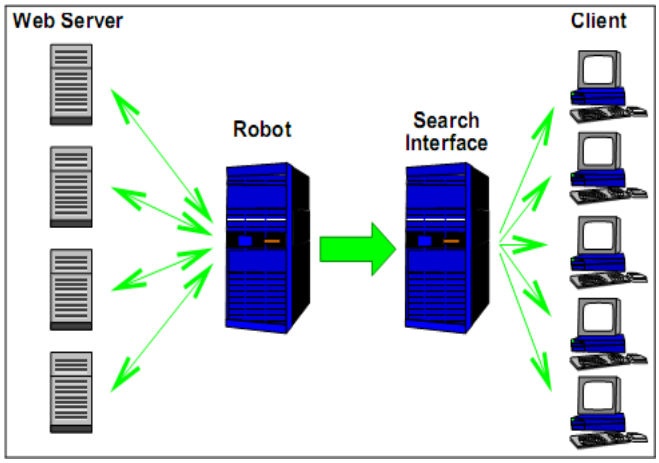

Fig 1 General structure of a search engine.

CRAWLER BASED SEARCH ENGINES

The most popular search engine to date. It consists of a robot/spider and a web server. The web server is a machine that is dedicated to the sole function of hosting the search engine website. The web server typically has large disc storage space of several Gigabytes or terabytes. When the robot searches for information, it returns with it and stores it in the web server. When a user now queries, in fact what he/she is doing is searching the database of the web server.

Robot

Is a software designed so as to maneuver from one link to another following the chain, summarizing the information it retrieved at the links and returning it back to be stored in the web server. On visiting a site, it searches within its code for tags, keywords, headings, hyperlinks and any other captivating information because there is where you will deduce what the site is all about. It then goes on to summarize that information and the summary is then stored in the database. As can be illustrated from the diagram below, the spider has a starting point and from that point it moves on to the next point and to the next and to the next until it has exhausted all the links. But for each link, it builds a list of keywords found on each website and that list is termed an index. That summary is then compressed so as to save disk space because in the long run as you index millions and millions of sites you might end up running out of disk space. The data is then stored in the database for users to access [5].

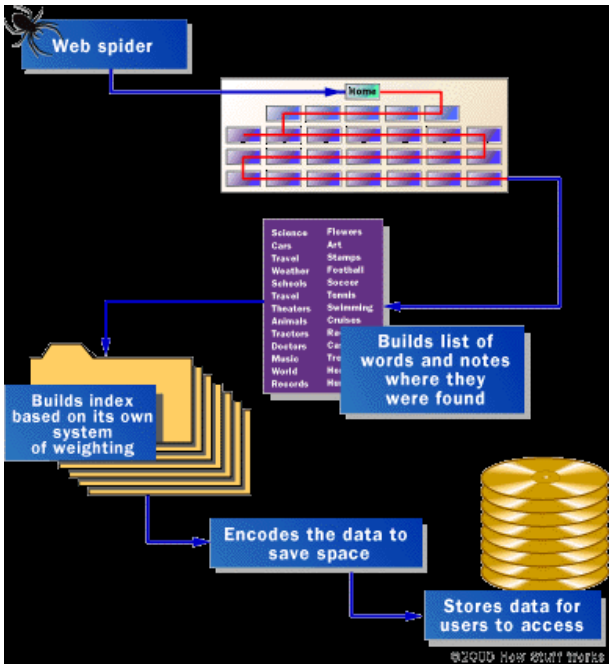

Figure 2. General Spider function adapted from Curt Franklin http://computer.howstuffworks.com/search-engine.htm/printable

One of the major problems with these search engines is the **quality of results**. Robots or spiders usually go all the way looking for all links, meta tags etc. This would mean to say more of garbage results may be retrieved. This points to the fact that there has to be some way to govern the extent to

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 01, January-2019

which the crawler can dig for information in a site, probably have it in levels representing the limit it can search up to at a given time. This is a major weakness in most search engines.

It can be noted that the number of documents being indexed are increasing in many orders of magnitude everyday but the user's ability to look at the documents has not. This implies that relevant results retrieved will be to a greater extent washed away in garbage results obtained. People usually only look at the first tens of results [2]. This research will not focus on quality of results for educational institutions.

Below is a discussion on one of the best search engines and its pitfalls:

Google
Is a crawler based search engine whose results are retrieved and displayed basing on what they call the PageRank algorithm. All the pages in the database are given a rank using Point awarding scale. It looks at how many pages link to that page. Page rank does not count all links from all pages equally and it normalizes the number of links on a page.
PageRank assumes that *page A* has pages ***T1***.....***Tn*** which point to it
The parameter d is a damping factor which can be set between 0 and 1 but is usually set to 0.85.
C(A) is defined as the number of links going out of *page A*.
The PageRank of a page is given as follows:

PR(A) = (1-d) + d (PR(T1)/C(T1) + ... + PR(Tn)/C(Tn)).

The Page Rank algorithm is the one that works wonders for Google's precision in relevant results. However it is not perfect because many times it also returns irrelevant results but most the time it brings about relevant results. This might be because of the fact that it indexes millions of pages and if you input a simple search query, say for example type in the word "virus effects" it will most likely return results mostly about computer viruses but maybe in fact you meant HIV effects on people, but with a mixture of virus effects in animals, plants and so on. PageRank algorithm works well with compound queries but is poor with simple queries, because if it was a compound query it would retrieve results exactly on the subject you want because your query might have been something like "Sexually transmitted Virus + effects in people". In this case no mistakes can be made, it is clear what exactly you are looking for and the result ranking will obviously be relevant according to the above algorithm. This is a pitfall in the PageRank algorithm and once addressed the search engine will be very precise in results. GMT seeks to rectify that part of the algorithm.

Also important is the issue of spider spamming. The more the spider moves to different sites and the more it searches the more it gathers both relevant and irrelevant data. This research proposes having a way to administer the movement of the spider. It should only go to and where it is sent. In this particular case the search engine will only be restricted to educational institutions only. Having mentioned about the first part we now move on the second part of the proposed GMT protocol. It focuses on peer to peer file sharing in institutions. Most of the concepts have been adopted from BitTorrent hence below is a discussion of how BitTorrent protocol functions.

BITTORRENT PROTOCOL

Is a peer to peer file sharing protocol based on the idea of several sources downloading and uploading too and from each other concurrently [8]. Large files are broken down into pieces and these pieces are then transmitted to the client who requires the resource. To promote sharing of bandwidth a Tit For Tat algorithm is implemented on each peer. This suggests that a peer must send data to another peer if it expects the other peer to send data back. Thus to successfully download from a BitTorrent network one has to allocate some of their upstream bandwidth to the network otherwise suffer very slow transfers. BitTorrent is interesting as it is currently used by many users to distribute large files. Originally used to distribute legal high quality bootleg recordings of live concerts, BitTorrent is now very popular with those who trade television, movies, arcade games, comic books and music illegally [9].

Below we will discuss some of the major terms used with the BitTorrent protocol:

- **Tracker**- A middleman who informs the peers of all the other peers in the network. It is a computer that is dedicated to monitoring the movement of resources, checking who has what and connecting a fellow peer to other peers which have the required resource at a particular point in time.
- **Torrent**- A file which provides a URL to the tracker as well contains a list of SHA1 hashes for the data being transferred. This is so that the hashes in the Torrent can be used to verify if the blocks received are valid or not. The user downloads this file. It usually contains url of the tracker

1. Pieces <hash1,hash2,….hashn>
2. Piece length
3. Name
4. Length

- **Peer** - A client to the network dedicated to a torrent.
- **Seeder** - a peer that provides the complete file.
- **Initial seeder** - a peer that provides the initial copy.
- **Choked** - A connection is choked if not file data is passed through it. Control data may flow but the transmission of actual blocks will not.
- **Interest** - indicates whether a peer has blocks which other peers want.
- **Leecher**- first locates the **.torrent** file that directs it to a **tracker**, which tells which other peers are downloading that file. As a leecher downloads pieces of the file, replicas of the pieces are created. More downloads mean more replicas available**.** As soon as a leecher has a complete piece, it can potentially share it with other down loaders. Eventually each leecher becomes a seeder by obtaining all the pieces, and assembles the file.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 01, January-2019

BITTORRENT PROTOCOL EXAMPLE

As can be seen from the diagram below, when a user wants a resource for example movie called “Harry Potter” they first of all have to obtain a file with a .torrent extension. That file will have information about The tracker and other peers who have the resource. The client informs the tracker who in turn informs connects the user to its seeders. Once it has connected to its peers then the download begins.The file is broken down into chunks and the many seeders will be supplying those chunks to the user. The purpose behind BitTorrent is to reduce the bandwidth load for the peer (the seeder) initially sharing the file. Peers who download from the seeding peer join the network and share their blocks of the file with other clients. Each file is split into blocks so a seeder can distribute blocks among the downloading peers such that peers can download the blocks off other peers. Thus a downloader effectively becomes an uploader. As more peers join they connect to the downloading peers and trade file blocks from them.

BitTorrent is subject to collusion [3]. This means that a set of peers can work together against a set of victims. The coalition could easily easily turn victims to seeders by uploading only a nominal amount of data to the victims and getting a full unchoke slot in return This clearly shows that the choking algorithm is only disastrous and at worst counter-productive. It is not really vital to the functioning of the prototype and thus can be removed.

TEST ENVIRONMENT

The experiment was conducted at University of Zimbabwe in a computer laboratory. Comparing the performance of the first part of the GMT protocol versus HTTP and FTP involved setting up 5 machines at University of Zimbabwe network.

The PC’s were P4 desktops with 1-1.8GHz processors and anything varying from 256-512MB RAM as system bottlenecks would affect all tests equally. It was performed during an academic break while university was closed. For the software utilized, all the machines had Linux, XAMP server PHP and SQL installed.

For the torrent part, one machine was set as tracker and for the tracker port 1000 for listening. This helped in that there would no conflicts with HTTP and FTP on the common ports generally used. HTTP uses port 80 and FTP uses port 21. If they had to conflict that would cause a distortion in the results obtained. Log files were used to determine the time it took to successfully transfer content to all machines. Tests were performed in an incremental fashion, starting out with the host server transferring the file to one other machine on the network. With the progression of time this meant to say the number of servers increased with time, from 1 server to 2 to 3 and so on until all the other machines had the copy of the resource being shared among the peers. Tests consisted of transferring zip files of approximately 10MB, 50MB and 500MB.This was to check if it would make any difference in protocol’s performance.

GMT ALGORITHM

The algorithm designed took into consideration some of the weaknesses from previous algorithms and systems discussed in the literature review. Various ideas were taken from the previous ones discussed but modified to some extent. The algorithm is as follows:

GMT Crawler Search Algorithm

Select the site which is to be visited

1. *Specify the indexing depth spider is to reach.*
   Either
   full depth (if it is to follow through all the links to the end of the chain).
   Else
   Spider to depth 1 to 10 (can be any number varying from 1 to10)
   Upon selection of indexing depth and site URL/ IP address then move on to next step Else go back to main menu and repeat process 1 or exit program.
2. *Spider to visit site and go into its source code. Look for all the meta tags in it, the hyperlinks, headings, most frequently used words.*
3. *Capture all those links, following through from one to the next (if a full indexing depth was set spider will follow through all the links, from a link to a sub link to sub of sub link up until there is nothing to follow anymore. If the depth was specified to a certain level spider will only stop on reaching the level).*
4. *Encode the index collection so that disk space is conserved. Return to GMT server with compilation, and store data in database table 1.*
5. *From table 1 where collected data is first stored rank the pages as follows:*
   *assume that page A has pages T**1**…..T**n** which point to it*
   The parameter d is a damping factor which can be set between 0 and 1 but is usually set to 0.85. C(A) is defined as the number of links going out of page A.
   The PageRank of a page is given as follows:
   PR(A) = (1-d) + d (PR(T1)/C(T1) + ... + PR(Tn)/C(Tn)).
6. *When page is evaluated, it is stored in table 2 with its rank topic etc.*
7. *If user searches on topic, retrieve from databse. Priority is given to pages with*
   Either
   Complete word(s) match. (PRIORITY 1)
   Else if
   Part of the query, either some of the words used in querying (PRIORITY 2)
   Else
   Part of a word matches on letters (PRIORITY 4)
   Else
   Output “NO RESULTS FOUND”
   Using the two criteria discussed in 6) and 7) results are retrieved basing on the two, but step 7) overidding 6) if need be.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 01, January-2019

For File sharing part
For file sharing part (BitTrust)
If willing to be a seeder

1. *Create .trust file include the file name with the extension. The .trust file is to be used by peers who want to download file you are sharing. Announce to the tracker about it through its address (Announcement URL). Tracker will be dedicated to the resource and were it is found.Tracker coordinates the events.*
2. *If peer is to download from the information of seeders and some who have part of the required file from the .trust file*
   Either
   Connect to peers with resource
   Or
   OUTPUT "connection failed"
3. *If connection succesful, divide file into manageable chunks say N chunks.Each should =<10MB Each seeder is to*
4. *supply one or more of the chunks. Allocate resources according to:*
   RAREST FIRST Algorithm. Determine the pieces that are most rare among your peers, and download those first.This ensures that the most commonly available pieces are left till the end to download.
5. *Each supplier to supply a chunk of file to client.*
6. *If download complete assemble chunks and arrange them according to chunk number and using SHA1 to determine if right portion or not.*
   END OF ALGORITHM

GMT ALGORITHM IMPLEMENTATION

The designed system had the following features as can be highlighted from the following screenshots. All the torrent files were named trust. So all **.torrent** extensions were in fact called **.trust**. For example a **harry_potter.torrent** in this implementation was a **harry_potter.trust**. For the peer to peer sharing part there were clients and a tracker. The page below shows the login page for the client part. Each machine had XAMP server installed on it. Each client hosted its own web pages rather than the traditional BitTorrent protocol clients whereby one had to install the software.

After the login, the client will meet the following page. On that page there exist many options either to seed, to download trust, to create a trust, to download file etc. The interface was kept as simple as possible for easy use.

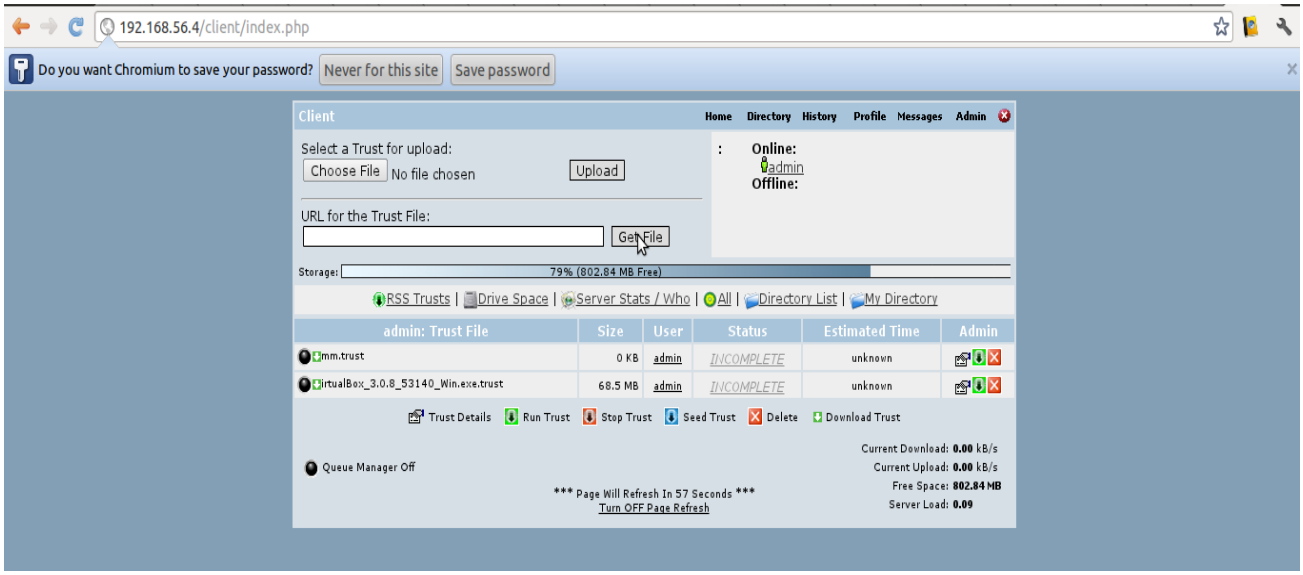

Fig 3 General interface for GMT.

If a client so wishes to seed a file, they can create a trust by naming the file and announcing to the tracker that they have such a file. The announcement URL is the address of the tracker, after inserting the information they will click on create and the .trust file is created.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 01, January-2019

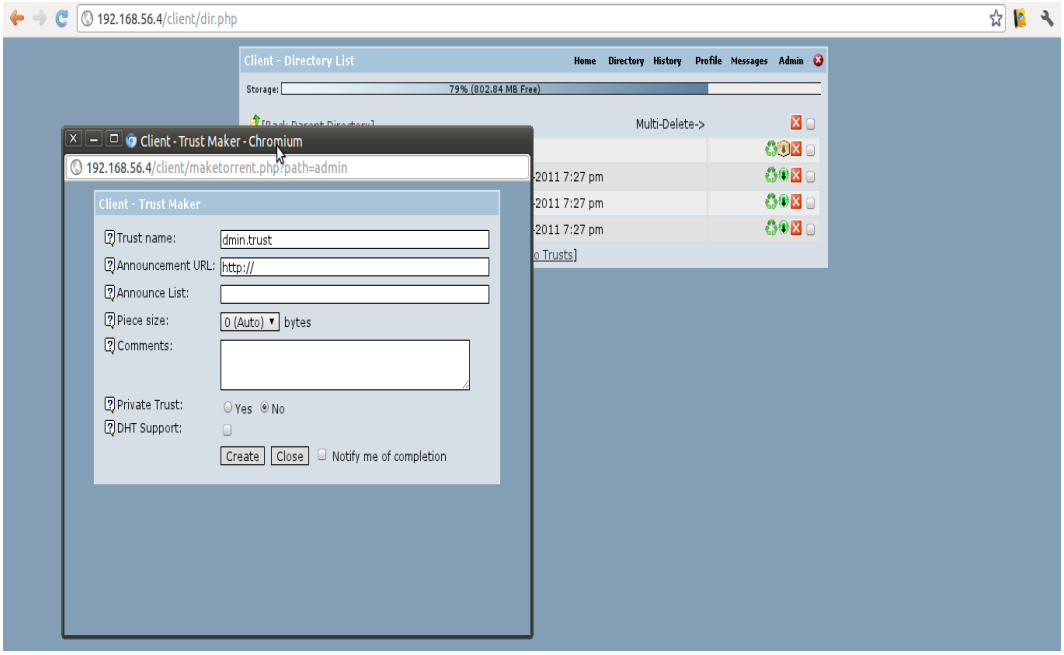

Fig 4. Creating a trust

If a client has the trust file, they are then able to download. Connection to fellow peers can be seen and the download progress represented as a percentage.

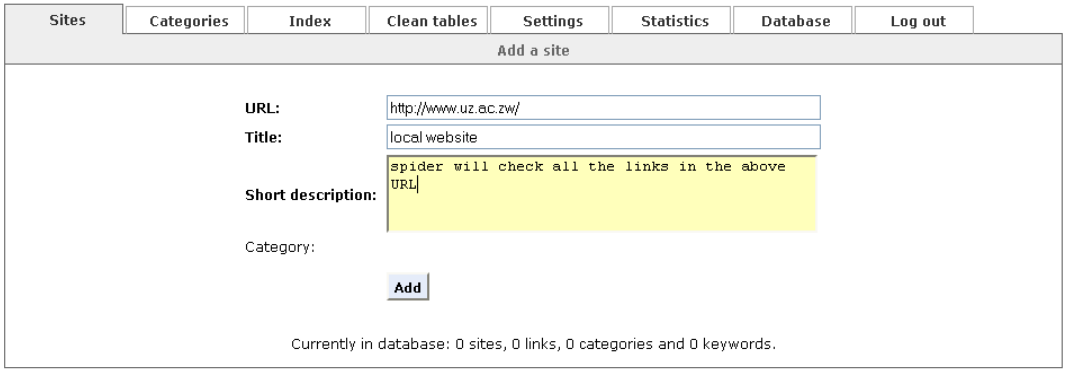

Fig 5. Assigning spider to navigate a website.

The page above is for the crawler part of the prototype. As can be seen, the administrator is able to enter the URL of the website which they want the crawler to spider and return a summary to be stored in the database. The spider does not just visit sites anyhow. It can only visit sites entered by the administrator. This helps to restrict crawler freedom and restrict it as in this case to function and research on educational websites only. This helps to curb a large percentage of junk results. Once the site is added, the above page allows them to start to index the entered site. On clicking on option they can now enter the index level which they want the spider to crawl as shown on the diagram below.

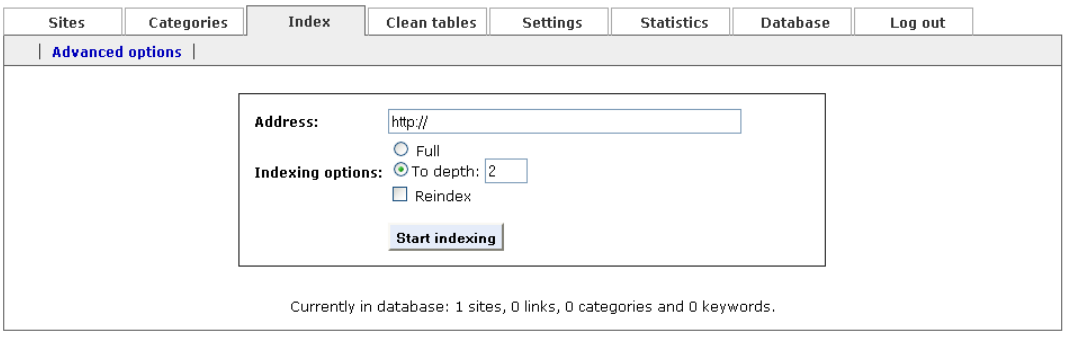

Fig 6. Indexing depth.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 01, January-2019

As shown above one can specify the index level to full or a number they so wish. Once done the spider will start moving, visiting the entered site. A list of the links found is displayed on the above page. Once done it will tell that it has finished indexing. Generally the time taken indexing is directly proportional to the indexing depth set. If a full depth is required then that means it will follow through every link till there are no more links. If the depth is a number say depth 2 it will only take a reasonably short amount of time.

This was just a discussion on the functioning of the spider. It was more of the back part of the design because users will not view those pages. Now coming to the front end or rather the user interface for the search engine, it is shown on the diagram below.

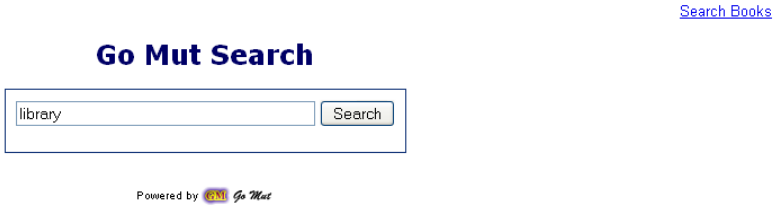

Fig 7. Front end search engine interface.

The user can enter the search query in the textbox below and click on search. The results are then retrieved from the database if they match to some extent what already was in the database.

DATA ANALYSIS

The time it took for each particular file transfer was recorded in log files. The values in the log files were compiled and stored in Microsoft Excel which was used for the analysis and data mining so as to better explain the obtained results.

LIMITATIONS AND ASSUMPTIONS

- The experiment was conducted using Unshieded Twisted Pair (UTP) cables. This was a limitation because had it been the case that other network mediums were accessible a comparison would have been made on the results to check if medium did not produce any distortion of results.
- The above can also be viewed as an assumption that the medium did not have any effects on the results produced.
- The delay in connecting peers on downloading files is in the range of +/-20 seconds.

RATIO	10MB GMT	10MB HTTP	10MB FTP
1-TO-1	2030	1719	3255
1-TO-2	2815	2144	3984
1-TO-3	3413	2993	4988
1-TO-4	4755	4133	6208
1-TO-5	5878	4988	7524

Table 1. Shows the simulation results on GMT, HTTP and FTP on file size 10MB.

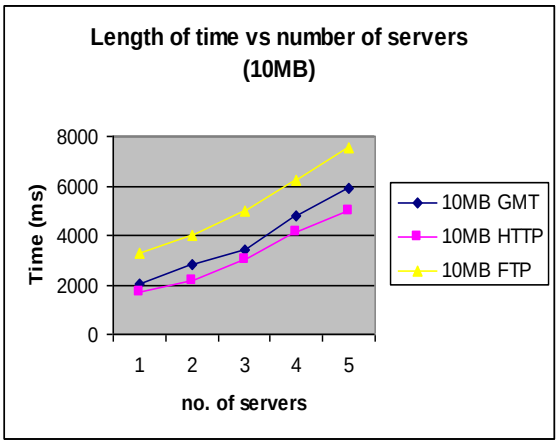

Figure 5. Shows the performance of HTTP, FTP and GMT on file of size 10MB .

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 01, January-2019

As highlighted from the table above for 10MB file size, generally HTTP had the least time in all cases, followed by GMT then lastly FTP. Performance of HTTP and GMT almost equaled when server ratio was 1-to-1 and 1-to-3. HTTP proved better for small file size than GMT by not so large a margin as highlighted from the graph above.

RATIO	50MB GMT	50MB HTTP	50MB FTP
1-TO-1	6177	4837	5708
1-TO-2	8509	5964	7193
1-TO-3	8966	6758	8807
1-TO-4	7749	8783	9708
1-TO-5	7109	8969	11655

Table 2. Shows the simulation results on GMT, HTTP and FTP on file size 50MB.

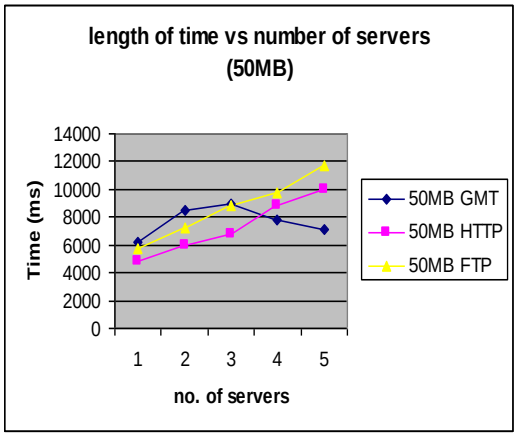

Figure 6. Shows the performance of HTTP, FTP and GMT on file of size 50MB .

For 50MB file sizes, both FTP and HTTP proved to be better than GMT during the first half of the tests when server ratio was 1-to-1 up to 1-to-3, FTP and GMT almost equaling at1-to-3, but from there onwards GMT drastically improved such that from server ratio 1-to-4 it had the best performance of the three, worse still there was a significant time difference between GMT and HTTP at 1-to-5 of approximately 1860 ms.

RATIO	500MB GMT	500MB HTTP	500MB FTP
1-TO-1	6177	4837	5708
1-TO-2	8509	5964	7193
1-TO-3	8966	6758	8807
1-TO-4	7749	8783	9708
1-TO-5	7109	8969	11655

Table 3. Shows the simulation results on GMT, HTTP and FTP on file size 500MB.

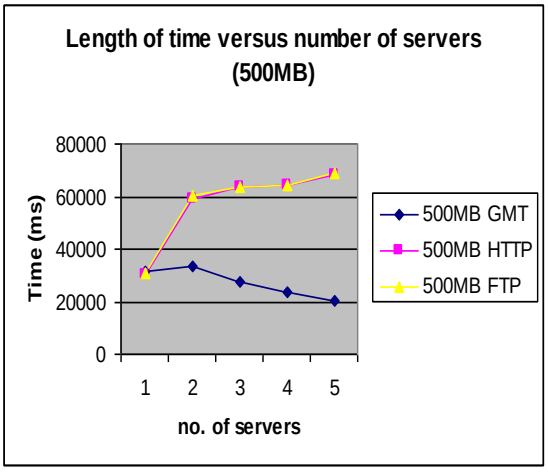

Figure 7. Shows the preformance of HTTP, FTP and GMT on file of size 500MB .

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 01, January-2019

For the largest file size, performance almost equaled only when server ratio was 1-to-1, but from 1-to-2 and onwards, GMT unanimously outclassed HTTP and FTP with an ever increasing gap. The collected results all seem to converge at the point that GMT performance improves with increased file size. It can be noted from the results for 50MB and 500MB files that from server ratio 1-to-3 and upwards, GMT provided the least time, followed by HTTP then lastly FTP. HTTP and FTP employ one server to one client ratio, which means at any given time there can only exist a single connection, unlike with GMT there can exist several connections to the same machine hence as hypothesized from 1-to-3 as more and more clients are graduating to being servers, the time it will take for a client to have a resource drastically reduces.

The poor performance for GMT from server ratio 1-to-1 to 1-to-2 can be accounted for by the fact that initially there is one server and one client hence GMT will be operating just like the ordinary HTTP and FTP, but when ratio is now 1-to-2, initially what happens is there is one server and two clients. When the first client finishes downloading it graduates to being a server hence now we have two servers but the other client is still downloading hence we now have two servers and one client. This means that once the client is connected to both servers, it will obtain some part from one server and the other from the other. The conclusion is because of connection time, that is why for small file (10MB) HTTP proved to perform better because generally it takes a small amount of time to transfer 10MB from one machine to another, hence for GMT because of the connection time which is almost constant, this added a delay to the normal transfer time.

EVALUATION OF THE CRAWLER PART

We can evaluate the second part of the search engine through a framework adopted from Paul de Vrieze [4]. According to him we can evaluate the prototype through analysis of the following 3 major categories:

- Understandability of the user interface
- Responsiveness
- Precision and Recall

Understandability of the user interface

GMT search engine offers more or less a similar type of interface to the already existing search engines. User tests were done with 5 peers on the interface and they all were comfortable with the interface and admitted it was just like that of google search engine. The only difference clearly noted was the combination of a type of torrent client with the normal search engine. 3 of 5 were of the opinion that there is a seaming less integration and it made life easier because both were web based unlike with normal torrent software which usually is not web based. The other difference noted is fact that with this one there is no need to install the client as with normal torrent clients software but rather, you just host the web page with a web server on the local; machine.

RESPONSIVENESS

The prototype was very much responsive. It was tested using the local site for the institution. If the page was already cached, it took a blink of an eye to retrieve the results of the search but if it was searched for the first time it took nothing more than 3 seconds to retrieve the result. The responsiveness could maybe be attributed to the fact that the tests were done in a LAN, so there was no need of routing.

Precision and Recall

We can also evaluate any search engine through recall and precision. Precision is defined as:

$$\textbf{Precision} = \frac{\text{Sum of the relevant documents retrieved by a search engine}}{\text{Total number of results evaluated}}$$

On precision, there were 2 major classifications that were made for the queries that were made. We had single and compound queries. The queries once submitted to the search engine, the first 12 hits were considered because generally only use the results retrieved on the first page and each page shows a maximum of 12 results. For a single query, the mean precision value obtained was 0.66 for 3 iterations and for a compound query the mean precision was 0.22 for 3 iterations as well as reflected on the graph below.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 8 Issue 01, January-2019

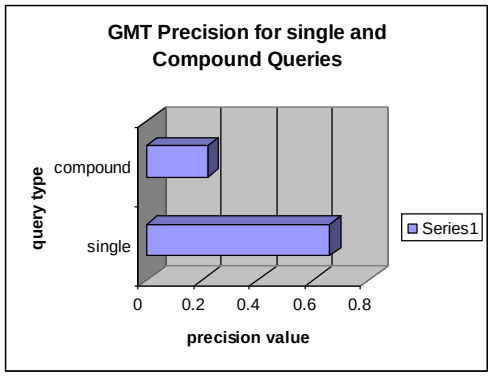

Figure 8. Shows GMT precision on query types single or compound.

The above diagram clearly shows that GMT is quiet flexible with single queries more than compound queries. More junk results were obtained when a compound query was issued. A single query produced results that were more or less what was required by the user of the system. This implies to say for every 12 hits, close to 8 retrieved links are relevant (precise) for a single query, and for every 12 hits, close to 3 links are relevant for a compound query. Recall is defined as the set of all relevant results over the set of all relevant web pages in existence at a certain point in time. We could not really perform. For example if you have stored a million documents, and of those 20 are relevant to your needs, you perform a search and you retrieve 12 on your first page. Of the 12, 8 are relevant to your needs, therefore you have obtained 8/20 of relevant books so your recall is 40%, but your precision is 8/12 which is 66.67%. According to Mansour Sarr [6], incorporating a spell checker into a search engine improves both precision and relative recall. GMT prototype did incorporate a spell checker which again improved its precision and recall.

Indexing level/depth

Indexing level is another key result area which can be noted. Unlike with other search engines, the usual result has low precision because it will be crowded with irrelevant material retrieved. With GMT the robot can index a page only to a level desired by the system administrator; one can specify the depth he or she wishes the spider to get to.

Specific Site Indexing

As can be reflected from the above figure, it is only the search engine administrator who can allow the spider to maneuver to certain sites by entering the address of the site to visit in the textbox named Address. This implies to say it is possible to restrict content found on the search engine to a scholarly nature only. It will turn out to be beneficial to students at large.

CONCLUSION

From the above results, it is evident that GMT works well with increased file size. This is because from principle the greater the file size, the greater the transfer time from one machine to another, the more the number of givers the lesser the time it takes a client to have the complete resource.

And also it can be noted that the algorithm is most precise with single queries than with compound queries. The GMT algorithm was designed and a prototype was developed to implement the designed algorithm. The obtained results show that the GMT algorithm works better than the already existing HTTP and FTP protocols for significantly large file sizes, but for small file sizes HTTP proved to be better by a slight margin and the difference was accounted for by the connection time from one machine to another.

Future Work

It might be the case that someone might delete the original file and rename another one to that same name. For example if there is a video tutorial on photoshop named **Changing_background.wma** of size 64MB, if the client removes that file and renames another video file to that same name which but the new file is 120MB, there is need to quickly identify that this is no longer the same resource as the new one. The research might have to look into MD5 (Message-Digest algorithm 5) which is a widely used cryptographic hash function. MD5 is usually applied in security applications and is usually used to check file integrity [7].

REFERENCES

[1] Sergey.Brin, Lawrence.Page. (1997). The Anatomy of a large-scale hyper textual web search engine, pages 1-2.
[2] Xiaoyun.Wang, Hongbo.Yu. (2009). How to break MD5 and other Hash Functions,page 1.
[3] Di Francesco.Paolo, (2009). Cucchillela.Stephano. Selfish strategies affecting the Bit Torrent Protocol, page 5.
[4] Paul.de Vrieze. (2002). Improving search engine technology (master thesis), pages 36-38.
[5] Heting. Chu, Marilyn. Rosenthal. (1996). Search Engines for the World Wide Web: A comparative study and Evaluation Methodology, pages.
[6] Mansour. Sarr. (2004). Improving Precision and Recall Using a Spellchecker in a Search Engine, page 26.
[7] S.M. Shafi, Rafiq.A. Rather. (2005). Precision and Recall of Five Search Engines for Retrieval of Scholarly Information in the Field of Biotechnology.
[8] Lincoln. Scully. (2007). Network File Distibution with the Bit Torrent Protocol.
[9] Abram. Hindle. (2004). Analysis of the P2P BitTorrent Protocol, page 1.