Framework for Green Search Engine Design

Kapil Kumar Nagwanshi
RCET Bhilai

Praval Kumar Jha
HCL Technologies Bangalore

Sipi Dubey
RCET Bhilai

ABSTRACT
Traditional search engines use a thin client, distributed model for crawling. This crawler based approach has certain drawbacks which could be removed with a proposed rich client based model. The rich client based search engine offers faster crawling and better updation time using lesser resources than thin client model, and it covers more of the World Wide Web than normal crawler based search engines. Although modern day search engine giants have improvised on various features such as ergonomics and utilities, along with several added goodies, little work is done to improve energy efficiency of such Large Scale Search Engines. As the Internet is increasing exponentially the search engines will involve more and more servers thus costing more and more energy. This ever increasing demand of search engines needs to be curbed down. Rather than multiplying server resources it is better to use existing servers which work in a congenial environment, using communication methods to reduce redundant downloading of data from different servers by the crawlers.This paper proposes a rich client based architecture for search engines along with analysis and comparison with present search engines. This could help into reducing the challenges of global warming, keeping up the speed and efficiency requirements.

General Terms
Search engine optimization.

Keywords
Search engines, thick client, rich client, updation delay, and crawler.

1. INTRODUCTION
Typically search engines use a crawler to crawl through URL's, extract links, and index mined data from the web-pages to build a web repository [1]. Huge Search engines maintain a gigantic web repository of web pages. This repository is indexed with help of data mining tools for faster searches. The crawlers reside on high performance servers spread throughout the world and crawling the web 24x7. When they find an un-indexed page they store it in repository and index it. Whereas when a web page is revisited during crawling, if the content is changed an update is made and the old page in repository is replaced by new page. Commercial search engines have periodic update policies for maintaining a relevant web repository [2].

The update frequency of search engine database for a particular site depends on number of parameters such as, (i) priorities decided by search engine, (ii) page rank of the site, (iii) frequency of occurrence in queue, and (iv) availability of the site on the WWW. The search engine may decide its priority for updating different sites which may be based on diverse criteria such as, (i) how popular the website is, (ii) what is the size of website, (iii) how often does site content transforms, and (iv) how many links point to that particular web site (which partially depends on the page rank) [6] The content change frequency and page-rank are used to schedule updates. Content changes are observed and an optimum refresh policy is obtained by ignoring too frequently changing pages [7].

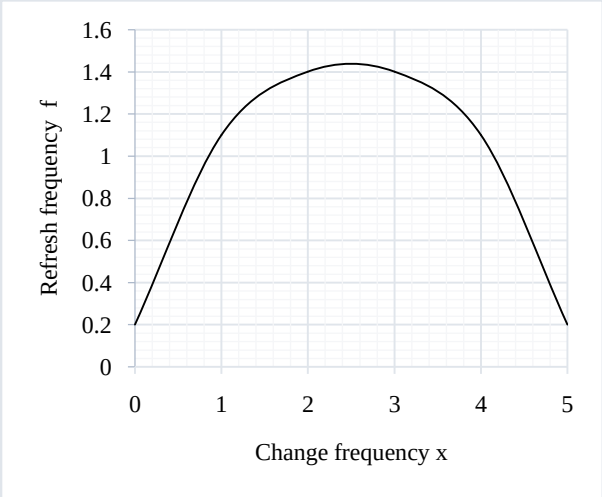

Fig. 1. Change frequency vs refresh frequency for freshness optimization

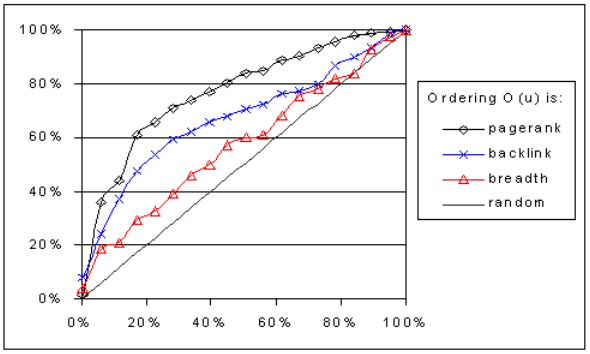

Fig. 2. Study result on URL Ordering with various methods of scheduling [10].

1.1 Traditional Search Engines
The drawbacks of traditional search engines includes (i) update delays, (ii) wastage of Storage space, (iii) incomplete data & less coverage, (iv) performance impacts on websites, and (v) contribution to global warming.

1.1.1 Update delays
The update delay is a major drawback to the traditional search engines. The update might be either steady or periodic [5]; in case of steady crawling we have to wait for a changed page to come again in the crawl path to be updated in the repository. Steady crawlers use page-rank to prioritize crawling. Periodic crawling is used along with change frequency estimation

details to schedule updating. Even then if a page is updated more than once between two updates only latest page is copied to repository [6]. And if a page change misses an update interval it will have to wait for more time.

1.1.2 Wastage of Storage Space

The pages crawled by the search engine are downloaded and saved into the web repository, and then indexed [2]. If the data already exists online what is the need to save it into repositories maintained by search engines doing this we are unknowingly wasting storage resources. A large scale search engines need not save all pages into its repository; but only abstract of the pages will do work.

1.1.3 Incomplete data & less coverage

Only very small no. of web pages is indexed amounting about 6-12 % [2]. This is due to following reasons:

- Search engines have limited bandwidth
- Crawlers may not reach important pages because they are too far.
- Depth of crawl is fixed, and page is located too deep.

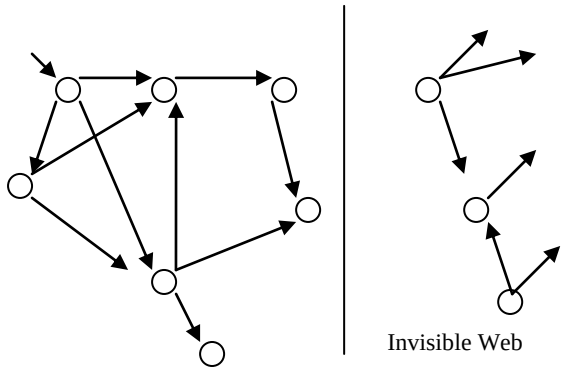

Fig. 3. Invisible web

The part of World Wide Web which is not restricted by dynamic content is also amalgamated into deep web due to inability of the crawler to find a link to it from the set of page it is crawling. This may be visualized from given Fig. 1.

1.1.4 Performance impact on websites

When the crawler revisits a website to gain updates, the web server uses its network bandwidth to serve the crawler; also it has to spend its CPU time and memory to entertain crawlers [5]. These resources could have been better utilized to serve users by the web server. All above resources are wasted if the site does not change frequently.

1.1.5 Contribution to global warming

When the crawlers are indefinitely crawling and downloading web pages a lot of energy is wasted in fetching of web pages from the World Wide Web as it involves transmission of packets all over the route from source to destination (i.e. Our Server)[7]. The pages are completely downloaded even when they are of the same date as of one we have in our repository. [8] , [9].

2. MATHEMATICAL MODELLING

A Page-rank is calculated based on number of pages that point to it. This is actually a measure based on the number of back-links to a page. A back-link is a link pointing to a page rather than pointing out from it. Measure is not purely a count of number of back-links because a weighting is used to provide more importance to back-links coming from important pages. Given a page p, it uses Bp to be the set of pages that point to p, and Fp be the links set out of p. Page-rank [3] of page p is defined as eq. (1)

$$R(p) = c * \sum_{q=0}^{n} \frac{R(q)}{N_q} \Big\|_{N_p = |F_p|}^{\forall q \in B_p} \quad (1)$$

Here *c* is a constant used for normalization whose value is between 0 and 1.

A problem called rank sink that exists with above page-rank calculation causes R value to increase for a cyclic reference. It is removed by adding additional term in eq. (1) and improved page rank *R'* is obtained as eq.(2)

$$R'(p) = c * \sum_{q=0}^{n} \frac{R(q)}{N_q} + c * E(v) \Big\|_{N_p = |F_p|}^{\forall q \in B_p} \quad (2)$$

Here *c* is maximized, *E(v)* is a vector that adds an artificial link. This simulates a random surfer who periodically decides to stop following links and jumps to a new page. *E(v)* adds links of small probabilities between every pair of nodes. This page-rank calculation takes place after the firing of a query, and sorted results are displayed with pages with highest page ranks first [4]. The update is scheduled after observing the frequency by which a site content changes. The pages changed daily are updated daily, while weekly changing sites are updated once in a week.

2.1 Reduced Time Complexity

The time required by the traditional crawlers to crawl a particular website will be as given below.

$$T_p = (\frac{n_p}{k}) * \left(\frac{S_p}{B} + D_n \right) \quad (3)$$

Where

n_p is the no. of pages

k is a constant defined by the techniques used for concurrency.

S_p is the average web page size

B is the bandwidth of the channel

D_n is the network delay involved.

Our time required to crawl the given website will be

$$T_p' = (\frac{n_p}{k}) * (\frac{S_p}{B_f}) + D_n \quad (4)$$

Here Bf = bandwidth of local browsing

$$\frac{B_f}{B} \approx 10^2$$

Since the XML sitemap file size increases with no. of pages using our technique, the D_n should be changed with the increase in size of file transferred. But for ease of simplicity we can omit this part, as there is no significant change in our comparison.

Comparing these two equations we see that the Network delay is multiplied each time, so for a practical channel the time required by a traditional search engine will be much higher than our proposed crawler.

The crawling of the site is said to be fruitful only if the content has changed, therefore the equations 3 & 4 need to be modified to find fruitful time aseq (5) and (6)-

$$T_f = \left\{ \left(\frac{n_p}{k}\right) * \left(\frac{S_p}{B} + D_n\right) * P(Change) \,\middle\|\, \forall P(Change) = \frac{(\lambda t)^n}{n!} e^{-\lambda t} \right\} \quad (5)$$

Where λ is frequency of change and *n* is the no. of arrivals (change events).

Assuming the change of page content is a Poisson process [6] the crawler will be effective only when the crawling is scheduled so that P (Change) is maximum which is exactly at λ.

$$T_f = \left\{ \left(\frac{n_p}{k} * \frac{S_p}{B_f} + D_n \right) * P(Change) \,\middle\|\, \forall P(Change) = 1 \right\} \quad (6)$$

P (change) in above equation become = 1, as the crawl event is fired by a change of content. These equations can be plot, and then the area covered by each curve gives the successful effort applied by each of them. The traditional crawler's performance varies with probability of change.

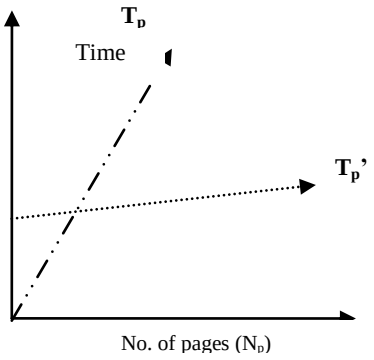

Fig. 1.Time Expenditure Comparison

2.2 Space Complexity

Our sitemap reduces the text content of the website by abstraction. Thus the space complexity will be reduced, the sitemap may be further space optimized by using text compression for large websites.

2.3 Lesser power consumption

While the traditional steady crawlers continuously crawls the web, further more continuous running of multiple servers too affects [6]. Using our architecture the updates become event fired and are analogous to Poisson's process [5]. Thus unnecessary wandering of crawlers, which wastes power, is removed. [8] , [9] .

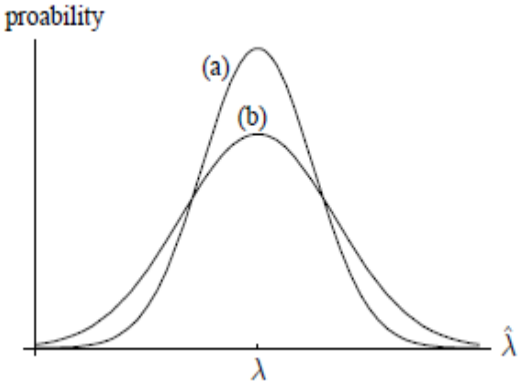

Fig. 2.Change in probability function with maxima at λ [6]

2.4 Better Coverage

The Deep web problem will be solved using this architecture. As each crawl is limited to a local resource, broken links will be less frequent. For dynamic contents the site administrator may provide URL to focus crawler on certain pages he wants to be indexed in the search engine.

3. CONCLUSIONS

Practicality of above architecture can be checked by a parallel run with traditional search engine. A multithreading enabled language should be used with the user agents; we suggest that Java should be the language for its implementation. With Java the user agent will become platform independent and could run on any type of web-servers. Database may be clustered for better performance and access time.

4. ACKNOWLEDGMENTS

Our sincere thanks to the RCET-Bhilai management for providing us essential infrastructural requirements and supports.

5. REFERENCES

[1] Guckian K., 2011. "Internet 202: The Invisible Web - Beyond Google", http://www.ire.org/resourcecenter/viewtipsheets.php?number=2347.

[2] Brin, S. & Page, L., 1998. "The anatomy of a large-scale hypertextual Web search engine", Computer networks and ISDN systems, Elsevier, Vol. 30(1-7), pp. 107-117.

[3] Dunham, M.,2003. "Data Mining: Introductory And Advanced Topics", 1/e , Pearson Education,.

[4] Brandman, O., Cho, J., Garcia-Molina, H. &Shivakumar, N., 2000. "Crawler-friendly web servers" , ACM SIGMETRICS Performance Evaluation Review, ACM, Vol. 28(2), pp. 14.

[5] Cho, J. & Garcia-Molina, H.,2003. "Estimating frequency of change" , ACM Transactions on Internet Technology (TOIT), ACM New York, NY, USA, Vol. 3(3), pp. 256-29.

[6] Boswell, D.,2003. "Distributed high-performance Web crawlers: a survey of the state of the art", Department of Electrical & Computer Engineering, University of California, San Diego, Citeseer.

[7] Wissner Gross, 2009. "A. Environmental Footprint Monitor For Computer Networks" , WO Patent WO/2009/076,667.

[8] Connolly, M. ,2009. "What Does A Google Search Really Cost?", Association of Small Computer Users in Education "Our Second Quarter Century of Resource Sharing", pp. 84.

[9] Junghoo, C., Garcia-Molina, H. & Page, L. ,1998. "Efficient crawling through URL ordering" , Computer Networks and ISDN Systems, Elsevier, Vol. 30(1), pp. 161-172.

[10] Arasu, A., Cho, J., Garcia-Molina, H., Paepcke, A. &Raghavan, S. , 2001. "Searching the web" , ACM Transactions on Internet Technology (TOIT), ACM, Vol. 1(1), pp. 43.