Fall 2010

Event detection from click-through data via query clustering

Prabhu Kumar Angajala

EVENT DETECTION FROM CLICK-THROUGH DATA VIA QUERY CLUSTERING

by

PRABHU KUMAR ANGAJALA

A THESIS

Presented to the Faculty of the Graduate School of the

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

In Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE IN COMPUTER SCIENCE

2010

Approved by:

Dr. Sanjay Kumar Madria, Advisor
Dr. Jennifer Leopold
Dr. Fikret Ercal

PUBLICATION THESIS OPTION

This thesis consists of the following article that has been submitted for publication as follows:

Pages 16-38 have been submitted to the 11th International conference on Web Information System Engineering (WISE 2010).

ABSTRACT

The web is an index of real-world events and lot of knowledge can be mined from the web resources and their derivatives. Event detection is one recent research topic triggered from the domain of web data mining with the increasing popularity of search engines. In the visitor-centric approach, the click-through data generated by the web search engines is the start up resource with the intuition: often such data is event-driven. In this thesis, a retrospective algorithm is proposed to detect such real-world events from the click-through data. This approach differs from the existing work as it: (i) considers the click-through data as collaborative query sessions instead of mere web logs and try to understand user behavior (ii) tries to integrate the semantics, structure, and content of queries and pages (iii) aims to achieve the overall objective via Query Clustering. The problem of event detection is transformed into query clustering by generating clusters - hybrid cover graphs; each hybrid cover graph corresponds to a real-world event. The evolutionary pattern for the co-occurrences of query-page pairs in a hybrid cover graph is imposed for the quality purpose over a moving window period. Also, the approach is experimentally evaluated on a commercial search engine's data collected over 3 months with about 20 million web queries and page clicks from 650000 users. The results outperform the most recent work in this domain in terms of number of events detected, F-measures, entropy, recall etc.

ACKNOWLEDGMENTS

I owe a debt of gratitude to all those who have helped me with this thesis. First of all, I would like to thank my advisor Dr. Sanjay Kumar Madria, who gave me an opportunity to work on this research. His suggestions and encouragement carried me through difficult times. His valuable feedback contributed greatly to this thesis. Secondly, I would also like to express my gratitude to Dr. Jennifer Leopold and Dr. Fikret Ercal for serving on my thesis committee and for taking time to review this work.

I'm grateful to my loving brother Srinivasa Rao Angajala for his professional help and inputs. This project is partially funded by the Air Force Research Laboratory (AFRL) in Rome, New York and I express my gratitude to them as well.

TABLE OF CONTENTS

Page

PUBLICATION THESIS OPTION iii

ABSTRACT iv

ACKNOWLEDGMENTS v

LIST OF ILLUSTRATIONS viii

LIST OF TABLES ix

SECTION

1. INTRODUCTION 1
    - 1.1. CLICK-THROUGH DATA 1
    - 1.2. AUTHOR-CENTRIC VS. VISITOR-CENTRIC DATA 2
    - 1.3. THE THREE WEB DATA TYPES 3
        - 1.3.1. Content. 3
        - 1.3.2. Structure. 3
        - 1.3.3. Web usage. 3
    - 1.4. MOTIVATION 3
        - 1.4.1 Dynamics of Click-through data. 3
        - 1.4.2. Query Space. 5
        - 1.4.3. Via Query Clustering. 7
        - 1.4.4. Query Sessions. 8
        - 1.4.5. Data Pruning. 8
2. RELATED WORK 10
    - 2.1. EVENT DETECTION 10
    - 2.2. QUERY CLUSTERING 11
    - 2.3. CLUSTERING 11
        - 2.3.1. Notation. 11
        - 2.3.2. Clustering Algorithms. 12

2.3.2.1. Partitioning Clustering. 13

2.3.2.1.1. DBSCAN Algorithm 13

2.4. ASSOCIATION RULES 14

PAPER

ECO: Event Detection from Click-through Data via Query Clustering 16

1. INTRODUCTION 16

1.1. MOTIVATION 18

2. RELATED WORK 20

3. EVENT DETECTION FRAMEWORK 21

4. DATA REPRESENTATION 22

5. DISTANCE FUNCTION 25

5.1. Similarity based on Query Contents 25

5.2. Similarity based on Session Feedback 25

6. CLUSTERING PROCESS 26

6.1. Event Detection Algorithm 28

7. WORKING EXAMPLE 29

8. PERFORMANCE STUDY 31

8.1. Data Set 31

8.2. Result Analysis 32

8.3. Experimental Analysis 34

8.4. Effect of α 36

9. CONCLUSION 37

REFERENCES 37

BIBLIOGRAPHY 40

VITA 42

LIST OF ILLUSTRATIONS

Figure Page

1.1. Frequency of query “Easter Eggs”....…………………………………………….. 4

1.2. Co-occurrence of query and page ………………………………………............... 4

1.3. Notation symbols ……………………….. 5

1.4. State change probability matrix...………………………………………................ 6

1.5. State change state-diagram with probabilities.………………………….................. 6

1.6. Demonstration of query-page pair dynamics for “Easter” over six week period........ 7

2.1. Core points and border points ………………………..…………….............. 13

2.2. Density-reachability and density-connectivity...………..………………….............. 14

PAPER

1. Demonstration of query-page pair dynamics for “Easter” over six week period……… 19

2. Event detection framework overview…....…………………………………………….. 21

3. Summarized bipartite graph…...………………………………………………………… 22

4. Hybrid Cover Graph.……………………….…………………………….................. 24

5. Overview of clustering process ……………………………………………………….. 27

6. Clustering on Hybrid Cover Graph ………………..……………………….............. 27

7. Illustration of “Easter” and “Easter Eggs” clustering ………………………………… 30

8. Comparison of ECO with DECK on number of events detected …………….............. 35

9. Precision, recall, F-measure and entropy of ECO and DECK ………………………... 36

10. Effect of α on number of events detected ………………………..……….............. 36

LIST OF TABLES

Table Page

1.1. Sample click-through data……………………….………………………………... 1

1.2. User behavior on timeline………………………………………………................ 6

1.3. Frequent query-page pairs of popular portals…..…………………………................ 9

PAPER

1. Sample click-through data……………………….…………………………………... 17

2. Frequent query-page pairs of popular portals……..…………………………................ 19

3. Co-occurrence of query-page pairs over a 6 week window period…………................. 29

4. Support of query-page pairs over a 6 week window period...………………................. 30

5. Complete list of events detected..……………………………………….................. 33

1. INTRODUCTION

The approximate size of today's indexed World Wide Web is at least 45.93 billion pages as per existing estimation [1] and is a rich collection of all the real world objects. Web is a great source of knowledge to be mined to learn about topics, stories, events etc. Event detection is becoming increasingly popular because of its applicability in several diversified areas. Therefore the interpretation of "event" definition is context-dependent. An event can be associated with a sensor at a door post reporting how many people/cars have entered a building/freeway, web access log, security log, object trajectory in video surveillance and business activity monitoring in Business Intelligence etc. In our perspective and from the viewpoint of the Web, an event can be understood as some real-world activity. It stirs large scale querying and browsing activity that is of more interest to users over a sizable window period, which is unusual relative to normal patterns of querying and browsing behavior. Web is the collaborative work of many people, a few publishing, and all of them querying and retrieving the information.

1.1. CLICK-THROUGH DATA

Search engines record every single query and click activity from every single user in the web logs; called the click-through data which reflects the query and clicks activities of the users. Click-through data is more or less in the format shown in the table 1.1 below:

Table 1.1: Sample click-through data

AnonID	Query	Query Time	Item Rank	Click URL
7	Easter	2006-03-01 23:19:52	1	http://www.happy-easter.com
7	Easter eggs	2006-03-01 23:19:58	1	http://www.eeggs.com

Observe that the click-through data has the fields:

AnonID: The anonymous User ID from whom the search engine received the request. Actually search engines record the IP addresses of users who issued the queries but due to privacy issues, an anonymous ID is assigned for the IP addresses while disclosing the data. Proprietary

Query: The query issued by the user

Query Time: The time at which the search engine received the request from the user

Item Rank: The rank of the page item clicked from the result set in response to the query issued by the user.

ClickURL: The page clicked from the result-set returned by the search engine.

Note that the click-through data format varies slightly from one search engine to the other. Each line in the data represents one of two types of use activities:

1. A query that was not followed by the user clicking on a result item.
2. A click through on an item in the result list returned from a query.

In the first case (query only) there is data in only the first three columns/fields, namely AnonID, Query, and QueryTime. In the second case (click through), there is data in all five columns. For click through events, the query that preceded the click through is included. Note that if a user clicked on more than one result in the list returned from a single query, there will be two lines in the data to represent the two click activities. Also if the user requested the next "page" or results for some query, this appears as a subsequent identical query with a later time stamp.

1.2. AUTHOR-CENTRIC VS. VISITOR-CENTRIC DATA

Web data types are previously classified into two types in [3] as: author-centric and visitor-centric. Author-centric data is created by web publishers for user browsing and represents web content and structure data. It refers to a set of hyperlinked web pages that describes certain object or event. On the other hand, the visitor-centric data is generated as a result of users’ browsing activities or query activities. Observe that author-centric data describes author’s point of view while visitor-centric data reflects the web visitor’s point of view. Traditionally, only the author-centric data is considered while the rich collection of visitor-centric data is ignored. Lately, beginning with [3], visitor-centric data is taken into account because of the following reasons: First, the increasing popularity of the web search engines has given rise to a large number of search engine users issuing huge volumes of queries. These queries often return links to high quality web pages. Consequently, there is a large volume of click-through data that can be potentially exploited for event detection. Second, as shown in table 1.1, the click-through data contains the query keywords that are created by users and links to web pages that often describe real world events. Specifically, these keywords and the corresponding pages clicked by the users often reflect their response to contemporary real world events.

1.3. THE THREE WEB DATA TYPES

The three web data types that are identified in previous [2] efforts are:

1.3.1. Content. Text and multimedia of the documents on the web that present knowledge stories, topics and information etc.

1.3.2. Structure. Links that form a graph. Several graph theories are in existence to represent the structure of the documents on the web as a graph or set of graphs.

1.3.3. Web usage. Transactions from the web log. Click-through data is an example for the same.

Web data mining encompasses broad range of research topics like improving page ranking, better indexing, query clustering, query similarity, query suggestions, extracting semantic relations and event detection etc. All these areas are inter-related and many use the click-through data as the start up source. The seamless flow of advancement in developing better approaches in individual areas can be pipelined to improve existing techniques in the inter-related fields. So our effort in this thesis is to integrate the three web data types and achieve the overall objective via query clustering. In the attempt to exploit all possible resources (from both author-centric and visitor-centric data) and to integrate all the three web data types, we believe that our event detection approach will do better.

1.4. MOTIVATION

1.4.1 Dynamics of Click-through data. The dynamics in click-through data was previously identified in [3]. The dynamic nature refers to the evolving nature of the queries and pages in the click-through data over time. Users may formulate new queries that were not queried before, or new web pages that were not available earlier may now be clicked by users. Users might click different pages for a same query because their page ranks might have changed etc. As a result, the frequencies of queries being issued and pages being clicked also their co-occurrences may change over time. The frequency of queries and page clicks grow very fast when a real-world event approaches and become weaker gradually after the event. The *co-occurrence* of a query-page pair in a given window period is the number of times the pair appear together in the same row of table 1.1 in that window period. The dynamics of co-occurrences can sense the arrival and pass over of the events. For instance, figure 1.1 shows how the frequency (y-axis) of the query "Easter Eggs" changes in six weeks (x-axis) window period. Also the co-occurrence of the query page pair ("Easter Eggs", www.eastereggs.com) is shown in figure 1.2.

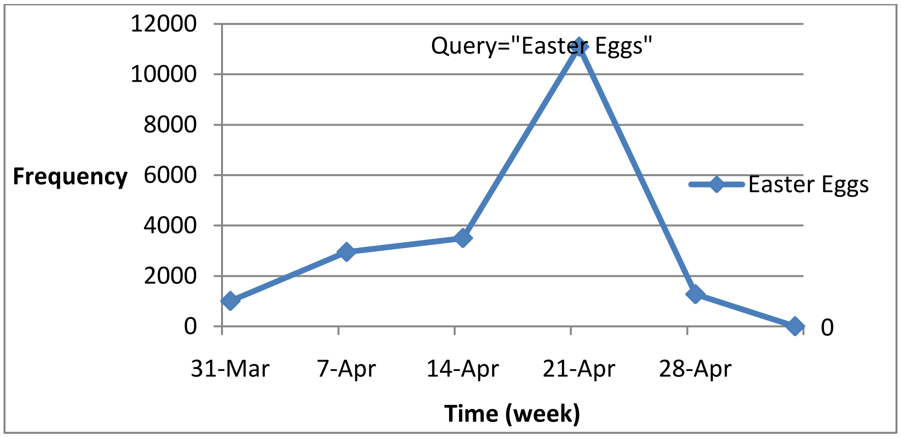


Figure 1.1 Frequency of query "Easter Eggs"

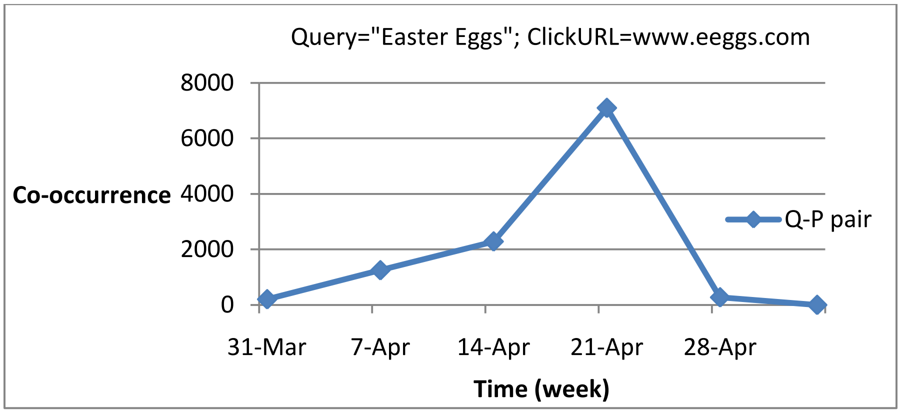


Figure 1.2: Co-occurrence of query and page

The frequency and co-occurrence increased gradually from last week of March to the third week of April (Easter was on April 16th) and then decreased at a faster rate. When a new event occurs, the number of related queries being issued and the number of related web pages being visited may increase drastically. At the same time, the co-occurrences are surprisingly strong. In our data analysis, it is observed that evolutionary patterns for related queries are similar.

1.4.2. Query Space. The work done by Greg [17] et al. gave an inside out perspective about query space, query sessions, user behavior and content space. Interesting facts were revealed: about 28% of all queries are reformulations of previous query. An average query is reformulated 2.6 times. Users formulate and reformulate a series of queries in pursuit of a single overall task; each time refining the query to obtain better pages that meet their information needs better. The possibilities of user actions in query formulation/ reformulation and click-through are: new query, add/remove word(s) to query, change word(s) in query, more results for same query, return to a previous query etc. The notation for corresponding actions is shown in figure 1.3.

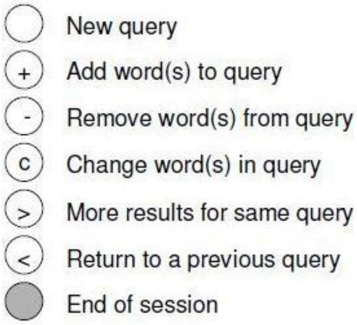
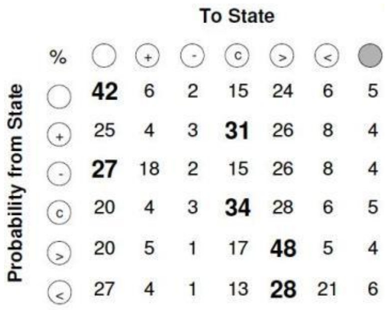

New query
+
Add word(s) to query
-
Remove word(s) from query
c
Change word(s) in query
>
More results for same query
<
Return to a previous query
End of session

Figure 1.3: Notation symbols

For example in the table 1.2, the user is looking for "John west salmon commercial", a famous commercial ad in 2006. The user started with the query "John west ad". Then changed the words in the query and re-queried as "John west salmon" and so forth. Finally, the user ended up the query "John west salmon commercial". Observe the timeline, the user spent 14:59 minutes querying, re-querying and clicking-through the result sets of different queries to get the information the user is looking for. The probability of moving from one state to another is as shown in figure 1.4.

Table 1.2: User behavior on timeline

Timeline(mm:ss)	Action	Query
00:00	New query	John west ad
02:55	©	John west salmon
04:23	©	Latest salmon ad
07:49	+	John west salmon bear ad
09:33	©	Salmon bear fight
14:59	+	John west salmon commercial



To State

Probability from State

%	○	+	-	c	>	<	●
○	**42**	6	2	15	24	6	5
+	25	4	3	**31**	26	8	4
-	**27**	18	2	15	26	8	4
c	20	4	3	**34**	28	6	5
>	20	5	1	17	**48**	5	4
<	27	4	1	13	**28**	21	6

Figure 1.4: State change probability matrix

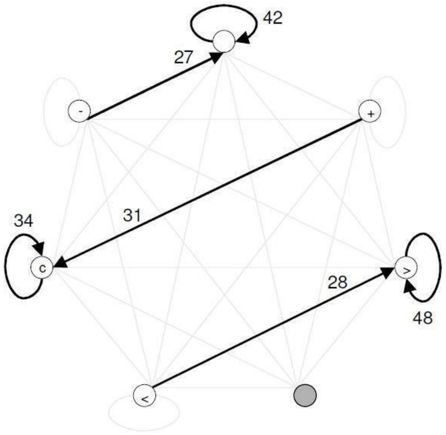


Figure 1.5: State change state-diagram with probabilities

It is reasonable to believe that highest probability 48% is to move for more results, clicking through the pages looking for more information. New queries 42% are always possible with change in information needs, dynamic content of the web and human behavior. The possibility to change the query keywords and re-framing the query is also high 31%. The state diagram for the same is shown in figure 1.5.

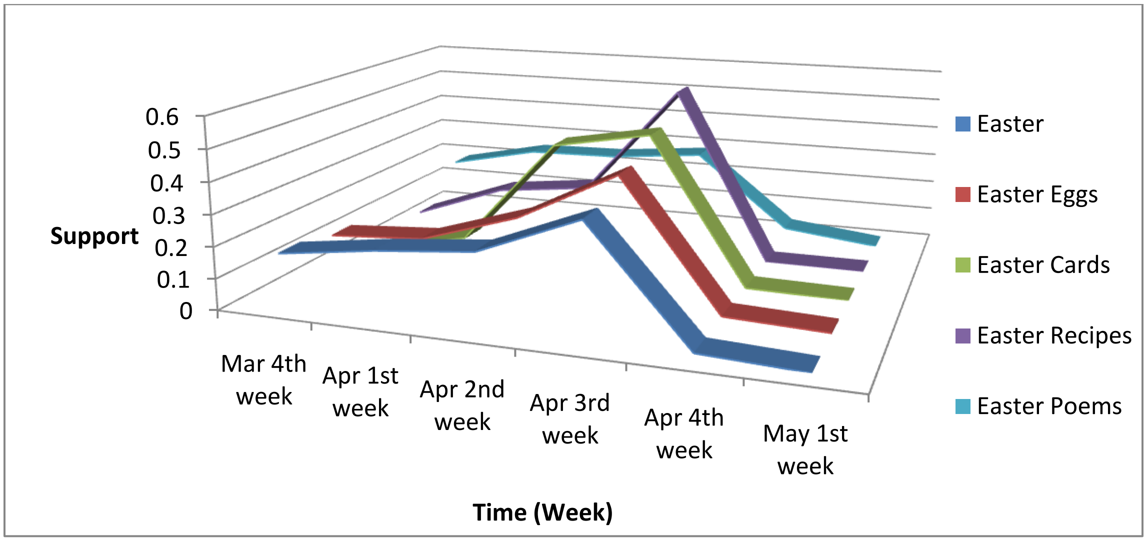


Figure 1.6: Demonstration of query-page pair dynamics for “Easter” over six week period

1.4.3. Via Query Clustering. The overall objective of event detection is achieved via query clustering. Event detection process ends with clusters of query-page pairs that are semantically and temporally related, corresponding to one or more events. Our approach begins with queries because the number of queries the search engine receives (number of ways in which real-times queries are framed) are far less than the size of the web i.e. Q<<P. By this obvious fact, the intuition is clustering can be done efficiently if the process begins with Q. Also query keywords as a summary, give insight about the events. Queries can be formulated in different ways in different contexts, although they all mean the same and correspond to the same event. For example, figure 1.6 shows the support of query-pairs {“Easter”, www.happy-easter.com}, {“Easter Eggs”, www.eeggs.com}, {“Easter Cards”, www.easter-cards.com}, {“Easter Recipes”, www.easter-recipes.com} and {“Easter Poems”, www.poemsforfree.com}. All the 5

query- page pairs have similar evolutionary pattern in the window period and correspond to the same event “Easter” on April 16, 2006.

Similar queries from query sessions tend to be closer in query space. As one can observe, the support increased gradually up to the 3rd week of April and then decreased gradually. By early detection of this kind of query clusters, event detection can be done efficiently. Lot of research has been done in the area of query clustering so by incorporating this work into the event detection framework, the event detection techniques can evolve as the query clustering techniques evolve.

1.4.4. Query Sessions. In this work, click-through data is considered as collaborative query sessions rather than collection of individual entries of query-page pairs as considered in [3]. A query session captures a series of user interactions with the search engine. For example, the first two entries in the table 1.1 will be considered as a query session because they indicate that after issuing the query ”Easter”, the user 7 issued the query “Easter Eggs”. For entries of a query session are temporally close to each other, the timestamp of the first entry is taken as the occurring time of the query session for simplicity. The advantage of this approach is: in most of the meaningful sessions, users issue a series of related queries and click through the web pages of the result set. Thus, instead of clustering these query-page pairs afterwards to discover events, the queries can be grouped into a query session. Usually the queries from same session are semantically and temporally related to one another. These meaningful query sessions, as initial clusters can correspond to real world events. User intensions are better understood by considering the click-through data as query sessions. Also, search engine click-through data is massive and the graphs generated from the click-through data are overwhelmingly large. By considering click-through data as collaborative query sessions, the complexity of the problem can be substantially reduced.

1.4.5. Data Pruning. As observed, not every entry in the click-through data corresponds to some real-world event. Navigational queries account for 21% of the total query frequency [17]. So pruning irrelevant data can prepare a better ground for the approach.

For example, just in a sample of data, the co-occurrences of query and page clicks of popular portal pages are found and shown in table 1.3. The co-occurrences are high but they really do not correspond to any real-world event. So in the data cleaning, preparation and transformation phases of the web data mining, filtering methods are incorporated to process the data. This step significantly improved the quality of the results.

Table 1.3: Frequent query-page pairs of popular portals

Query	Click URL	Co-occurrence
Google	http://www.google.com	14236
Yahoo	http://www.yahoo.com	181820
AOL	http://www.aol.com	4774
MySpace	http://www.myspace.com	17104
Ask.com	http://www.ask.com	2213

Similarities based on query contents and query sessions represent two different points of view. The two criteria have their own advantages and shortcomings. In general, content-based criterion tends to cluster queries with the same or similar terms. Session-based criterion tends to cluster queries related to the same or similar topics. So our motivation is to take combined measures to cluster such similar queries with similar evolutionary patterns corresponding to real world events.

2. RELATED WORK

In this section, a review of significant works in the literature on event detection, query clustering, clustering techniques and association rules is presented.

2.1. EVENT DETECTION

The beginning of event detection originates from the initial works done on (TDT) Topic Detection and Tracking [11] to automatically detect topically related stories within a stream of news media. It consists of three major issues: segmenting the text corpus into events, tracking the development of the detected events, and detecting new events. The objective of the work done on retrospective and on-line detection [12] is to detect stories based on two tasks: retrospective detection and online detection. The retrospective detection aims to discover previously unidentified events in accumulated collection while the on-line detection tries to identify the on-set of news events from live news feeds in real-time. This work belongs to the category retrospective detection. Attempt for bursty event detection was done by Fungs et al. [13] from chronologically ordered documents as text streams. A parameter-free probabilistic approach called feature-pivot clustering was proposed to fully utilize the time information to determine set of bursty features in different time windows.

The work done by Zhao et al. [16] introduced the dynamic behaviors idea to cluster web access sequences (WASs), based on their evolutionary patterns of support counts. The intuition is that often WASs are event/task- driven and partitioning WASs into clusters result in grouping of similar/closer WASs. Later their work in [3] laid a foundation for visitor-centric approach to detect events by using click-through data. The query-page relationship is represented as a bipartite graph, which is later summarized as the vector-based graph. The dual graph of vector-based graph is deduced on which, a two-phased graph cut algorithm is used to partition the dual graph based on (i) semantic-based similarity and (ii) evolution pattern-based similarity to generate query-page pairs that are related to events.

Later, a novel approach was introduced by Chen et al. [4] by transforming the click-through data to the 2D polar space by considering the semantic and temporal dimensions of queries. Then perform a robust subspace estimation to detect subspaces such that each subspace corresponds to queries of similar semantics. Uninteresting subspaces are pruned which do not contain queries corresponding to real events by simultaneously considering the respective distribution of queries along the semantic dimension and the temporal dimension in each subspace. Finally, a non-parametric clustering technique is used to detect events from interesting subspaces.

2.2. QUERY CLUSTERING

Significant work has been done on the topic Query Clustering previously by Wen et al. [7] aiming at grouping users' semantically related, not syntactically related, queries in a query repository. Their approach was based on the two principles: (1) if users clicked on the same documents for different queries, then the queries are similar (2) if a set of documents are often selected for a set of queries, then the terms in these documents are related to the terms of the queries to some extent. In the effort of extracting semantic relations from query logs, Baeza-Yates et al. [8] proposed a model to project queries in a vector space and deduced some interesting properties in large graphs. According to which, non-binary weights are assigned to index terms. The weights are used to calculate the degree of similarity to consider documents that match the queries. Therefore, the resulted ranking is more precise than the Boolean model (in which requests are represented as Boolean expressions carrying precise meaning).The term-weighting scheme improved the retrieval performance.

2.3. CLUSTERING

Clustering is a division of data into groups of similar objects [18]. Each group, called cluster consists of objects that are similar among themselves and dissimilar to objects of other groups. Certain fine details will be lost on representing data by fewer clusters necessarily but simplification is gained. Clustering represents many data objects by few clusters, and hence, it models data by its clusters.

2.3.1 Notation.

To clarify the prolific terminology, consider a dataset X consisting of data points (or synonymously, objects, instances, cases, patterns, tuples, transactions) $x_i = (x_{i1} \dots x_{in}) \in A$ in attribute space A, where $i = 1 \dots N$, and each component

$x_i \in A_i$ is a numerical or nominal categorical attribute (or synonymously, feature, variable, dimension, component, field). Note that in this work, data points are tuples of transactions from query session and attributes are fields in the click-through data. The simplest attribute space subset is a direct Cartesian product of sub ranges called a segment (also cube, cell, and region). A unit is an elementary segment whose sub-ranges consist of a single category value, or of a small numerical bin. Describing the numbers of data points per every unit represents an extreme case of clustering. This is a very expensive representation, and not at all a very revealing one with massive data sets like the one used in this work.

The objective of clustering is to assign points to a finite system of *k* subsets, clusters. Usually clusters do not intersect but in this work this assumption is surpassed. Because a query can belong to multiple clusters (can be related to one or more events) and the page contents are highly dynamic. The union of all the clusters is the full dataset with possible exceptions of outliers i.e. $X = C_1 \cup C_2 \cup \ldots. C_k \cup C_{outliers}$

2.3.2. Clustering Algorithms. Categorization of clustering algorithms is neither straightforward, nor canonical. The categories of clustering algorithms overlap but traditionally clustering techniques are broadly categorized as *hierarchical* and *partitioning*. There are several challenges for a clustering algorithm.

It should:

- Handle different types of attributes
- Be scalable on large datasets
- Have reasonable Time Complexity
- Be parameter-free
- Be independent of data order
- Find clusters of irregular shape
- Handle outliers
- Work with high dimensional data
- Produce interpretable results

Hierarchical algorithms build clusters gradually and on the other hand, partitioning algorithms learn clusters directly. In doing so, they either try to discover clusters by iteratively relocating points between subsets, or try to identify clusters as areas highly populated with data.

2.3.2.1. Partitioning Clustering. Partitioning clustering algorithms divide data into several subsets. Relocation schemes iteratively reassign points among the clusters. Unlike hierarchical methods, in partitioning clustering the intermediate clusters are revisited and improved. K-means [Hartigan & Wong 1979] and DBSCAN [10] are the widely used clustering techniques in this category. K-means requires initial parameter k to start. DBSCAN [10] meets all the challenges and our algorithm is inspired by this work. DBSCAN is density-based whereas our algorithm is distance-based.

2.3.2.1.1. DBSCAN Algorithm

Definition 1: (Eps-neighborhood of a point p), denoted by $N_{Eps}(p)$, is defined as $N_{Eps}(p) = \{q \in D \mid dist(p, q) \leq Eps\}$ i.e. for each point in a cluster there should be atleast a minimum number (MinPts) of points in Eps-neighborhood of that point.

The definition does not suffice for border points of the cluster but works for the core points.

Definition 2: (Directly density-reachable) A point p is directly density-reachable from a point q wrt. Eps and MinPts if 1) $p \in N_{Eps}(q)$ and 2) $|N_{Eps}(q)| \geq$ MinPts (core point condition)

Evidently, this is not symmetric if one core point and one border point are involved. Both are shown below in figure 2.1.

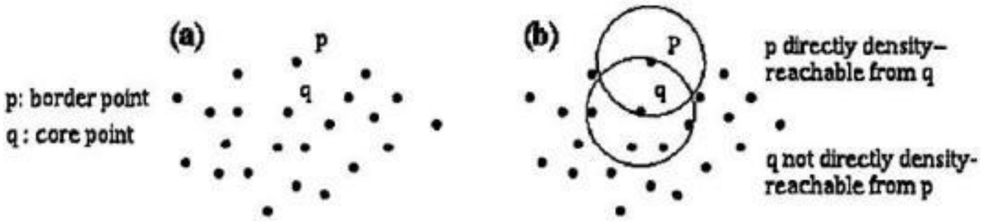

(a)
p
q
p: border point
q : core point
(b)
P
q
p directly density–reachable from q
q not directly density-reachable from p

Figure 2.1: Core points and border points

Definition 3: (density-reachable) A point p is density reachable from a point q wrt. Eps and MinPts if there is a chain of points p_1... p_n, p_1 = q, p_n = p such that p_{i+1} is directly density-reachable from p_i.

Two border points of the same cluster C are possibly not density reachable from each other because the core point condition might not hold for both of them. However, there must be a core point in C from which both border points of C are density-reachable.

Definition 4: (density-connected) A point p is density-connected to a point q wrt. Eps and MinPts if there is a point o such that both, p and q are density-reachable from o wrt. Eps and MinPts. Both are shown below in figure 2.2.

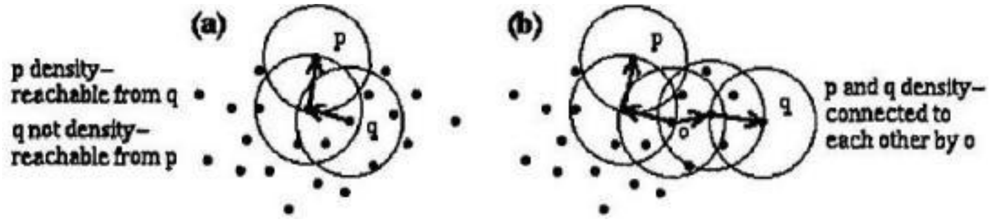

(a)
p density-reachable from q
q not density-reachable from p
p
q
(b)
p
o
q
p and q density-connected to each other by o

Figure 2.2: Density-reachability and density-connectivity

Intuitively, a cluster is defined to be a set of density connected points which is maximal wrt. density-reachability. Noise will be defined relative to a given set of clusters. Noise is simply the set of points in D not belonging to any of its clusters.

Definition 5: (cluster) let D be a database of points. A cluster C wrt. Eps and MinPts is a non-empty subset of D satisfying the following conditions:

1) ∀ p, q: if p ∈ C and q is density-reachable from p wrt. Eps and MinPts, then q ∈ C.

2) ∀ p, q ∈ C: p is density-connected to q wrt. Eps and MinPts.

2.4 ASSOCIATION RULES

Association rules are widely used in several areas of data mining. Work done by Fonseca et al [10] is an attempt to automatically generate suggestions of related queries submitted to web search engines. The method extracts information from the log of past submitted queries to search engines using algorithms for mining association rules.

Notation

Let I = $\{I_1, I_2 \ldots I_m\}$ be a set of queries from log files and T is the set of user sessions t. For each t there is a binary vector $t[k]$ such that $t[k]$ =1 if session t searched for query I_k, and $t[k]$ =0 otherwise.

By an association rule it means the implication $X \Rightarrow Y$ where $X \subset I$, $Y \subset I$ and $X \cap Y = \emptyset$. The rule $X \Rightarrow Y$ has a confidence factor of c% if c% of the transactions in T that contains X also contains Y. Classical notation $X \Rightarrow Y \mid c$ is used to specify that the rule $X \Rightarrow Y$ has a confidence factor of c. The rule $X \Rightarrow Y$ has a support factor of s% if s% of the transactions in T contains $X \cap Y$. The problem of mining association rules is to generate all association rules that have a support greater than a specified minimum support (also called *minsup*).

PAPER

ECO: Event Detection from Click-through Data via Query Clustering

Prabhu K. Angajala[1] and Sanjay K. Madria

Department of Computer Science
Missouri University of Science & Technology Rolla, MO, USA
Email: {panr5@mail.mst.edu, madrias@mail.mst.edu}

Abstract. In this paper, we propose an algorithm to detect real world events from the click-through data. Our approach differs from the existing work as we: (i) consider the click-through data as collaborative query sessions instead of mere web logs (ii) try to integrate the semantics, structure, and content of queries and pages (iii) aim to achieve the overall objective via Query Clustering. The problem of event detection is transformed into query clustering by generating clusters - hybrid cover graphs; each hybrid cover graph corresponds to a real-world event. The evolutionary pattern for the co-occurrence of query-page pairs in a hybrid cover graph is imposed for the quality purpose over a moving window period. Finally, we experimentally evaluate our proposed approach using commercial search engine's data collected over 3 months with about 20 million web queries and page clicks from 650000 users. Our results outperform the most recent work in this domain in terms of number of events detected, F-measures, entropy, recall etc.

1. INTRODUCTION

The approximate size of today's indexed World Wide Web is at least 45.93 billion pages as per existing estimation [1] and is a rich collection of all the real world objects. Web is a great source of knowledge to be mined to learn about topics, stories, events etc. Event detection is becoming increasingly popular because of its applicability in several diversified areas. Therefore the interpretation of "event" definition is context-dependent. An event can be associated with a sensor at a door post reporting how many people/cars have entered a building/freeway, web access log, security log, object trajectory in video surveillance and business activity monitoring in Business Intelligence etc. In our perspective and from the viewpoint of the Web, an event can be understood as some real-world activity. It stirs large scale querying and browsing activity that is of more interest to users over a sizable window period, which is unusual relative to normal patterns of querying and browsing behavior. Web is the collaborative work of many people, a few publishing, and all of them querying and retrieving the information. Search engines record these activities in the web logs; called the click-through data and reflects the query and clicks activities of users. Click-through data is more or less in the format shown in the table 1 below:

Table 1: Sample click-through data

AnonID	Query	Query Time	Item Rank	Click URL
7	Easter	2006-03-01 23:19:52	1	http://www.happy-easter.com
7	Easter eggs	2006-03-01 23:19:58	1	http://www.eeggs.com

To briefly explain the fields, we begin with AnonID, which is the anonymous User ID from whom the search engine received the request, followed by the query issued by the user, the time at which the search engine received the request, the rank of the page item clicked, the page clicked in response to the result-set returned by the search engine. Note that the click-through data format varies slightly from one search engine to the other.

The three web data types that are identified in previous [2] efforts are: content (text and multimedia), structure (links that form a graph) and web usage (transactions from the web log). Web data mining encompasses broad range of research topics like improving page ranking, better indexing, query clustering, query similarity, query suggestions, extracting semantic relations and event detection etc. All these areas are inter-related and many use the click-through data as the start up source. The seamless flow of advancement in developing better approaches in individual areas can be pipelined to improve existing techniques in the inter-related fields. So our effort in this paper is to integrate the three web data types and achieve the overall objective via query clustering. In our attempt to exploit all possible sources to detect events, we believe that our approach will do better.

1.1 MOTIVATION

The dynamics in click-through data was previously identified in [3]. The frequency of queries and page clicks grow very fast when the real-world event approaches and become weaker gradually after the event. The co-occurrence of a query-page pair in a given window period is the number of times the pair appear together in the same row of table 1 in that window period. The dynamics of co-occurrences can sense the arrival and pass over of the events. The work done by Greg [17] et al. gave an inside out perspective about query space, query sessions, user behavior and content space. Interesting facts were revealed: about 28% of all queries are reformulations of previous query. An average query is reformulated 2.6 times. Users formulate and reformulate a series of queries in pursuit of a single overall task. The possibilities are: new query, add/remove word(s) to query, change word(s) in query, more results for same query, return to a previous query etc. So our motivation is to cluster such similar queries with similar evolutionary pattern corresponding to a real world event.

Our work differs from the existing work in one or more of the following ways:

(1) We consider the click-through data as collaborative query sessions rather than collection of individual entries of query-page pairs considered in [3, 4]. A query session captures a series of user interactions with the search engine. The advantage of this approach is in most of the meaningful sessions, users issue a series of related queries and click through the web pages of the result set. They are semantically and temporally related to one another. These meaningful query sessions, as initial clusters can correspond to real world events. User intensions are better understood by considering the click-through data as query sessions. Search engine click-through data is massive and the graphs generated from the click-through data are overwhelmingly large. By considering click-through data as collaborative query sessions, we can substantially reduce the complexity of the problem.

(2) As we see, not every entry in the click-through data corresponds to some real-world event. Navigational queries account for 21% of the total query frequency [17]. So pruning irrelevant data can prepare a better ground for the approach. For example: just in a sample of data, we found the frequency of queries and page clicks of popular portal pages. The frequencies are shown in table 2.

The frequencies are high but they really do not correspond to any real-world event. So in the data cleaning, preparation and transformation phases of the web data mining, we incorporate filtering methods to process the data. This step significantly improved the quality of the results.

Table 2: Frequent query-page pairs of popular portals

Query	Click URL	Frequency
Google	http://www.google.com	14236
Yahoo	http://www.yahoo.com	181820
Aol	http://www.aol.com	4774
Myspace	http://www.myspace.com	17104
Ask.com	http://www.ask.com	2213

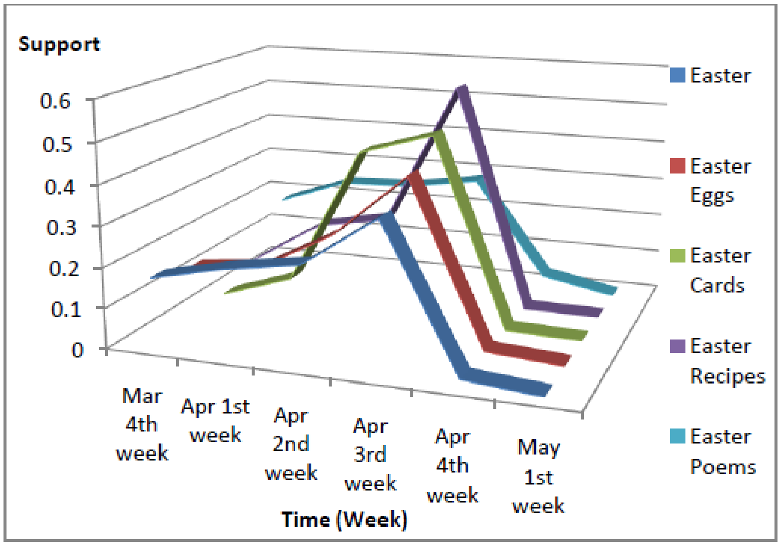


Figure 1: Demonstration of query-page pair dynamics for “Easter” over six week period

(3) We achieve the overall objective of event detection via query clustering. Event detection process ends with clusters of query-page pairs that are semantically and temporally related and corresponding to one or more events. We begin this process with queries because the number of queries the search engine receives (number of ways in which real-times queries are framed) are far less than the size of the web i.e. Q<<P. By

this obvious fact, we believe that clustering can be done efficiently if we begin the process with Q. Also query keywords as a summary, give insight about the events. Queries can be formulated in different ways in different contexts, although they all mean the same and correspond to the same event. For example, figure 1 shows the support of query-pairs {"Easter", www.happy-easter.com}, {"Easter Egg", www.eeggs.com}, {"Easter Cards", www.easter-cards.com}, {"Easter Recipes", www.easter-recipes.com} and {"Easter Poems", www.poemsforfree.com}.

All the four query-page pairs have similar evolutionary pattern in the window period and correspond to the same event "Easter" on Aril 16, 2006. As one can observe, the support increased gradually up to the 3rd week of April and then decreased gradually. By early detection of this kind of query clusters, event detection can be done efficiently. Lot of research has been done in the area of query clustering so by incorporating this work into the event detection framework, the event detection techniques can evolve as the query clustering techniques evolve.

2. RELATED WORK

In this section, we review the significant works in the literature on event detection and query clustering. The beginning of event detection originates from the initial works done on (TDT) Topic Detection and Tracking [11] to automatically detect topically related stories within a stream of news media. The objective of the work done on retrospective and on-line detection [12] is to detect stories based on two tasks: retrospective detection and online detection. The retrospective detection aims to discover previously unidentified events in accumulated collection while the on-line detection tries to identify the on-set of news events from live news feeds in real-time. Attempt for bursty event detection was done by Fungs et al. [13] from chronologically ordered documents as text streams. They proposed a parameter-free probabilistic approach called feature-pivot clustering to fully utilize the time information to determine set of bursty features in different time windows. The work done by Zhao et al. [16] introduced the dynamic behaviors idea to cluster web access sequences (WASs), based on their evolutionary patterns of support counts. The intuition is that often WASs are event/task- driven and partitioning WASs into clusters result in grouping of similar/closer WASs. Later their work in [3] laid a foundation for visitor-centric approach to detect events by using click-through data. The query-page relationship is represented as the vector-based graph. On the dual graph of vector-based graph, a two-phased graph cut algorithm is used to partition the dual graph based on (i) semantic-based similarity and (ii) evolution pattern-based similarity to generate query-page pairs that are related to events. Later, a novel approach was introduced by Chen et al. [4] by transforming the click-through data to the 2D polar space by considering the semantic and temporal dimensions of the queries. Then perform a subspace estimation to detect subspaces such that each subspace corresponds to queries of similar semantics.

Significant work has been done on the topic Query Clustering previously by Wen et al. [7] on the Encarta encyclopedia. Their approach was based on the two principles: (1) if users clicked on the same documents for different queries, then the queries are similar. (2) If a set of documents are often selected for a set of queries, then the terms in these documents are related to the terms of the queries to some extent. In the effort of extracting semantic relations from query logs, Baeza-Yates et al. [8] proposed a model o project queries in a vector space and deduced some interesting properties in large graphs.

3. EVENT DETECTION FRAMEWORK

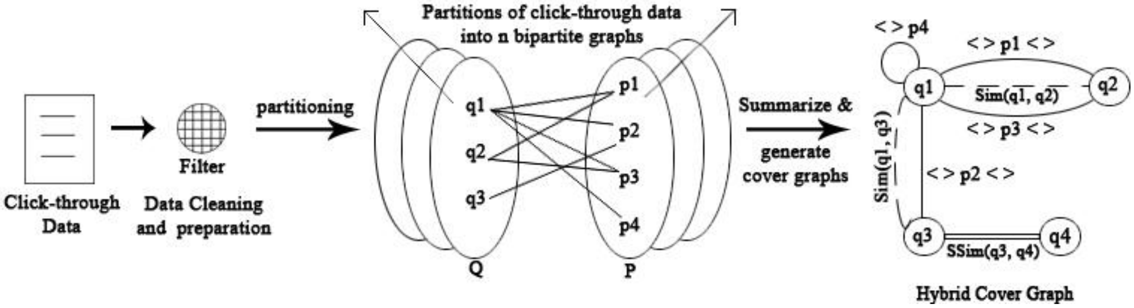


Figure 2: Event detection framework overview

The overview of our proposed event-detection framework is shown in figure 2 and is briefly explained in this section. Given the click-through data, we perform the data cleaning, preprocessing and transformation tasks to refine the data. As shown in table 2, some portion of the click-through data does not correspond to real-world events. Filtering this noise is a better step to prepare ground for further process. In order to analyze the dynamics of increase and decrease of co-occurrences of query-page pairs, we partition the click-through data into a sequence of collections based on user-defined time granularity. Time granularities can be like a day, week, month etc. Different time granularities are required to detect events over moving window sizes. Each collection can be represented by a bipartite graph. We summarize the co-occurrences of query-page pairs from all the collections into a summarized bipartite graph. Then we transform the problem of event detection into query clustering while capturing the relationship among queries and pages. For this purpose, we use the hybrid cover graph and employ a distance-based function that includes the semantics of the query and pages to define the criteria for clustering. The summarized support from bipartite graph is used to emphasize the dynamics of the queries and pages in the clusters to detect the event.

4. DATA REPRESENTATION

Click-through data is collected as raw web logs from the search engines. As mentioned earlier, we consider the click-through data as collaborative query sessions instead of individual query-page records. The reason for the same is explained earlier in Section 1.1. A query session is essentially wrapped by time boundaries, the beginning and the end time. We segment user's streams into sessions based on anonymous ID. Another widely used technique [14] is based on the idea: two consecutive actions (either query or click) are segmented into two sessions if the time interval between them exceeds 30 minutes.

Definition1: (Query session) A query session S= (Q, P), where Q={q_1, q_2…q_m} is a bag of queries issued to the search engine and P = {p_1, p_2….p_n} is the set of corresponding pages clicked by the user from the search result set.

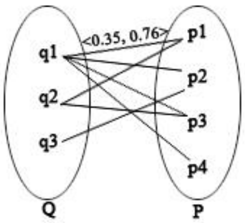


Figure 3: Summarized bipartite graph

A bipartite graph, G = (V, E) where nodes in V represent queries and web pages and edges in E represent the strengths of the query-page pairs. Bipartite graphs are widely used in the web data mining domain [5, 6] to represent the relationship between queries and pages. An edge between a query and page is formed if the page is clicked in response to the query. Bipartite graphs can be visualized as mapping between the query set (Q) and the page set (P) as shown in figure 3. We do like [3] to partition the click-through data C into sequence of collections <C_1, C_2... C_n> based on user-defined time granularity like hour, day, week and month etc.

Definition2: (**Strength**) of a query-page pair $P_{s,t} = (q_s, p_t)$ in collection C_i is $S_i(P_{s,t})$ = $\frac{|Ps,t(Ci)|}{\sum_{i=1}^{n} |Ps,t(Ci)|}$ where $1 \leq i \leq n$ and s, t is a query-page pair. Strength is the ratio of the co-occurrence of the query-page pair in collection C_i to C. The ratio actually keeps the value ≤ 1 and is easy to process than showing actual high co-occurrence values. Note that in figure 3 the strength of (q1, p1) is summarized as <0.35, 0.76> for two collections. Noisy query-page pairs that appear sporadically and potentially not related to any event have substantially low strengths.

In order to cluster queries with consideration for pages clicked, we need efficient data structure and representation. Several graph theories are in existence for this purpose. Baeza-Yates et al. [15] identified several types of query graphs. In all cases, the queries are nodes and an edge is drawn between two nodes if: (i) the queries contain the same word(s) – word graph (ii) the queries belong to the same session – session graph (iii) users clicked on the same URLs from the result sets – cover graph. Word graph is hard to use because users formulate queries in different ways but word graph is essential to capture the query semantics. Not all the queries from a session correspond to some event so session graph is not the choice of option for us. Both word and session graphs fail to capture the semantics of pages clicked. Cover graph can be efficient because for two queries with a commonly clicked page, the edge is represented only once. Reducing the complexity of the graph structure with emphasis on page clicks can simplify the problem and helps for easy representation. We extend the notion of cover graph to hybrid cover graph, which is explained shortly. The notion of commonly clicked documents [15] is as follows:

Definition3: Query Instance is a query (set of words or sentences) plus zero or more clicks related to that query. Formally: QI = (q, u*) where q = {words or phrase}, q being the query, u a clicked URL and denoted by QI_q and $QI_{c(u)}$ denotes the set of its clicked URLs.

Definition4: URL Cover is set of all URLs clicked for a query. So for a query p, $UC_p = \bigcup_{QIq=p} \mathrm{QIc(u)}$

The nodes in the hybrid cover graph are queries from the click-through data. Three types of edges are possible between any two nodes: 1. Cover edge (represented by normal line) is drawn if a page is clicked in common to both the queries 2. Similarity edge (represented by dotted line) represents the similarity of the two queries, page content and user feedback. 3. Session similarity edge (represented by double line ==) is drawn if two queries are related to each other in inference from most of the sessions. The criterion for similarity over the similarity edge is based on distance function and session inferences.

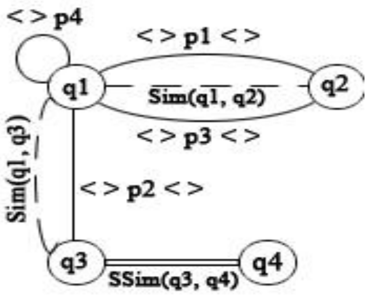


Figure 4: Hybrid Cover Graph

The hybrid cover graph as shown in figure 4 is formed by incorporating the features of word and session graphs into the cover graph. Sim(q1, q3) is the similarity edge that represents the similarity between the queries q1 and q3, which have common URLs clicked in response to them. The vectors on each side of the page p2, represented as <>p2<> indicate the summarized support of p2 with the corresponding query nodes. SSmin(q3, q4) is the session similarity between q3 and q4, which will be explained in Section 5.

5. DISTANCE FUNCTION

Similarity between two queries i.e. nodes in a graph is based on our approach to integrate the semantics, structure, and content of queries and pages. Our distance criterion is based on work done by Wen et al. [7] to cluster queries.

5.1. Similarity based on Query Contents

Although low length queries are harder to understand, queries are better understood by considering them as keywords, words in their order and phrases. We perform the stemming, stop words elimination, phrase recognition and synonym labeling while adding a query to the query semantics dictionary of a cluster. Let p, q are two queries.

Similarity based on Keywords or Phrases:

$$\mathrm{Sim}_{\mathrm{keyword}}\ (p, q) = \mathrm{KN}\ (p, q)/\mathrm{Max}\ (\mathrm{kn}\ (p), \mathrm{kn}\ (q))$$

KN (p, q) = number of common keywords in the queries p and q.

kn (p) = number of keywords in p.

Similarity based on String Matching:

The comparison is the string-matching problem and can be computed by edit distance i.e. number of edit operations required to unify two strings:

$$\mathrm{Sim}_{\mathrm{edit}}\ (p, q) = 1\text{- } (\mathrm{EditDistancte}(p, q)\ /\ \mathrm{Max}\ (\mathrm{kn}\ (p), \mathrm{kn}\ (q))$$

$$\mathrm{Similarity}_{\mathrm{content}} = \mathrm{Sim}_{\mathrm{keyword}}\ /\ \mathrm{Sim}_{\mathrm{edit}}$$

5.2. Similarity based on Session Feedback

A query can be expressed as a point in high-dimensional space [15], where each dimension corresponds to a unique URL i.e. a query can be given a vectorial representation based on all the different URLs in its cover. If p and q are two queries then $\mathrm{Sim}_{\mathrm{vector}}$ is computed as:

$$\mathrm{Sim}_{\mathrm{vector}} = \frac{\overline{p}}{|p|} . \frac{\overline{q}}{|q|}$$

Session feedbacks from meaningful query sessions can help to relate topically similar URLs. A simple way to take user feedback into consideration is by taking the normalized value to see the similarity in terms of the commonly clicked URLs for the queries.

$$\mathrm{Sim}_{\mathrm{doc}}= \mathrm{RD}\ (p, q)\ /\ \mathrm{Max}\ (|\mathrm{Cover}\ (p)|, \mathrm{Cover}\ (q)|)$$

where RD (p, q) is the number of commonly clicked URLs and |Cover (p)| is the number of URLs clicked for query p.

$$\text{Similarity}_{\text{feedback}} = \text{Sim}_{\text{vector}} * \text{Sim}_{\text{doc}}$$

Content-based measures tend to cluster queries with same or similar terms whereas session feedback-based measures tend to cluster page clicks related to the same or similar topics.

$$\text{Similarity (p, q)} = \alpha\ \text{Similarity}_{\text{content}} + (1- \alpha) * \text{Similarity}_{\text{feedback}}$$

Where α is the weight factor and $\alpha \in [0, 1]$.

$$\text{Distance (p, q)}\ \alpha\ 1 / \text{Similarity (p, q)}$$

Larger the similarity, smaller the distance and the weights for content and session feedback similarities are adjusted to obtain better results. An edge between two queries p and q in the hybrid cover graph is drawn if Distance (p, q) $\leq D_{min}$, where D_{min} is the minimum distance.

Association Rules [9] can be applied to find queries that are asked together in most of the query sessions. In the problem of finding related queries from query set Q, we are interested in associations like $X \Rightarrow Y$, where X, Y are subsets of Q, $X \cap Y = \emptyset$. The rule $X \Rightarrow Y$ should have a support $\geq S_{min}$ and confidence $> C_{min}$, which S_{min} and C_{min} are minimum support and confidence values. Suppose the rule $q1 \Rightarrow q4 \mid S, C$ where $S \geq S_{min}$ and $C \geq C_{min}$ is found then include the rule in the hybrid cover graph.

6. CLUSTERING PROCESS

The overview of clustering process is shown in figure 5. First the query sessions are extracted from the click-through data then we do some data cleaning and preprocessing. Then the query-page pair relationships are represented internally as summarized bipartite graphs. The clustering algorithm computes the similarity functions and based on distance threshold, clusters are formed. The clusters are represented as hybrid cover graphs. Association rules mined are also embedded into the hybrid cover graph. For each query $q \in Q$, find the clusters (among the clusters obtained so far) with which the minimum distance condition is satisfied. Assign q to those clusters. If the minimum distance condition is not satisfied with any of the existing clusters then start a new cluster beginning with q.

For example, as shown in figure 6 when a new query q5 comes in, its content is compared with the semantics of the query dictionary formed from existing queries - q1, q2, q3, q4. Then its page clicks from the summarized bipartite graph are compared with the session feedback library of all the pages - p1, p2, p3, p4 for a given cluster. If the distance D is ≤ D_{min} then the query is added to the cluster, the query semantics are added to the query semantics dictionary and its page clicks are added to the session feedback library. If not the query begins forming a new cluster.

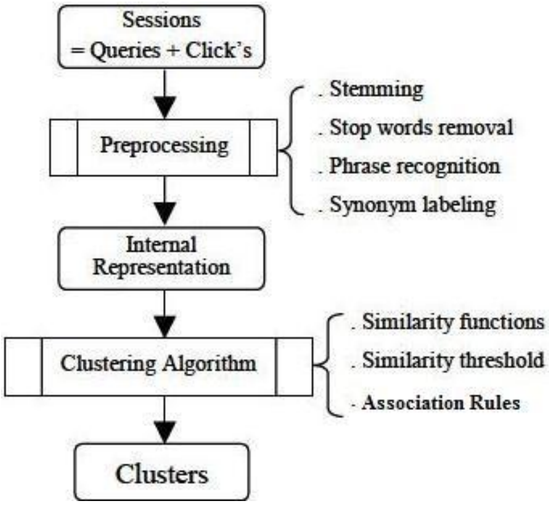


Figure 5: Overview of clustering process

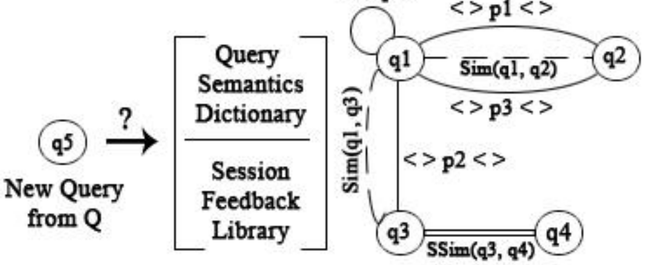
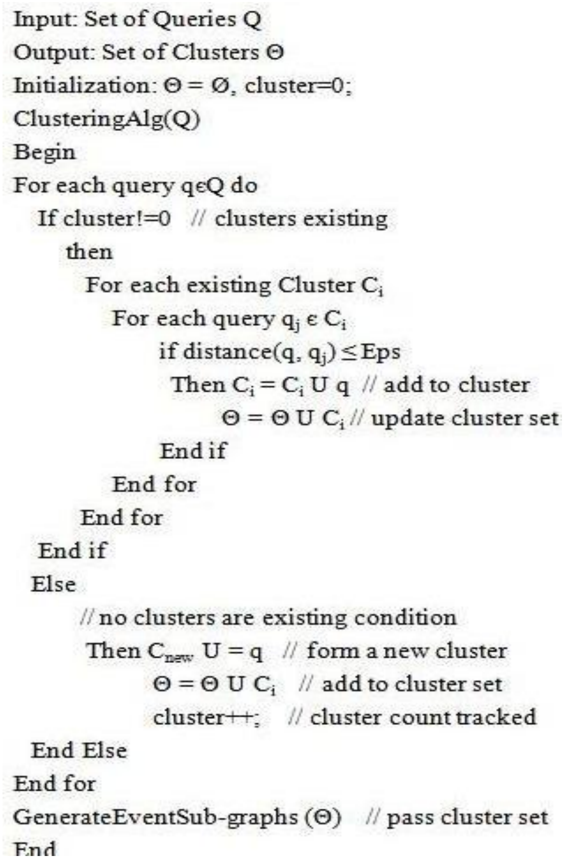


Figure 6: Clustering on Hybrid Cover Graph

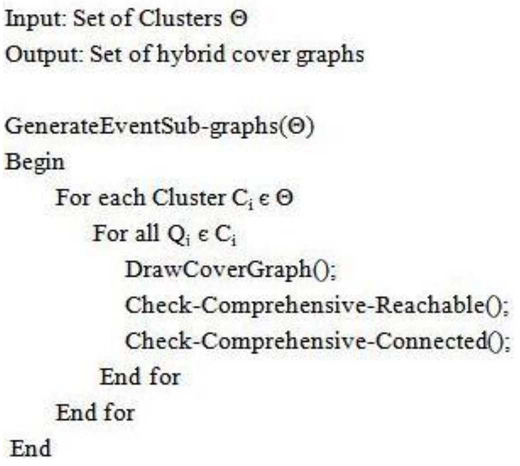
6.1 Event Detection Algorithm

There are several challenges in query clustering technique. It should be able to handle all types of attributes, be scalable on massive datasets, work with high dimensional data, handle outliers, have reasonable time complexity, be independent of data order, and start without initial parameters (for example, the number of clusters). DBSCAN [10] algorithm and its incremental version meet all the required conditions and its average time complexity is O (n*log n). But the distance function in our approach is not density-based but distance-based.



```
Input: Set of Queries Q
Output: Set of Clusters Θ
Initialization: Θ = Ø, cluster=0;
ClusteringAlg(Q)
Begin
For each query q∊Q do
  If cluster!=0   // clusters existing
    then
      For each existing Cluster $C_i$
        For each query $q_j$ ε $C_i$
            if distance(q, $q_j$) ≤ Eps
              Then $C_i$ = $C_i$ U q   // add to cluster
                  Θ = Θ U $C_i$ // update cluster set
            End if
        End for
      End for
  End if
  Else
      // no clusters are existing condition
      Then $C_{new}$ U = q   // form a new cluster
           Θ = Θ U $C_i$   // add to cluster set
           cluster++;   // cluster count tracked
  End Else
End for
GenerateEventSub-graphs (Θ)   // pass cluster set
End
```

Algorithm1: ECO – Clustering Process



```
Input: Set of Clusters Θ
Output: Set of hybrid cover graphs

GenerateEventSub-graphs(Θ)
Begin
    For each Cluster $C_i$ ε Θ
      For all $Q_i$ ε $C_i$
          DrawCoverGraph();
          Check-Comprehensive-Reachable();
          Check-Comprehensive-Connected();
        End for
    End for
End
```

Algorithm2: Event Detection ECO – Hybrid Cover Graph

Our algorithm inspired by the DBSCAN algorithm differs significantly from DBSCAN and requires only one scan of the queries through the click-through data. The criterion for distance function is explained previously in Section 5. The event detection algorithm is presented in two steps. Algorithm1 is for the clustering process and the later is for generating the hybrid cover graphs. The hybrid cover graphs are drawn with respect to the comprehensive-reachable and comprehensive connected conditions of the DBSCAN algorithm for the terminal nodes. The algorithm runs at different time granularities to detect events of different window sizes like day, week and month etc. The summarized support values for the query-page pairs are analyzed using histograms to ensure that the hybrid cover graph has an evolutionary pattern. The higher ranking of the nodes in the hybrid cover graph can be given for the connected dominating set (nodes that essentially connect the graph), nodes with least distance and with higher supports with their corresponding edges. The page rank of the edge can be obtained as the ItemRank from the click-through data. The edges with better ranks can be regarded as high quality web pages clicked in relation to events.

Pruning irrelevant data is very important because the click-through data has millions of queries and pages. We reduced the size of the graphs qualitatively and quantitatively by eliminating: 1. Queries and pages that have low support values. By doing so, some edges and nodes can be removed from the graph. These queries and pages can be seen sporadically in the data. 2. Multi-topical URLs (pages that talk about several topics or a very generic topic) by removing edges of low weight obtained from criteria in section 5. Low weight edges are more likely to represent poor quality semantic relations.

7. WORKING EXAMPLE

In this section we explain the overall process by continuing the example initiated in section 1.1.

Table 3: Co-occurrence of query-page pairs over a 6 week window period

	31-March	7-April	14-April	21-April	28-April	04-May
P1	7000	8700	9900	1510	600	0
P2	9200	10500	16900	2740	1000	200
P3	300	1500	8200	9300	100	0
P4	1000	2900	3500	6900	0	0
P5	9100	8300	8500	9500	1200	0

Table 4: Support of query-page pairs over a 6 week window period

	31-March	7-April	14-April	21-April	28-April	04-May
P1	0.169	0.210	0.239	0.365	0.014	0
P2	0.141	0.161	0.259	0.420	0.015	0
P3	0.015	0.077	0.422	0.479	0.005	0
P4	0.058	0.170	0.205	0.564	0	0
P5	0.181	0.247	0.252	0.282	0.035	0

Figure-1 shows the support of query-pairs P1 {“Easter”, www.happy-easter.com}, P2 {“Easter Egg”, www.eeggs.com}, P3 {“Easter Cards”, www.easter-cards.com}, P4 {“Easter Recipes”, www.easter-recipes.com} and P5 {“Easter Poems”, www.poemsforfree.com}. The co-occurrence, support for the query page pairs for the 6 week window period is shown in tables 3 and 4. As one can see, the evolutionary patterns for the query-page pairs are similar in the given window period.

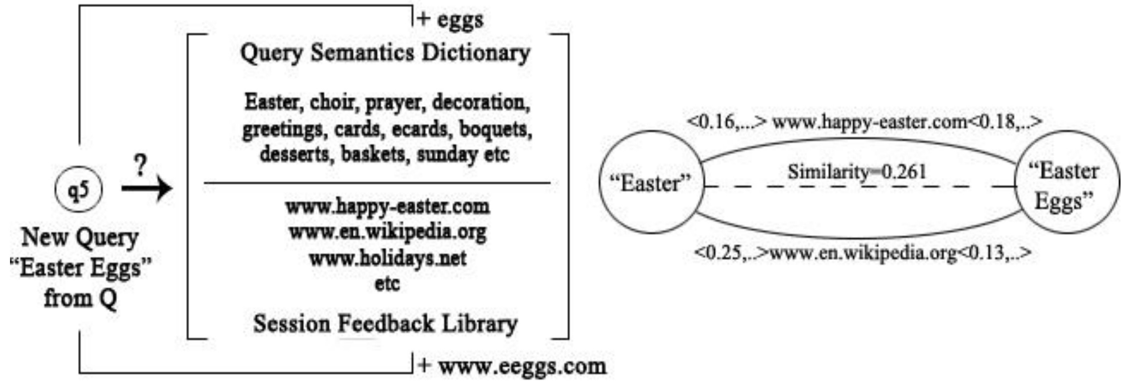

+ eggs
Query Semantics Dictionary
Easter, choir, prayer, decoration, greetings, cards, ecards, boquets, desserts, baskets, sunday etc
www.happy-easter.com
www.en.wikipedia.org
www.holidays.net
etc
Session Feedback Library
+ www.eeggs.com
q5
?
New Query
“Easter Eggs”
from Q
<0.16,...> www.happy-easter.com<0.18,..>
“Easter”
Similarity=0.261
“Easter Eggs”
<0.25,..>www.en.wikipedia.org<0.13,..>

Figure 7: Illustration of “Easter” and “Easter Eggs” clustering

$Sim_{keyword} = 1/2 = 0.5$

$Sim_{edit} = 4$

$Similarity_{content} = Sim_{keyword} / Sim_{edit} = 0.125$

We computed $Sim_{vector} = 1.2$

Sim_{doc}= 177/569=0.311

$Similarity_{feedback}$ = Sim_{vector}* Sim_{doc}= 0.373

Similarity (“Easter”, “Easter Eggs”) = α $Similarity_{content}$ + (1- α) * $Similarity_{feedback}$, where α is the weight factor.

Assume α=0.45. Similarity (“Easter”, “Easter Eggs”) = 0.261

Distance (p, q) α 1 / Similarity (p, q)

Let Distance = 1/0.261=3.83.

Assume D_{min}=3 then the queries “Easter” and “Easter Eggs” should fall into the same cluster. The process is illustrated in figure 7. Note that only the portion of hybrid cover graph with nodes “Easter” and “Easter Eggs” is shown because of the complexity of the graph. All the four query-page pairs are semantically and temporally related and have similar evolutionary patterns in the window period and correspond to the same event “Easter” on Aril 16, 2006. As one can observe, the support increased gradually to 3^{rd} week of April and then decreased gradually. The criterion for distance function is explained in section 5 and the clustering process is explained in section 6.

8. PERFORMANCE STUDY

In this section, we study the performance of our event detection approach. Firstly, we describe the characteristics of the dataset used for our experiments. Then we present the experimental results and compare with some of the existing work.

8.1. Data Set

A real click-through dataset obtained from AOL search engine is used in our experiments. The data is from March 2006 to May 2006, comprised of 500k query sessions, consisting ~20 web million queries and click-through activities from 650k users. As described in [17], each line in the data represents one of two types of activities: (i) a query that was not followed by the use clicking on a result item. (ii) a click through on an item in the result list returned from a query. In the later case, the pages appear as successive entries in the data. In our approach, as a query session is obtained as successive pages corresponding to the same query from the same user. The timestamp of the first page click in a query session is taken as the start time of the session.

8.2. Result Analysis

Our approach can also detect pre and post period events, where the current period is referred to March through May, 2006. As discussed in Section 1.2 the co-occurrence of query page pairs corresponding to an event do not stop abruptly right after the event but slow down at a faster rate. So pre and post period events can be detected by analyzing such kind of behavior. For example pre-period event “Winter Olympics Torino, Italy” happened during February 10 through 26. We observed significant interest decreasing at a faster rate in regard of this in early March data. Post-period event “FIFA World Cup, Germany” during June 9 through July 9 is detected with increasing interest at the end of the May data.

Our algorithm can detect events of different time granularity like day, week and month. For an event, the traffic spreads around the event juncture like few days, weeks, and months in time granularity before and after the event. Day events like the death of Jack Wild, a famous British actor on March 1, the St. Patrick’s Day on March 17 etc are detected. Week events like the Philadelphia flower show, (a big indoor flower show) during the week March 5 through 12, the Fleet week (public can see USA Navy and Coast guard ships) during the week May 24 through 30 etc. Monthly events span across bigger time frames and appeared throughout the data. The famous American Idol 5 episode appeared March 1 through May 24 (finale), the Internal Revenue Service (IRS) tax filing appeared March 1 through 31.

Note that some of the events are regular and previously known like the St. Patrick’s Day; Good Friday etc recur every year. Some are previously unknown; like Simon Lindley, an Organist received the “Coveted Spirit of Leeds” award on May 3, the release of the movie “V for Vendetta” on March 17 etc. These events are not periodic and do not recur. Our approach can detect both types of events of different time granularities. Our approach detected a lot of events that are not recognized previously by the existing work [3, 4] on the same dataset. The complete list of events detected is shown in table 5.

Table 5: Complete list of events detected

Event	Timestamp
Pre-period events	
Winter Olympics (Torino 2006)	February 26th
Current-period events	
Ash Wednesday	March 1st
Jack Wild died	March 1st
World Baseball Classic	March 3rd-20th
48th Annual Heard Museum Fair	March 4th, 5th
78th Academy Awards	March 5th
Triple Six Mafia won Academy Award	March 5th
Philadelphia flower show	March 5th-12th
Dubai Tennis Open ends	March 6th
Big 12 Women's Basketball Championship	March 7th-12th
Big Ten Conference Men's Basketball Tournament	March 9th-12th
NCAA men's Division I Basketball Tournament	March 14th-April 3rd
Ides of March	March 15th
John West salmon commercial	March 15th
Ram Bahdur Bomjon disappeared	March 16th
V for Vendetta movie released	March 17th
Saint Patrick’s day	March 17th
NCAA Women's Division I Basketball Tournament	March 18th-April 4th
Los Angeles Marathon	March 19th
Washington D.C. Cherry Blossom Festival	March 25th
27th Annual Young Artist Awards	March 25th
Buck Owens died	March 25th
Rocio Durcal died	March 25th
Bataan Memorial Death March	March 26th
Indy racing league season started	March 26th
Solar eclipse in North Africa	March 29th
Basic Instinct 2 movie released	March 31st
April fool’s day	April 1st
Liberty Bell Classic	April 2nd
140th anniversary of Baptist Union Baptist Church	April 2nd
Good Friday	April 14th
Scary movie 4 released	April 14th
Easter	April 16th
Boston Marathon	April 17th
Stanley Cup Playoffs	April 21st
Launch of lucky lines by Oregon Lottery	April 23rd
Italian Social Republic	April 25th
Dolphins Massacre at Zanzibar	April 28th
Steve Howe died	April 28th
The 33rd Annual Daytime Emmy Awards	April 28th
Pleasant valley baseball tournament	April 29th
The Hobbit movie started	April 31st
27th Sports Emmy Awards	May 1st
David Blaine performance at Lincoln Center	May 1st
Brooklyn Academy added to NHRP	May 2nd
10000 days album release	May 2nd
Simon Lindley received "Coveted Spirit of Leeds" award	May 3rd
National Teachers day	May 4th
Advanced Placement Test	May 1st-10th
Cindo de Mayo	May 5th
Men's World Ice Hockey Championship	May 5th-21st
132nd Kentucky Derby	May 6th
29th Annual Five Boro Bike Tour	May 7th
Fort Collins Old Town Marathon	May 7th
Chris Daughtry eliminated from American Idol 5	May 10th
Alligator attacks	May 14th
Mother’s day	May 14th
Tony Awards nominations	May 16th
The Amazing Race finale	May 17th
Penny saved 1000$ worth	May 17th
Cannes Film Festival	May 17th-28th
Big Island Film Festival	May 18th-21st
The Davinci Code movie release	May 19th
82nd Air Borne Division show	May 20th
NASCAR Sprint All-Star Challenge	May 20th
Strawberry Festival	May 21st, 22nd
10.5 Apocalypse Movie release	May 21st

41st Annual Country Music Awards	May 23rd
American Idol 5 ends	May 24th
Fleet week	May 24th-30th
Africa day	May 25th
31st Annual Million Dollar Beauty Ball	May 26th
Ultimate Fighting Championship 60: Hughes vs. Gracie	May 27th
The 90th Indianapolis 500	May 28th
Memorial day	May 29th
Post-period events	
The Omen movie release	June 6th
06/06/06 Doomsday	June 6th
FIFA World Cup (Germany)	June 9th
National Golden glove boxing championship	June 9th-13th
60th Annual Tony Awards	June 11th
Juneteenth Day	June 17th
Antique car show in Alabama	June 20th
USA Outdoor Track and Field Championships	June 21st-25th
Air shows New England	June 24th, 25th
Ann Arbor art fair	July 19th-21st
58th Annual Primetime Emmy Awards	August 27th
Albuquerque Baloon Festival	October 6th-15th
Month events	
NBA Basketball playoff	March, April
The Shoe show series aired on Resonance FM	March, April, May
American Idol	March, April, May
Annual walleye run in Fremont Ohio	March, April, May
IRS tax filing	March, April
Greenland ice melt by 250%	March, April
College Student Survey	March, April
1199 home care worker pay increase negotiations	March, April
Business Opportunities	
Summer - restaurants, resorts, cruises, islands etc	April, May

Table 5: Complete list of events detected (continued)

8.3. Experimental Analysis

DECK [4] outperformed two-phase-clustering algorithm [3] so we compare the performance of ECO with the DECK, DECK-NP [4] and DECK-GPCA [4] on the same dataset. Number of events detected is a simple way to compare approaches. ECO could detect 96 events where as DECK detected only 35 events previously. ECO could not detect 5 events in the list of 35 events detected by the DECK. On the other hand, DECK did not detect 61 events that ECO could detect. On time granularity comparison, ECO could detect 80 day events, 8 week events and 8 month events. In the events listed by DECK, 32 are day events, 3 are week events and no month events. As mentioned earlier, our approach could detect 1 pre-period, 83 current period and 12 post period events. The experimental results are shown in figure 8.

The evaluation metrics precision, recall, F-measure (F-1 score) and entropy are used along with the number of events detected to compare the performance. Precision is the ratio of number of correctly detected events to the overall discovered clusters. Recall is the ratio of number of correctly detected events to the total number of events. F-measure is computed based on the precision and recall as the weighted harmonic mean of precision and recall.

F-measure = 2 * precision * recall / (precision + recall)

For each generated cluster *i*, we compute Pij as the fraction of query-page pairs (query sessions) representing the true event *j*. Then the entropy of the cluster *i* is:

$$E_i = - \sum_j Pij \log Pij.$$

The total entropy can be calculated s the sum of the entropies of each cluster weighted by the size of each cluster: E $= \sum_i^m \frac{ni*Ei}{n}$, where *m* is the number of clusters, *n* is the total number of query-page pairs (query sessions) and *ni* is the size of the cluster *i*. The experimental results are shown in figure 9. ECO did fairly well in terms of precision and recall up to half of the data size. As the number of query sessions increased, the number of query patterns increased so the number of noisy query clusters increased which resulted in slight down fall of precision but not recall and increase in entropy.

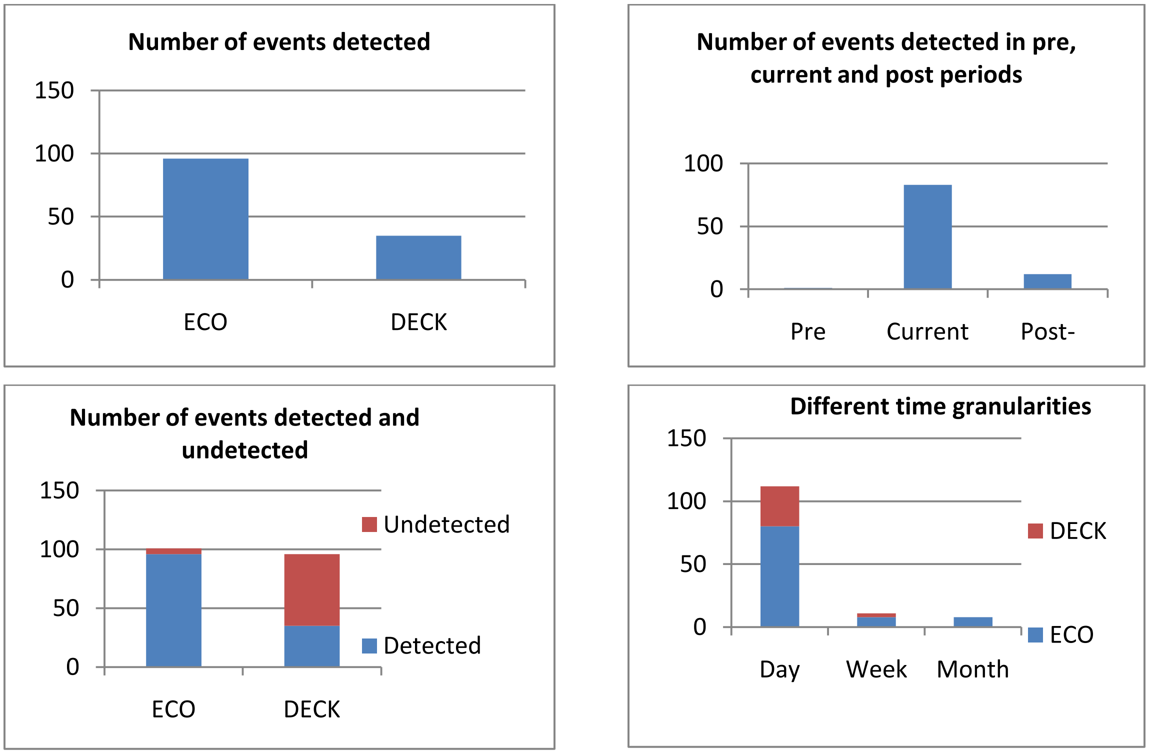


Figure 8: Comparison of ECO with DECK on number of events detected

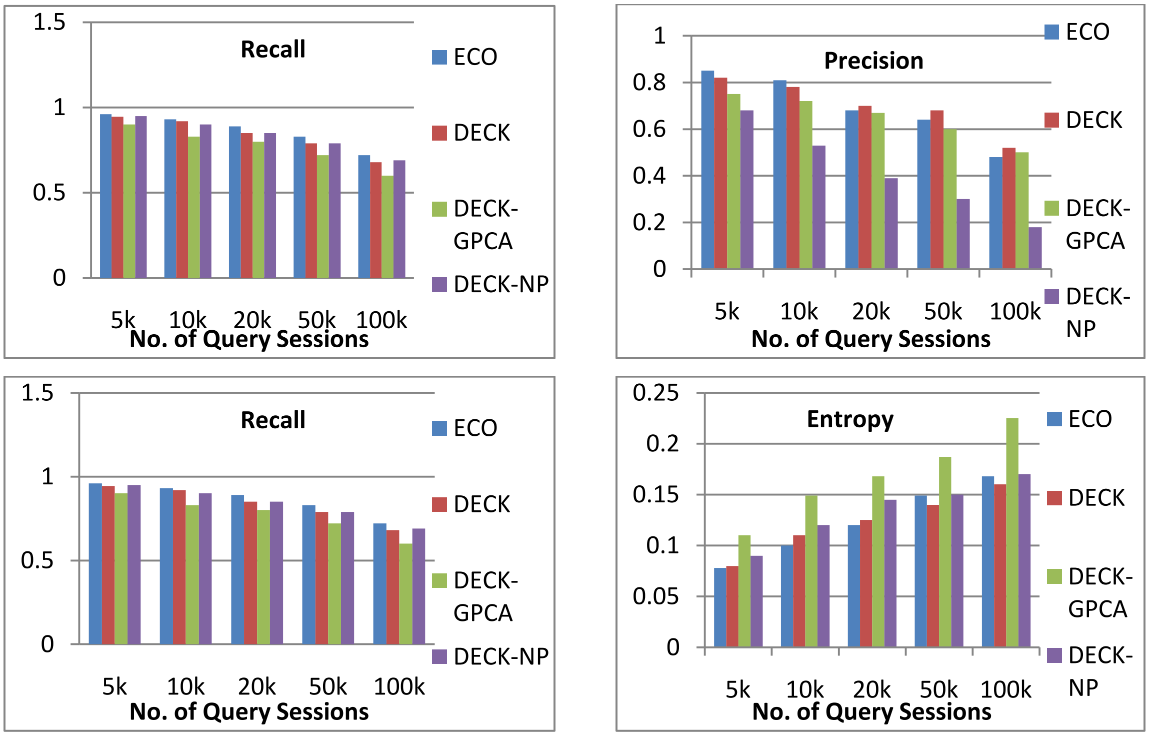


Figure 9: Precision, recall, F-measure and entropy of ECO and DECK

8.4. Effect of α

The factor α decides the weights for content-based similarity and feedback-based similarity. We ran experiments varying the value of α, which is shown in figure 10 below. The number of events detected varied accordingly. At α=0.15 31 events are detected. As the weight for feedback-based similarity increased we started identifying new clusters of events. At α=0.45 we got the best results. As the weight for feedback-based similarity increased further, the performance degraded.

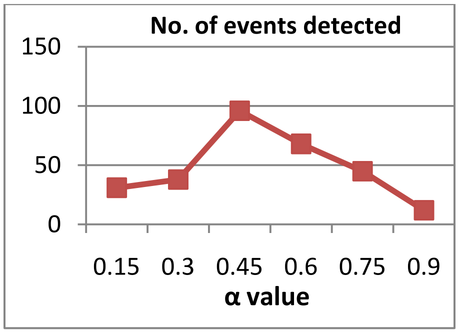


Figure 10: Effect of α on number of events detected

9. CONCLUSION

In this paper, we proposed an approach called ECO for detecting events from the click-through data. Firstly we performed data cleaning, transformation and preparation process to filter the noise and then portioned the click through data into collections of user defined granularity. Then we transformed the problem into query clustering, simultaneously trying to integrate the content, structure and semantics of the queries and click URLs. We introduced the hybrid cover graph to efficiently represent the clusters of query, page pairs. The evolutionary pattern of the query page pairs is embedded into the hybrid cover graph as vectors over the edges to sense the dynamics. Our results outperform most recent existing work [3, 4] in terms of the number of detected events, entropy measure, F-measure and recall.

REFERENCES

1. **Kunder, Maurice De.** The size of the World Wide Web. *World Wide Web Size.* [Online] 07 09, 2010. [Cited: 09 04, 2009.] http://www.worldwidewebsize.com/.

2. *Web mining in search engines.* **Baeza-Yates, Ricardo.** Dunedin, New Zealand : Australian Computer Society, Inc., 2004. Proceedings of the 27th Australasian conference on Computer science. Vol. 56, pp. 3-4.

3. *Event Detection from Evolution of Click-through Data.* **Zhao, Qiankun, et al.** Philadelphia, PA, USA : ACM New York, NY, USA, 2006. Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 484 - 493 .

4. *Using subspace analysis for event detection from web click-through data.* **Chen, Ling, Hu, Yiqun and Nejdl, Wolfgang.** Beijing, China : ACM New York, NY, USA , 2008. Proceeding of the 17th international conference on World Wide Web. pp. 1067-1068.

5. *Agglomerative clustering of a search engine query log.* **Beeferman, Doug and Berger, Adam.** Boston, Massachusetts, United States : ACM New York, NY, USA, 2000. Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 407 - 416 .

6. *Optimizing web search using web click-through data.* **Xue, Gui-Rong, et al.** Washington, D.C., USA : ACM New York, NY, USA, 2004. Proceedings of the thirteenth ACM international conference on Information and knowledge management. pp. 118 - 126 .

7. *Clustering user queries of a search engine.* **Wen, Ji-Rong, Nie, Jian-Yun and Zhang, Hong-Jiang.** Hong Kong, Hong Kong : ACM New York, NY, USA, 2001. Proceedings of the 10th international conference on World Wide Web. pp. 162 - 168 .

8. *Extracting semantic relations from query logs.* **Baeza-Yates, Ricardo and Tiberi, Alessandro.** San Jose, California, USA : ACM New York, NY, USA, 2007. Proceedings of the 13th ACM SIGKDD international conference on KDD. pp. 76 - 85 .

9. *Using Association Rules to Discover Search Engines Related Queries.* **Fonseca, Bruno M., et al.** s.l. : IEEE Computer Society Washington, DC, USA, 2003. Proceedings of the First Conference on Latin American Web Congress. p. 66.

10. *Density-Based Clustering in Spatial Databases.* **Sander, Jörg, et al.** s.l. : Kluwer Academic Publishers Hingham, MA, USA, 1998. 2nd International Conference on Knowledge Discovery. pp. 169 - 194.

11. *On-line new event detection and tracking.* **Allan, James, Papka, Ron and Lavrenko, Victor.** Melbourne, Australia : ACM New York, NY, USA, 1998. Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval. pp. 37 - 45 .

12. *A study of retrospective and on-line event detection.* **Yang, Yiming, Pierce, Tom and Carbonell, Jaime.** Melbourne, Australia : ACM New York, NY, USA, 1998. Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval . pp. 28 - 36.

13. *Parameter free bursty events detection in text streams.* **Fung, Gabriel Pui Cheong, et al.** Trondheim, Norway : VLDB Endowment , 2005. Proceedings of the 31st international conference on Very large data bases. pp. 181 - 192.

14. *Investigating behavioral variability in web search.* **White, Ryen W. and Drucker, Steven M.** Banff, Alberta, Canada : ACM New York, NY, USA, 2007. Proceedings of the 16th international conference on World Wide Web. pp. 21 - 30 .

15. *Graphs from Search Engine Queries.* **Baeza-Yates, Ricardo.** Harrachov, Czech Republic : Springer-Verlag Berlin, Heidelberg, 2007. Proceedings of the 33rd conference on Current Trends in Theory and Practice of Computer Science. pp. 1 - 8 .

16. *Evolutionary Patern-based Clustering of Web Usage Data.* **Zaho, Qiankun, Bhowmick, Sourav S. and Gruenwald, Le.** 2006. PAKDD. pp. 323-333.

17. *A picture of search.* **Pass, Greg, Chowdhury, Abdur and Torgeson, Cayley.** Hong Kong : ACM New York, NY, USA, 2006. Proceedings of the 1st international conference on Scalable information systems.

18. **Berkhin, Pavel.** *Survey of clustering data mining techniques.* San Jose, CA : s.n., 2002.

BIBLIOGRAPHY

1. **Kunder, Maurice De.** The size of the World Wide Web. *World Wide Web Size.* [Online] 07 09, 2010. [Cited: 09 04, 2009.] http://www.worldwidewebsize.com/.

2. *Web mining in search engines.* **Baeza-Yates, Ricardo.** Dunedin, New Zealand : Australian Computer Society, Inc., 2004. Proceedings of the 27th Australasian conference on Computer science. Vol. 56, pp. 3-4.

3. *Event Detection from Evolution of Click-through Data.* **Zhao, Qiankun, et al.** Philadelphia, PA, USA : ACM New York, NY, USA, 2006. Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 484 - 493 .

4. *Using subspace analysis for event detection from web click-through data.* **Chen, Ling, Hu, Yiqun and Nejdl, Wolfgang.** Beijing, China : ACM New York, NY, USA , 2008. Proceeding of the 17th international conference on World Wide Web. pp. 1067-1068.

5. *Agglomerative clustering of a search engine query log.* **Beeferman, Doug and Berger, Adam.** Boston, Massachusetts, United States : ACM New York, NY, USA, 2000. Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 407 - 416 .

6. *Optimizing web search using web click-through data.* **Xue, Gui-Rong, et al.** Washington, D.C., USA : ACM New York, NY, USA, 2004. Proceedings of the thirteenth ACM international conference on Information and knowledge management. pp. 118 - 126 .

7. *Clustering user queries of a search engine.* **Wen, Ji-Rong, Nie, Jian-Yun and Zhang, Hong-Jiang.** Hong Kong, Hong Kong : ACM New York, NY, USA, 2001. Proceedings of the 10th international conference on World Wide Web. pp. 162 - 168 .

8. *Extracting semantic relations from query logs.* **Baeza-Yates, Ricardo and Tiberi, Alessandro.** San Jose, California, USA : ACM New York, NY, USA, 2007. Proceedings of the 13th ACM SIGKDD international conference on KDD. pp. 76 - 85 .

9. *Using Association Rules to Discover Search Engines Related Queries.* **Fonseca, Bruno M., et al.** s.l. : IEEE Computer Society Washington, DC, USA, 2003. Proceedings of the First Conference on Latin American Web Congress. p. 66.

10. *Density-Based Clustering in Spatial Databases.* **Sander, Jörg, et al.** s.l. : Kluwer Academic Publishers Hingham, MA, USA, 1998. 2nd International Conference on Knowledge Discovery. pp. 169 - 194.

11. *On-line new event detection and tracking.* **Allan, James, Papka, Ron and Lavrenko, Victor.** Melbourne, Australia : ACM New York, NY, USA, 1998. Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval. pp. 37 - 45 .

12. *A study of retrospective and on-line event detection.* **Yang, Yiming, Pierce, Tom and Carbonell, Jaime.** Melbourne, Australia : ACM New York, NY, USA, 1998. Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval . pp. 28 - 36.

13. *Parameter free bursty events detection in text streams.* **Fung, Gabriel Pui Cheong, et al.** Trondheim, Norway : VLDB Endowment , 2005. Proceedings of the 31st international conference on Very large data bases. pp. 181 - 192.

14. *Investigating behavioral variability in web search.* **White, Ryen W. and Drucker, Steven M.** Banff, Alberta, Canada : ACM New York, NY, USA, 2007. Proceedings of the 16th international conference on World Wide Web. pp. 21 - 30 .

15. *Graphs from Search Engine Queries.* **Baeza-Yates, Ricardo.** Harrachov, Czech Republic : Springer-Verlag Berlin, Heidelberg, 2007. Proceedings of the 33rd conference on Current Trends in Theory and Practice of Computer Science. pp. 1 - 8 .

16. *Evolutionary Patern-based Clustering of Web Usage Data.* **Zaho, Qiankun, Bhowmick, Sourav S. and Gruenwald, Le.** 2006. PAKDD. pp. 323-333.

17. *A picture of search.* **Pass, Greg, Chowdhury, Abdur and Torgeson, Cayley.** Hong Kong : ACM New York, NY, USA, 2006. Proceedings of the 1st international conference on Scalable information systems.

18. **Berkhin, Pavel.** *Survey of clustering data mining techniques.* San Jose, CA : s.n., 2002.

VITA

Prabhu Kumar Angajala was born on January 3, 1986 in Vijayawada, India. He received distinction in Bachelor of Technology degree in Computer Science and Engineering from Jawaharlal Nehru Technological University, India in 2008. He has been a graduate student in the Computer Science Department at Missouri University of Science and Technology since August 2008 and worked as a Graduate Research Assistant under Dr. Sanjay Kumar Madria from August 2008 to May 2010. He received his Master’s in Computer Science at Missouri University of Science and Technology in December 2010.