

# STOCHASTIC DIFFUSION PROCESS FOR A WEB SEARCH ENGINE

Mark James Talbot

Bsc Computer Science and Cybernetics  
siu00mjt@rdg.ac.uk

## ABSTRACT

The internet has grown from a small military communications network to a large dynamic repository of knowledge. As the internet gets larger people now longer can simply navigate around the linked structure of the web but require a system to do that search for you. Prior attempts at this have required a cache of the internet to search from but this is now becoming prohibitively expensive. The system described in this paper uses a soft focused web crawler to search intelligently for the best page in the internet from a query given by the user dynamically with no cache.

## 1. INTRODUCTION

As the internet grows search engines like google require ever increasing amounts of storage space. When google was first started in 1997 they reported in [1] there index contained 26 million pages at a total size of 147 GB of textual content. According to [2] the portion of the web indexed by google has grown to 3.3 billion pages. This number of pages would take up about 14 TB of space uncompressed based on the pages being the same size between the two reports. This also means that pages take a long time to get updated into the index because of the time taken to crawl through the internet in its entirety before re rating pages.

A new approach is to use on-line searching systems. These systems search the internet directly without the need for a cache. This makes them more bandwidth costly but means that the returned results are from the current page. The performance of the on-line search can be improved by directing the system to pages of perceived higher relevance first. This can be achieved in either one of two ways, a hard focused or soft focused crawlers. A hard focused crawler always picks the best page to move onto where as a soft focused crawler only makes it more probable to pick a better page. This allows for some exploration into the weaker information space in case that there is good data hidden in it.

The system uses a relatively new neural network system called the stochastic diffusion process. This system was first proposed in [4]. It uses agents to perform basic tests on sections of the search space to identify the best fit for a model in this space. Because of its random and statistical nature this system is much faster at finding inexact match's in the search space than liner search methods like inexact indexed string matching.

## 2. PAGE SELECTION

The system described here required modifications to the basic sds system to allow a the separate page downloaders and the rating system to work as a single agent directing the system around the internet. This was achieved by using a probability distribution that was based upon the page ratings and then randomly picking a new page based upon this. To generate a rating for a unviewed page the average rating of all the pages that link to that page is used. The formula for working out the rating is thus

$$R_x = \frac{\sum_{i=0}^n L_x^i}{n} \quad (1)$$

where  $R_x$  is the rating for page  $x$  and  $n$  is the number of links to a page and  $L_x^i$  is the rating for the page  $i$  linking to  $x$ . This is then used in the following algorithm to calculate the pages probability of being the next page to view.

$$\text{If } R_x > 0 \text{ then } R_x = R_x + t \quad (2)$$

$$R_x = R_x + 1 \quad (3)$$

$$T = \sum_{i=0}^t R_i \quad (4)$$

$$P = R_x / T \quad (5)$$

(2) distorts the already rated pages by the length  $t$  of the list of pages to traverse. This means that the total probability from the zero rated pages is always less than for a rated

page. (3) makes all the pages have a minimum rating of 1. This has to be done because at the point that there are no rated pages then (5) fails as  $T = 0$  which cause a problem when calculating it with a computer but with (3) in place  $T \geq t$ . These probability's are implemented differently in the system to increase speed. This is done by not actually working out the probability's for each item but using a first past the post method. A random number is generated and then the total of ratings is worked out down the list until it became larger than the random number.

$$r = \text{random}(0..1) \quad (6)$$

$$r * T \geq \sum_{i=0}^j R_i < \sum_{i=0}^{j-1} R_i \quad (7)$$

In (7)  $j$  is the index of the next page to view in the list. This system allows for the rating agents to pass information of active pages (e.g. those with appropriate content to the search) to the agents fetching content from the internet.

### 3. PAGE RANKING

The system also uses sds in its page ranking algorithm. Because the system is dynamic it can't really on link structure based search systems like googles page rank system defined in [1]. Instead it uses the text of the page to work out a ranking. This is done by defining a search string as a sentence and then looking at the linking structure of the words in the document.

This linking of words can be defined by the distance between the words forming the document. The rating for a particular grouping of words is defined by

$$f(x) = 1 - s/x \quad (8)$$

$$R = f(w_1) + \sum_{i=0}^W f(w_i) \quad (9)$$

Equation (9) gives a sum of the modified distances between words. An exponential decay was chosen to invert the distances because it allowed only the larger distances to be heavily penalized whilst keeping short distances with a very similar weight. A value of thirty was decided upon to generate a 50% scaling at about 40 characters which is the approximate length of a sentence. To find the closest word using sds a penalty was applied to the activation of the agents. The same scaling was used as a probability and then a random number between 0 and 1 is generated. This means that an agent becomes more likely to become active the closer it is to the root word of the search. This in turn means that through the diffusion stage the other agents tend towards the closest match to the root word of the search.

The implication of this is that the search system awards higher ratings to pages that have a target match at the beginning of the page which scalars to being  $\frac{100}{W}$  less at the end of the file. It also rates high the exact match but only decreases by a small amount by small deviations in the local surrounding of the text. This means that the system identifies the local context and rates the page based upon this. As people tend to place the most important information in the upper section of the document and side subjects towards the bottom then this scaling of the rating by the distance into the document works very well on correctly identifying close match pages.

### 4. CLUSTER ORGANIZATION

The system was designed so that it could be clustered over a number of machines. This was decided so that the large load of doing the text processing could be done over a number of machines and clearing the machine running the interface away from this load so they still remained responsive whilst a search was in operation. The system was split up into 4 parts being the mcp, a page grabber, a page rater and a database. Only the page grabbers require any external connections to the internet to perform. The page rater only does the rating of pages and all the text processing is off loaded to the page grabbers. This was decided because the rater will have a far heavier load on them due to the construction of a new sds network for every page and to reduce the amount of data need to be transmitted around the network. The page grabber behaves like a standard crawler except that the page to move on to is controlled by the mcp.

The mcp controls the entire system and negotiates the data flow in the network. It also contains the list of URLs that have been found in the pages that have currently been found and then decides on which one the ratters should view next. Any imbalance in the data rate being absorbed by the page raters and page grabbers is cached inside the mcp. The a message based system was decided on and XML was used to encode this. Each message contains a name and a data section. The destination of each message can be interpreted based on where it came from and its name. Each message is interpreted by the mcp and some operations are done on it there. This is mainly because of the choice of language used and the availability of api calls in the different language's.

The whole system is interfaced to by a set of web pages. These web pages start up each instance of the search and then read the results from the database where they have been stored by the mcp. This interface was chosen to make it closer to the standard interface for the major search engines on the net.

## 5. FUTURE WORK

The system so far can only support a basic subset of the HTML standard defined in [3] and so the more advanced features need to be supported. This includes frames and iframes as the system can only extract links from the anchor tag. It would also be adventitious to add support for the other common textual media types like postscript and pdf. The system that reads the pages for rating needs to be enhanced so that it can use a thesaurus and also understand the words in between the identified keywords and apparently modify the rating.

## 6. CONCLUSIONS

The system works as a single user system but as soon as multiple searches are performed it starts to reach the limits of a reasonable connection and download the same pages multiple times. Some form of cache is probably still needed but this is closer to a normal caching HTTP proxy

rather than the full caching of google and its counterparts. This level of caching is not unreasonable as the majority of searches will be for the same locations or similar ones. It is also not a difficult think to add to a network. It is also feasible to covert the rating method to a full result caching text based system like altavista to improve there results.

## 7. REFERENCES

- [1] The Anatomy of a Large-Scale Hypertextual Web Search Engine (1998): Sergey Brin, Lawrence Page
- [2] Search Engine Sizes (2003): Danny Sullivan ([www.searchenginewatch.com](http://www.searchenginewatch.com))
- [3] The 'text/HTML' Media Type(2000):L.Masinter
- [4] Stochastic Searching Networks. Proc. 1st IEE Conf. on Artifical Neural Networks, pp 329-331, London (1989)J.M. Bishop