Diversifying with Few Regrets, But too Few to Mention

Zaeem Hussain
The University of Queensland
Queensland, Australia
z.hussain@uq.edu.au

Hina A. Khan
The University of Queensland
Queensland, Australia
h.khan3@uq.edu.au

Mohamed A. Sharaf
The University of Queensland
Queensland, Australia
m.sharaf@uq.edu.au

ABSTRACT

Representative data provide users with a concise overview of their potentially large query results. Recently, diversity maximization has been adopted as one technique to generate representative data with high coverage and low redundancy. Orthogonally, regret minimization has emerged as another technique to generate representative data with high utility that satisfy the user's preference. In reality, however, users typically have some pre-specified preferences over some dimensions of the data, while expecting good coverage over the other dimensions. Motivated by that need, in this work we propose a novel scheme called ReDi, which aims to generate representative data that balance the tradeoff between regret minimization and diversity maximization. ReDi is based on a hybrid objective function that combines both regret and diversity. Additionally, it employs several algorithms that are designed to maximize that objective function. We perform extensive experimental evaluation to measure the tradeoff between the effectiveness and efficiency provided by the different ReDi algorithms.

1. INTRODUCTION

Apart from the prevailing query-answer paradigm for data access, emerging data exploration platforms typically apply novel post-processing techniques on both the queries and their respective answers to provide users with guidance and insights. Generating representative data is one such technique that aims to provide meaningful summary of a potentially large query answer (e.g., [2, 14, 9, 1]).

Early examples of representative data generation methods include the well-studied *top-k* and *skyline* queries [13]. In top-k, the user's preference is captured by means of a utility function over the different dimensions of data, whereas in skyline, that preference is captured by applying the dominance property over those dimensions. Recently, regret minimization has been proposed as a practical alternative for both queries [9]. In regret minimization, a small representative set is generated by considering the universe of all possible utility functions. Hence, a user does not need to specify a specific utility function, as it is the case in top-k, but are still provided with a small and concise representative set, unlike the skyline query, in which the result can be arbitrarily large. In the absence of preference, however, diversification methods have been recently employed to generate representative sets that provide high coverage of the accessed data, while minimizing redundancy (e.g., [2, 14]).

ExploreDB '15, May 31-June 04, Melbourne, VIC, Australia
Copyright 2015 ACM ISBN 978-1-4503-3740-3/15/05 $15.00
DOI: http://dx.doi.org/10.1145/2795218.2795225

Table 1: Car Database

Car	MPG	HP	Weight	Height
p_1	51	134	1760	52.4
p_2	40	110	2945	48.8
p_3	41	191	1875	54.3
p_4	35	198	2050	56.3
p_5	30	140	2215	50.6

In many applications, however, the user might have some notion of preference associated with some dimensions of the data, while other dimensions are neutral. In that case, it is desired to select representatives that: 1) minimize regret over the preference dimensions (i.e, high utility), and 2) maximize diversity over the neutral dimensions (i.e., low redundancy).

For example, consider a tourist visiting downtown Melbourne for SIGMOD and is looking for few restaurants to try during her visit. That user might have some preference for restaurants with low price and high rating. At the same time, she might not want all restaurants to be cluttered in one location so that she gets to see more of the city during her short visit.

To capture that tradeoff between preference and coverage, we propose a novel scheme called ReDi, which aims to generate representative data that balance the tradeoff between regret minimization and diversity maximization. Our proposed scheme ReDi is based on a hybrid objective function formulated as the linear weighted combination of the diversity and regret objectives. To that end, ReDi incorporates two alternative novel algorithms that are based on two different algorithmic design approaches. In particular, we propose the ReDi-Greedy algorithm, which is a constructive based heuristic, and we also propose ReDi-SWAP, which is a local search based algorithm. Further, we study the tradeoff those two algorithms exhibit in terms of efficiency and effectiveness.

The rest of this paper is organized as follows. We provide preliminaries and related work in Section 2. Next we formulate our hybrid objective function in Section 3, and present our ReDi scheme in Section 4. Our evaluation testbed and results are reported in Section 5. We conclude in Section 6.

2. PRELIMINARIES

2.1 Representative Data

In many application domains, a user is more interested in a concise answer rather than a potentially large query result. That motivated the need for developing effective methods for generating representative datasets, in which post-processing techniques are applied to select a small set of representative tuples from a rather large query result (e.g., [2, 14, 9, 1]). Such representatives data allow users to quickly assess the usefulness of the returned results and their relevance to their needs

In this work, we assume a typical data exploration model, in which a user explores a D-dimensional database by posing *range* queries. Such queries retrieve a number of results, or tuples, from the database. Database results can be generally viewed as a set of attribute values represented as data points in a multi-dimensional space. For instance, consider a database $\mathcal{DB}$, which consists of a set of D-dimensional tuples. Each tuple $p_i = < p_i[1], p_i[2], ..., p_i[D] >$ is basically a point in a D-dimensional space, where the value of $p_i[j]$ is drawn from the domain of attribute A_j.

A range query is simply represented as a multi-dimensional box, which is also known as hyper-rectangle. Given a range query Q, let $P = \{p_1, \ldots, p_n\}$ be the set of data points that fall within the multi-dimensional range specified by Q. For instance, Table 1 shows a P subset of a Cars database, in which there are $n = 5$ data points, and each data point has $D = 4$ dimensions. Given a query result P, data exploration platforms employ different representative data extraction methods to retrieve a small subset $S \subseteq P$, which represents the original result P.

Early examples of such methods include the well-studied *top-k* (e.g., [5, 4]) and *skyline* (e.g., [13, 12]) queries. While both queries provide significant advantages in preference and personalized databases, their application is limited in the absence of preference parameters, which is the case in data exploration [11]. This motivates the need for novel parameter-free methods for extracting meaningful representative tuples from a query answer. In this work we focus on two such methods, namely: *diversification*, and *regret minimization*. In particular, data diversity has been adopted as one method to ensure maximum coverage while minimizing redundancy in the representative set [2, 14], whereas regret minimization has been adopted as the method to maximize the relevance of representative data to the posed query [9]. In the next sections, we describe those two methods in details, and in Section 3, we define our hybrid objective function, which balances the tradeoff between minimizing regret and maximizing diversity.

2.2 Maximizing Diversity

The aim of diversification methods is to select a small *diverse* subset of a query result set. In particular, given a query result P, the goal is to select a subset S^*, where $|S^*| = k$, such that the diversity of the results in S^* is maximized. While the diversity of a set can have different definitions [14], in this work we focus on the widely used content-based definition of diversity. Content diversity is an instance of the *p-dispersion* problem [3], in which the objective is to maximize the overall dissimilarity within a set of selected objects.

In particular, given a metric d that measures the distance between two results, e.g., the Euclidean distance among two data points, the diversity of a set S is measured by a *diversity function* $f(S, d)$ that captures the dissimilarity between the results in S. To that end, a number of different diversity functions have been employed in the literature, among which previous research has mostly focused on measuring diversity based on either the *average* or the *minimum* of the pairwise distances between results [2, 14]. The former is known as max-sum diversity and the latter as max-min diversity. Without loss of generality, in this paper we focus on max-sum diversity. Putting it together, the diversification problem is defined as follows: Let P be the set of results that satisfy a user query Q and k be a positive integer such that $k \leq |P|$. Let also d be a distance metric and f a diversity function. Then, diversification is defined as selecting a subset S^* of P, such that:

$$S^* = \arg \max_{\substack{S \subseteq P \\ |S| = k}} f(S, d)$$

The max-sum diversity function is defined as:

$$f(S, d) = \frac{1}{k(k-1)} \sum_{i=1}^{k} \sum_{j>i}^{k} d(p_i, p_j) \text{ and } p_i, p_j \in S \quad (1)$$

Identifying an optimal diverse subset S^* has been shown to be NP-hard [3], therefore greedy-based heuristics are typically employed to select a near optimal diverse subset. For instance, in *Greedy Construction* the diverse subset S is constructed iteratively, where in each iteration t, the point with the maximum distance from the current diverse set S_t is selected and added to that set, until k points are selected [2, 14]. Further optimizations have also been proposed to improve the efficiency of that greedy approach, including our previous work [7, 8, 6].

2.3 Minimizing Regret

In the presence of some preference over some of the data dimensions, representatives are typically selected based on those preferences. For instance, in a *top-k* query, the user specifies a utility function g, such that the utility of a D-dimensional point p is given by $g(p)$. Hence, the top k points based on their values under that utility function are then selected as representatives. Alternatively, in a *skyline* query, any point for which there is no other point with better values in all D dimensions is selected. However, in a *top-k* query the user is required to precisely define a utility function, which is often unknown, whereas in a *skyline* query there is no control on the size of the output, which can be arbitrarily large. Such drawbacks motivated the recent regret minimization methods [9].

Particularly, the goal of regret minimization is to select a subset of size k, which minimizes the maximum regret ratio for any class of utility functions. This captures how disappointed any user could be had they seen k representative tuples instead of the whole database [9]. Specifically, for a certain user with a utility function $g \in G$ and a subset of points $S \subseteq P$, the regret of that user is $max_{p \in P} g(p) - max_{p \in S} g(p)$, where $max_{p \in P} g(p)$ is the maximum utility if the user saw the entire database P, whereas $max_{p \in S} g(p)$ is the maximum utility if the user saw only the representative set S. Accordingly, given a class of utility functions G, the maximum regret ratio of a subset S, denoted $rr_P(S, G)$, is defined as:

$$rr_P(S, G) = sup_{g \in G} \frac{max_{p \in P} g(p) - max_{p \in S} g(p)}{max_{p \in P} g(p)}$$

In this work, we restrict G to be the class of linear functions with positive weights, as in [9]. Given that restriction, consider again the dataset of table 1 and suppose there is some notion of preference associated with attributes MPG and HP. For simplicity, further assume $G = \{g_{\{0.2,0.8\}}, g_{\{0.4,0.6\}}, g_{\{0.6,0.4\}}, g_{\{0.8,0.2\}}\}$ where: $g_{\{x,y\}}(MPG, HP) = MPG.x + HP.y$ If $S = \{p_1, p_2\}$, then $rr_P(S, G) = 0.29$. If, however, $S = \{p_3, p_4\}$, $rr_P(S, G) = 0$. However, if the goal is to select a set of 2 cars that is diverse in weight and height, then the set $S = \{p_1, p_4\}$ is the most diverse under the diversity definitions formulated in the previous section.

In general, computing the value $rr_P(S, G)$ for a class G is based on finding the "worst" point. That is, the point that contributes to the currently perceived maximum regret ratio after S points have been selected. Hence, that computation is achieved by running a linear program to compute $rr_{S \cup \{p\}}(S, G)$ for each $p \in P \backslash S$ to find the one point p' that is responsible for the current maxi-

Algorithm 1 Regret Greedy Algorithm

```
Input: A set of $D$ dimensional points $P = \{p_1, p_2, ..., p_n\}$ and an integer $k$ for output size
Output: A subset of $P$ of size $k$ denoted by $S$
$S \leftarrow \{p\}$ such that $p = \underset{p_i \in P}{argmax}\, p_i[1]$
for $i = 1$ to $k-1$ do
    $r* = 0$
    $p* = null$
    for all $p \in P \backslash S$ do
        if $r* < rr_{S\cup\{p\}}(S)$ then
            $r* \leftarrow rr_{S\cup\{p\}}(S)$
            $p* \leftarrow p$
        end if
    end for
    $S \leftarrow S \cup \{p*\}$
end for
```

mum regret ratio [9]. Hence, $rr_P(S,G) = rr_{S\cup\{p'\}}(S,G)$, where $rr_{S\cup\{p\}}(S,G)$ is evaluated using the following linear program [9]:

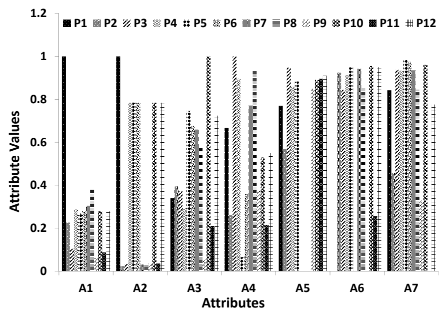
$$\begin{aligned} & \max \; x \\ \text{s.t.} \; & \sum_{i=1}^{D}(p[i] - p'[i])v[i] \geq x \quad \forall p' \in S \\ & \sum_{i=1}^{D} p[i]v[i] = 1 \\ & v[i] \geq 0 \quad \forall i \leq D \\ & x \geq 0 \end{aligned} \tag{2}$$

Notice that in the linear program above, each possible utility function is represented by a vector v in D dimensions with non negative coordinate values. Hence, the linear program finds the vector v that maximizes the regret ratio of S relative to $S \cup p$, and the value x returned by the linear program is precisely $rr_{S\cup\{p\}}(S,G)$.

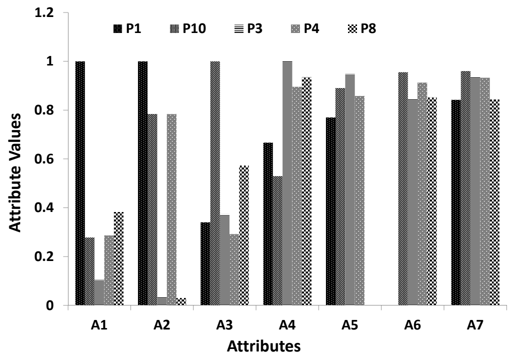
Similar to maximizing diversity, the problem of regret minimization has also been shown to be NP-hard [1]. Hence, several greedy heuristics have been proposed to find near-optimal solutions for regret minimization [9]. Those heuristics are based on the Greedy algorithm proposed in [9], in which S is constructed iteratively, where in each iteration, the point that contributes to the maximum regret ratio is selected and added to S, until k points are selected. The pseudo code for this algorithm is given in algorithm 1, where $rr_{S\cup\{p\}}(S)$ is evaluated using the above linear program.

3. COMBINING DIVERSITY AND REGRET

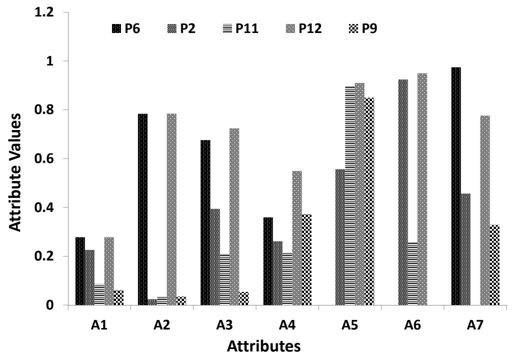
Clearly, the methods for diversity maximization and regret minimization compute different representative sets that optimize their respective objective functions. For example, consider Figure 1a, which shows a small data set consisting of 12 points $p_1, p_2, ..., p_{12}$, each with 7 dimensions $A_1, A_2, ..., A_7$. For each point p_i, the normalized attribute value for each of its 7 attributes is shown on the y-axis. Further, assume there is some notion of preference associated with the first four attributes A_1 to A_4 (e.g., the higher the value, the better), whereas there is no such notion defined for the remaining three attributes $A5$ to A_7. Hence, in generating a representative set S, it is desired to select points that: i-minimize regret over the first four dimensions (i.e, high utility), and ii-maximize diversity over the last three dimensions (i.e., low redundancy). Figure 1b shows a set S of size 5 points selected by the Greedy algorithm for regret minimization (called *Reg-Greedy* hereafter). The figure shows that selected points have high values under one or more of the first four attributes (i.e., high utility), but have very similar values in the last three attributes (i.e., high redundancy). Alternatively, Figure 1c shows the set S selected by the Greedy algorithm for diversity maximization (called *Div-Greedy* hereafter). In contrast to



(a) All Records



(b) Regret minimized set (k=5)



(c) Diverse set (k=5)

Figure 1: Impact of Regret Minimization and Diversification

Figure 1b, Figure 1c shows that the selected points have diverse values under the last three attributes (i.e., low redundancy), but miss the user preference for higher values along the first four dimensions (i.e., low utility). Hence, neither of the two algorithms manages to achieve both high utility and low redundancy at the same time.

To capture the conflict illustrated in the previous example, we utilize a hybrid function that considers both diversity and regret. Specifically, for a subset $S \subseteq P$, an objective function is formulated as the linear weighted combination of the scaled diversity and regret objectives, which is defined as:

$$\begin{aligned} \mathcal{F}(S,G,P,\lambda) = \lambda \frac{\sum_{i=1}^{k}\sum_{j>i}^{k} d(p_i,p_j)}{\underset{p_i,p_j \in P}{max} d(p_i,p_j)} + \\ (1-\lambda)\frac{k(k-1)}{2}(1 - rr_P(S,G)) \end{aligned} \tag{3}$$

where $k = |S|$, and λ is the weight parameter for balancing the tradeoff between diversity and regret, such that $0 \leq \lambda \leq 1$. Notice that the sum of the pairwise distances is divided by the maximum distance between any pair of points in the whole set so as to normalize the value of the distances between 0 and 1. Similarly, the regret objective is scaled up by $\frac{k(k-1)}{2}$ which is the total number of pairs considered in the sum of the distances for diversity.

When diversity and regret are associated with different dimensions, our objective function defined above is easily modified accordingly. In particular, if $V_{div} \subseteq \{A_1, ..., A_D\}$ represents the dimensions designated for diversification and $V_{reg} \subseteq \{A_1, ..., A_D\}$ represents the dimensions on which regret minimization is required, our objective function is reformulated as:

$$\mathcal{F}(S, G, P, \lambda, V_{div}, V_{reg}) = \lambda \frac{\sum_{i=1}^{k} \sum_{j>i}^{k} d(p_i, p_j, V_{div})}{\underset{p_i,p_j \in P}{max}\ d(p_i, p_j, V_{div})} +$$
$$(1-\lambda)\frac{k(k-1)}{2}(1 - rr_P(S, G, V_{reg}))$$

Hence, the goal is to find a set S^*, which balances the tradeoff between diversity and regret by maximizing the objective function $\mathcal{F}$ defined above. Formally:

$$S^* = \underset{S \subseteq D, |S|=k}{argmax}\ \mathcal{F}(S, G, D, \lambda, V_{div}, V_{reg}) \qquad (4)$$

Note that in this work we focus on the case where diversification and regret minimization are desired on different dimensions, which means $V_{div} \cap V_{reg} = \phi$. Referring back to the sample data in table 1, assume $V_{div} = \{Weight, Height\}$, $V_{reg} = \{MPG, HP\}$, and it is required to select a representative set S of size 2 (i.e., $k = 2$). For that data, $S = \{p_1, p_3\}$ is the the set which minimizes the maximum regret ratio in MPG and HP, but provides very low diversity in Weight and Height. On the other hand, $S = \{p_2, p_4\}$ maximizes the diversity in Weight and Height, but has a very high maximum regret ratio in the first 2 dimensions. Giving equal weight to both diversity and regret by setting $\lambda = 0.5$ in $\mathcal{F}$, we find that the subset of size $k = 2$ which maximizes $\mathcal{F}$ is $S = \{p_1, p_4\}$, where the points in S provide the desired balance between minimizing regret and maximizing diversity.

4. THE ReDi SCHEME

In this section, we present our ReDi scheme for balancing the tradeoff between minimizing regret and maximizing diversity. In particular, we present two algorithms that aim to achieve that goal as it is captured by the hybrid function presented in Section 3. Towards this, we present two algorithms: ReDi-Greedy, and ReDi-SWAP, and study the tradeoff they exhibit in terms of both efficiency and effectiveness.

4.1 ReDi-Greedy

ReDi-Greedy follows the same general design adopted by the class of constructive algorithms for solving different optimization problems, including those for regret minimization, and diversity maximization. In constructive algorithms, a final solution is achieved incrementally in steps, where in each step a local decision is made based on some criteria, where the choice of such criteria depends on the target optimization problem. ReDi-Greedy also constructs the result set S iteratively by selecting a new point in each iteration, where the criteria for selecting such point is based on the objective function $\mathcal{F}$ defined in Section 3. Particularly, based on $\mathcal{F}$, it is desirable to select in each iteration a point that can potentially contribute the most to decreasing the regret of the current set S and also increasing its diversity. In order to locate such point, ReDi-Greedy employs a heuristic priority function, where each point in P that is not in S is assigned a score, which is defined as follows:

Algorithm 2 ReDi-Greedy

```
Input: A set of $D$ dimensional points $P$, a set $V_{div}$, a set $V_{reg}$, an integer $k$ and $\lambda$
Output: A subset of $P$ of size $k$ denoted by $S$
$S \leftarrow \{p\}$ such that $p = \underset{p_i \in P}{argmax}\ p_i[1]$
for $i = 1$ to $k - 1$ do
    $maxScore = 0$
    $p* = null$
    for all $p \in P \setminus S$ do
        if $maxScore < Score(p, S, , V_{div}, V_{reg})$ then
            $maxScore \leftarrow Score(p, S, V_{div}, V_{reg})$
            $p* \leftarrow p$
        end if
    end for
    $S \leftarrow S \cup \{p*\}$
end for
```

Table 2: $Score()$ vs. $\mathcal{F}()$ at $S = \{p_2\}$

Point	p_1	p_3	p_4	p_5
$Score(p)$	0.99	0.96	1	0.528
$F(S \cup p)$	0.627	0.726	0.724	0.395

$$Score(p, S, V_{div}, V_{reg}) = \lambda \frac{SetDist(p, S, V_{div})}{max_{p_i \in P \setminus S} SetDist(p_i, S, V_{div})} +$$
$$(1-\lambda)\frac{rr_{S \cup \{p\}}(S, G, V_{reg})}{rr_P(S, G, V_{reg})}$$

Where SetDist(p,S), between a point p and a set of points S is defined based on its distance from the points in S, as: $SetDist(p, S) = \frac{1}{|S|}\sum_{p_j \in S} d(p, p_j)$.

Thus each candidate point p is assigned a score, which is the weighted sum of its set distance from S and the maximum regret ratio of set S with respect to p. Both the set distance and regret components are normalized by dividing the first term by the maximum set distance of S and the second component by the maximum regret ratio of S with respect to the whole set P. The score of each candidate point p thus measures the potential contribution of p in increasing the objective function value for set $S \cup p$. Therefore in each iteration the point with maximum score is added to set S (the pseudocode for ReDi-Greedy is presented in algorithm 2).

Although ReDi-Greedy is very efficient in practice, there is no guarantee that the point $p*$ picked in each iteration is actually the best point (i.e., local minimum) for the objective function $\mathcal{F}$ given the current representative set. This is because the point $p*$ that has the highest score may not necessarily be the one that improves the combined function value the most. Consider again the example in table 1 and assume that our current set $S = \{p_2\}$ and $\lambda = 0.5$. Table 2 lists the scores of each of the other points based on the above scoring function as well as the actual function values obtained by adding that particular point to the current S. As the table shows, the highest scoring point, p_4 does not yield the highest function value when added to S, which is given by p_3.

To address the limitations of ReDi-Greedy, next we present the ReDi-SWAP heuristic, in which the selection of points is based on the actual improvement in the value of the objective function $\mathcal{F}$ instead of the expected improvement.

4.2 ReDi-SWAP

The ReDi-Greedy algorithm presented in the previous section is of the constructive type. That is, it starts without a representative set and incrementally constructs it by adding one point at a time. To the contrary, ReDi-SWAP presented in this section falls under the local search type of algorithms. In general, a local search algorithm

Algorithm 3 ReDi-SWAP Algorithm

```
Input: A set of $D$ dimensional points $P$, a set $V_{div}$, a set $V_{reg}$, an integer $k$ and $\lambda$
Output: A subset of $P$ of size $k$ denoted by $S$
$S \leftarrow regretGreedy(P, k, V_{reg})$
$S' \leftarrow S$
for all $p \in P \setminus S'$ do
    $S_{temp} \leftarrow S$
    $p^* = \underset{p \in P \setminus S'}{argmax} \sum_{p_i \in S} d(p, p_i, V_{div})$
    $S' \leftarrow S' \cup \{p^*\}$
    for all $p' \in S$ do
        if $F(S_{temp}, G, P, \lambda, V_{div}, V_{reg}) < F(\{S \setminus p'\} \cup p^*, G, P, \lambda, V_{div}, V_{reg})$ then
            $S_{temp} = \{S \setminus p'\} \cup p^*$
        end if
    end for
    if $F(S_{temp}, G, P, \lambda, V_{div}, V_{reg}) > F(S, G, P, \lambda, V_{div}, V_{reg})$ then
        $S \leftarrow S_{temp}$
    end if
end for
```

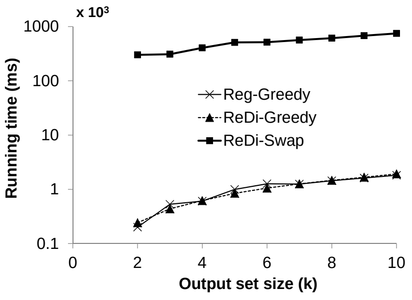
starts out with a complete initial solution and then attempts to find a better solution in the neighborhood of that initial one. Like constructive algorithms, local search algorithms are also widely used in solving optimization problems including generation of representative data. For instance, the SWAP local search method has been utilized to maximize diversity [2, 14], and in this paper, we further expand into our ReDi-SWAP method for balancing the tradeoff between regret and diversity.

The basic idea underlying ReDi-SWAP is to start with an initial set S of size k and then iteratively modify the set S in order to improve the value of the objective function $\mathcal{F}$. One of the main design criteria in local search algorithms is the choice of the initial solution. In ReDi-SWAP, we opt to initialize S with the k points which minimize the regret ratio as selected by the traditional Reg-Greedy algorithm [9]. Another important criterion is the neighborhood definition of local search together with the search process. Since we initialize ReDi-SWAP with the regret minimization set, the neighborhood of the local search is explored based on the second component of our objective function (i.e., diversity). In particular, the points in $P \setminus S$ are visited according to their distance from S, such that the points with higher distance are visited first. Accordingly, in each iteration, from the set of points that haven't yet been visited, given by $P \setminus S'$, the point with the highest distance from the current set S is tested, denoted by $p*$. This point is tested against each point in S by removing that point p' from S and adding $p*$ to the set. After going through all the points in S one by one, the swap that results in the highest function value is made if it also is an improvement over the set S before the swaps were made. The intuition is to replace the point in S which contributes the least to its diversity with a point from $P \setminus S$ which would improve that diversity while at the same time maintaining regret very close (or equal) to that of S, which is easily captured by evaluating the hybrid objective function $\mathcal{F}$.

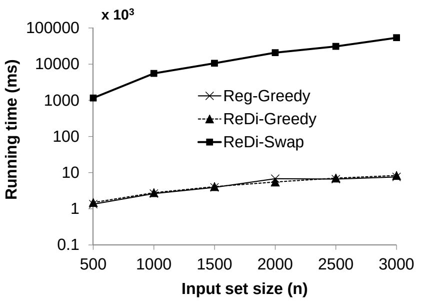
Notice that for any candidate point p, the decision made by ReDi-SWAP is based on evaluating the objective function $\mathcal{F}$ if p is selected to join S. Evaluating $\mathcal{F}$ requires $O(n)$ calls of the linear program to compute the maximum regret ratio and $O(k^2)$ distance computations to calculate the diversity of set S. Under ReDi-Greedy, however, the decision on a candidate point p is based on assigning a priority value (i.e., score) to p. For that decision, it is enough to evaluate the regret contributed by p but not the actual regret achieved when p is added to S, which requires one call of the linear program. Hence, in total, ReDi-Greedy performs $O(nk)$ calls to the linear program, whereas ReDi-SWAP performs $O(n^2k)$. The tradeoff between the efficiency and effectiveness provided by these two algorithms is evaluated experimentally in the next section.

Table 3: Evaluation Setting.

Parameter	Range	Default
Number of Dimensions (D)	2–10	10
Database Size	3k	300
Representative Set Size (k)	2–10	5
Weighting Factor (λ)	0–1	0.5
Number of Regret Dimensions $\lvert V_{reg} \rvert$	2–8	5
Number of Diversity Dimensions $\lvert V_{div} \rvert$	2–8	5



(a)



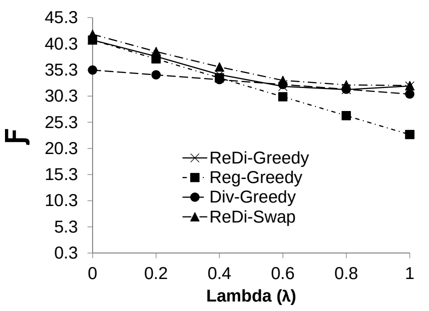
(b)

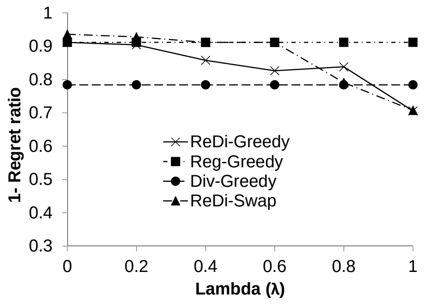
Figure 2: Cost of different methods in terms of running time

5. EXPERIMENTAL EVALUATION

We perform several experiments to evaluate the performance of the different schemes discussed in this paper, namely: i) Div-Greedy ii) Reg-Greedy, iii) ReDi-Greedy, and iv) ReDi-SWAP. In particular, we compare those algorithms in terms of effectiveness, which is measured in terms of the objective function $\mathcal{F}$, and efficiency, which is measured as total time spent in executing the linear program optimization. Experiments are conducted on a simple synthetic 10-dimensional dataset of size 3K tuples, in which the attribute values of each dimension are generated uniformly in the range [0–1]. Table 3 summarizes the database settings together with the other parameters considered in our evaluation.

Figure 3a shows the impact of λ on the value of the objective function $\mathcal{F}$. As shown in the figure, ReDi-SWAP provides the highest values for $\mathcal{F}$ followed by ReDi-Greedy. Meanwhile, Reg-Greedy and Div-Greedy are both oblivious to any tradeoff between regret and diversity, which translates into lower $\mathcal{F}$ values. This is further illustrated in Figures 3b and 3c. Figure 3b shows the happiness measure of a set S computed as 1-regret ratio(S). As expected, the figure shows that Reg-Greedy provides the highest happiness, whereas Div-Greedy provides the lowest. The figure also shows


(a) Function Value


(b) Regret Ratio

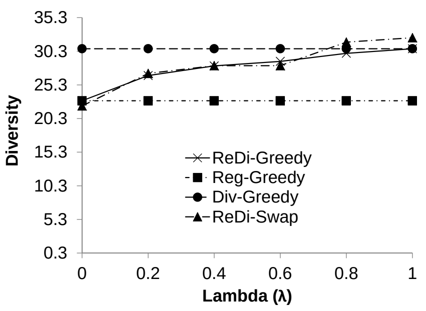

(c) Diversity

Figure 3: Impact of λ

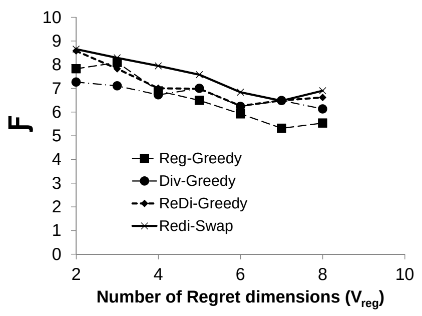


Figure 4: Impact of dimensions split

that ReDi-Greedy and ReDi-SWAP provide slightly lower happiness than Reg-Greedy but much higher than Div-Greedy. The same behavior is exhibited in Figure 3c with respect to diversity.

The improvement in the $\mathcal{F}$ value provided by ReDi-SWAP over ReDi-Greedy (as shown in Figure 3a) comes at the expense of increasing the processing time, as shown in Figure 2, which shows the time spent in executing the LP optimizations and distance computations. This is to be expected since ReDi-SWAP has higher complexity than ReDi-Greedy, as discussed in the previous section.

In all the previous experiments, the 10 dimensions were split evenly between regret and diversity (i.e., $|V_{reg}| = 5$ and $|V_{div}| = 5$), as in the default setting. In this experiment, we measure the impact of changing that split as shown in Figure 4. The figure shows the $\mathcal{F}$ value as we increase the number of dimensions considered for regret minimization, or equivalently decreasing the number of dimensions for diversity maximization. As the figure shows, the overall trend for all algorithms appears to be a decrease in $\mathcal{F}$ value as $|V_{reg}|$ is increased. This is consistent with results in literature on the impact of dimensions on regret minimization and diversity maximization. In particular, maximum regret ratio has been observed to worsen, or increase, with the increase in dimensionality whereas diversity increases with increase in dimensionality. Both of these observations are consistent with the trend in figure 4, where the $\mathcal{F}$ value decreases as the number of dimensions for regret increase and those for diversity are reduced.

All the previous results indicate that ReDi-SWAP is a more effective scheme than ReDi-Greedy. However, that improvement in effectiveness comes at the expense of a higher computational cost. Such a high cost would be prohibitive when applying ReDi-SWAP on large databases. Hence, from a practical standpoint, ReDi-Greedy presents a more attractive solution due to its scalability, while at the same time providing $\mathcal{F}$ values close to those achieved by ReDi-SWAP.

6. CONCLUSIONS AND FUTURE WORK

We considered the problem of simultaneous regret minimization and diversity maximization on multi-dimensional data where both these objectives are desired on different dimensions. To that end, we captured the tradeoff between those two objectives by means of a hybrid function, which also forms the basis of our new ReDi scheme. As part of our ReDi scheme, we proposed two alternative solutions that fall under two major classes of optimization algorithms, namely: local search optimization, and constructive optimization. Our experimental evaluation studies the efficiency and effectiveness of those algorithms under different parameters.

This work is an initial step towards integrating diversity maximization and regret minimization. In the future, we plan to further investigate the role of the weighing parameter λ together with automated methods for setting it. We also plan to expand the ReDi scheme to include the geometry-based approach for regret minimization [10] towards achieving higher efficiency.

Acknowledgement: This work is partially supported by Australian Research Council grant LP130100164.

7. REFERENCES

[1] S. Chester et al. Computing k-regret minimizing sets. *VLDB*, 7(5), 2014.

[2] M. Drosou and E. Pitoura. Search result diversification. *SIGMOD Record*, 39(1), 2010.

[3] E. Erkut, Y. Ülküsal, and O. Yeniçerioglu. A comparison of *p*-dispersion heuristics. *Computers & OR*, 21(10), 1994.

[4] R. Fagin, A. Lotem, and M. Naor. Optimal aggregation algorithms for middleware. In *PODS*, 2001.

[5] I. F. Ilyas et al. A survey of top-*k* query processing techniques in relational database systems. *ACM Comput. Surv.*, 40(4), 2008.

[6] H. A. Khan, M. Drosou, and M. A. Sharaf. Scalable diversification of multiple search results. In *CIKM*, 2013.

[7] H. A. Khan and M. Sharaf. Progressive diversification for column-based data exploration platforms. In *ICDE*, 2015.

[8] H. A. Khan, M. A. Sharaf, and A. Albarrak. Divide: efficient diversification for interactive data exploration. In *SSDBM*, 2014.

[9] D. Nanongkai et al. Regret-minimizing representative databases. *VLDB*, 3(1-2):1114–1124, 2010.

[10] P. Peng and R. C.-W. Wong. Geometry approach for k-regret query. In *Data Engineering (ICDE), 2014 IEEE 30th International Conference on*, pages 772–783. IEEE, 2014.

[11] A. D. Sarma et al. Beyond skylines and top-k queries: representative databases and e-commerce product search. In *CIKM*, 2013.

[12] S.Borzsony, D.Kossmann, and K.Stocker. The skyline operator. In *ICDE*, 2001.

[13] Y. Tao et al. Efficient skyline and top-k retrieval in subspaces. *IEEE Trans. Knowl. Data Eng.*, 19(8), 2007.

[14] M. R. Vieira et al. On query result diversification. In *ICDE*, 2011.