Selecting Representative Objects Considering Coverage and Diversity

Shenlu Wang[†], Muhammad Aamir Cheema[‡], Ying Zhang[§], Xuemin Lin[†]
[†]*School of Computer Science and Engineering, The University of New South Wales, Australia*
[‡]*Faculty of Information Technology, Monash University, Australia*
[§]*Centre for Quantum Computation and Intelligent Systems, University of Technology, Sydney, Australia*
swan398@cse.unsw.edu.au, aamir.cheema@monash.edu, ying.zhang@uts.edu.au, lxue@cse.unsw.edu.au

ABSTRACT

We say that an object o attracts a user u if o is one of the top-k objects according to the preference function defined by u. Given a set of objects (e.g., restaurants) and a set of users, in this paper, we study the problem of computing a set of representative objects considering two criteria: *coverage* and *diversity*. Coverage of a set S of objects is the distinct number of users that are attracted by the objects in S. Although a set of objects with high coverage attracts a large number of users, it is possible that all of these users have quite similar preferences. Consequently, the set of objects may be attractive only for a specific class of users with similar preference functions which may disappoint other users having widely different preferences. The diversity criterion addresses this issue by selecting a set S of objects such that the set of attracted users for each object in S is as different as possible from the sets of users attracted by the other objects in S. The existing work on representative objects considers only one of the coverage and diversity criteria. We are the first to consider both of the criteria where the importance of each criterion can be controlled using a parameter. Our algorithm has two phases. In the first phase, we prune the objects that cannot be among the representative objects and compute the set of attracted users (also called reverse top-k) for each of the remaining objects. In the second phase, the reverse top-k of these objects are used to compute the representative objects maximizing coverage and diversity. Since this problem is NP-hard, the second phase employs a greedy algorithm. For the sake of time and space efficiency, we adopt MinHash and KMV Synopses to assist the set operations. We prove that the proposed greedy algorithm is ϵ-approximate. Our extensive experimental study on real and synthetic data sets demonstrates the effectiveness of our proposed techniques.

1. INTRODUCTION

In the Web 2.0 era, increasing amount of spatial data are manually or algorithmically annotated. This results in a rich body of information associated with objects rather than a minimal flat list of keyword descriptions in the pre-Web 2.0 era. This rich information usually brings about its own substructures (e.g., detailed menu of a restaurant), and may contain hyperlinks to external documents (e.g., reviews of the restaurant at various review web sites or social network sites). Such rich information allows users to refine their search based on various criteria. For example, a user may issue a query to find a near by restaurant based on various criteria such as number of Foursquare check-ins (indicating its popularity), average rating on Yelp, and its average cost per person. Similarly, a user looking to buy a house may issue a query to find houses based on price, size and location (e.g., profile of the suburb, distance to schools, shopping centres and transport etc.).

In this paper, we study the problem of selecting t representative spatial objects. Consider the example of a real estate company that maintains a database containing the properties to be sold and the preferences of the purchasers. Suppose that the company wants to choose t houses to display their ads on the walls of the shop or publish on the home page of their website. These t houses must be chosen such that they attract a large number of purchasers considering their preferences such as locations, prices, construction dates of the houses.

Consider another example where the tourism department of a city wants to prepare a brochure to promote near by tourist attractions. Each tourist site has different attributes such as distance from the city centre, average cost to visit the site, rating on reviewing websites, and time required to visit etc. Different tourists may have different preferences, e.g., some tourists may prefer to visit only near by sites and may not be too concerned about the associated costs whereas others may want only the cheap options. The tourism department needs to choose t tourist sites such that they attract a large number of tourists considering that different tourists may have widely different preferences.

In this paper, we formalize this problem and propose algorithms to compute representative products. We model each object (e.g., house, tourist site) as a d dimensional point where d is the number of attributes associated with the object. A user preference can be described by a d dimensional weighting vector w, where $0 \leq w[i] \leq 1$ indicates the importance of the attribute i to the user and $\sum_{i=1}^{d} w[i] = 1$. This model is widely adopted by existing works related to top-k queries [18, 22], e.g., a top-k query returns k objects ranked highest according to a user defined preference function w.

Similar to the existing work, we use reverse top-k queries

GeoRich'15, May 31 - June 04 2015, Melbourne, VIC, Australia
 ISBN 978-1-4503-3668-0/15/05 $15.00
DOI: http://dx.doi.org/10.1145/2786006.2786012.

Table 1: Coverage

Objects	Reverse Top-k
o_1	$\{w_1, w_2, w_3, w_4\}$
o_2	$\{w_3, w_4, w_5, w_6\}$
o_3	$\{w_6, w_7\}$

Table 2: Diversity

Objects	Reverse Top-k	centroid
o_1	$\{w_1\}$	(0,1)
o_2	$\{w_2, w_3\}$	(0.3,0.7)
o_3	$\{w_4, w_5\}$	(0.7,0.3)
o_4	$\{w_6\}$	(1,0)

in the definition of representative objects. Given an object q, a reverse top-k query returns every user preference w for which q is one of the top-k objects. Note that a reverse top-k query returns the users that are attracted by the object q. The existing work has focused on using reverse top-k queries in selecting the representative objects. Specifically, Koh *et. al* [11] consider *coverage* and select a set of t representative objects that maximizes the total number of users attracted by the objects. In contrast, Gkorgkas *et. al* [8] consider *diversity* and select a set of objects such that the objects are different and attract users having a wide range of preferences. Below, we describe the limitations of both approaches.

Koh *et. al* [11] consider coverage and aim at maximizing the number of users attracted by the selected objects. Consider the example of Table 1 that shows the reverse top-k (i.e., the attracted users) for three objects o_1, o_2 and o_3. Assuming $t = 2$, the set of objects $\{o_1, o_2\}$ has the maximum coverage as it attracts the users w_1 to w_6. However, note that these two objects attract similar users. This is because the users attracted by an object (i.e., its reverse top-k) usually have similar preference functions, e.g., w_1 to w_4 are similar because they are the reverse top-k of the object o_1. Also, w_3 to w_6 are similar because they are the reverse top-k of o_2. Hence, although the set $\{o_1, o_2\}$ attracts a large number of users, these users have similar preferences. We argue that it is better to choose $\{o_1, o_3\}$ as the representative set of objects. This is because these two objects attract two different sets of users and, therefore, are attractive for the users having a wider range of preferences. Thus, it is important to consider not only the total number of reverse top-k users of the objects but also the dissimilarity of the reverse top-k of the selected objects.

Gkorgkas *et. al* [8] has taken diversity into consideration. Their work maximizes the sum of pair-wise dissimilarities, where dissimilarity between two objects is the distance between the centroid of their reverse top-k preferences. Since they only consider diversity, the returned set of objects may have a low coverage. Consider the example of Table 2 that shows the reverse top-k of four objects. Assume that the weighting vectors are $w_1 = (0, 1)$, $w_2 = (0.2, 0.8)$, $w_3 = (0.4, 0.6)$, $w_4 = (0.6, 0.4)$, $w_5 = (0.8, 0.2)$, $w_6 = (1, 0)$. Table 2 also shows, for each object o_i, the centroid of its reverse top-k preferences. Considering the diversity as in [8], the two representative products are o_1 and o_4 because the distance between their centroids is maximum. Although o_2 and o_3 cover more user preferences, these objects are not selected because the distance between their centroids is smaller than that of o_1 and o_4. Hence, their work may return the representative objects that are attractive only for a small number of users.

Inspired by the above, in this paper, we study the problem of computing representative objects considering both coverage and diversity. The importance of the coverage and diversity criteria can be controlled using a parameter α. We remark that representative skylines [13, 17, 14] also study a similar problem in the sense that a set of t most significant objects is returned. However, representative skylines return the objects considering the pair-wise comparison between the objects and do not consider the preferences of the users at all. As a result, it is possible that the objects returned by the representative skyline may be attractive for a very small set of users (or no user at all).

Our contributions in this paper can be summarized as follows:

- We are the first to propose a model for selecting the representative objects that considers both coverage and diversity based on the user preference functions.
- We formulate the problem as an optimization problem of an objective function. We propose to solve the problem in two phases. We propose three different algorithms for the first phase and two algorithms for the second phase. The problem addressed in the second phase is NP-hard and we show that one of our proposed algorithms for the second phase is an ϵ-approximate algorithm. The other algorithm for the second phase does not have the accuracy guarantee but it significantly improves the efficiency using MinHash and KMV Synopses.
- We conduct extensive experiments on both real and synthetic data sets and show the effectiveness of our proposed techniques. The results demonstrate that our algorithms are efficient and have high accuracy.

The rest of the paper is organized as follows. Section 2 reviews the related works. Section 3 provides the necessary preliminaries. In section 4 we formally define the t-representative problem. We present a greedy algorithm in Section 5 and prove it is ϵ-approximate in Section 6. Finally, we present experimental results in Section 7.

2. RELATED WORK

Top-k queries. Top-k queries [18, 22, 16] have been extensively studied in the past. Tao *et. al* [18] process top-k queries based on branch-and-bound search on R-tree index. Xin *et. al* [22] propose index method to boost run time performance of top-k query processing. Ilyas et al. [9] give a comprehensive survey of top-k query processing techniques.
Reverse top-k queries. Reverse top-k queries is first introduced by Vlachou *et. al* [19]. The size of a product's reverse top-k indicates its potential market and possible profit. Ge *et. al* [7] evaluate batch of top-k queries by grouping similar queries, which in turn enables the computation of a large number of reverse top-k queries. Yu *et. al* [24] address the problem of processing a large number of continuous top-k queries. The proposed method relies on a pre-calculated index of the k-th ranked objects. Vlachou *et. al* [21] propose a branch-and-bound algorithm for reverse top-k queries that based on R-tree index. Cheema *et. al* [4] propose an algorithm to compute reverse top-k queries in dual space using the idea of k-upper envelope.

Reverse k nearest neighbors (RkNN) query is a widely studied problem and is a variant of the reverse top-k query that only considers the distance attribute. Yang *et. al* [23] provide a survey of the most notable RkNN algorithms.

Representative objects. Lin *et. al* [13] select a set of skyline representatives, which collectively dominate as many distinct points as possible. This work maximize coverage. Tao *et. al* [17] select a set of representative skyline points that best describe the skyline contour. This work maximize diversity. Magnani *et. al* [14] take both the significance (sigmoid significance) of all the records and their diversity (skyline edit distance, namely the number of skyline records in between) into account, and select representative skyline points that maximize the sum of significance and minimum distance to other skyline points.

Top-m influential query [20] finds the m most influential products based on the influence score, which is the cardinality of a product's reverse top-k user preference. The selected products have significant impact in the market individually. Lin *et. al* [12] study the problem of determining a set of most demanding products. In this work, products are not ranked according to user preferences. Instead, a product is acceptable as long as the product is below a threshold defined by a user. They also assume all qualified products have equal probability to be selected by a user, and calculate the potential profit of a product by summing up the chances it is selected across all users. This work considers coverage only. Koh *et. al* [11] select a set of products that collectively covers the maximum possible number of reverse top-k. This work considers coverage only. Gkorgkas *et. al* [8] select a set of products that are most diverse to each other based on centroid of reverse top-k. This work considers diversity only.

Other related works. Diversification of the results also has received significant attention in the past few years [25, 5, 10]. Drosou *et. al* [5] study the problem of result diversification based on dissimilarity and coverage. In their definition, the neighbors of an object, i.e., the objects lying at distance no more than a threshold are considered as similar to the object. A diverse subset selected for a query result contains objects such that: 1) each object in the result is represented by at least one similar object in the diverse subset, and 2) none of the objects in the diverse subset are similar to each other. Indyk *et. al* [10] consider efficient construction of composable core-sets for diversity and coverage maximization problems. A core-set is representative in the sense that an approximate solution to the whole data set can be obtained from the core-set only.

3. PRELIMINARIES

3.1 Top-k and reverse top-k queries

Given a d-dimensional space $\mathcal{R}^d$, we have a set of data objects $\mathcal{P}$, where each object $p \in \mathcal{P}$ is described by a d-dimensional point $p = \{p[1], \cdots, p[d]\}$ in $\mathcal{R}^d$. Here, $p[i] \geq 0$ for $1 \leq i \leq d$ describes an attribute of the data object p.

In addition, we have a set of weighting vectors $\mathcal{W}$, where each weighting vector $w \in \mathcal{W}$ is described by a d-dimensional vector $w = \{w[1], \cdots, w[d]\}$ in $\mathcal{R}^d$. Here, $w[i] \geq 0$ for $1 \leq i \leq d$ describes the relative weight of the i-th attribute of data objects in $\mathcal{P}$. As a widely accepted convention, we assume $\sum_{i=1}^{d} w[i] = 1$. The score of an object p considering a preference function w is $s(p, w) = \sum_{i=1}^{d} p[i] \cdot w[i]$.

DEFINITION 1 : **Top-k query.** Given a set of data objects $\mathcal{P}$, and a query weighting vector w, a top-k query retrieves a set of data objects $\mathcal{S}_k(w)$ such that $\mathcal{S}_k(w) \subseteq \mathcal{P}$, $|\mathcal{S}_k(w)| = k$, and $\forall p_1, p_2 : p_1 \in \mathcal{S}_k(w)$, $p_2 \in \mathcal{P} - \mathcal{S}_k(w)$ it holds that $s(p_1, w) \leq s(p_2, w)$.

DEFINITION 2 : **Reverse top-k query.** Given a set of data objects $\mathcal{P}$, a set of weighting vectors $\mathcal{W}$, and a query object $p \in \mathcal{P}$, a reverse top-k query retrieves a set of weighting vectors $\mathcal{S}_k(p) \subseteq \mathcal{W}$ such that a weighting vector $w \in \mathcal{S}_k(p)$ if and only if $\exists p' \in \mathcal{S}_k(w)$ for which $s(p, w) \leq s(p', w)$.

3.2 MinHash

MinHash is a technique for estimating Jaccard similarity, firstly introduced by Border [3]. Given a set of elements $\mathcal{V}$ and a hash function $\mathcal{F} : \mathcal{V} \rightarrow \mathcal{V}$ that defines a random ordering over $\mathcal{V}$. Let $\mathcal{F}^{min}(\mathcal{V})$ be the minimum element in $\mathcal{V}$ with respect to $\mathcal{F}$. Given two randomly chosen sets $\mathcal{V}_1$ and $\mathcal{V}_2$, and let $\mathcal{J}(\mathcal{V}_1, \mathcal{V}_2)$ be the Jaccard similarity of $\mathcal{V}_1$ and $\mathcal{V}_2$. The probability that $\mathcal{F}^{min}(\mathcal{V}_1) = \mathcal{F}^{min}(\mathcal{V}_2)$ is equal to $\mathcal{J}(\mathcal{V}_1, \mathcal{V}_2)$. Let r be a random variable that is 1 when $\mathcal{F}^{min}(\mathcal{V}_1) = \mathcal{F}^{min}(\mathcal{V}_2)$, and 0 otherwise, then r is an unbiased estimator of $\mathcal{J}(\mathcal{V}_1, \mathcal{V}_2)$. However, r has too high a variance to be useful for estimating $\mathcal{J}(\mathcal{V}_1, \mathcal{V}_2)$. The idea of MinHash scheme is to reduce the variance by averaging several variables constructed in the same way. In other words, given N hash functions $\mathcal{F}_i$, where $i \in [1, N]$, and let n be the number of $\mathcal{F}_i$s that has $\mathcal{F}_i^{min}(\mathcal{V}_1) = \mathcal{F}_i^{min}(\mathcal{V}_2)$, then $\mathcal{J}(\mathcal{V}_1, \mathcal{V}_2)$ can be estimated by $\frac{n}{N}$.

3.3 KMV synopsis

The k minimal values (KMV) technique is first proposed by Bar-Yossef *et al.* [1] to estimate the number of distinct values in a data stream. Suppose h is a pair-wise independent hash function which randomly maps the values onto the range $[0, 1]$ and $h(v_i) \neq h(v_j)$ for any two different values v_i and v_j. A KMV synopses of a set $\mathcal{D}$ of values, denoted by $\mathcal{L}_{\mathcal{D}}$, keeps k smallest hash values of the elements in $\mathcal{D}$. Then the number of distinct values in $\mathcal{D}$, denoted by $\hat{D}$, can be estimated by $\hat{D} = \frac{k}{\mathcal{L}^{(k)}}$ where $\mathcal{L}^{(k)}$ is k-th smallest hash value. Bayer *et al.* [2] systematically investigate the problem of distinct value estimation under multi-set operations. They show that $\hat{D} = \frac{k-1}{\mathcal{L}^{(k)}}$ is an unbiased estimator.

Multi-set Union. Consider two sets $\mathcal{A}$ and $\mathcal{B}$, with their KMV synopses $\mathcal{L}_{\mathcal{A}}$ and $\mathcal{L}_{\mathcal{B}}$ of size k_A and k_B, respectively. Let $\mathcal{L}_{\mathcal{A}} \oplus \mathcal{L}_{\mathcal{B}}$ be the set comprising the k smallest values in $\mathcal{L}_{\mathcal{A}} \cup \mathcal{L}_{\mathcal{B}}$, where $k = min(k_A, k_B)$. Then the set $\mathcal{L} = \mathcal{L}_{\mathcal{A}} \oplus \mathcal{L}_{\mathcal{B}}$ is the KMV synopses of $\mathcal{A} \cup \mathcal{B}$. The number of distinct values in $\mathcal{A} \cup \mathcal{B}$, denoted by $\hat{D}_{\cup}$, can be estimated as follows:

$$\hat{D}_{\cup} = \frac{k-1}{\mathcal{L}^{(k)}} \tag{1}$$

4. PROBLEM DEFINITION

Consider a set of objects $\mathcal{P}$ and a set of user preferences represented by weighting vectors $\mathcal{W}$. Let $p_1, p_2 \in \mathcal{P}$ be two objects. Given a positive integer k, let $\mathcal{S}_k(p_1), \mathcal{S}_k(p_2) \subseteq \mathcal{W}$ be the sets of reverse top-k user preferences of p_1 and p_2, respectively. We define the distance $d(p_1, p_2)$ between p_1 and p_2 as:

$$d(p_1, p_2) = 1 - \frac{|\mathcal{S}_k(p_1) \cap \mathcal{S}_k(p_2)|}{|\mathcal{S}_k(p_1) \cup \mathcal{S}_k(p_2)|} \tag{2}$$

Equation (2) is defined based on Jaccard similarity that is a widely used similarity measure for sets. Note that

$d(p_1, p_2) \in [0, 1]$. When $\mathcal{S}_k(p_1)$ and $\mathcal{S}_k(p_2)$ has no common elements (i.e., user preferences), $d(p_1, p_2) = 1$, and when they are identical, $d(p_1, p_2) = 0$.

PROBLEM 1. ***t-diversity problem.*** *Given a set of data objects $\mathcal{P}$, a distance function d that measures the dissimilarity between two data objects, t-diversity problem is to select a subset $\mathcal{S}_d \subseteq \mathcal{P}$ of size t such that:*

$$\mathcal{S}_d = \arg \max_{\substack{\mathcal{S}_d \subseteq \mathcal{P} \\ |\mathcal{S}_d|=t}} d_{min}(\mathcal{S}_d) \tag{3}$$

where $d_{min}(\mathcal{S}_d)$ is the minimum pair wise distance between any two data objects in $\mathcal{S}_d$, defined as:

$$d_{min}(\mathcal{S}_d) = \min_{\substack{p_i, p_j \in \mathcal{S}_d \\ i \neq j}} d(p_i, p_j) \tag{4}$$

An optimal solution of t-diversity problem selects t data objects $\mathcal{S}_d$ maximizing $d_{min}(\mathcal{S}_d)$. Intuitively, t maximum diverse objects should serve t groups of users with their unique preferences. t-diverse problem can be seen as a dispersion problem [6] which is to select a number of points out of a set of given candidate points, such that the minimum distance between pairs of the selected points is maximized. It is proved that p-dispersion problem is NP-complete [6] by transformation to the clique problem.

PROBLEM 2. ***t-coverage problem.*** *Given a set of data objects $\mathcal{P}$, a set of weighting vectors $\mathcal{W}$, a positive integer k, t-coverage problem is to select a subset $\mathcal{S}_d \subseteq \mathcal{P}$ of size t such that:*

$$\mathcal{S}_c = \arg \max_{\substack{\mathcal{S}_c \subseteq \mathcal{P} \\ |\mathcal{S}_c|=t}} c(\mathcal{S}_c) \tag{5}$$

where $c(\mathcal{S}_c)$ is the proportion of $\mathcal{W}$ covered by $\mathcal{S}_c$ (e.g., belong to some $S_k(p)$ for $p \in \mathcal{S}_c$), formally defined as:

$$c(\mathcal{S}_c) = \frac{|\bigcup_{p_i \in \mathcal{S}_c} \mathcal{S}_k(p_i)|}{|\mathcal{W}|} \tag{6}$$

An optimal solution of t-coverage problem selects t data objects $\mathcal{S}_c$ maximizing $c(\mathcal{S}_c)$. Intuitively, t maximum coverage objects should collectively be favoured by as many users as possible. t-coverage problem can be seen as a set cover problem. As each data object p is associated with a set of weighting vectors $\mathcal{S}_k(p)$, which is a subset of $\mathcal{W}$, selecting $\mathcal{S}_c$ is equivalent to selecting a set of subsets of $\mathcal{W}$ such that as many weighting vectors as possible should belong to at least one $\mathcal{S}_k(p)$ for $p \in \mathcal{S}_c$. Hence, t-coverage problem is NP-hard.

PROBLEM 3. ***t-representative problem.*** *Given a set of data objects $\mathcal{P}$, a set of weighting vectors $\mathcal{W}$, a positive integer k, a non-negative weight α, t-representative problem is to select a subset $\mathcal{S}_r \subseteq \mathcal{P}$ of size t such that:*

$$\mathcal{S}_r = \arg \max_{\substack{\mathcal{S}_r \subseteq \mathcal{P} \\ |\mathcal{S}_r|=t}} \left(\alpha \cdot d_{min}(\mathcal{S}_r) + (1 - \alpha) \cdot c(\mathcal{S}_r) \right) \tag{7}$$

Note that t-representative problem is the same as t-coverage problem when $\alpha = 0$ and is the same as t-diversity problem when $\alpha = 1$. Hence, t-representative problem is also NP-hard.

5. ALGORITHMS AND TECHNIQUES

A fundamental component for computing representative objects is to compute reverse top-k sets of the objects. Hence we propose a two phases approach: 1) in the first phase, we compute reverse top-k sets for all objects excluding the objects that have empty reverse top-k sets (Section 5.1); 2) in the second phase, we use the reverse top-k sets to select t-representative objects (e.g., $\mathcal{S}_r \subseteq \mathcal{P}$) via a greedy algorithm (Section 5.2).

5.1 Computing reverse top-k

We use R*-tree to index $\mathcal{P}$ and $\mathcal{W}$. It is well known that only the objects that are in k-skyband can have a non-empty reverse top-k set [4]. Hence, we first use **BBS** [15] algorithm to compute k-skyband points from $\mathcal{P}$, and store these objects in a main memory R*-tree. Next, we present three algorithms to compute reverse top-k sets using the k-skyband objects.

Reverse top-k query based algorithm. A straight forward algorithm is to calculate reverse top-k using the existing branch-and-bound reverse top-k algorithm [20], for each k-skyband object. This can be improved. Given a point p, let $\mathcal{D}(p)$ denote the set of points that dominate p. Note that if $\mathcal{S}_k(p')$ for any $p' \in \mathcal{D}(p)$ is empty, then $\mathcal{S}_k(p)$ is also empty (see Theorem 1 in [20]). Therefore, we calculate reverse top-k of the k-skyband points p in the order that they are reported by BBS, and we proceed to reverse top-k calculation via BBRA [21] algorithm only if $\mathcal{D}(p)$ is not empty.

Top-k query based algorithm. We note that BBRA [21] needs to calculate top-k objects for many preferences to get $\mathcal{S}_k(p)$ for a single object. When BBRA is invoked for multiple objects, there are redundant computations of top-k objects. Hence, a better way is to calculate top-k $\mathcal{S}_k(w)$ for all the preferences $w \in \mathcal{W}$, and add w to $\mathcal{S}_k(p)$ if $p \in \mathcal{S}_k(w)$. In this algorithm, we compute top-k objects for each preference w using a branch-and-bound search on R*-tree.

Near brute force algorithm. We notice that the number of k-skyband objects is significantly smaller than the total number objects. We compute the reverse top-k of these objects as follows. For each w, we iterate through all k-skyband objects p, and calculate the score $s(p, w)$. We maintain a size k heap, and keep only the best k objects. At the end, we add w to $\mathcal{S}_k(p)$ if p remains in the heap. This algorithm is almost a brute force algorithm except that it considers only the objects in the k-skyband.

5.2 Selecting representative objects

Since t-representative problem is NP-hard, we propose a greedy algorithm that iteratively selects a object that maximizes the value of $\alpha \cdot d_{min}(\mathcal{S}_r) + (1-\alpha) \cdot c(\mathcal{S}_r)$. Note that both $d_{min}(\mathcal{S}_r)$ and $c(\mathcal{S}_r)$ require a lot of set operations. When $|\mathcal{S}_k(p)|$ for candidate point p is large, these set operations would become the bottleneck of $\mathcal{S}_r$ computation. In our first algorithm, we compute the exact set operations. Whereas, in the second algorithm, to speed up these set operations, we employ MinHash for $d_{min}(\mathcal{S}_r)$ calculation, and employ KMV Synopses for $c(\mathcal{S}_r)$ calculation. Note that the second algorithm approximates the set operations but is more efficient.

6. THEORETICAL ANALYSIS

In this section, we analyse the approximation ratio of the

proposed greedy algorithm. We remark that this analysis is applicable only for the case when the exact computation of set operations is conducted and not when MinHash and KMV Synopses are used to approximate the set operations.

THEOREM 1. *The proposed greedy algorithm is ϵ-approximate for some $\epsilon \in [min(\frac{1}{t}, \frac{1}{|\mathcal{S}_k(p_1)|}), 1]$.*

PROOF. When $t = 1$, t-representative problem becomes t-coverage problem. Since the proposed greedy algorithm select the object p that has $|\mathcal{S}_k(p)| \geq |\mathcal{S}_k(p')|$ for all $p' \in \mathcal{P}$, the solution is optimal (i.e., $\epsilon = 1$).

For $t > 1$, let $\mathcal{S}_r = \{p_1, \cdots, p_t\}$ be the set of t objects selected by the proposed greedy algorithm, and let $\mathcal{S}_r^o = \{p_1^o, \cdots, p_t^o\}$ be the optimal solution to the t-representative problem. The proximation ratio ϵ can be obtained by:

$$\epsilon = \frac{\alpha \cdot d_{min}(\mathcal{S}_r) + (1-\alpha) \cdot c(\mathcal{S}_r)}{\alpha \cdot d_{min}(\mathcal{S}_r^o) + (1-\alpha) \cdot c(\mathcal{S}_r^o)} \quad (8)$$

$$= \frac{\alpha \cdot d_{min}(\mathcal{S}_r) + \frac{(1-\alpha)}{|\mathcal{W}|} \cdot |\bigcup_{p_i \in \mathcal{S}_r} \mathcal{S}_k(p_i)|}{\alpha \cdot d_{min}(\mathcal{S}_r^o) + \frac{(1-\alpha)}{|\mathcal{W}|} \cdot |\bigcup_{p_i^o \in \mathcal{S}_r^o} \mathcal{S}_k(p_i^o)|} \quad (9)$$

If we can find a pair p_i and p_j in $\mathcal{S}_r$ such that $\mathcal{S}_k(p_i)$ and $\mathcal{S}_k(p_j)$ are identical, we know that $c(\mathcal{S}_r) = c(\mathcal{S}_r^o) = 1$, and $d_{min}(\mathcal{S}_r) = d_{min}(\mathcal{S}_r^o) = 0$. Hence we have $\epsilon = 1$.

If $\mathcal{S}_k(p_i)$ differ each other by at least one user preference for all $p_i \in \mathcal{S}_r$, $\mathcal{S}_r$ may or may not be the optimal solution. Without lose of generality, let $|\mathcal{S}_k(p_1)| \geq |\mathcal{S}_k(p_i)|$ for $i \in [1, t]$, and $|\mathcal{S}_k(p_1^o)| \geq |\mathcal{S}_k(p_i^o)|$ for $i \in [1, t]$. We can get:

$$\epsilon \geq \frac{\alpha \cdot (1 - \frac{|\mathcal{S}_k(p_1)|-1}{|\mathcal{S}_k(p_1)|}) + \frac{(1-\alpha)}{|\mathcal{W}|} \cdot |\mathcal{S}_k(p_1)|}{\alpha \cdot 1 + \frac{(1-\alpha)}{|\mathcal{W}|} \cdot (t \cdot |\mathcal{S}_k(p_1^o)|)} \quad (10)$$

$$\geq \frac{\alpha \cdot (1 - \frac{|\mathcal{S}_k(p_1)|-1}{|\mathcal{S}_k(p_1)|}) + \frac{(1-\alpha)}{|\mathcal{W}|} \cdot |\mathcal{S}_k(p_1)|}{\alpha \cdot 1 + \frac{(1-\alpha)}{|\mathcal{W}|} \cdot (t \cdot |\mathcal{S}_k(p_1)|)} \quad (11)$$

$$= \frac{|\mathcal{S}_k(p_1)|^2 + \alpha \cdot (|\mathcal{W}| - |\mathcal{S}_k(p_1)|^2)}{t \cdot |\mathcal{S}_k(p_1)|^2 + \alpha \cdot (|\mathcal{W}| \cdot |\mathcal{S}_k(p_1)| - t \cdot |\mathcal{S}_k(p_1)|^2)} \quad (12)$$

The derivation[1] is:

$$\frac{|\mathcal{W}| \cdot (t - |\mathcal{S}_k(p_1)|)}{(t \cdot |\mathcal{S}_k(p_1)| + (|\mathcal{W}| - t \cdot |\mathcal{S}_k(p_1)|) \cdot \alpha)^2} \quad (13)$$

Now we have 3 cases:

1. When $t = |\mathcal{S}_k(p_1)|$, ϵ is lower bounded by $\frac{1}{t}$.
2. When $t > |\mathcal{S}_k(p_1)|$, this is a monotonically increasing function, and ϵ is lower bounded by a value $\in [\frac{1}{t}, \frac{1}{|\mathcal{S}_k(p_1)|}]$, depending on the value of α.
3. When $t < |\mathcal{S}_k(p_1)|$, this is a monotonically decreasing function, and ϵ is lower bounded by a value $\in [\frac{1}{|\mathcal{S}_k(p_1)|}, \frac{1}{t}]$, depending on the value of α.

□

7. EXPERIMENTS

In this section, we present our experimental evaluation. All algorithms are implemented in C++, compiled by g++ with flag -O3. The experiments are run on a 32-bit PC with Intel Xeon 2.40GHz dual CPU and 4GB memory running Debian Linux. As moderate data sets may be completely loaded in main memory, we use main memory R*-tree with 4kB pagesize, and we evaluate CPU time mainly. For the cases where disk-based index may be used, we also report the number of page requests.

[1]Obtained from quotient rule, $\frac{d}{d\alpha}\left(\frac{u}{v}\right) = \frac{v\frac{du}{d\alpha} - u\frac{dv}{d\alpha}}{v^2}$.

Table 3: Experimental settings

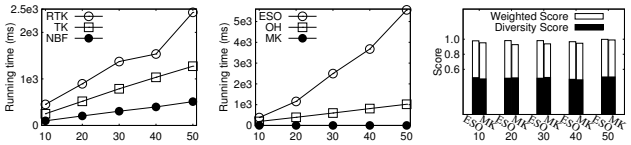
Parameter	Values	
$\|\mathcal{W}\|$	**10k**, 20k, 30k, 40k, 50k	(small)
	100k, 200k, 500k, 1000k	(large)
	100k, **200k**, 500k	(house)
$\|\mathcal{P}\|$	**10k**, 20k, 30k, 40k, 50k	(small)
	100k, 200k, 500k, 1000k	(large)
	147043	(house)
d	**3**, 4, 5	
	6	(house)
k	5, **10**, 20, 30, 50	
t	5, **10**, 15, 20, 30	
α	0.2, **0.5**, 0.8	

Algorithms We compare Reverse top-k query based algorithm (**RTK**), Top-k query based algorithm (**TK**) and Near brute force algorithm (**NBF**) for phase one. We also compare greedy selection using exact set operation (**ESO**) and MinHash and KMV Synopses based set operation (**MK**) for phase two. Note that the overhead of creating MinHash and KMV Synopses for use by the process of MK is also evaluated, and denoted as **OH**.

Data sets We use both synthetic and real data similar to those used in [8]. Specifically, we use two synthetic data sets, namely uniform and anti-correlated. For the uniform data set, the data object values for all dimensions are generated independently using a uniform distribution. The anti-correlated data set is generated in the same way as in [8]. We also use the same real data set **HOUSE** that consists of more than 147,043 6-dimensional tuples, representing the percentage of an American family's annual income spent on 6 types of expenditure: gas, electricity, water, heating, insurance, and property tax. For the data set $\mathcal{W}$, we use a clustered distribution to simulate different types of user preferences.

Parameters Main parameters and values used are specified in Table 3, default settings are in bold. We have 100 randomly generated hash functions for MinHash. We use Golden ratio multiplication as the hash function for KMV Synopses and we keep 17 smallest values.

7.1 Effect of $|\mathcal{W}|$



(a) phase 1 (b) phase 2 (c) score

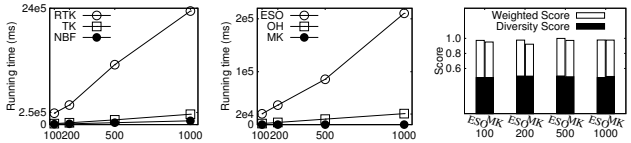
Figure 1: Effect of $|\mathcal{W}|$ (in 1000) (small uniform)

Figure 1 shows the effect of $|\mathcal{W}|$. This experiment is conducted on small uniform data set. In phase 1 (Figure 1(a)), NBF is several times faster than RTK, the fastest among the 3 algorithms. It also scales the best compared with the other two. Table 4 shows the number of page requests in phase 1. We can see that the IO cost demonstrates similar trends as

Table 4: Number of page requests

$\|\mathcal{W}\|$	RTK	TK	NBF
10,000	327123	642	642
20,000	586712	954	954
30,000	908755	1272	1272
40,000	918044	1603	1603
50,000	1492885	1930	1930

in CPU time. We omit report of IO cost in later experiments as they are always similar to the CPU time of phase 1. In phase 2 (Figure 1(b)), despite the overhead of computing MinHash and KMV Synopses (OH), MK runs much faster than ESO. Although OH demonstrates linear growth with $|\mathcal{W}|$, it scales much better than ESO. MK process demonstrates almost no growth. The score of the results selected by MK is very close to ESO, and ESO achieves almost the maximum possible score (note that the maximum possible score is one). As our default setting for α is 0.5, the diversity part(e.g., $\alpha \cdot d_{min}(\mathcal{S}_r)$ is around half of the weighted score. In this experiment, the use of MinHash and KMV Synopses does not affect the quality of selected results much.



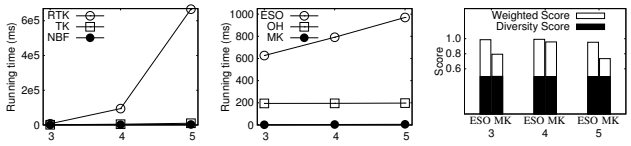
(a) phase 1 (b) phase 2 (c) score

Figure 2: Effect of $|\mathcal{W}|$ (in 1000) (large anti-correlated)

Figure 2 shows the effect of $|\mathcal{W}|$ on large anti-correlated data set. We can see that this group of figures demonstrates similar trend to Figure 1. The reason that NBF and MK scale even better in this experiment is not because of the cardinality, but the data distribution. We remark that $\sum_{p \in \mathcal{P}} |\mathcal{S}_k(p)| = k \cdot |\mathcal{W}|$ is the same for any data distribution. For anti-correlated data sets, the number of points in k-skyband is much larger compared with uniform data sets. Although $|\mathcal{S}_k(p)|$ is smaller on average, the dominating factor is still the number of k-skyband points. Therefore, both in phase 1 and 2, the scalability of RTK and ESO are even worse compared with the experiments on uniform data set.

Due to space limitation, we omit experiments on effect of $|\mathcal{P}|$ as increasing $|\mathcal{P}|$ with fixed $|\mathcal{W}|$ is equivalent as decreasing $|\mathcal{W}|$ with fixed $|\mathcal{P}|$.

7.1.1 *Effect of* d

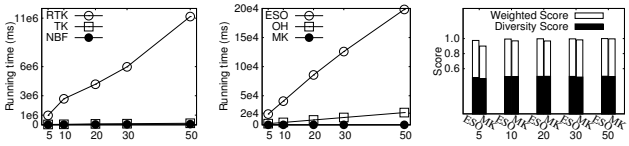


(a) phase 1 (b) phase 2 (c) score

Figure 3: Effect of d (small anti-correlated)

Figure 3 shows the effect of d. We can see that the cost of RTK grows significantly compared with the other two. ESO slightly grows as the number of non-empty $\mathcal{S}_k(p)$ grows with the number of k-skyband points when d increases. Due to the inaccuracy of KMV Synopses, the coverage score of MK results are sometimes only half of the ones selected by ESO. In contrast, the diversity scores of MK looks identical to the ones of ESO. This is because, as d grows, diversity is easier to achieve, especially on anti-correlated data sets, while coverage is harder to achieve.

7.1.2 *Effect of* k

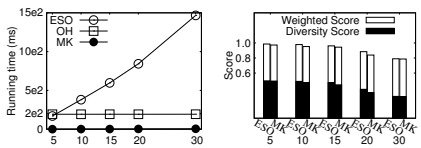


(a) phase 1 (b) phase 2 (c) score

Figure 4: Effect of k (house)

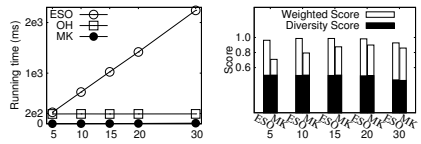
Figure 4 is conducted on the real data set, HOUSE, and shows the effect of k. Both NBF and TK are not much influenced by k. However, RTK grows fast with k. In phase 2, MK including overhead scales much better than ESO. In Figure 4(c), there is a slight growth of coverage score along with k for both ESO and MK. This is because with larger k, the size of $\mathcal{S}_k(p)$ is larger on average, and hence easier to achieve coverage.

7.1.3 *Effect of* t



(a) phase 2 (b) score

Figure 5: Effect of t (small uniform)

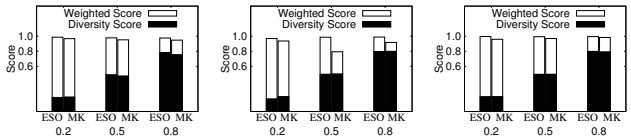


(a) phase 2 (b) score

Figure 6: Effect of t (small anti-correlated)

Figure 5 (conducted on small uniform data set) and Figure 6 (conducted on small anti-correlated data set) show the effect of t. ESO is faster on uniform data set because of the smaller number of k-skyband points, despite the larger size of $\mathcal{S}_k(p)$ on average. We can also notice that it is easier to achieve coverage but harder to achieve diversity on uniform data sets, and that it is easier to achieve diversity but harder to achieve coverage on anti-correlated data sets. Hence, with increasing t, diversity score drops while coverage score remains high on uniform data set, and coverage score increases while diversity score remains high on anti-correlated data set.

7.1.4 Effect of α

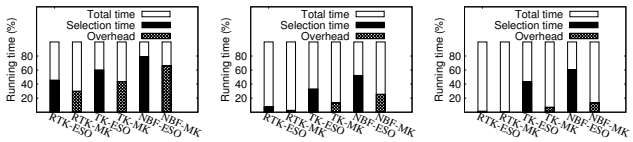


(a) uniform (b) anti-correlated (c) house

Figure 7: Effect of α

Figure 7 shows the effect of α on uniform data set, anti-correlated data set, and real data set. The figures clearly demonstrates the effectiveness of α on controlling the weight of diversity score and coverage score. It is clear that coverage scores were sacrificed for diversity scores with large α.

7.1.5 Significance of phase 2 time



(a) uniform (b) anti-correlated (c) house

Figure 8: Percentage of phase 2 time

Although we introduced MinHash and KMV Synopses to speed up set operations and save space, we also encourage exact set operation in some cases. For example, space limitation may not be an issue with nowadays hardware when the data set is of moderate size. Also, when the proportion of CPU time on set operation is not dominant, exact set operation may be considered as it has theoretical guarantees (Section 6).

Figure 8 shows the percentage of phase 2 time of all the 6 algorithms[2] on uniform data set (Figure 8(a)), anti-correlated data set (Figure 8(b)), and real data set (Figure 8(c)). Note that, selection time is identical for RTK-ESO, TK-ESO and NBF-ESO, overhead is identical for RTK-MK, TK-MK and NBF-MK. Selection time for RTK-MK, TK-MK and NBF-MK are also identical. However, their proportions are too small to be visible. We distinguish the overhead time for computing MinHash and KMV Synopses because this part is critical when considering reduce time and space cost by sacrificing accuracy.

Within the range of our experiments, it is clear that NBF is faster than both RTK and TK. The fancy pruning techniques of RTK and TK does not seem to justify their performance, and this is because they are designed for single query processing, but here when applied in our problem, they actually introduce redundant computations compared with NBF.

8. CONCLUSION

Given a set of objects and a set of user preferences, we address the problem of selecting a set of t representative objects that maximize the number of attracted users having widely different preference functions. The existing working consider only one of diversity and coverage criteria whereas we consider both of the criteria. We divide the problem in two phases and propose several algorithms for each phase. Specifically, we propose three algorithms for the first phase and show that a bruteforce approach works better than existing fancier algorithms. For the second phase, we propose two algorithms. One computes exact set operations and have higher accuracy. While the other computes approximate set operations sacrificing the accuracy but has better efficiency. The second phase of the problem is NP-hard and we prove that our proposed greedy algorithm is ϵ-approximate. We conduct extensive experiments to demonstrate the effectiveness of our proposed techniques.

[2](3 for phase 1) $\times$ (2 for phase 2)

Acknowledgements. Muhammad Aamir Cheema is supported by ARC DE 130101002 and DP 130103405. The research of Xuemin Lin is supported by NSFC 61232006, NSFC 61021004 and ARC (DP 120104168, DP 140103578, DP 150102728). Ying Zhang is supported by ARC DP 130103245 and DP 110104880.

9. REFERENCES

[1] Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, D. Sivakumar, and Luca Trevisan. Counting distinct elements in a data stream. In *Randomization and Approximation Techniques, 6th International Workshop, RANDOM 2002, Cambridge, MA, USA, September 13-15, 2002, Proceedings*, pages 1–10, 2002.

[2] Kevin S. Beyer, Peter J. Haas, Berthold Reinwald, Yannis Sismanis, and Rainer Gemulla. On synopses for distinct-value estimation under multiset operations. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, Beijing, China, June 12-14, 2007*, pages 199–210, 2007.

[3] Andrei Z. Broder. On the resemblance and containment of documents. In *Compression and Complexity of Sequences, 1997, Salerno, Jun 11-13, 1997, Proceedings*, pages 21–29, 1997.

[4] Muhammad Aamir Cheema, Zhitao Shen, Xuemin Lin, and Wenjie Zhang. A unified framework for efficiently processing ranking related queries. In *Proceedings of the 17th International Conference on Extending Database Technology, EDBT 2014, Athens, Greece, March 24-28, 2014.*, pages 427–438, 2014.

[5] Marina Drosou and Evaggelia Pitoura. Disc diversity: result diversification based on dissimilarity and coverage. *PVLDB*, 6(1):13–24, 2012.

[6] Erhan Erkut. The discrete p-dispersion problem. *European Journal of Operational Research*, 46(1):48–60, 1990.

[7] Shen Ge, Leong Hou U, Nikos Mamoulis, and David W. Cheung. Efficient all top-k computation - a unified solution for all top-k, reverse top-k and top-m influential queries. *IEEE Trans. Knowl. Data Eng.*, 25(5):1015–1027, 2013.

[8] Orestis Gkorgkas, Akrivi Vlachou, Christos Doulkeridis, and Kjetil Nørvåg. Finding the most diverse products using preference queries. In *Proceedings of the 18th International Conference on Extending Database Technology, EDBT 2015, Brussels, Belgium, March 23-27, 2015.*, pages 205–216, 2015.

[9] Ihab F. Ilyas, George Beskales, and Mohamed A. Soliman. A survey of top-k query processing techniques in relational database systems. *ACM Comput. Surv.*, 40(4), 2008.

[10] Piotr Indyk, Sepideh Mahabadi, Mohammad Mahdian, and Vahab S. Mirrokni. Composable core-sets for diversity and coverage maximization. In *Proceedings of the 33rd ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS'14, Snowbird, UT, USA, June 22-27, 2014*, pages 100–108, 2014.

[11] Jia-Ling Koh, Chen-Yi Lin, and Arbee L. P. Chen. Finding k most favorite products based on reverse top-t queries. *VLDB J.*, 23(4):541–564, 2014.

[12] Chen-Yi Lin, Jia-Ling Koh, and Arbee L. P. Chen. Determining k-most demanding products with maximum expected number of total customers. *IEEE Trans. Knowl. Data Eng.*, 25(8):1732–1747, 2013.

[13] Xuemin Lin, Yidong Yuan, Qing Zhang, and Ying Zhang. Selecting stars: The k most representative skyline operator. In *Proceedings of the 23rd International Conference on Data Engineering, ICDE 2007, The Marmara Hotel, Istanbul, Turkey, April 15-20, 2007*, pages 86–95, 2007.

[14] Matteo Magnani, Ira Assent, and Michael L. Mortensen. Taking the big picture: representative skylines based on significance and diversity. *VLDB J.*, 23(5):795–815, 2014.

[15] Dimitris Papadias, Yufei Tao, Greg Fu, and Bernhard Seeger. Progressive skyline computation in database systems. *ACM Trans. Database Syst.*, 30(1):41–82, 2005.

[16] Zhitao Shen, Muhammad Aamir Cheema, Xuemin Lin, Wenjie Zhang, and Haixun Wang. A generic framework for top-k pairs and top-k objects queries over sliding windows. *IEEE Trans. Knowl. Data Eng.*, 26(6):1349–1366, 2014.

[17] Yufei Tao, Ling Ding, Xuemin Lin, and Jian Pei. Distance-based representative skyline. In *Proceedings of the 25th International Conference on Data Engineering, ICDE 2009, March 29 2009 - April 2 2009, Shanghai, China*, pages 892–903, 2009.

[18] Yufei Tao, Vagelis Hristidis, Dimitris Papadias, and Yannis Papakonstantinou. Branch-and-bound processing of ranked queries. *Inf. Syst.*, 32(3):424–445, 2007.

[19] Akrivi Vlachou, Christos Doulkeridis, Yannis Kotidis, and Kjetil Nørvåg. Reverse top-k queries. In *Proceedings of the 26th International Conference on Data Engineering, ICDE 2010, March 1-6, 2010, Long Beach, California, USA*, pages 365–376, 2010.

[20] Akrivi Vlachou, Christos Doulkeridis, Kjetil Nørvåg, and Yannis Kotidis. Identifying the most influential data objects with reverse top-k queries. *PVLDB*, 3(1):364–372, 2010.

[21] Akrivi Vlachou, Christos Doulkeridis, Kjetil Nørvåg, and Yannis Kotidis. Branch-and-bound algorithm for reverse top-k queries. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2013, New York, NY, USA, June 22-27, 2013*, pages 481–492, 2013.

[22] Dong Xin, Chen Chen, and Jiawei Han. Towards robust indexing for ranked queries. In *Proceedings of the 32nd International Conference on Very Large Data Bases, Seoul, Korea, September 12-15, 2006*, pages 235–246, 2006.

[23] Shiyu Yang, Muhammad Aamir Cheema, Xuemin Lin, and Wei Wang. Reverse k nearest neighbors query processing: Experiments and analysis. *PVLDB*, 8(5):605–616, 2015.

[24] Albert Yu, Pankaj K. Agarwal, and Jun Yang. Processing a large number of continuous preference top-k queries. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2012, Scottsdale, AZ, USA, May 20-24, 2012*, pages 397–408, 2012.

[25] Chengyuan Zhang, Ying Zhang, Wenjie Zhang, Xuemin Lin, Muhammad Aamir Cheema, and Xiaoyang Wang. Diversified spatial keyword search on road networks. In *Proceedings of the 17th International Conference on Extending Database Technology, EDBT 2014, Athens, Greece, March 24-28, 2014.*, pages 367–378, 2014.