# Improving Query Search Results by Using Click Patterns

Vishwanath D. Chavan
*Assistant Professor*
*Dept.Computer Science and Engineering*
*WIT Solapur,Maharastra, india*

Anil S. Naik
*Assistant Professor*
*Dept.Information Technology*
*WIT Solapur, Maharastra ,india*

Santosh C Pawar
*Mtech Student*
*Dept. Computer science and Engg*
*AHCET Hyderabad., india*

**Abstract - Understanding user search intent is critical component of modern search engines. A key limitation made by most query log analyses is the assumption that each clicked web result represents unique intent. However, there are many search tasks, where user intent is to explore many documents. In these cases, the assumption of a one to one co-occurrence between clicked documents and user intent breaks down.**

**To capture and understand such behaviors, we propose the use of click patterns. Click patterns capture the relationship among clicks on search results by treating the set of clicks made by a user as a single unit. We present an approach for the faster search on the basis of clicks using click pattern.**

**Keywords: Query search,clustering,click pattern**

## I. INTRODUCTION

### *A. Existing Systems*

Understanding and interpreting a user query is the first step a web search engine must take to full the user needs. To help in this endeavor, web search engines routinely analyze the click behaviors of past searchers. These past searchers' clicks provide crucial information about query intents and are used to measure query ambiguity, influence ranking decisions and result presentation, as well as generate query recommendations. Current query analysis techniques assume that each clicked web result provides evidence of a distinct intent.

Such a simplifying assumption is problematic, however, as it ignores them any reasons why users with the same semantic intent may click on more than one web result: they may have a high recall information need, such as when user is comparison shopping or completing a research task; or they may have an exploratory intent with no specific predefined interest. When query analysis techniques ignore such multi-click intents, they lose important evidence of relationships among documents and their representation of user’s intents.

### *B. Proposed System*

A benefit of using click patterns as an empirical estimation of intents is that, no matter what the true intent is, our clustered click patterns allow us to capture the similarity of intent accurately and robustly. Each individual user behaves in different ways when they are presented with the same search results. However, we hypothesize that underneath the noisy click behaviors each search query corresponds to an underlying behavior model, where user obey a set of common behavioral pattern. By identifying the common pattern of user click behaviors, we want to filter out the noises in user behaviors and achieve more accurate user models [3].

### *C. Need*

Understanding and interpreting a user query is the first step a web search engine must take to fulfill the user needs. To help in this, web search engines routinely analyze the click behaviors of past searcher. Current query analysis techniques assume that each clicked web result provides evidence of a distinct intent. Such a simplifying assumption is problematic, however, as it ignores the many reasons why user with the same semantic intent may click on more than one web result. When query analysis techniques ignore such multi-click intent, they lose important evidence of relationships among documents and cloud their representation of user intent.

We argue for explicitly representing multi-click intent by making click patterns a first-class abstraction in query analyses. We identify the common patterns of user behaviors using a clustering algorithm and treat these most common click patterns as a proxy representation of the user intent underlying a query. One of the interesting conceptual implications is that this allows us to represent query intent as a mixture of multiple navigational and multi-click intent.

### *D. Problem Statement*

Generally we get bunch of unrequired results in search. Here we improve the searching by understanding the user requirements by help of his clicks. By help of click pattern user gets more accurate results according to his requirements

### *E. System Architecture*

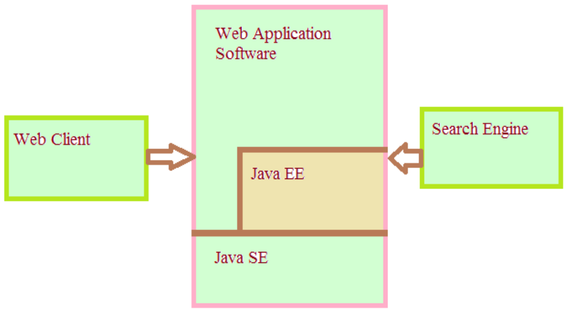


Fig 1.Basic Architecture Diagram

## II. SYSTEM DESIGN

### *A.Use Case Diagram*

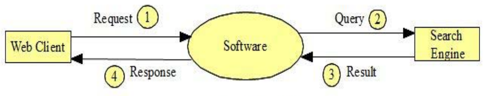


Figure a-1: Level 0

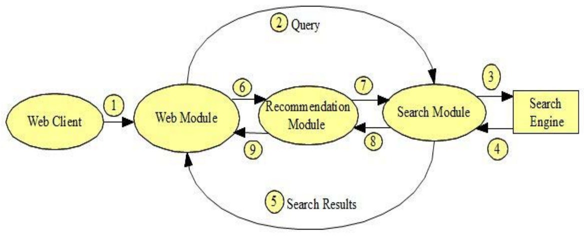


Figure a-2: Level 1 - Software

*B.Sequence Diagram*

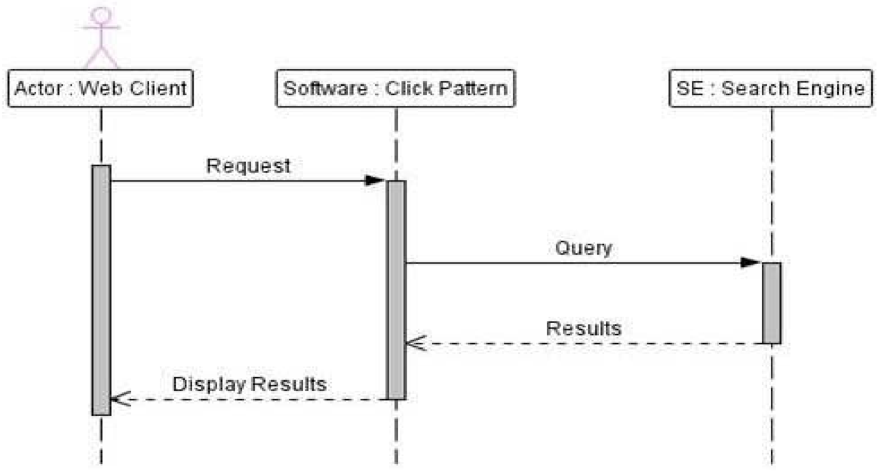


Fig .2.(a) Basic Sequence Diagram

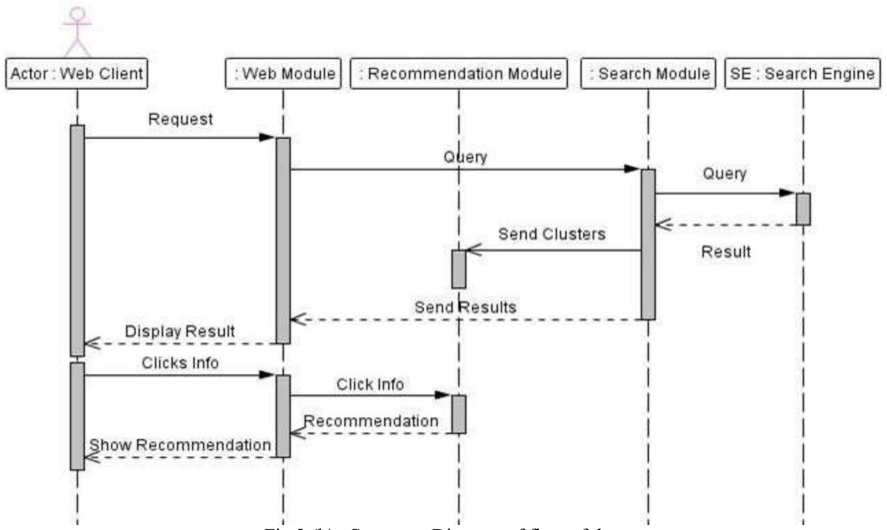


Fig 2.(b) Sequence Diagram of flow of data

## III. IMPLEMENTATION

### *A.Search Module*

Search Module is the heart of Click Patterns. It provides many search results with related searches from three different search engines such as Etools.ch, Google.com & Bing.com.The search query given by the user is passed through web module to the search module. The search module then checks search results for given query in its own cache. If it is not present then it passes query to the search engines for getting search results. After that search engine fetches the results in Html page form. Here we parse out these Html pages using Html parser API. Also, we fetch related searches from google.com & bing.com.Search engine Etool.ch fetches search results in Xml document format, which we parsed out using DOM parser which is inbuilt in Java API.

These results are returned to search module where they are combined, duplicated & redundant search results are removed.Similar to Search results the related searches are also filtered.The related searches obtained are compared with search query using cosine angle with threshold 0.65. Only the related searches above this threshold are taken into consideration. Further, these search results are forwarded for clustering. The hierarchical clustering algorithm creates clusters from these search results and attaches appropriate clusters to the respective search results. These attached search results are thenreturned to search module. Finally these search results are added to the cache and forwarded back to web module for representation.

*B. Hierarchical Clustering Algorithm*

When designing a web search clustering algorithm, special attention must be paid to ensuring that both content and description(or labels) of the resulting labels are meaningful to us. As stated, "A good cluster—or document grouping—is one, which possesses a good, readable description" [5]. The majority of open text clustering algorithms follows a scheme where cluster content discovery is performed first, and then, based on the content, the labels are determined.

We first attempt to ensure that we create a cluster label and only then assign documents to it. Specifically, we extract frequent phrases from the input documents, hoping they are the most informative source of search results. Finally, we match cluster labels with the extracted search results and assign relevant search results to them [4].

Input: Get search query & all search results from Search Module.

a) *Pre-processing:* Obtain keywords & frequent words by using TFIDF algorithm.
b) *Frequent Phrase Extraction:* Remove stop words from set of words obtained in step 2.
c) *Labeling Clusters:* Create cluster of these words & frequent phrases from words.
d) *Attaching Clusters:* Then attach clusters to corresponding search results.

Pre-processing:

Tf–idf is term frequency and inverse document frequency. In the case of the term frequency tf (*t*, *d*), the simplest choice is to simply use the *raw frequency* of a term in a document, i.e. the number of times that term *t* occurs in document *d*. If we denote the raw frequency of *t* by f (*t*, *d*), then the simple tf scheme is tf (*t*, *d*) = f (*t*, *d*).

The inverse document frequency is a measure of whether the term is common or rare across all documents. It is obtained by dividing the total number of documents by the number of documents containing the term, and then taking the logarithm of that quotient. Then tf–idf is calculated as

$$\mathrm{tfidf}(t,d,D) = \mathrm{tf}(t,d) \times \mathrm{idf}(t,D)$$

The aim of the preprocessing phase is to prune from the input all characters and terms that can possibly affect the quality of cluster labels [6].In tfidf words generated are in ascending order, 1/4th of words are taken in consideration for good clusters where they are best match with the search query.

$$\mathrm{idf}(t,D) = \log \frac{|D|}{|\{d \in D : t \in d\}|}$$

Frequent Phrase Extraction:

Stop words removal are very important operations in Information Retrieval. The input snippets are automatically generated summaries of the original documents and hence are usually very small (one or two sentences). Although TFIDF is capable of dealing with noisy data, without sufficient preprocessing, the majority of discovered abstract concepts would be related to meaningless frequent terms. So, we used stop words removal for next cluster formation.

Labeling Clusters:

We define frequent phrases as recurring ordered sequences of terms appearing in the input documents. Intuitively, when writing about something, we usually repeat the subject-related keywords to keep a reader's attention. Single subjects or groups of related subjects those are cognitively different from other abstract concepts.

Frequent words retrieved from stopwords removal are directly taken for cluster formation. The Search results related to these clusters are attached. Then, these clusters are taken recursively to form new frequent phrase-clusters with search results as intersection of these clusters. We consider those new frequent phrase-clusters with result size greater than 10. Also, we reduced the redundancy in phrase cluster titles by avoiding permutation & combination of titles.This process goes on till there is no new frequent phrase-clusters are obtained.

Attaching Clusters:

In this phase, the classic mapping method is used to assign the input documents to the cluster labels induced in the previous phase& vice versa. In a way, we re-query the input document set with all induced cluster labels.

3.3 Click Pattern Algorithm

Ideally, we want to discover a small set of underlying click patterns that capture the common click behavior of user[3]. Users obeying the same click pattern are expected to show very similar click behaviors, while users obeying different click patterns shall behave in quite different manners. Because the clicked documents are usually quite different for each query, obtaining direct supervision is difficult. Therefore, we resort to unsupervised methods. Specifically, we employ the divisive hierarchical clustering algorithm to find the patterns among the noisy sample of observed click behaviors. The detail of the algorithm is shown in Algorithm 1.

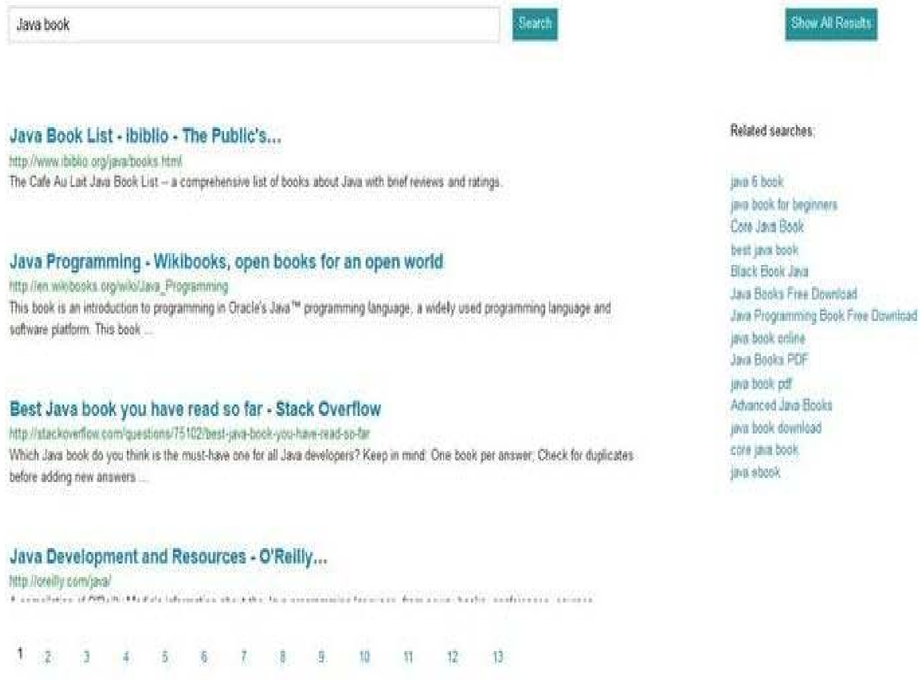
Input: Load all search results with attached clusters.

a) Obtain visited URL from Web Client.
b) Find search results from visited URL & attached clusters from search results.
c) Find common cluster set from clusters & return associated search results.
d) Show these recommended search results.
e) Repeat step 2 to 4 until Web Client visits search results.

After finding the common cluster set to present associated search results, we divide common cluster set into two parts by using threshold of search result size of 45. Intersection of search results from common clusters with search result size above threshold is taken. The search results of remaining common clusters below threshold are directly taken for associated search results.

## IV. TESTING

- Results for Search: Java Book



Java book

Java Book List - ibiblio - The Public's...
http://www.ibiblio.org/java/books.html
The Cafe Au Lait Java Book List – a comprehensive list of books about Java with brief reviews and ratings.

Java Programming - Wikibooks, open books for an open world
http://en.wikibooks.org/wiki/Java_Programming
This book is an introduction to programming in Oracle's Java™ programming language, a widely used programming language and software platform. This book ...

Best Java book you have read so far - Stack Overflow
http://stackoverflow.com/questions/75102/best-java-book-you-have-read-so-far
Which Java book do you think is the must-have one for all Java developers? Keep in mind: One book per answer; Check for duplicates before adding new answers ...

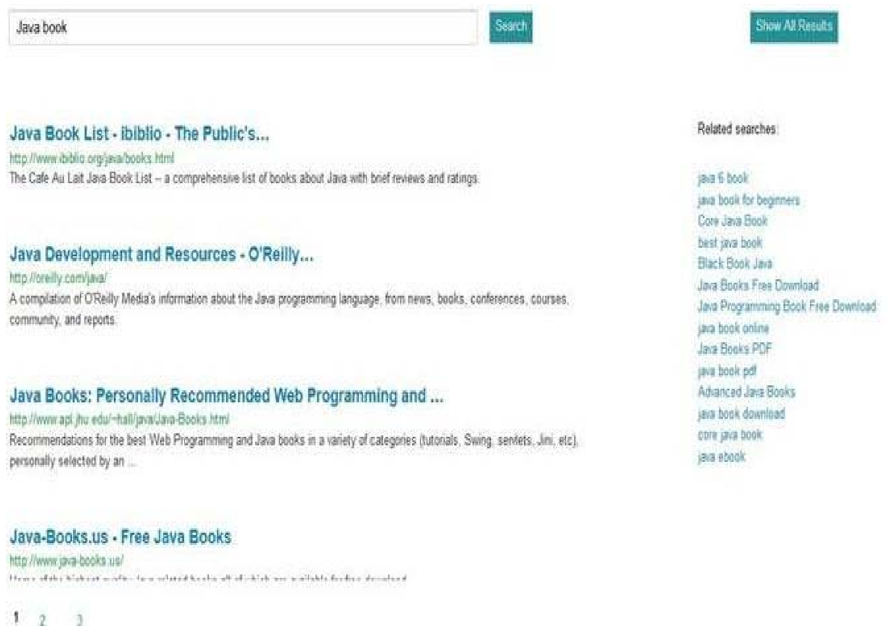
Java Development and Resources - O'Reilly...
http://oreilly.com/java/
[illegible]

1 2 3 4 5 6 7 8 9 10 11 12 13

Related searches:

java 6 book
java book for beginners
Core Java Book
best java book
Black Book Java
Java Books Free Download
Java Programming Book Free Download
java book online
Java Books PDF
java book pdf
Advanced Java Books
java book download
core java book
java ebook

Fig.3 (a) Before Applying Click Pattern



Java Book List - ibiblio - The Public's...
http://www.ibiblio.org/java/books.html
The Cafe Au Lait Java Book List -- a comprehensive list of books about Java with brief reviews and ratings.

Java Development and Resources - O'Reilly...
http://oreilly.com/java/
A compilation of O'Reilly Media's information about the Java programming language, from news, books, conferences, courses, community, and reports.

Java Books: Personally Recommended Web Programming and ...
http://www.apl.jhu.edu/~hall/java/Java-Books.html
Recommendations for the best Web Programming and Java books in a variety of categories (tutorials, Swing, servlets, Jini, etc), personally selected by an ...

Java-Books.us - Free Java Books
http://www.java-books.us/

1 2 3

Related searches:

java 6 book
java book for beginners
Core Java Book
best java book
Black Book Java
Java Books Free Download
Java Programming Book Free Download
java book online
Java Books PDF
java book pdf
Advanced Java Books
java book download
core java book
java ebook

Fig. 3(b) After Applying Click Pattern

## V .CONCLUSION

In this project, we used click patterns as an empirical representation of user intent in search engines, and a first-class abstraction for query analyses. We showed how click patterns can be extracted from logs of user behavior and demonstrate that click patterns provide a richer representation of query intents, from multi-click intents such as high-recall research tasks to navigational intents, and mixtures of query intents of various kinds. We are examining real query logs and providing the richer representation of search intents afforded by click patterns. We are also demonstrating the integration of click patterns into existing search analyses by adapting traditional search ambiguity and search recommendation tasks to use click patterns as the fundamental unit of user behavior.

As search engines continue their advancement from simple document retrieval to supporting higher-level question answering and aiding task completion, developing richer and more complete representations of search intent is of paramount importance. We believe click patterns represent a significant advance as an empirical representation of user intent, and have the potential to impact a broad set of information retrieval technologies, from the ranking to the presentation of results, where modeling user intent is critical.

## VI.FUTURE WORK

Click Pattern Algorithm is implemented for one user clicks. It has unique direction for user intention. It does not store the user clicks for a search query. So, it can be further extended as multiple users' search clicks for one search query. Then we can store clicks of users' for better search results.

For more accurate results we can change the Clustering Algorithm & Click Pattern Algorithm, where the Clustering Algorithm does not have classified clusters & Click Pattern Algorithm depends on it.

This project does not support for multiple browser tabs search.

## REFERENCES

[1] Data Mining: Concepts and Techniques, Third Edition (The Morgan Kaufmann Series in Data Management Systems)
[2] Data Mining: Introductory And Advanced Topics by Margaret H Dunham
[3] http://research.microsoft.com/apps/pubs/default.aspx?id=169639
[4] D. N. Aurelio and R. R. Mourant. The effects of web search engine query ambiguity and results sorting method on user performance and preference. Proceedings of the Human Factors and Ergonomics Society Annual Meeting.
[5] X. Li, Y. yi Wang, and A. Acero. Learning query intent from regularized click graphs.
[6] E. Yilmaz, M. Shokouhi, N. Craswell, and S. Robertson. Expected browsing utility for web search evaluation. In Proceedings of the 19th ACM international conference on Information and knowledge management
[7] http://project.carrot2.org/publications/osinski-2003-lingo.pdf\