An Efficient Incremental Indexing Mechanism for Extracting Top-k Representative Queries Over Continuous Data-streams[*]

Y.S.Horawalavithana
School of Computing
University of Colombo
Colombo, Sri Lanka
sam2010ucsc@acm.org

D.N.Ranasinghe
School of Computing
University of Colombo
Colombo, Sri Lanka
dnr@ucsc.cmb.ac.lk

ABSTRACT

Top-k publish/subscribe (pub/sub) models have gained traction as an expressive alternative to extend the binary notion of matching. In our study, we focus on the problem of extracting the k-most representative set of publications in the dynamic case where the results are updated over a stream of matching publications. This can be observed as the minimum independent dominating set problem in graph theory, when streaming publications are represented as dynamic graph spaces. Due to the inherent complexity of solving this problem over continuous data, an incremental indexing mechanism is proposed for handling a stream of publications. The proposed mechanism is based on Locality Sensitive Hashing (LSH) to avoid the overhead of recalculating neighborhoods over consecutive sliding windows. The experimental results show that the incremental version of LSH indexing mechanism reduces the computational cost of naive greedy approach significantly, while producing Top-k representative results at 70% accuracy compared to the naive optimal method.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Filtering

General Terms

Theory,Algorithms,Performance

Keywords

Dynamic diversification, Indexing, Publish/subscribe

ARM '15 December 07-11, 2015, Vancouver, BC, Canada
Copyright 2015 ACM ISBN 978-1-4503-3733-5/15/12
DOI: http://dx.doi.org/10.1145/2834965.2834975 ...$15.00.

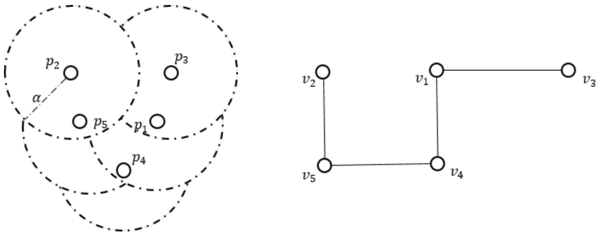

Figure 1: Graph representation of publication space with neighborhood α

1. INTRODUCTION

Inherently, Top-k pub/sub systems adopt an expressive stateful query processing nature which aim to overcome the drawbacks in Boolean models while controlling the number of delivered publications by the parameter k in a given delivery instance. However previous research has mainly considered threshold based schemes, which results in similar sets of relevant publications being delivered [3, 4]. In our study, we focus on a diversity aware ranking mechanism for extracting Top-k representative queries.

In fact, diversification models have attracted considerable attention in modern information processing systems where the possibility of getting dis-similar sets of relevant results has exploited in different models, but fails to maximize the representativeness of information content as a mean of increasing user satisfaction. Our study is based on a specialized form of k-diversity problem called DisC [2] to deal with continuous data-streams. Given the set of relevant publications P, DisC attempts to find the smallest set of objects that represent all relevant objects. Mathematically, DisC computes an answer set Q, such that $\forall p_i \in S, \exists p_j \in Q$, where $dist(p_i, p_j) \leq \alpha$ and $\forall p_i, p_j \in Q,\ p_i \neq p_j,\ dist(p_i, p_j) > \alpha$. To model the objectives behind above diversification method, let's consider a set of publications P represented by an undirected graph $G_{P,\alpha}(V, E)$ (Figure 1) such that each vertex $v_i \in V$, there is a publication $p_i \in P$ and an edge $(v_i, v_j) \in E$ iff $d(p_i, p_j) \leq \alpha$ for $N_\alpha(p_i)$ neighborhood where $N_\alpha(p_i) = \{p_j | p_i \neq p_j \wedge d(p_i, p_j) \leq \alpha\}$. Distance function d can be any p-norm for finite-dimensional vector spaces.

By aligning with graph definitions, the dominance condition ensures that all publications in the set P are represented by at least one similar publication in the selected set Q and, independent condition ensures that the publications in the

[*]2015 Association for Computing Machinery. ACM acknowledges that this contribution was authored or co-authored by an employee, contractor or affiliate of a national government. As such, the Government retains a nonexclusive, royalty-free right to publish or reproduce this article, or to allow others to do so, for Government purposes only.

selected set Q are dis-similar to each other. An independent dominating set for G, is a subset $V' \subseteq V$ such that for all $v \in V - V'$ there is a $v' \in V'$ for which $(v, v') \in E$, and such that no two vertices in V' are joined by an edge in E. The problem of locating DisC diverse subsets in static space can be formalized as an independent dominating set problem on graphs [1].

In our study, we address the selection of a DisC set of results under a dynamic setting where the subsets need to be updated frequently across the stream. Hence, we exploit a novel incremental index mechanism based on Locality Sensitive Hashing (LSH) to present a randomized approximation for the efficient computation along with performance results. The rest of the paper is structured as follows. The diversification model is explored in Section 2 which introduces the novel concept of adaptive diversification. In Section 3, we introduce an incremental indexing mechanism to efficiently handle streaming publications for solving the dynamic diversification problem. We present our experimental results in Section 4 and finally, Section 5 concludes the paper.

2. ADAPTIVE DIVERSIFICATION

We extend the static notion of DisC to handle streaming publications by exploiting adaptive diversification based on dominance and independence conditions.

Definition 2.1 (Dynamic DisC). *Let $P = \{P_1, ..., P_j\}$ be a stream of publications grouped into corresponding sets over sliding windows $w = \{w_1, ..., w_i\}$, and let any two consecutive windows be w_{i-1}, w_i and P^*_{i-1} is the diverse subset of P_{i-1}, and given the relevancy metric u depicting the relevance of a publication which is calculated according to the degree of user interest, and the distance metric d expresses the dissimilarity between publication points, s.t. $d(p_i,\ p_j) \leq \alpha$ where $\alpha > 0$ is the neighborhood parameter, and $\lambda > 0$ is a parameter that tunes the importance of diversification, then selecting the diverse subset P^*_i of P_i at window w_i such that $P_{i-1} \cap P_i \notin \emptyset$ to satisfy dynamic DisC,*

$$P^*_i = arg\ max\ f_\alpha(Q_i,\ d,\ u);\ Q_i \subseteq P_i;\ |Q_i| = k;\ k \geq 0;\ \alpha \geq 0$$

$$s.t.\ f_\alpha(Q_i,\ d,\ u) = \lambda.\ \frac{g_\alpha(Q_i,\ d,\ u)}{h_\alpha(Q_i,\ d,\ u)}$$

$$where\ g_\alpha(Q_i,\ d,\ u) = \frac{1}{|Q_i|}.\sum_{p_i,\ p_j \in Q_i} \frac{u(p_j)}{u(p_i)}.d(p_i, p_j)\ and,$$

$$h_\alpha(Q_i,\ d,\ u) = \frac{1}{|P_i - Q_i|}.\sum_{p_i \in Q_i,\ p_j \in (P_i - Q_i)} \frac{u(p_j)}{u(p_i)}.d(p_i, p_j);$$

$$iff\ \forall p_i, p_j \in Q_i, d(p_i, p_j) > \alpha; (independence)$$

$$\forall p_i \in P_i, \exists p_j \in Q_i\ s.t.\ d(p_i, p_j) \leq \alpha; i \neq j; (dominance)$$

where it also must satisfy the continuity conditions [2] which are defined as durability and ordering to suit the dynamic settings. It's been already proved that solving DisC over static publication space is NP-Hard [1]. It keeps the dynamic version of DisC in the same family with the need to satisfy dominance and independence conditions at each sliding window instance. Thus, the minimum DisC diversified subsets are computed over sliding windows of length w. In the worst case, any addition of single publication to the set P or a removal of any publication from the set P may result in a completely different set of diversified items.

	$minhash_i$	publication A	publication B	publication C	publication D	publication E
L_1	h_1	1	0	0	1	4
	h_2	6	3	3	6	7
	h_3	0	6	6	0	9
L_2	h_4	4	1	1	4	1
	h_5	8	2	2	8	2
	h_6	0	5	5	0	5
L_3	h_7	3	4	1	1	4
	h_8	4	1	3	3	1
	h_9	0	0	2	2	0
L_4	h_{10}	5	0	1	0	5
	h_{11}	0	0	1	0	0
	h_{12}	1	1	1	1	1

Table 1: Segmented signature matrix

In streaming windows, the performance bottleneck occurs when locating neighborhoods. The straightforward way to solve the dynamic DisC problem is to apply any greedy algorithm [1] that solve DisC at each sliding window instance by assuming the publication space is static. Due to the overhead of re-calculating neighborhoods, we propose a hash based indexing mechanism to compute the k diversified results over consecutive windows.

3. INCREMENTAL INDEXING

Typically publications are represented by a characteristic matrix that depicts the existence of categorical values. The columns of the matrix correspond to the publications while the rows correspond to the universal set of category values which the publications are characterized with. Any cell in the table is represented by 1 or 0 based on the existence of the categorical value in the given publication.

MinHashing[1] is a technique to construct a signature that represents the given set of categorical values (i.e. publication). It's common to permute rows of the characteristic matrix and, take the index of the first row, in the permuted order, in which the column has a 1 for the correspondent column of publications. We can have $m > 0$ number of permutations to construct the vector of minhash signatures for any publication. For any publication X, we can construct the vector of minhash signatures $[h_1(X), ...h_m(X)]$ by applying m number of permutations to the characteristic matrix. We can form a signature matrix where the vector of minhash signature for a given publication is represented by the corresponding column. Then, we can evenly segment the minhash signature of any publication into L hash tables. The size of a signature segment is denoted by r where $r \leq m$ and $L \times r = m$. Each table consists of b number of buckets on average. For such a hash table, there is a hash function that takes a column vector of size r and, maps them into a bucket.

The signature matrix $(m \times n)$ in Table 1 shows $n = 5$ dummy publications A to E where each publication has been represented by a size $m = 12$ minhash signature. The signature matrix is divided into $L = 4$ hash tables where each hash table contains b arbitrary number of buckets. A bucket is denoted by a key which is calculated by taking a vector of minhash signature of size $r = m/L = 12/4 = 3$. In hash table L_1, the pair of publications A and D is mapped into the same bucket at L_1 bucket array, since their columns are identical. That similarity is estimated based on the hash

[1] http://en.wikipedia.org/wiki/MinHash

function for the corresponding hash table L_1, regardless of column vectors in other hash tables. This similarity mapping will repeat until all $n = 5$ publications are projected into buckets at L_1. The same process is performed on all hash tables simultaneously. For all L hash tables, the probability $1 - (1 - (1 - \alpha)^b)^L$ denotes that any closely similar publications will have the chance to be projected into at least one bucket among all hash tables.

Each bucket is considered as a neighborhood of similar publications. We pick the most relevant publication that has the highest relevancy score as the winner from each bucket. Any winner is dominant in it's bucket neighborhood. All the hash tables vote for such winners to be in the Top-k publications. We select k number of independent winners that have a majority of votes to be in the final Top-k publications. For streaming publications, it is not efficient to re-construct LSH index repeatedly over sliding window instances. Thus, we construct the LSH index incrementally as new items arrive and old ones expire. For a new publication, the system generates a new minhash signature to represent it. Then it is projected into at least one bucket at each L hash tables where the similar set of publications reside. We only consider the buckets which hold the new publication to update the list of votes. Only those buckets forward their votes for a Top-k candidate, such that a winning publication is being voted by only L number of buckets.

4. EVALUATION

All algorithms were implemented in Java & experiments were performed on a Linux node with 2.3 GHz processing power, 8GB memory. The data set contains information of $100K$ e-commerce products that were on sale at Amazon on-line marketplace during $November\ 17 - 19^{th}$ 2014. For the evaluation, we use products as high-dimensional publications and content-based subscriptions that follow zipfian properties. For simplicity, we refer the batch-wise LSH index as *BLSH* & incremental LSH index as *ILSH*. Also we evaluate the performance of index mechanisms with the naive greedy approach ($NAIVE$) [1] that solves DisC which is not equipped with an indexing mechanism.

ILSH produces ranked results incrementally over the current window while incorporating the computed neighborhood over the previous over-lapped window. Table 2 shows the average update cost of ILSH when a new publication arrives over a window of size D dimensions. Publications are inserted into the specific bucket at each hash table by pruning less probable candidate solutions. The pruning rule is optimistic at each hash table. For high-dimensional publications, ILSH incremental construction cost is slightly increased, as ILSH needs to maintain a universal characteristic matrix over the publication space. For simplicity, we periodically refresh the characteristic matrix to handle streaming publications. Figure 2 shows the performance comparison of ILSH with both BLSH and naive methods over the same stream of high-dimensional publications. ILSH avoids recomputing previous neighborhoods, and thus, fewer computations are required to update Top-k publications in consecutive windows. For solving dynamic DisC problem, ILSH performs faster than BLSH or naive as expected.

By keeping the results produced by naive greedy algorithm that solves DisC [1] as a benchmark, the accuracy of index based results is depicted by the probability of producing the same diverse set of Top-k results. We have explored the accuracy of results produced by LSH index under different sliding windows and, it is further refined by LSH similarity threshold. On average, the probability of getting the identical diverse set of results is around 0.7 under different similarity thresholds. Also we observe that, when the similarity threshold (s) ≈ 0.5, the accuracy of producing diversified results is increased. It ensures that our mechanism produces diversified set of publications by preserving the properties of S-curve as well.

D	50	250	500	1000
ILSH update cost (seconds)	0.165	0.314	0.523	0.917

Table 2: ILSH update cost

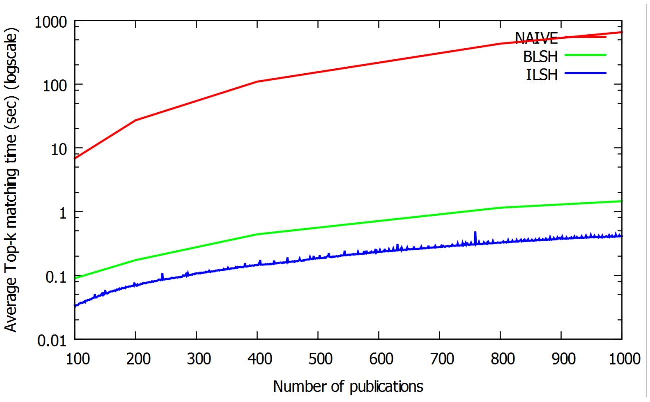

Figure 2: Top-k matching time at ILSH vs. BLSH vs. NAIVE when D=500

5. CONCLUSION

We extended the notion of DisC to produce dynamic representative sets across a stream of publications by introducing adaptive heuristics for computing approximate solutions. We also presented an incremental indexing mechanism based on LSH that increases the efficiency of the Top-k matching process by reducing the processing time significantly over the naive greedy approach. We are currently developing indexing methods to exploit the overlap among representative results of users who have similar interest in a multi-threaded distributed environment.

6. REFERENCES

[1] M. Drosou and E. Pitoura. Disc diversity: Result diversification based on dissimilarity and coverage. *Proc. VLDB Endow.*, 6(1):13–24, Nov. 2012.

[2] M. Drosou and E. Pitoura. Diverse Set Selection Over Dynamic Data. *IEEE Transactions on Knowledge and Data Engineering*, 26(5):1102–1116, 2014.

[3] K. Pripužić, I. Podnar Žarko, and K. Aberer. Top-k/w publish/subscribe: A publish/subscribe model for continuous top-k processing over data streams. *Information Systems*, 39:256–276, Jan. 2012.

[4] A. Shraer, M. Gurevich, M. Fontoura, and V. Josifovski. Top-k Publish-Subscribe for Social Annotation of News. *Proceedings of the VLDB Endowment*, 6(6):385–396.